1.0: Executive Overview [B3]

The original need for rain simulators came about in the 1930s. Kansas had experienced horrible dustbowls many times during 1934 and 1935. These dustbowls were a result of the severe erosion of the soil due to run-off chemicals and substances from various materials in the area. This severe erosion of the soil transformed the once healthy soil into a sand or dust-like consistency. The community reacted and there was a research station established in order to learn about the run-off chemicals of materials and their impact on their surroundings. Below is an excerpt depicting the end results of chemical run-off and sequential erosion of the soil.

> “Noonday became like night. A person couldn't see anything over 200 yards away. Battering waves of fine soil particles almost smothered people, livestock, fences, and farm buildings. Drifts of constantly shifting fine soil particles filled roadside ditches and roads, covered fences, buried farm machinery, and destroyed plant growth with their abrasive action. Much of the soil's fertility was in the fine dust that was picked up by the wind and carried hundreds and even thousands of miles. The heavier infertile sand particles were left behind. Many people and animals developed respiratory problems. Dust filtered through closed doors and windows. Crop and livestock farming became difficult to impossible. Most of the benefit of heavy rains in May and June of 1935 was lost because there was little vegetation and few structures to hold the moisture where it fell.”

Unfortunately, once the research station was set up, it did not rain frequently enough to do the testing needed on the materials. Thus, the need for rain simulators was born. However, Kansas was not the only place that had a need for a rain simulator. Other rain simulators began to emerge nationwide once people became aware of the deterioration of the farming soil around them. President Theodore Roosevelt declared, "When the soil is gone, man must go. And the process does not take long." History seemed doomed to repeat itself since this was the very reason for the fall of the Babylonian Empire and the great migration of the first people of America inland after they had exhausted the nutrients of the soil on the east coast. It was not until people realized the possible future consequences for their current actions that they started to react and think about the erosion the valuable soil was going through and the fact that they needed to do something about it. Whether it was due to a severe dustbowl or awakening from ignorance, rain simulators began to emerge in various places across the United States. Along with the birth of rain simulators also came the need for a dependable control system that could accurately represent the rainfall activity unique to that area. This fact is the very basis for our senior design project.

1.1: Project Summary [B1] [B2]

The UCF Stormwater Lab located on campus has geared its research towards protecting the state's valuable natural resources. They have housed projects such as the green roof, pervious pavements research, and the notorious rain simulator. This lab features the largest rainfall simulator in North America and second largest in the world next to Japan's large-scale rain simulator. The Stormwater Lab has been in operation for 3 years now and they are responsible for testing and evaluating the level of run-off chemicals coming from many different materials. As Chad Binette, from the UCF news team reported, "The simulator allows researchers to study how water falls on different surfaces, how it runs off and how it dislodges particles that can become sediment and clog up drains". Below in Figure 1.1a, is a picture of the rain simulation system.

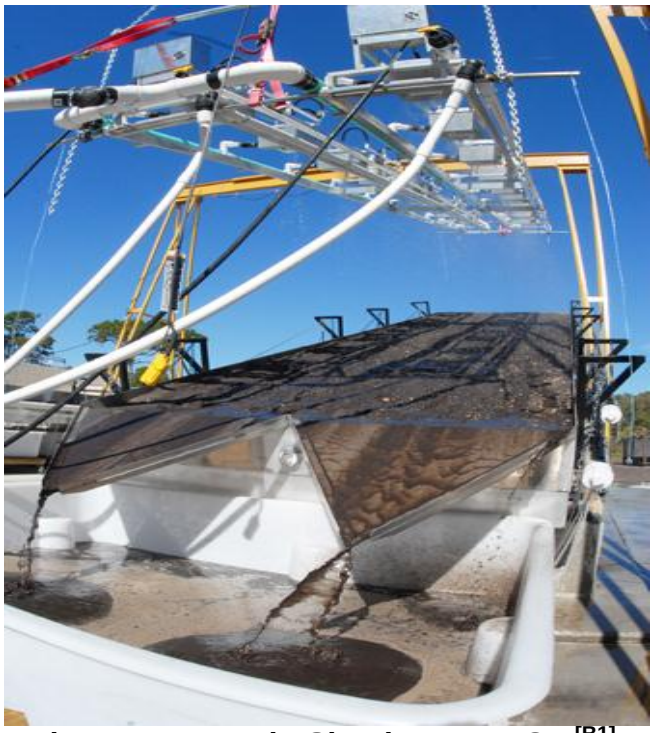

Figure 1.1a - Rain Simulator at UCF [B1]

The Stormwater Lab has mainly done work for the state of Florida Department of Transportation (FDOT) and the Florida Department of Environmental Protection (FDEP). They have also done work for private industries mainly companies that make products to control sediment and prevent erosion by testing the effectiveness of their products. When the need for a rain simulator was initially established, the Stormwater lab turned to Advance Design and Machine, an external company, to design and build the system for them. The resulting rain simulator had a massive rain bed measuring 8 feet wide and 30 feet long and a 1,500-gallon water tank to provide the artificial rainfall. The design also included two massive stepper motors controlling two rods extending down the length of

the simulator with pressurized nozzles that release the rain. The entire simulation is controlled through a central computer loaded with the proprietary software of the third party company. The complete system cost the Stormwater lab approximately $100,000. In summary, the rain simulator is an advanced computerized system that is capable of mimicking the torrential downpours unique to our Floridian environment. In addition to creating the different rainfall intensities, it also disperses the rainfall evenly over the large test beds. The simulations run on these test beds have been very valuable to the stormwater management research team in helping them measure the chemical run-off from different materials.

Throughout the years, there have been some reported issues with the control system of the simulator that have caused significant downtime to the lab's testing schedule. The first issue arose with the power supply for the system. At times, the power supply would not function at all while other times it would function nominally. This unpredictable behavior left the Stormwater lab with no choice but to have it replaced and repaired multiple times. Another issue occurred with the serial communication from the control unit to the computer. They opted for swapping the previous PC serial port for a USB/RS232 adapter instead and this has been working fine but they are still tied to using the computer to control their system. The final major problem that the Stormwater lab has experienced was with the proprietary software used to control the rain simulator. The sweeping motion of the water nozzles and the resulting rainfall intensity created do not match up with the user's desired rainfall intensity. The exact cause behind this problem remains unknown but the software and communication port are the two biggest suspects.

1.2: Project Description

The rain simulation system is a combination of electrical, computer, and mechanical engineering sub-systems. Our team was challenged with the task of completely redesigning and rebuilding the entire control system for the rain simulator. Since we all majored in electrical engineering, the software integration was definitely the biggest challenge for us.

1.3: Goals & Objectives

Over the years, the rain simulator has encountered the aforementioned issues that have made it difficult for the lab to fulfill their delivery dates to their customers. Through our senior design project, we desired to find and implement solutions to solve the problems as well as add some extra features to avoid problems in the future and to improve the ease of use of this rainfall simulator.

First off, the faulty power supply would require us to either completely redesign it or replace it with a pre-existing model. If we were to design and build an entirely

new power supply, we would have to perform a power analysis and we would need to make sure that we have the appropriate voltage outputs needed for our control system. If we were to replace it with a pre-existing design, we would need to be sure to learn all of the details of the design. Either way, we would need to ensure that the power supply was dependable and would be able to be used for any simulation at any time. Our plan was to research different power supply designs, take the best attributes from each of them, and combine them to create our own original design in order to meet the power needs of the system throughout the entire simulation period.

Secondly, the user interface needed to be reconsidered. The existing control system connected to the lab's main computer via serial port where the custom software acts as the graphical user interface between the user and the control system. The software was intended to make it possible for the user to effortlessly adjust critical parameters for the rainfall simulation event. Some of the parameters include stepper motor speed, direction, and timing. The software conveniently calculates these low-level details from only two inputs coming from the user: intensity of the rainfall and the simulation time for the required test. Unfortunately, achieving the desired rainfall intensity has not been an easy task for the user. Due to the fact that an external company specifically designed the software for the rain simulator, the details and inner workings of the architecture used remain unknown and were deemed difficult to decipher. Also, the very fact that the software was developed for a specific computer platform also forced the research lab to keep their main computer outdated only so that they were able to run the software for the simulator. Any updates made to a new operating system could be hazardous to the functionality of the software. With that being said, the team decided it would be better to remove the computer's role in the entire control system. The interface between the user and the simulator in our design was replaced with a simple LCD and keypad which was easily mounted onto the control system enclosure. This came in the form of a 12 digit push pad in which the user could enter the desired duration and rainfall intensity. The most important aspect of the user interface was the ease of use for the lab workers. By eliminating the computer, we added great mobility to the control system. If needed, the researchers are able to actually pick up the casing with the control system and bring it anywhere that is convenient for them whether it be outside next to the simulator or to a future site of the rain simulator if they decide to change their location later. Additionally, if any maintenance needs to be performed on the control system, a team member is able to take the control system home and work on it without having to worry about taking the computer and specific software needed to troubleshoot the system. And as for the serial communication with the computer, it is longer needed seeing that the computer is completely removed. The topic of concern was longer the communication from the control to the computer but from the LCD and keypad user interface to the microcontroller.

Lastly, we had to consider the fact that we were not building this control system as a prototype for future systems. We were building this control system as the one and only final build of the full-scale project. We were forced to consider factors of convenience and reliability for the researchers at the lab. We also had to consider the environmental concerns since the electronics could be exposed to all of the elements: water, wind, and constant humidity. This includes procuring wires that are rated waterproof enough for this kind of application. As far as convenience goes, we would like to go above and beyond for the Stormwater lab. We wanted to take tasks that were done manually and automate them. An example of this would be to take their manually read rain gauges and make them wireless with the data being displayed on the LCD. This would not only eliminate the need to manually check the rain gauges but it would also guarantee better accuracy in reading the water level and provide live updates. In the end, this project was not just about meeting the three requirements given to us, it was about engineering a dependable and user-friendly product that the Stormwater research lab team could enjoy using and that enabled them to meet their customer's needs easily and efficiently.

1.4: Requirements & Specifications

In essence, the team was only given three specific requirements for the final control system:

- Need to have a fully functioning software program to accurately control the simulation duration and rainfall intensity with a tolerance of $\pm$ 0.5 inches per hour, with a range of 0.0 to 12.0 inches per hour.

- Need a new circuit board with effective communication to the motor drivers and eventually the stepper motors.

- To provide a dependable power supply to ensure performance of the simulator under orders from the lab's customers.

Because these requirements were so broad, we really had a lot of freedom to do what we thought was best. There was no specific microcontroller requested or even user interface requested. While this gave us more flexibility, it also made the decisions on choosing each component much more difficult. It also lengthened the research that needed to be done since we needed to explore a greater variety of components. All in all, the team tried to at least research three different parts for each component. The differences among these parts could have been due to their technical specifications and performance or maybe just their manufacturer and the ease of their procurement. When it came down to it, we made sure to emphasize the importance on the ease of integration ease of replacement in order to prepare for future mishaps where the team may not be around.

1.5: Existing Similar Projects [B7] [B8]

Although the Stormwater lab's rain simulator is one of a kind, there are other weather simulators that are similar and comparable. The University of Florida has developed a simulator, which can reproduce the effects of hurricanes, rain, and high-speed winds. Their simulator is mainly used to test for the effect of hurricanes on different types of construction. Their simulator consists of eight fans, each measuring five feet by five feet, stacked four wide and two tall. The fans are powered by marine diesel engines that all together produce 2800 horsepower. This simulator can reach wind speeds up to 130mph which would be equivalent to a category three hurricane. It can also double as a rain simulator and can produce a rainfall intensity of up to 35 inches per hour. This simulator is similar in the aspect that it can also simulate rain, but it does not serve the same purpose as the rain simulator at UCF, it is depicted in Figure 1.5a.

Figure 1.5a - Hurricane Simulator at UF [B7]

Japan has also developed their own rain simulator to test a variety of things such as fiber optics and radar communication in bad weather, mudslide probability and possible preventative measures, and drainage inspection of sewer and piping access ports as well as erosion oriented issues. Figure 1.5b is a representation of Japan's simulator.

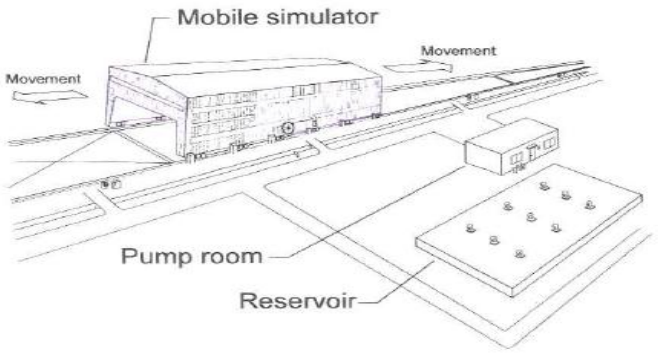

Figure 1.5b - Japan's rain simulator [B8]

Japan's rain simulator is the largest in the world measuring at 44 meters by 72 meters with a possible rainfall intensity of 15-200 millimeters per hour. The testing space of this simulator can be divided into multiple parts, enabling it to be used efficiently for small-scale models and actual size models alike. They have also adopted a unique mobile system where the simulation warehouse can be rolled to 5 different locations enabling them to set up 5 full size simulations without having to take one down before setting the next one up. Lastly, they also have a fully automated system where they only need one operator to run the entire simulation.

1.6: Team Introduction

After four years of electrical engineering classes, our team came together for our final senior design project. Our senior design team consists of four Electrical Engineering students of various ages and backgrounds. We all share a common passion to work diligently and persevere in the goals we have set for ourselves. Senior design was the perfect opportunity to choose a challenging project and use one another's various talents to provide a convenient solution.

Luke Falls

Luke Falls, see figure 1.6a, is an Electrical Engineering student at UCF with a determination for success.

Figure 1.6a - Luke Falls

Originally from Wisconsin, Luke moved to Florida for a fresh start and found work in the area of air conditioning and refrigeration systems. After 7 years in the field, he began his undergraduate degree at UCF. His experience in the field has made him an expert in troubleshooting, planning, and problem

solving. During his junior year at UCF, Luke began a co-op at United Space Alliance at the Kennedy Space Center. At his co-op, Luke worked on projects like power systems planning and layout. Luke's main focus in the senior design project has been in the area of microcontrollers and their various characteristics and applications to the rain simulator. Additionally, Luke has also taken the major task in putting together the smaller sub-systems into one central communicating control system, much like a systems engineer. In the future, Luke hopes to work in the area of Power Systems.

David Levy

David Levy, see figure 1.6b, is an Electrical Engineering student at UCF with a passion for learning.

Figure 1.6b - David Levy

Born and raised in Malabar Florida, David first attended Brevard Community College for his Associate's degree and continued his education at UCF to complete his undergraduate degree. During his junior year at UCF, David had an internship at the United Space Alliance at the Kennedy Space Center. During his co-op, he worked on maintaining hazardous gas detection equipment and gas spectrometry systems. He also performed projects focused on control systems that implemented Allan Bradley programmable logic controllers. Also, he earned his certification as a shuttle launch and recovery systems processing engineer. David's main focus in the senior design project has been the stepper motors and the power supply design. David has successfully dissected the inner workings of the stepper motor and applied their form, fit, and function to the specific application of the rain simulator. Additionally, David has done extensive research and design of the power supply needed for the control system. In the future, David hopes to obtain a career as an electrical engineer, specializing in control system design.

Adam Cutting

Adam Cutting, see figure 1.6c, is an Electrical Engineering student at UCF with a hunger for all things electrical.

Figure 1.6c - Adam Cutting

Adam has lived in Florida all of his life and started his educational career at the University of Central Florida. During his junior year, Adam participated in the co-op program at United Space Alliance at Kennedy Space Center. His projects at the space coast included a three-phase portable solar power system and other energy saving projects. Adam's focus in the senior design project is the user Interface and its integration to the rest of the system. He has also done extensive research on the motor drivers and their specific application to the rain simulator. In the future, Adam hopes to continue enjoying life wherever that may take him.

Brenda Garcia

Brenda Garcia, see figure 1.6d, is an Electrical Engineering student at UCF with a minor in International Engineering. Brenda grew up in Denver, Colorado and move to Orlando, Florida to pursue her undergraduate degree at UCF. Brenda has participated in a co-op experience at Lockheed Martin where she worked with the turreted systems program. She is currently working at Siemens Energy in generator supply chain management. Brenda's focus in this senior design project has been in the area of the wireless integration of the rain gauges and various administrative tasks. In the future, Brenda hopes to attain a position at Siemens Energy and work with power systems.

Figure 1.6d - Brenda Garcia

2.0: Strategic Components

The following research corresponds to the main components that were necessary for the control system for the rain simulator. The stepper motors, motor drivers, user interface, microcontroller, power supply, spray nozzles, and transmission lines make up the core design of the original rain simulator.

2.1: Stepper Motors [D5]

One of the primary considerations to be taken into perspective for this project was what type of motor is to be used for the control of the nozzle swing arm. After some research, we found that there are a wide variety of motors in existence today. However, in general there are about six different domains of motors, brushed, brushless, servo, linear and all of their alternating-current and direct-current variants, and finally the DC stepper motor. Fortunately for us, when we were presented with the task, the UCF Storm Water Management Academy provided us with the entire mechanism, including the motors. The motors used for the rain simulator were in fact stepper motors. Thus, our crucial concern was

to learn the fundamentals of how a stepper motor operates, how to control the motors and what the capabilities of our motors were.

During our ensuing research of the rudiments of a stepper motor, we found that there are three main types of stepper motors. Each type of motor has its own set of advantages and disadvantages. However, each type of stepper motor exhibits the essential property of being able to translate the various input signals into specifically defined increments of shaft rotation, or position. In simpler terms, the purpose of the stepper motor is to be able to precisely control the position of the shaft, and inherently the position of whatever mechanism that may be attached to the shaft, by using an input signal.

The stepper motor is indeed set apart from all of the other domains of motors in that the stepper motor allows the user to precisely control the position of the shaft, in an open loop control configuration; with every other type of motor, the position of the shaft cannot be determined simply by the input signal, a feedback loop must be implemented. This made the stepper motor the ideal motor for our design, namely because the design of the control system could be simplified to an open loop system.

As with any type of motor, we found that the stepper motor is composed of two fundamental parts. The stationary part of the motor, the stator, is composed of the conductive winding mechanism that provides an electromagnetic force that excites motion in the rotational part of the motor, the rotor. Thus, the casing of the motor houses the stator and the shaft that performs the desired mechanical rotation is attached to the rotor.

Stepper motors are classified as doubly salient machines, which means that the materials that both the stator and rotor are composed of allow a magnetic field to be transmitted through them. Without this doubly salient configuration, the stepper motor would not exist; the very ability to be able to move the rotor to, or hold the rotor in, a specific position is reliant on this doubly salient property. The ability to hold the position of the rotor and move the rotor, in steps, hinges on the basic electromagnetic property of magnetic reluctance. This property is comparable to the properties of resistance in an electrical circuit. Just as an electric field will cause electrical current to always flow along the path of least resistance, a magnetic field will always influence magnetic flux to flow in the path of least magnetic reluctance. The operation of a stepper motor capitalizes on this property in that the winding mechanisms, which are housed in the stator, produce polarized magnetic fields in a specific sections of the stator, which then attract the oppositely polarized magnetic fields present in the rotor; this magnetic attraction is indeed the force that induces the desired rotation of the shaft of the motor. To visualize this property, a diagram is shown in Figure 2.1a.

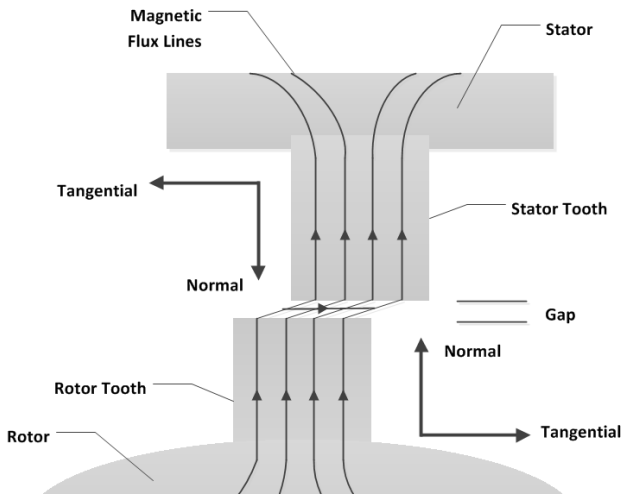

Figure 2.1a – Magnetic Field
Depiction of the magnetic forces upon the stepper motor teeth

The magnetically permeable materials have both normal and tangential magnetic forces acting upon them. These forces will influence the two bodies to position themselves such that a path of least magnetic reluctance will be formed. When that path is formed the two bodies in the Figure will have their edges aligned such that the tangential forces acting upon them will cancel and the magnetic flux will flow freely from one body to the other. This flow of magnetic flux incites the normal force that aids in holding the two bodies in position, relative to one another. The magnetic circuit is then complete and the magnetic flux flows from the north-polarized magnetically permeable material to the south-polarized material. This method of using the attractive forces of oppositely polarized magnetic fields is the basic principle that all three types of stepper motors use to accomplish the task of accurately positioning the rotor, and shaft, of the motor. However, each type of stepper motor achieves this goal by using a different mechanism for manipulating the magnetic fields. The three differing types of motors are called permanent magnet, variable reluctance and hybrid stepper motors.

As previously stated, and implied by the name, the stepper motor allows the ability to hold the rotor and shaft in a specified location and to change the position of the rotor and shaft by stepping from one position to the next. A sequential change in the position of the rotor can be achieved by applying the correct signals in order. By reversing the order of the input signal, we are able to rotate the rotor in either the clockwise or counter clockwise directions.

The resolution for the positioning of the rotor, of any type of stepper motor, is determined by how many steps per full rotation of the shaft that the motor is capable of producing. There are three modes of stepping for the motors, full, half

and micro-stepping. For now, we will focus on full stepping, because half and micro-stepping are simply improvisations implemented in the controlling signals to the motor that allow for better resolution on the shaft positioning. Whereas full stepping is the fundamental method for producing a change in the shaft position, and it is dependent upon the actual fabrication of the motor

Perhaps the easiest way to visualize how a stepper motor performs a full step is to examine an extremely simple version of the permanent magnet stepper motor. If we had a rotor that was composed of a shaft that was magnetically north-polarized on the top half and magnetically south-polarized on the bottom half, down the entire axis, and if we had a stator that simply had two windings, with four poles to hold the windings, that could induce a magnetic field in the stator, we would have a very simple stepper motor, as illustrated in Figure 2.1b.

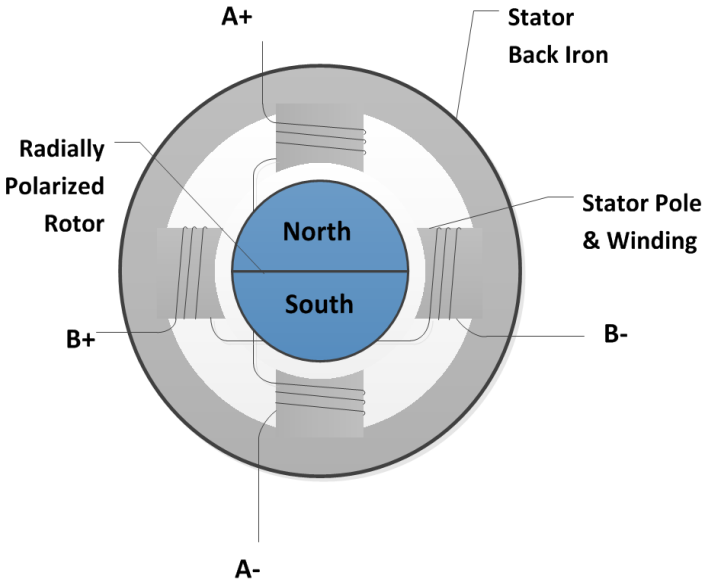

Figure 2.1b – Stepper Motor
Depiction of the permanent magnet stepper motor configuration

To hold the rotor in the position that it is currently in, all that would have to be done is apply an electrical current to winding A such that a south-polarized magnetic field is induced in the stator at the A^+ side and a north-polarized magnetic field is induced in the A^- side of the stator; to accomplish the two fields with one winding, the winding is wound in one direction on the A^+ side and in the opposite direction on the A^- side. In order to cause the rotor to take a full step and change its position, we would have to turn off winding A and winding B would simply have to be excited with electrical current that would induce a polarized magnetic field in the stator across the B winding. The direction that the rotor will rotate simply depends on how the next winding is polarized. If the B winding is polarized such that the B^+ side is south-polarized and the B^- side is north-polarized, the rotor would be influenced to rotate in a clockwise direction to complete the magnetic circuit and allow the magnetic flux to flow in the path of

least magnetic reluctance. In essence, this is the simplest example of how any stepper motor performs a full step to rotate the shaft.

Obviously, this crude stepper motor has a very low degree of resolution in the stepping angle. We see that for every full step the rotor has a 90-degree change in position. This is not very helpful if you need a motor to position the shaft more accurately than a quarter of a turn per step. The stepping angle can be improved simply by adding more pairs of north and south-polarized teeth on the rotor and by adding more windings to the stator. To further improve the angle, sets of teeth can be attached to the windings in the stator; this allows a shorter distance to be covered for every full step, which in turn provides greater precision in the stepping angle. Each of the three types of stepper motors applies this technique of using multiple teeth on both the rotor and stator to produce a smaller stepping angle.

How the actual magnetization of the rotor is performed and how the next pair of north-south magnetic couplings is achieved, to perform a step with the motor, is where the three different types of stepper motors come from. Each type has a unique way of performing the steps. A brief overview of the method for each type is discussed in the appropriate section below.

2.1.1: Permanent Magnet Stepper Motor [D5]

The first, and perhaps the simplest, version of the stepper motor is the permanent magnet stepper motor, as discussed above. The main source of the magnetic field vector on the rotor and the magnetic flux in this stepper motor simply comes from the permanent magnet that is implemented as the rotor. Most permanent magnet stepper motors utilize the very simple north-south radially polarized rotor and two winding configuration, as shown in Figure: 2.1b. This configuration provides good torque performance; however, the resolution in the stepping angle is poor, due to the fact that the rotor does not have any teeth for better step definition and that there are only four poles to house the windings in the stator.

2.1.2: Variable Reluctance Stepper Motor [D5]

The second type of stepper motor is known as the variable reluctance stepper motor. The only sources for the magnetic flux that drive this motor are the windings that are housed in the stator. The rotor of the VR stepper motor is composed of a metal that simply conducts the magnetic flux straight through. Because the rotor is not already magnetically polarized the rotor does not produce any magnetic vector. Thus, in order to complete the magnetic circuit, while the rotor is lined up in a position, the rotor and stator must have an equal amount of teeth. The magnetic circuit is completed such that the flux flows from the north-polarized winding in the stator through the pole/teeth of the stator and into the teeth of the rotor; then the flux is simply transferred through the rotor and

into the south-polarized pole of the stator on the opposite side of the rotor from the first pole of the stator.

Furthermore, since the rotor is not inherently polarized, which means that the tangential force on the rotor tooth completely depends on how much magnetic flux is flowing through the rotor, in order to cause the shaft to move to the next desired position it becomes necessary to use multiple pairs of stators and rotors in separate sections of the motor casing. These sections are known as stacks. Thus, this type of motor is commonly referred to as the multi-stack variable reluctance motor. Stepping action can be performed with a single-stack VR motor; but, there must be a proper combination of rotor teeth versus stator teeth. For the sake of brevity, only the multi-stack will be discussed. An illustration of the multi-stack motor is shown in Figure: 2.1.2a.

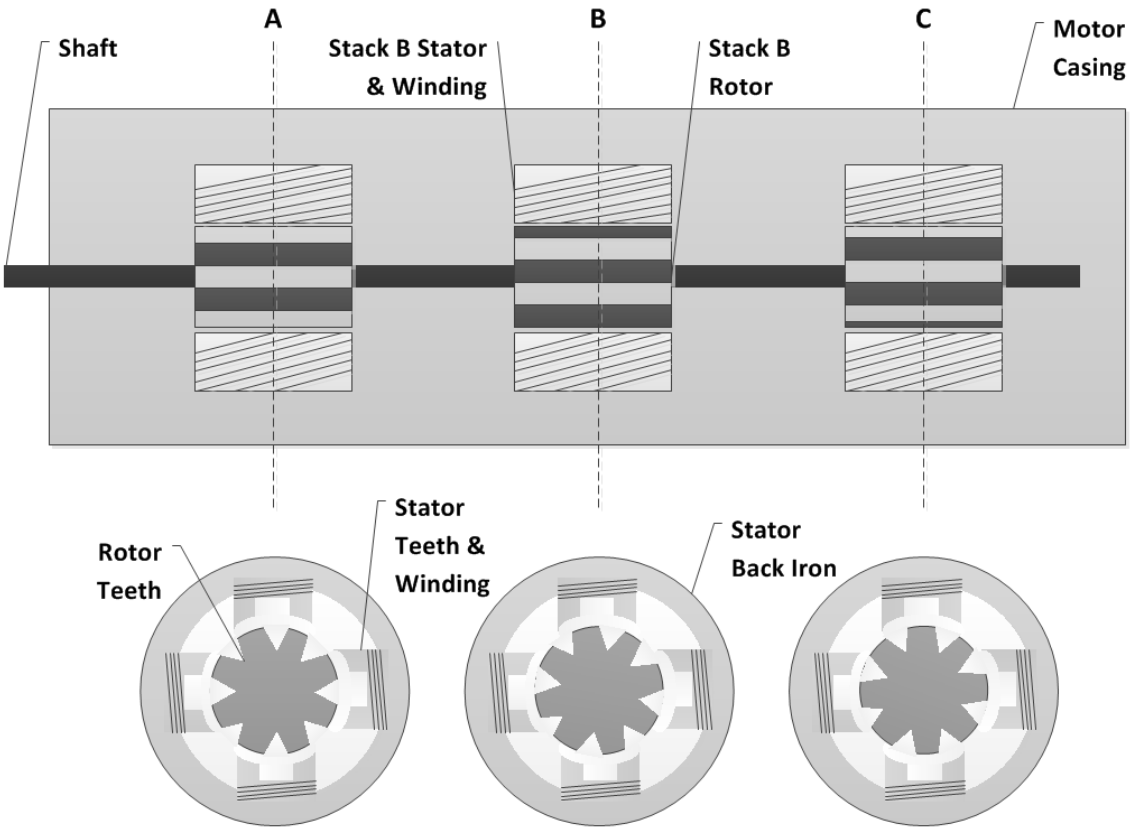

Figure 2.1.2a – Variable Reluctance Stepper Motor
Depiction of the Variable Reluctance Stepper Motor Construction

As can be seen from Figure 2.1.2a, this motor has three stacks, each with its own rotor, stator and one winding. Each rotor is attached to the motor shaft, thus each stack contributes to the motion of the shaft. In each stack, the stator poles and teeth are offset in position from the other stacks. When the rotor teeth are aligned with the stator in one stack, the rotor teeth in the other stacks are

misaligned with their stator teeth pairings. When each stack is excited, the magnetic field attracts the rotor teeth to align with the stator teeth such that the flux can from the north-polarized poles to the south-polarized poles in the stack. In Figure 2.1.2a, the rotor is aligned with the stator in stack A. If the winding in stack B was energized, the rotor would conduct flux and experience tangential forces that would cause the rotor to rotate in a counter-clockwise direction. If the windings in C were energized, the rotor would rotate in a clockwise direction. If the stacks are energized in the correct sequence, full rotation can be induced in the shaft of the motor. For instance, if the desired rotation were in the counter-clockwise direction, the stacks would have to be energized in the A, B, C, A, B, C, A... sequence. For a clockwise direction the stacks would have to be energized in the A, C, B, A, C, B, A... sequence. In order to have a bi-directional multi-stack VR stepper motor it must be fabricated with at least three stacks.

This multi-stack VR stepper motor is superior to the permanent magnet stepper in that the resolution of the stepping angle is much greater. Multi-stack motors typically have stepping angels from about 2 to 15 degrees. However, the downside of the multi-stack VR is that, to decrease the stepping angle, more and more stacks must be added, which inherently makes the motor much larger.

2.1.3: Hybrid Stepper Motor [D5]

The hybrid stepper motor is unique in that it was designed as a combination of the variable reluctance and the permanent magnet stepper motors, which is the reason that it is called a hybrid. The hybrid uses a permanent magnet placed axially along the shaft. Each side of the magnet is attached to a different rotor in a separate stack; each rotor has its own set of polarized teeth, but both stacks share the same windings and stator teeth. The rotor teeth in section X are set up to be out of line with the rotor teeth in section Y, the reason for this will be explained later. A Figure of a simple hybrid stepping motor is shown below in Figure 2.1.3a.

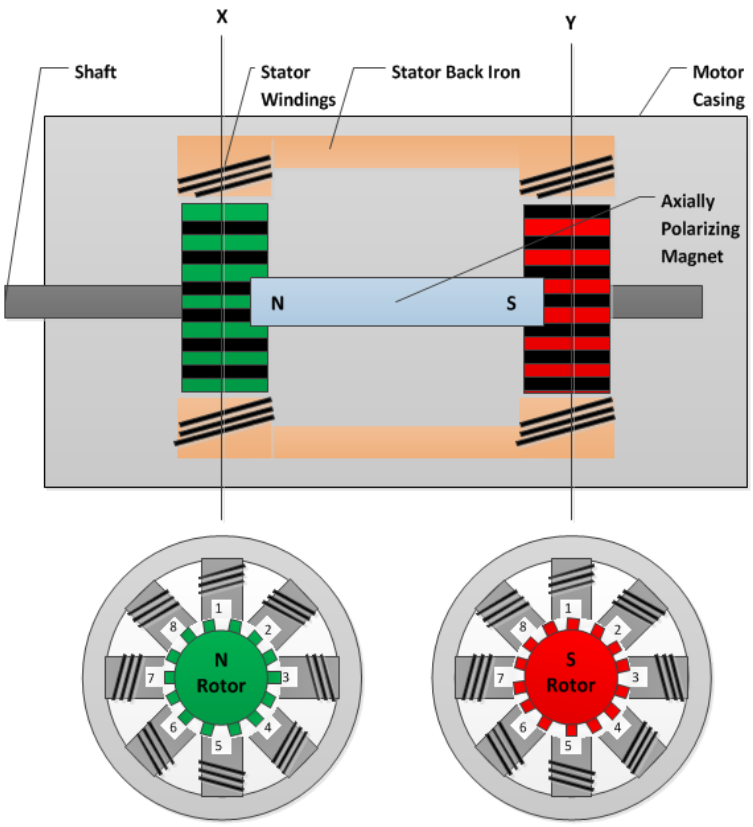

Figure 2.1.3a – Hybrid Stepper Motor
Depiction of the Hybrid Stepper Motor construction

This simple hybrid has two windings, 14 rotor teeth and 8 stator poles which each have two teeth. The first winding, A, is mounted on the stator poles 1,3,5 and 7, while the second winding, B, is mounted on poles 2,4,6 and 8. Every other stator pole for each winding is wound in the opposite direction as the previous pole for that winding. By doing this, the resulting magnetic field is north-polarized for two poles of the winding and south-polarized for the other two poles of the winding, when the winding is excited by electrical current. Thus, while referring to Figure 2.1.3, when a winding is excited, the north-polarized rotor will align its teeth up with the poles of the stator that have an electrically induced south-polarized field, in section X, and the rotor teeth in section Y will align with the north-polarized stator teeth of the same winding, this is the reason that the rotor teeth for both rotors are misaligned.

If winding A is wound such that, when it is excited by a positive current, poles 1 and 5 have an induced south-polarized magnetic field, and poles 3 and 7 have an induced north-polarized magnetic field, the rotor teeth in section X will align themselves with poles 1 and 5 when A is positively excited, and in section Y the rotor teeth will align with the north polarized poles, 3 and 7. When A is negatively excited, the rotor in section X will then align its teeth with the stator teeth in poles 3 and 7. The same methodology applies for winding B.

Rotational direction for the hybrid motor is unlike the variable reluctance motor, where the direction of rotation of the motor simply relied on the order in which the stacks were excited. To cause the shaft of the hybrid to rotate in the opposite direction, the current in the windings must be reversed. By reversing the current in the winding, the induced magnetic field on each set of poles is reversed. And the rotor, in each section, then becomes attracted to the new set of poles that are inducing the polar-opposite magnetic field. The rotor then performs rotation and alignment in the opposite direction.

To complete the magnetic circuit, a winding is excited in the stator, which induces a polarized magnetic field, and the permanent magnet causes the rotor sections align themselves to be in agreement with the induced magnetic vector; the magnetic flux then flows from the north-polarized section of the rotor, in section X, into and through the stator back-iron to south-polarized rotor in section Y. By exciting the windings in the correct sequence, rotation of the shaft is achieved. To obtain a clockwise rotation for the motor the windings can be excited in the order: A^+, B^+, A^-, B^-, A^+, B^+… For a counter-clockwise direction, the windings should be excited in the following sequence: A^+, B^-, A^-, B^+, A^+, B^-…

This hybrid design of the stepper motor is beneficial in that it allows clearly defined steps, with the larger number of rotor and stator tooth pairings, while only using two stack sections. This larger number of tooth pairs provides a much smaller stepping angle than that of the variable reluctance or permanent magnet stepper motors. The advantages of the hybrid motor are described well by Paul Acarnley:

> Hybrid motors have a small step length (typically 1.8°), which can be a great advantage when high resolution angular positioning is required. A survey or manufacturers' data by Harris et al. (1997) revealed that the torque-producing capability for a given motor volume is greater in the hybrid than in the variable-reluctance motor, so the hybrid motor is a natural choice for applications requiring a small step length and a high torque in a restricted working space. When the windings of a hybrid motor are unexcited the magnet flux produces a small 'detent torque', which retains the rotor at the step position. Although detent torque is less than the motor torque with one or more windings fully excited, it can be a useful feature in applications where the rotor position must be preserved during a power failure. **[D5]**

Therefore, it is easy to determine that the hybrid stepper has inherited the best qualities of each of the other stepper motors. It is capable of respectable torque performance with a small casing size and it is able to produce a small, yet strong, stepping angle. To our approval, the motors that we were given to use to control the rain simulator are, indeed, hybrid stepper motors.

2.1.4: Motor Winding Configurations [D5]

There are many ways to configure the windings, otherwise known as phases, within the stepper motor. Each different configuration provides a unique input to output transfer characteristic for the motor. The two main methods for configuring the windings are in a unipolar or bipolar configuration. If each winding is split into multiple sections more options become available.

The difference between the unipolar and bipolar configuration is implied by their names. The unipolar configuration only accounts for current flowing in one direction across the winding, while the bipolar configuration assumes that the current in the winding will be induced in both the positive and negative directions. A unipolar configuration is used mainly in the variable reluctance motors, because the rotational direction of the variable reluctance stepper depends only on the sequence in which the stacks are excited with positive current. The hybrid motors typically use a bipolar winding configuration, because the rotational direction is determined by exciting the winding with both positive and negative current. However, the hybrid can be operated in a unipolar configuration if the windings are configured correctly.

Depending on how the windings are configured when the motor is manufactured, its casing will have 4, 5, 6 or 8 wires protruding from it. The user can then attach these wires to the appropriate driving circuit that will induce current through the windings, thus causing the motor to perform. The three most common winding configurations, for a hybrid stepper motor, are shown in Figure 2.1.4a below.

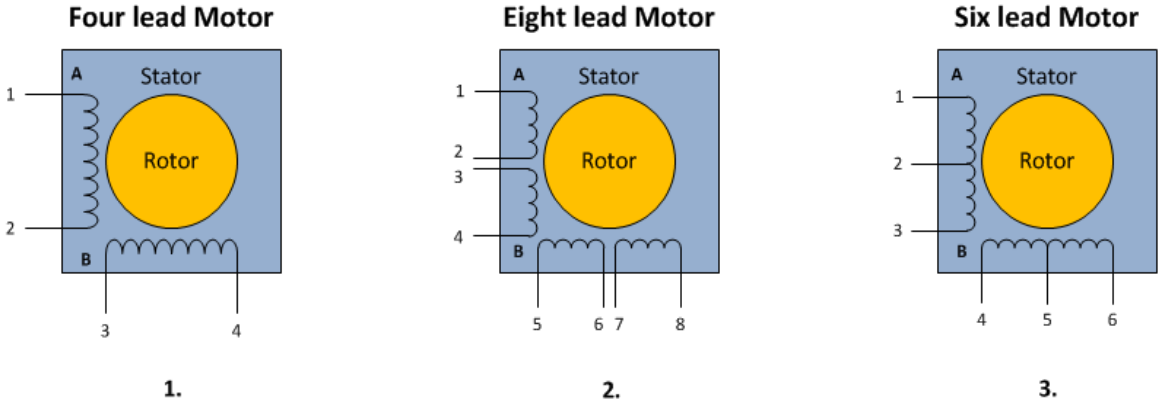

Figure 2.1.4a – Winding Configurations
Illustration of various options for stepper motor winding configurations

Obviously, the simplest representation is depicted in picture 1 of Figure 2.1.4a. Here there are simply two phases, A and B, with 4 wires protruding from the casing. To perform the stepping action, the phases would simply have to be energized in the proper sequence, as described in the previous section. The windings for this Figure are configured in the bipolar method only; to have

negative current induced through any part of either winding, the whole winding must have a polarity change.

The second setup, shown in picture 2 of Figure 2.1.4a, appears to have 4 windings instead of 2. However, the windings shown are always configured to work in pairs. So, effectively, there are only 2 windings that have been split into two smaller windings each. The windings are split so that the hybrid motor can be configured to operate with either a unipolar or bipolar drive circuits or to operate with the same torque performance for either high-current / low-voltage or high-voltage / low-current applications; for either of the current or voltage options, the overall power through the motor remains the same.

The main advantage of operating the hybrid in a unipolar winding configuration is that it allows the motor to perform at higher speeds. All windings in any stepper motor have very high inductance. Therefore, the current in a winding resists instantaneous changes in direction strongly. To avoid the task of absolutely reversing the current in a winding, as is done in the bipolar configuration; the winding can be split into two sections. One section is used to induce current in the positive direction and one section is used to induce current in the negative direction, with respect to the polarity of the overall winding. When positive current for the overall winding is desired, the section for the negative current is turned off and the section for the positive current is turned on. While the negative section allows the stored current to dissipate, the positive section of the winding is already ramping up current and inducing the desired magnetic field.

In the bipolar winding, when the polarity on the winding is switched to induce a negative current a high voltage spike occurs and it takes time for the current to reverse directions. Thus, the unipolar configuration gives better speed performance than the bipolar configuration. However, because the unipolar method requires the winding to be split in two, and only one half of the winding is being purposefully used at a given time, the torque performance of the unipolar configuration is much less than that of the bipolar method. Because the bipolar configuration utilizes the entire winding, it is capable of inducing a much stronger magnetic field to position the rotor of the motor. Thus, the bipolar configuration should be used for low speed / high torque applications and the unipolar configuration should be used in high speed / low torque applications.

An example of the unipolar configuration is shown in picture 3 of Figure 2.1.4a. This configuration is typically called the center-tap configuration. Normally the center is attached to the common voltage source and when current is desired in either section of the winding the correct side will be polarized. It can easily be seen that the motor in picture 2 of Figure 2.1.4a, with 8 wires, can be configured in either the bipolar or unipolar configurations. Furthermore, for the bipolar configuration, the sections of each winding can be placed in parallel for high-current / low-voltage applications or the sections can be placed in series for high-voltage / low-current applications. Thus, the 8 wire motor allows a variety of options; the motor can be configured for many different control circuit conditions.

The stepper motors provided to us by the Storm Water Management Academy were the 8 lead hybrid stepper motors. Thus, we knew that we were able to configure the motor for whatever application that we determined necessary in controlling the position of the nozzle swing arm. A list of the specifications of the hybrid motors, to be implemented in this design, is shown below in Table 2.1.4b.

Performance Specifications	
Voltage per Winding	2.7 **V**
Current per Winding	5.6 **A**
Holding Torque	1284 **In-oz**
Rotor Inertia	0.0566 **oz-in-sec^2**
Motor Type	
Stepper Construction	Hybrid
Step Angle	1.80^o
Number of Leads	8
Shaft Orientation	In-line; Single-ended
Housing / Enclosure	
Width	3.38 **inches**
Length	4.96 **inches**
NEMA Frame	34
Environment	
Operating Temperature	-4 to 122^o**F**
Environment	Dust-Proof

Table 2.1.4b - List of stepper motor specifications [D6]

2.2: Stepper Motor Driver Circuits

Another imperative part in the control of stepper motors was the actual circuits that induce the current in the motor windings; these circuits are commonly known as motor driver circuits. There are a wide variety of circuits to perform this task and each design comes with many advantages and disadvantages. In our discovery of this topic we researched the basic types of drivers and then determined which style would serve the purpose of our design the best. The following is a brief discussion on the basic concepts behind the design of the different types of stepper motor driver circuits.

2.2.1: Unipolar Driver Circuits [D5]

As discussed earlier, the windings in a stepper motor can be configured such that the current is only meant to flow in one direction; this is the unipolar configuration. The control circuit for any motor in this configuration must then appropriately deliver the current to the windings to cause the motor to operate correctly. Unipolar winding configurations are typically seen in variable reluctance motors and are a greater rarity among hybrid motors. For each phase of the

variable reluctance motor, or some hybrids, the current needs only to be turned on or off. An example of a three-phase unipolar drive circuit for a variable reluctance motor is shown below in Figure 2.2.1a.

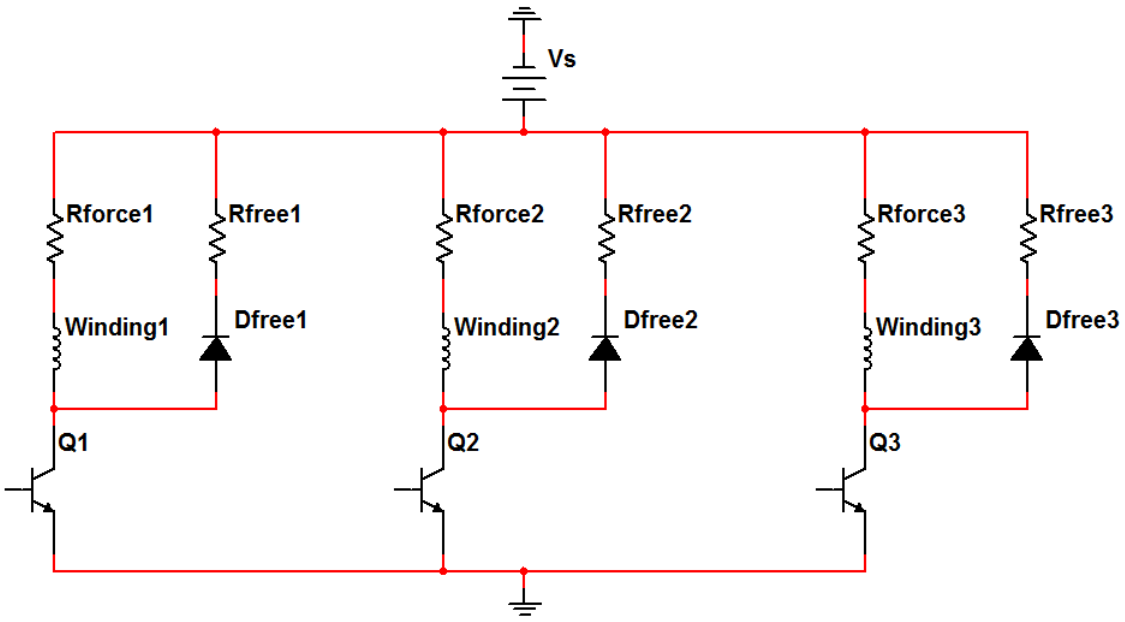

Figure 2.2.1a - Depiction of a Unipolar Driver Circuit

In Figure 2.2.1a, each phase has its own driving circuit. The basic design and concepts of this unipolar drive circuit are best described by the author, Paul Acarnley:

The phase winding is excited whenever its switching transistor is saturated by a sufficiently high base current. Under these conditions the full dc supply voltage is applied across the series combination of phase winding and forcing resistance, since the voltage drop across the saturated transistor is small (typically 0.2 V). The dc supply voltage (V_S) is chosen so that it produces the rated winding current (I) when applied to the total phase circuit resistance, which is equal to the sum of the phase winding (r) and the forcing (R) resistances:

$$V_S = I(r + R) \quad \text{(Eqn. 2.1)}$$

In general the phase winding has a considerable inductance, so its natural electrical time constant (inductance/resistance) is long. The build-up of phase current to its rated value would be too slow for satisfactory operation of the motor at high speeds. By adding the forcing resistance, with a proportional increase in supply voltage, the phase electrical time constant can be reduced, enabling operation over a wider speed range. [D5]

Furthermore, because the windings in the motor have high inductance the current in the winding will not stop instantly. If the transistor turns off immediately the current in the winding will continue flowing and will amass a large voltage at

the collector of the transistor that will inevitably blow out the device and damage the circuit.

This possibility is avoided by providing an alternative current path - known as the freewheeling circuit - for the phase current. When the switching transistor is turned off the phase current can continue to flow through the path provided by the freewheeling diode and freewheeling resistance. If the phase current is established at its rated value then the maximum voltage ($V_{ce\ max}$) across the switching transistor occurs in the instant after the transistor switch is opened. The current (I) has not started to decay and flows through the freewheeling resistance (R_f), so the maximum collector-emitter voltage (neglecting the forward voltage drop across the freewheeling diode) is

$$V_{ce\ max} = V_S + R_f I \quad \text{(Eqn. 2.2)}$$

The phase current therefore decays in the freewheeling circuit and the magnetic energy stored in the phase inductance at turn-off is dissipated in the freewheeling circuit (winding + forcing + freewheeling) resistances. [D5]

Thus, it is imperative to have this freewheeling circuit along with the actual switching transistor in the design of a proper motor driving circuit. Also, the supply voltage is dependent upon the combined impedance of the winding, forcing resistance and freewheeling resistance. Because of the inherent properties of the motors that the rain simulator utilizes we choose not to use the unipolar configuration for the motor driver circuitry.

2.2.2: Bipolar Driver Circuits [D5]

The motor driver circuit that was most pertinent to our design was the bipolar driver circuit. When a permanent magnet or hybrid motor has its windings configured for bipolar operation the driver circuit must be able to cause current to flow in both directions through the winding. To accomplish this task an H-Bridge assembly is commonly used. The general concept of an H-Bridge is that the polarity across the winding can be reversed by exciting the proper set of switching transistors. The transistors are attached to the winding such that the circuit forms an H, in the schematic at least. Figure 2.2.2a depicts a common H-Bridge assembly for a bipolar configuration of a motor winding.

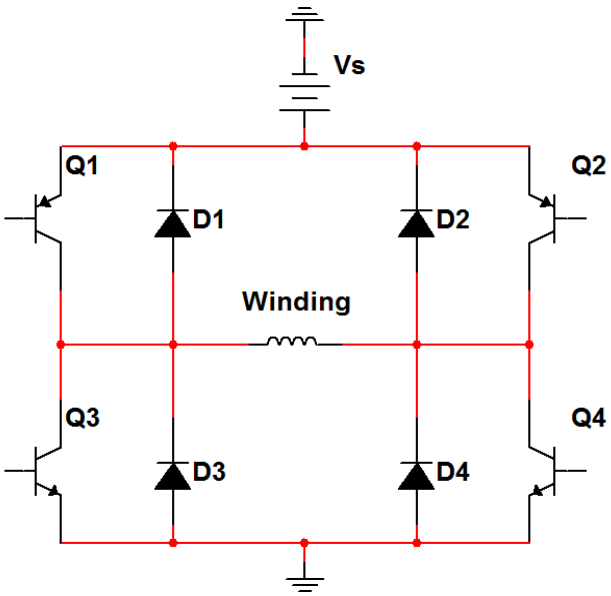

Figure 2.2.2a - Depiction of a Bipolar H-Bridge Driver Circuit

In Figure 2.2.2a, if transistors Q1 and Q4 are excited a path will be formed such that the current can flow from the power source through the winding from left to right. If transistors Q1 and Q4 are turned off, and Q2 and Q3 are excited, the current will flow in the opposite direction, right to left. However, just as with the unipolar circuit, the current through the winding cannot be stopped instantaneously. Thus, a freewheeling circuit is constructed with diodes that bridge across the switching transistors, as seen in Figure 2.2.2a. If Q1 and Q4 are suddenly shut off the current will continue to flow through the winding, yet be diverted up through D2, utilizing the freewheeling circuit as the path is completed between ground, D3, the winding, D2 and the power supply.

One thing that sets the freewheeling circuit for the bipolar driver apart from that of the unipolar driver is that the current that continues to flow from the winding will flow back to the power supply after the transistor pairing is shut off. In the unipolar circuit, the current will flow back to the supply and then back down through the winding, dissipating more power. This H-Bridge setup allows the bipolar driver to have better power efficiency than the unipolar circuit

An important aspect to be considered when choosing what driver circuit to use is what type of transistor is used to perform the switching. In Figures 2.2.1a and 2.2.2a bipolar junction transistors (BJTs) were used. However, other options such as the MOSFET or the IGBT can be used; each has its own advantages and disadvantages. The MOSFET is highly advantageous because it can be turned on by applying a small voltage to the gate and the current draw to maintain the MOSFET in the on-state is minimal; the BJT requires a much higher current draw to maintain the transistor in saturation. This aspect is important when considering what device is controlling the excitation of the switching transistors within the driving circuit. It is important to make sure that the

microcontroller is capable of maintaining the power requirements to keep at least two of the switching transistors saturated, per driver circuit. Due to the fact that we used the motors in the bipolar configuration and a single microcontroller was implemented to send the signals to the drivers, we choose to use MOSFET motor driver circuits that utilize the H-Bridge configuration.

2.2.3: Motor Driver – DRV8432 [A1]

In the search for a microcontroller, we also discovered that Texas Instruments manufactured motor driver chips. While looking through their motor drivers, they had made two drivers that were able to handle the current that our motors would draw. The motors that are in place at the rain simulator will draw up to 5.6 Amps per phase each and the two chips that Texas Instruments manufacturers are capable of handling 6 amps and 7 amps. Either chip would be capable of driving the motors, but the DRV8412 would not allow for maximum torque. The other major difference between the DRV8412 and the DRV8432 is that the thermal pad for the DRV8412 is on the bottom of the device and for the DRV8432, the thermal pad is on the top of the device. For these reasons as well as the reasons discussed in the Heat-Sink section of this report, the DRV8432 is the motor driver device that we are planning on using in the final design of the project.

The DRV8432 is a full bridge motor driver with advanced protection devices built in to the device. It is up to 97% efficient with low resistance MOSFETs and is capable of powering motors up to 50 volts and 14 amps while in parallel mode. In dual full bridge mode, the motor driver is capable of driving a motor with 50 volts and 7 amps. While in dual full bridge mode, each device is capable of driving two windings of a motor. While in parallel mode, each device can only drive one winding of a motor. However, the motors that are in use at the rain simulator are only rated for 5.6 Amps per phase and the motor drivers will not need to be used in the parallel mode. As was stated above, the motor drivers have built in protection systems that protect the device from under voltage, over temperature, overload, and short circuit protections. In addition to those protections, the device does not require any external snubber or Schottky diodes. As discussed in the power supply section of this paper, there is a need for two supply voltages for the motor driver chips. They require motor supply and a digital supply for the motor driver chips itself. Table 2.2.3a, shows the recommended operating conditions as provided by the manufacturer.

VDD – Digital supply voltage	12V
GVDD_X – Logic supply voltage	12V
PVDD_X – Half bridge supply voltage	50V
Peak current per output pin	15A
Continuous current per output pin	7A
PWM switching frequency	500kHz
Ambient operating temperatures	-40°C to 85°C

Table 2.2.3a - Recommended Operating Conditions[A1]

These are the recommended operating conditions of the DRV8432 as provided by the Texas Instruments.

Basic Operating Modes: The device is designed to run in one of four operating modes; the dual full bridge mode with two pulse width modulation inputs each full bridge and cycle-by-cycle current limiting, dual full bridge mode with two pulse width modulation inputs each full bridge and over current latching shutdown, parallel full bridge mode with cycle-by-cycle current limiting, and dual full bridge mode with one pulse width modulation input each full bridge and cycle-by-cycle current limiting. For the first two modes, the PWM_A input controls half bridge A, PWM_B input controls half bridge B, and so on. The parallel mode is not necessary for our project and will not be discussed. The fourth mode of operation of the motor driver has one PWM input controlling one full bridge reducing the number of I/O pins necessary for the microcontroller. In this mode, PWM_A input controls both half bridge A and B and PWM_C input controls both half bridge C and D. Also, using this mode of operation half bridge A and B are complimentary to each other and half bridge C and D are complimentary to each other. PWM_B and PWM_D are not used and are recommended to be connected to ground. Also, because each half bridge has separate power and ground pins, a current sensing resistor can be inserted between PVDD and PVDD_X or GND_X and GND.

Power Supply: For the power provided to the device, the only requirement was a 12Vdc supply and a motor voltage supply. An internal voltage regulator provides the necessary voltage levels for the digital and analog circuitry. Decoupling capacitors needed to be placed as close as possible to the to the pins on the motor driver and a short ground path back to the device. A ceramic capacitor is required between each bootstrap (BST_X) pin and the power stage output pin (OUT_X) for proper bootstrap loader functionality. While the output is low, the bootstrap capacitor is charged and while the output is high, the bootstrap capacitor is shifted above the output potential and provides a suitable voltage supply for the high side gate driver. This ensures sufficient energy is available during minimal PWM input cycles to keep the high side FET turned on during the remaining part of the PWM cycle. The recommended value for the bootstrap loader capacitor depends on PWM switching frequencies. For 10kHz to 500kHz frequencies a 100nF capacitor is recommended and for frequencies below 10kHz a higher value may be necessary. For optimal performance and reliability, it was

necessary for a decoupling capacitor to be placed between each PVDD_X pin and ground. The 12 volt supply needed to be a low-noise, low output impedance source and the 50V power-stage supply needed to be low output impedance and low noise. A nice feature of the DRV8432 chips is that the power supply turn on sequence is not important as the internal power on circuit handles this. Even though a power up sequence is not necessary, it was recommended to hold RESET_AB and RESET_CD low while the device is powered up as this allows the device to charge the external bootstrap capacitors. During power down, again, the motor driver does not require a special sequence. And, again, while a specific power down sequence is not necessary, it was recommended to hold RESET_AB and RESET_CD low during power down.

Protection Devices: Another nice feature of the DRV8432 devices was the error reporting functionality of the device. It has two pins that can be connected to the microcontroller to provide protection mode signaling to the microcontroller. Faults that result in a shutdown of the motor driver are signaled by the FAULT pin going low. The faults that can result in a driver shutdown are over temperature, over current, and under voltage. The other fault pin is the OTW pin. This pin goes low when the motor driver junction temperature rises above 125°C. By monitoring the OTW and reducing the load current when the OTW pin goes low, the system can continue to operate even though it will be in a reduced functionality. A nice feature that has been built in to the device is that there are internal pull up resistors on the OTW and FAULT pins. Along with these two error reporting pins, the DRV8432 has internal protections that will immediately set the half bridge outputs to a high impedance state and sets the FAULT pin low. As well as reporting the error to the FAULT pin, the device is able to recover from faults automatically, except for over current and over temperature faults. When running at low frequencies without a sufficiently high enough bootstrap loader capacitor, the voltage may not be able to maintain a high enough level for the gate driver. The bootstrap under voltage protection circuit will prevent a failure of the MOSFET.

The over current protection of the DRV8432 has a programmable trip threshold on all high side and low side FETs in two different modes, either cycle-by-cycle (CBC) current limiting mode or over current latching mode. In CBC mode the detector outputs are monitored by two protection systems. One protection system controls the power stage to prevent the output current from increasing further by limiting the current in each and every cycle preventing a motor driver shut down. The second protection system would be used during a short to ground or short to power condition. In either of these conditions, the CBC limiting circuit might not be able to maintain the current at an appropriate level. This second protection system triggers a latching shutdown and sets the half bridge in a high impedance state. Each half bridge has is own independent current limiting and over current protection. The figures below show examples of the cycle-by-cycle current limiting operation with a high side over current event and a low side over current event. In over current shut down latching mode, the CBC current limit and error

recovery circuits are disabled and the device will shut down. After shutting down, the RESET_AB or RESET_CD lines must be set to low to restore the motor driver to normal operating mode. The over current protection threshold value can be set using an external resistor connected between the OC_ADJ pin and GND. For the over current protection circuitry to function properly, an inductor or power ferrite bead needs to be connected at the power stage output pins. Without this inductor or ferrite bead, the device may not be able to protect itself from short circuits.

The device has multiple thermal sensors on different areas of the device to monitor the temperature of the device. The over temperature protection of the device also has two different stages of thermal protection. The first stage is just a over temperature warning and motor driver will set the over temperature warning (OTW) pin low when the junction temperature rises above 125°C. The second stage of the thermal protection in the DRV8432 is the thermal shut down mode. If the junction temperature rises above 150°C the motor driver will go into a thermal shutdown mode which sets all the half bridge outputs to a high impedance state. The FAULT pin is also set low and in order to begin normal operation again, the RESET_AB and RESET_CD lines must be set low to clear the over temperature fault. Another critical protection circuit in the motor driver is the under voltage protection. The reason under voltage protection is critical is because of the need to avoid the loss of control of the driver gates. It is critical to avoid a shoot through condition, where the power supply essentially becomes shorted to ground. The under voltage and power on reset protections are used in any power up, power down, or brownout situation. During power up the power on reset circuit resets the over current circuit and verifies that the VDD and GVDD_X supplies are above 9.8 volts before the device will begin operations. If the VDD or GVDD_X voltages drop below the threshold, then the device will set all half bridge outputs to the high impedance state and will set the FAULT pin low. The device is capable of automatically recovering from under voltage faults once the bootstrap capacitor voltage rises back above the threshold level.

Device Reset: The two reset pins, RESET_AB and RESET_CD are provided for independent control of the two sets of half bridges. When RESET_AB is set low, all four FETs in half bridges A and B are set to high impedance states. RESET_CD operates in the same method. Also, setting the RESET_XX pins to low will clear the FAULT pin and allow the device to continue operation.

Output Inductor: During normal operations, the inductance in the motor is high enough to slow the current change during minor faults and will allow the CBC current limiting circuit to operate normally. However, during a fault condition, the motor inductance may not be present any more and the current in the motor driver may rise to a high level too quickly and rise above the max rating of the device before the over current shut down can occur. In order to allow for safe operation of the device, even in short conditions, an inductor needs to be placed in series with the motor winding as close as possible to the motor driver.

PCB Layout: Texas Instruments has recommended PCB design considerations. One of these was that the PCB be made of FR-4 Glass Epoxy with 2-oz copper on both top and bottom layers for improved thermal performance. For the DRV8432, we used an external heat-sink so this recommendation was considered but not utilized. The ground plane was recommended to be as large as possible and for ground traces to be as short and wide as possible before connecting to this ground plane. The decoupling capacitors on the PVDD_X pins needed to be placed as close as possible to the pins and with a short as possible ground path. Because of the decision to use the DRV8432 instead of the DRV8412, the design considerations for the PowerPad™ of the DRV8412 did not need to be considered.

How to Use: A major design consideration was in the switching frequency of the MOSFETs of the motor driver circuit. A normal design would keep the switching frequency above the audible levels because if the driver is switched in the audible range, you can hear the switching which can be annoying. Another consideration was that the faster the switching frequency is, the lower the ripple current which directly relates to how smoothly and quietly the motors run. The downside to raising the switching frequency too high is that the switching losses increase and the efficiency goes down. The DRV8432 has diodes built in to handle the re-circulation currents and by using synchronous delay, it is possible to control the re-circulating currents. The two benefits to using synchronous decay is that the losses in the motor driver are reduced and therefore the heat loss in the motor driver is reduced. The downside to synchronous decay is that the current may not decay fast enough for proper motor control. For our purposes, the added complexity of synchronous decay was not necessary, however, if we begin having issues with heat in the motor drivers, this feature may have to be implemented to prevent motor driver failures. For the DRV8432, the PWM input signals directly control each FET in the motor driver in the default mode. As discussed earlier, the cycle-by-cycle current limitation can be set by an external resistor. As shown in Table 2.2.3b, the different cycle-by-cycle current levels are set by placing a resistor between the OC_ADJ pin and ground. The stepper motors that are connected to the rain simulator are rated to handle 5.2 amps and therefore we are going to need a resistor value of approximately 52,000 Ohms. This was one of the great things about the DRV8432 motor driver circuit, the simplicity of setting a current limit and allowing the motor driver to handle the actual process of limiting current.

Maximum Current	OC Adjust Resistor Value (kOhms)
1.4	200
1.9	150
2.4	120
2.8	100
3.4	82
4.1	68
4.9	56
5.8	47
6.3	43
6.9	39
7.4	36
8.8	30
9.7	27
10.7	24

Table 2.2.3b – OC Adjust Resistor Table[A1]
This table is the recommended resistor values for CBC current limiting within the motor drivers.

2.3: User Interface

2.3.1: Light Emitting Diode Display

A display technology that has become quite common, especially in small battery powered devices, is a light emitting diode screen, or LED screen. LED screens use one of two techniques for building an LED panel. The first type of panel uses discrete red, blue, and green LEDs individually mounted to form a display pixel. The second type of panel uses surface mounted red, blue, and green LEDS combined into a single package with each diode being smaller than a pinhead. The discreet LED screens are typically used in large displays such as the jumbo-trons at sports arenas. The surface mounted LEDs are used in smaller screens such as cell phones and televisions. The use of LEDs for displaying images is a relatively simple method of display. Each pixel is made up of the three different LEDs and they are individually controlled in a matrix fashion, same as the active matrix LCD screen. The main difference between LED and LCDs is that LCDs require an external light source in order to be seen. LED screens do not need an external light source because each pixel can emit light and changes the color of the light by varying the intensity of each color LED in the pixel. LED screens can consume much less power than LCD screens, however, are much more

expensive than an LCD that is the same size. The primary use of LED displays is in portable electronics because of the much lower power consumption of LED screens. Because LED screens did provide much benefit for the purposes of this project, along with the high cost of even a small LED screen, they were not used.

2.3.2: Plasma Display

A plasma display panel is common in large screen displays, especially televisions. The plasma display uses small cells that contain an ionized gas, this type of cell is very similar to a fluorescent lamp. A plasma display is similar to an LED display, in that each individual cell can produce its own light and does not require an external light source like an LCD. Even though plasma displays and LED displays are similar in that neither type requires an external light source, a plasma display uses much more energy than an LED display. The reason for this increase in power use of a plasma screen over an LED is a fundamental characteristic of the operation of the screen. When the cell is energized, the mercury in the cell emits UV light, this UV light passes through a phosphor coating on the cell. However, about 60% of the energy released from the cell is in the form of infrared light and therefore 60% of the power used is wasted in heat. Each pixel is also individually controllable using an active matrix control in the same method as LCD or LED screens. The main benefit of plasma display technology over other display technologies is that a plasma screen is able to generate more accurate colors as well as having a higher contrast ratio. Plasma screens are also able to display moving objects better than LCD screens. Our project does not use moving images, nor does it require a color display. For these reasons a plasma display was not implemented for our uses.

2.3.3: Liquid Crystal Display

The current system being used at the Rain Water Simulator utilizes a laptop computer to perform input and display. The only information that is displayed to the screen during a simulation is the set/desired rain intensity and the duration of the test. However, the laptop that they are currently using has already been a source of some test failures. Along with the unreliability of the laptop, a computer screen would be considered overkill for displaying two pieces of information in text form and therefore, in our design, we looked for a simpler and lower cost option for displaying that information to the users. Using the current system of input and display, if the laptop fails, the users would have to purchase another laptop at high cost. By moving to a low cost, simple interface, there will no longer be a need for a computer for input and display. One of the most common low cost displays is a liquid crystal display, more commonly known as an LCD. LCDs are also low power devices, which while not a requirement of our design, it is always a good idea to reduce power consumption when possible. In the liquid crystal layer of a LCD, the liquid crystal molecules are misaligned in their unexcited state. Due to the way the light passes through the different layers of

the screen, when these molecules are electrically aligned, light passes through the screen easily. Very basic LCDs are built with a mirror at the back of the device and it utilizes the light from the surrounding area. By blocking the light from passing through certain areas the device is able to produce a simple image. This type of display is typically seen in digital watches and alarm clock displays. These cheap LCDs need an external light source in order to be legible and that is the purpose of the mirror at the rear of the LCD, it reflects the ambient light in the room through the screen. The shape of the electrode on the upper level of the screen can be nearly any shape. This is more commonly seen on inexpensive devices that need to display special characters. If the device needs to display where ambient light is not available, then a light source is added behind the screen to provide the light necessary for displaying images.

Simple LCD screens are made using a technology called passive matrix. Passive matrix LCDs are made using two substrates, one used for rows and one used for columns. When the microcontroller sends a positive charge down one of the columns and a ground connection to one of the rows, the corresponding pixels at that location in the matrix are aligned and the pixel turns on. This kind of control has a slow response time, as well as imprecise voltage control. Imprecise voltage control results in reduced resolution as some of the voltage in the column leaks into other columns, turning on more than the intended pixels. In low cost screens, this is not necessarily an issue. The other form of LCD screen control is active matrix control. Active matrix control allows for much more precise control of the liquid crystal layer of the screen as well as being able to only partially turn on the pixel. Partially turning pixels on or off is what creates a gray scale display. Active matrix control uses thin film transistors on the glass substrate of the screen in a matrix arrangement. Because of the use of transistors to control the pixels, the resolution of the screen is greatly increased. With recent advances in manufacturing, active matrix control rivals passive matrix in cost and therefore passive matrix control is all but obsolete.

One of the biggest breakthroughs in power saving for LCDs is the ability to maintain a pixel in either the on or off state without any power. The only time the screen needs power is to change the pixel from one state to the other. This technology is most widely used in portable devices such as E-Books. This is a great feature for battery-powered devices. The display in our project does not need to display information while it's not powered and a simple LCD will suffice.

2.3.4: Keypad

In designing the small-scale implementation of the rain simulator, we knew we would need a way to enter the desired rainfall intensity and the duration. We began to source components, and one of the companies that had a good reputation and easy to work with parts was www.SparkFun.com. As we were browsing their website, we discovered that they sold a very inexpensive 4x3 matrix keypad. As we researched a little more, we discovered that the matrix

keypad input was very easy to implement and along with the low cost of the keypad, we decided to go with it for the small-scale model. Not only is matrix keypad input fairly easy to implement, Texas Instruments has sample code available for it. The keypad that we found at SparkFun is an AK-304-N-BWB with a 24Vdc contact rating at 20mA. The rated life of the device is 1,000,000 button cycles per key with an operating temperature of -20°C to 60°C[A2]. As this device was only going to be used in the small-scale model, it did not need to have any environmental protections. However, when we buit the final product, the keypad needed to be protected from both water and dust and therefore this keypad was only used for the small scale implementation.

Interfacing the matrix style keypad was pretty straightforward. For the small scale model, the keypad used was a 3x4 style keypad and the final product also used a 3x4 matrix style keypad. The difference between the two keypads is that the keypad used in the final product needed to be protected from the elements while the scale-model did not. There are some additional buttons that were needed for inputs on the final product but these additional inputs were located on the display bezel. There is only a small difference in the software needed to make the change from a 3x4 keypad only to a 3x4 keypad and the additional buttons on the display bezel. These additional buttons were used for functionality buttons (Start, Pause, etc) rather than numerical inputs. These additional buttons can be handled as if there was another column of buttons on the keypad.

In the small scale implementation, the keypad was used to input intensity and duration and one of the buttons was used to begin the simulation. By using a matrix style keypad in the small scale model, we were able to write the microcontroller code necessary to implement the input. Again, there is only a minor change in the code needed for the final product. The only difference in the code was that there was a 3x1 keypad added on the display bezel. This 3x1 keypad is used for the Start, Pause, and Reset of the simulation.

2.4: Microcontroller

Undoubtedly, the most integral part of our control system design is the central processing unit. A control system without intelligence is not a very good control system at all. Thus, much of our research went into determining what would be an optimal solution for implementing the intelligence in the rainfall simulator.

To optimize our system we had to decide on what would be sufficient to use for controlling the operation of the motors, handling the user interface, displaying any fault conditions and computing variables from the user inputs and rainfall sensors. The current system was designed using a laptop that handles the aforementioned requirements in a simple program within the operating system. However, when considering efficiency of this design, a laptop operating system is far more than what is needed to complete the task. Thus, we decided to base our

design around a single chip that is capable of implementing all of the necessary logic for our application.

We determined that our design needed to meet the required specifications at an efficient cost and still be flexible enough so that when we run into any unforeseen obstacles that may require more computing power, code size or input and output capabilities we will be able to use the same chip and not have to acquire another. Therefore, we wanted to choose the chip that could satisfy the need at little or no cost. To begin our search for the proper chip we determined a list of qualities that needed to be met by the chip in order to satisfy the need of our system. In order to meet our requirements, we deemed that the minimal capabilities that the chip must have are:

1. To have at least 6 Analog to Digital Conversion Pins
2. Have at least 30 General Purpose I/O Pins
3. To be able to perform serial communication
4. Have an affordable development kit
5. Have at least 2 Timers, for keeping track of motor delay and simulation time

One option taken into consideration was using a field programmable gate array, otherwise known as an FPGA. This was considered because we already had some experience with FPGAs in our digital systems class. FPGAs are extremely flexible and could perform all of the requirements with ease. However, we determined that even an FPGA would be overkill for this project and there are more cost effective options available in the domain of microcontrollers.

Naturally, another option taken into consideration was the Motorola 68HC11 microcontroller. We considered this controller because we had a good amount of experience with it in our embedded systems course. Also, we knew how to program it; and the development kit was readily available to us. However, after a small and very limited amount of research, we quickly found that this controller is completely obsolete. There are many more options available that have far greater capabilities and that can be implemented into our system with greater ease.

Obviously, there were many options available that were capable of optimally meeting the requirements of our design and many manufacturers to choose those microcontrollers from. We narrowed our search down to three main manufacturers that provide solutions: Texas Instruments, Atmel and Microchip. We selected certain chips that met our requirements and compared what each chip had available. The main aspects that we compared in the chips were how many A/D converters, GPIO pins, Serial Communication pins, timers, and the programming methods that were available. We also considered how much

memory the chip had, any special features and the price of each controller and its associated development kit. The following sections document the comparisons that we made with some available chips from each company, and the choice that we made for the optimal chip.

2.4.1: I/O, Timers, and Serial Communication

Some of the foremost characteristics that lead to our choice of and overall design of the microcontroller are what general-purpose input/output capabilities the chip has. We need enough flexibility within the chip to be able to accommodate parallel communication to the LCD of the user interface, and possibly the rain sensors. There also needs to be some form of serial communication for the rain sensor interface. Additionally, the number of timers available for delay time calculations also needs to be taken into consideration. Table 2.4.1a compares the available facets of the microcontrollers considered for this project.

Manufacturer	Part Number	GPIO Pins	Serial Peripherals	Timers
Texas Instruments	MSP430F4132	56	UART, I^2C, SPI, LIN, IrDA	5
	MSP430F5435	67	UART, I^2C, SPI, LIN, IrDA	5
	MSP430F5510	47	UART, I^2C, SPI, LIN, IrDA	6
Microchip	dsPIC33FJ16GS404	35	1xUART, 1xI^2C, 1xSPI	4
	dsPIC33FJ32GS610	85	2xUART, 2xI^2C, 2xSPI	5
	PIC24HJ16GP304	35	1xUART, 1xI^2C, 1xSPI	4
Atmel	ATxmega192A3	50	7xUSART, 2xTWI, 3xSPI	7
	ATxmega128D4	34	2xUSART, 2xTWI, 2xSPI	4
	ATxmega256D3	50	3xUSART, 2xTWI, 2xSPI	5

Table 2.4.1a - Comparisons of I/O Capabilities for Microcontrollers [D7][D8][D9][D10]

2.4.2: ADCs and Special Features

Another important aspect of the microcontroller is how many analog to digital conversion channels the chip has available on it. Because we plan to have at least 4 rain sensors that can communicate an analog signal, that represents the rainfall level, we need to have 4 ADC channels at absolute minimum. Also considered in this section is what special features each chip has on it that separates it from the others; table 2.4.2a considers these characteristics.

Manufacturer	Part Number	ADCs	Special Features
Texas Instruments	MSP430F4132	8	Optimized For LCD Control
	MSP430F5435	14	Optimized For ZigBee/RF
	MSP430F5510	8	USB Compatible
Microchip	dsPIC33FJ16GS404	8	C Optimized Instruction Set
	dsPIC33FJ32GS610	24	On-Chip LDO Voltage Regulator
	PIC24HJ16GP304	13	12 Bit ADC Resolution
Atmel	ATxmega192A3	16	2048-Byte EEPROM
	ATxmega128D4	12	Fast Debugging Interface
	ATxmega256D3	16	256 kB Flash Memory

Table 2.4.2a - Comparison of the ADCs and Special Features of each chip [D7][D8][D9][D10]

2.4.3: Programming Methods and Pricing

Furthermore, what it will cost us to develop the software on the chip and the actual chip pricing weighed greatly on the choice of the chip. Also, since we designed the system to be programmable in real time, the final size of the code was undeterminable. Thus, the size of the memory on the chip needed to be considered in the final decision; table 2.4.3a displays features for each chip.

Manufacturer	Part Number	Programming Method	Memory (kB)	Chip Price	Kit Price
Texas Instruments	MSP430F4132	C, Assembly	8	Free	MSP430 Development Board: $4.30
	MSP430F5435	C, Assembly	192	Free	
	MSP430F5510	C, Assembly	32	Free	
Microchip	dsPIC33FJ16GS404	C	16	$2.77	PICKit3: $44.95
	dsPIC33FJ32GS610	C	32	$4.41	
	PIC24HJ16GP304	C	16	$2.42	
Atmel	ATxmega192A3	C	192	Free	AVRISP mkII : $34.00
	ATxmega128D4	C	128	Free	
	ATxmega256D3	C	256	Free	

Table 2.4.3a - Comparison of Price, Programming and Memory Space [D7][D8][D9][D10]

2.5: MSP430F5435 Microcontroller [L8]

Due to the high likelihood that the Texas Instruments MSP430F5435 (will be referred to as F5435) microcontroller would be used in the final design, it was decided that research was needed on the specific capabilities of this microcontroller that apply to this project. Peripherals that were used such as the LCD screen, the keypad, the wireless rain sensors, and the motor drivers all require different functions of the microcontroller. Some of the specific functions that are applicable to this project include:

- Unified Clock System that can be sourced by external crystal oscillators
- Universal Serial Communication Interface
- Real Time Clock (which can also be used as an interval timer)
- Several built in timers with extensive interrupt capabilities
- Direct Memory Access
- Pulse Width Modulation Outputs

The microcontroller is capable of other functions that could also be used in the project, but at this stage, a thorough understanding of these fundamental capabilities was necessary before the final project could be built and the software written. The purpose of writing this research was to have some reference material that could be used when building and programming the final design. The following paragraphs will give an overview of the above topics and explain how to use each one.

2.5.1: Unified Clock System

The F5435 microcontroller has a very flexible clock system which can be quite confusing to set up. Three internal clock signals are available for the user to select to source the microcontroller's built in modules and the main processor. For the F5435 microcontroller, each of the three internal clock signals can be driven by up to four clock sources. The sources can be either internal or external to the microcontroller. By having all of these options, the microprocessor is able to be used for a wide variety of applications and most notably low power applications. For the purposes of this project, most of these clock options are unnecessary, but some of them could be put to use with an understanding of their purpose and how they operate. In the following paragraphs it is important to distinguish between the words *source* and *signal*. The use of the word *signal* is meant as a final clock signal that is directly used by the processor or internal/external peripheral modules. The use of the word *source* is meant as the initial generation of clock pulses that are manipulated in several ways until they finally produce a useable clock signal.

The three clock signals that are available for use are the Master Clock, the Sub-Master Clock, and the Auxiliary Clock. The master clock is always used to

source the CPU, and the other two are usually used to source the peripherals or internal modules. As previously stated, each of the three clock signals can be sourced by one of four clock sources. The four sources that can be used are:

- XT1CLK (External Clock Source 1)
- XT2CLK (External Clock Source 2)
- VLOCLK (Low frequency internal oscillator)
- REFOCLK (High precision internal reference oscillator)
- DCOCLK (Digitally controlled internal oscillator)

The XT1CLK can be an external crystal oscillator or a resonator in high frequency mode or low frequency mode. It is typically a 32 kHz watch crystal. A crystal can be connected to the XIN and XOUT pins without the need for any external capacitors. Software selectable built in load capacitors make it easy to interface to a crystal. The internal capacitors that are available are 2 pF, 6 pF, 9 pF, and 12 pF. If these are not sufficient, then external capacitors can be used. XT1CLK can support external high frequency crystals by changing a few register values with software. If a crystal or resonator is not used, the XT1 oscillator can be sourced by some other external clock connected to only the XIN pin. The XIN and XOUT pins can also be used as general purpose input/output (GPIO) if desired. Several other configurations of external clocks are possible for a broad range of low power applications, but since they don't apply directly to this project it is not necessary to discuss them. The default setup of the XIN and XOUT pins is for the XT1 oscillator operation in low frequency mode, but it is disabled until the XT1 port is configured for XT1 operation. The XT2CLK is very similar to the XT1CLK, but it is primarily used for high frequency crystals or resonators. When higher frequencies are needed power consumption goes up because the driving voltage must be increased.

The next clock source is the VLOCLK. This is an internal very low frequency, low power oscillator with the ability to output a 10 kHz signal. It is generated without the use of a crystal and is therefore not very accurate. It is typically used for low power applications that do not require a high degree of accuracy. It would not be suitable for sourcing a timer and thus likely to not be used in this project.

The next clock source is the REFOCLK which is an internal frequency reference oscillator. It outputs a typical 32,768 Hz signal which is internally trimmed for a high degree of accuracy. This is a very useful feature because it can be used to accurately source clock signals in the microcontroller without the need for an external crystal. When used in conjunction with the Frequency Locked Loop Reference Clock (FLLREFCLK), very stable clock signals can be produced. The FLLREFCLK is an extension of the XT1 and REFOCLK. The frequency locked loop continuously counts up or down a frequency integrator and is used to further stabilize a clock source so that the effects of temperature and voltage fluctuations are minimal.

The final clock source that is available in the F5435 is the Digitally Controlled Oscillator (DCO). This oscillator is useful because it can be adjusted by software. It can be a very accurate clock source if it is stabilized by the FLL; however this limits the number of selectable frequencies to integer divisors of the FLLREFCLK. In addition if the FLL is used, it means that another clock source must also be used to source the FLLREFCLK. As previously discussed the two clock sources that can be used for this are the XT1 clock and the REFOCLK. If high accuracy is not needed then the FLL can be disabled and the DCO is sourced by the DCO clock. The DCO is a very fast startup clock source. It is used as the default source for the Master Clock (which runs the processor). When the processor starts up, it only takes 5 microseconds for the DCO to stabilize and instruction execution can begin. Ironically, the first instructions that are executed are often instructions to change the DCO.

To further complicate the clock system, the sources can be adjusted with the use of prescalers. As an example, the REFOCLK can be used to source the Master, Sub-Master, or Auxiliary clocks. It can be independently divided by 1, 2, 4, 8, 16, or 32 creating a wide range of different frequencies for different clock signals. Fail safes are also built into DCO and XT1 clock sources. If a crystal is being used as the XT1 clock source and the FLLREFCLK and a fault occurs in the crystal, the REFOCLK is automatically selected as the FLLREFCLK. Similarly if a fault is detected on the clock sourcing the Master Clock, the Master Clock will automatically be sourced by the DCO or the REFOCLK depending on the frequency setting of the Master Clock. The default operation of the Unified clock system module is:

- Auxiliary Clock sourced by the XT1CLK oscillator in low frequency mode
- Master Clock is 1 MHz sourced by the DCOCLKDIV
- Sub-Master Clock is 1 MHz sourced by the DCOCLKDIV
- DCO is 2 MHz sourced by FLLREFCLK
- DCOCLKDIV is at 1 MHz sourced by FLLREFCLK
- FLL is enabled and FLLREFCLK is sourced by XT1CLK
- XIN and XOUT pins are set for general purpose I/O and XT1CLK is disabled

Since the XT1CLK is initially disabled, a fault will be detected for the XT1 oscillator and cause the Auxiliary clock to be sourced by the REFOCLK. As a result, the Auxiliary clock will default to 32,768 Hz.

2.5.2: Universal Serial Communication

UART: Communicating to and from a microcontroller in any embedded application is very important. The MSP430F5435 microcontroller has a built in Universal Serial Communication Interface (USCI) module that allows it to communicate in a variety of ways. The first of these ways is UART (Universal Asynchronous Receiver Transmitter). As the name implies this is an asynchronous form of communication meaning that transmitting and receiving

devices do not have to have a synchronous bit rate. This allows for fewer pins to be used and easier programming techniques. It is designated universal because the data format and transmitting speeds can be configured to cover a wide range of values. UART communication requires only two pins, one for transmitting and one for receiving. In this regard it is a full duplex system because it can communicate in both directions. The UART module on the F5435 microcontroller allows the user to set up the communication parameters easily in software by storing values in specific registers. When communicating in UART mode the main parameters that must be considered for transmitting characters are: start bit, stop bit, parity bit, number of data bits, MSB first or LSB first, address bit, and the baud rate.

The start bit tells the receiver that data is coming and the stop bit lets it know when the data transmission is over. The parity bit is used for error detection. It is a way for the transmitter to tell the receiver if an even number of bits or an odd number of bits will be transmitted. If the parity bit is a one, the receiver will expect even parity. If it receives an odd number of ones, it can set an error flag to indicate that a problem has occurred. The parity detection can also be disabled completely if desired. If data is to be transmitted one byte at a time then 8 data bits will be selected. The other option is to use 7 data bits. Data can be transmitted in either MSB (most significant bit) or LSB (least significant bit) first. In UART communication, LSB first is more common. The address bit is used when three or more devices are communicating asynchronously. This is known as master/slave or multidrop and is one of the “multiprocessor” formats supported by the F5435 USCI module. All devices share the same transmit line, so each device needs to have its own identification address so that it will know if it was the desired recipient of the data being sent. Setting the address bit in a transmission indicates that the data is the address of one of the desired devices. The other devices will then ignore the next transmission. The address bit is not needed if only two devices are communicating. The F5435 microcontroller also has a baud rate generator with additional modulation capabilities that allow for fractional and more accurate baud rates to be generated from a multitude of different source clock speeds. Baud is by definition the number of symbols per second or the number of discrete signal events per second that occur on a communication channel. It is related to the number of bits transmitted per second (bit/s), but not always equal. If more than one bit is required per signal event then the baud rate will be less than the bit rate. The UART module on the F5435 has independent transmit and receive shift registers which allows for faster speeds and independent flags and interrupts for transmitting and receiving.

After setting up the Universal Serial Communication Module for UART mode (writing values to specific registers for start bit, stop bit, parity,…) transmitting and receiving is rather simple. If a character is to be transmitted, the character needs to be written directly in to the transmit buffer. As soon as the write takes place, the baud rate generator automatically starts and transmits the character at the selected baud rate. The transmit buffer places the data into a shift register,

so when a transmit takes place, the bits are shifted out serially according to the baud rate. When the last bit has been shifted out (the stop bit), an interrupt flag is set indicating that the transmission is complete and the buffer is ready for another character. The interrupt flag can be handled with conditional statements or if more complex action is needed, it can be handled in an interrupt handler routine. When the transmission completes, the baud rate generator automatically turns off to conserve power. As soon as the next character is written into the transmit buffer, the interrupt flag automatically clears, the baud rate generator starts back up and the cycle continues.

The receive buffer works in a very similar manner. The receiving device must be configured to receive at the same baud rate as the transmitting device. While the rate needs to be the same, their clocks do not need to be synchronized. The F5435 has an automatic baud rate detection capability. The transmitting device must transmit a special sequence of characters to the receiving device. The receiving device will then automatically write the proper values to its baud rate register to match the transmitter. Of course if the transmitting baud rate is known, it is easier to manually configure the receiving baud rate to match it. If the need arose to transmit at multiple baud rates, this capability might come in handy. The receive sequence is as follows: the falling edge of the start bit starts the baud rate generator and a character is shifted in one bit at a time. When the stop bit has been read, the receive flag is set along with any other error flags. The data can then be read directly out of the receive buffer. When the data has been read the receive interrupt flag is automatically cleared and another character can be shifted in. Several errors can occur when receiving the data. A framing error occurs when the stop bit is read as a low value instead of a high value. A parity error occurs when the number of ones in a character does not match what the parity bit said it should be. An overrun error occurs when another character is shifted into the receive buffer before the previous one has been read out. In each of these cases individual error flags are set. They can trigger an interrupt handler to deal with the problem or they can be ignored. How the interrupts are dealt with is entirely up to the programmer.

SPI: Serial Peripheral Interface (SPI) is the next mode of communication that is supported by the MSP430F5435 microcontroller. SPI is another form of serial communication, but unlike UART, SPI is synchronous and uses three or four pins. One of the pins sends a clock signal to the receiving device so the sending and receiving devices will have a synchronous clock. SPI has some of the same features as UART, but does not require a start bit, stop bit, or parity bit. The main features of the SPI are: 7 or 8 bits of data, LSB or MSB first, master or slave modes, independent transmit and receive shift registers and buffers, continuous transmit and receive capability, selectable phase control and clock polarity, and independent interrupts for receiving and transmitting. Figure 2.5.2a shows the typical connections between an MSP430 master and an SPI slave. The four pins that are used in SPI communication are:

- SIMO – This is the Slave In Master Out pin. On the master controller this is the output and on the slave this is the input.
- SOMI – This is the Slave Out Master In pin. On the slave controller this the output and on the master controller this is the input.
- CLK – This is the Clock pin. The master controller is the source of the clock, so the CLK pin is an output on the master. On the slave the CLK signal is used to synchronize the communication so the CLK signal is an input.
- STE – This is the Slave Transmit Enable pin. It allows the master to enable or disable the external SPI devices. Separate pins can be used to individually disable or enable slaves in a multiple slave configuration. The STE pin can also be driven by the slave. In this configuration the slave becomes the master.

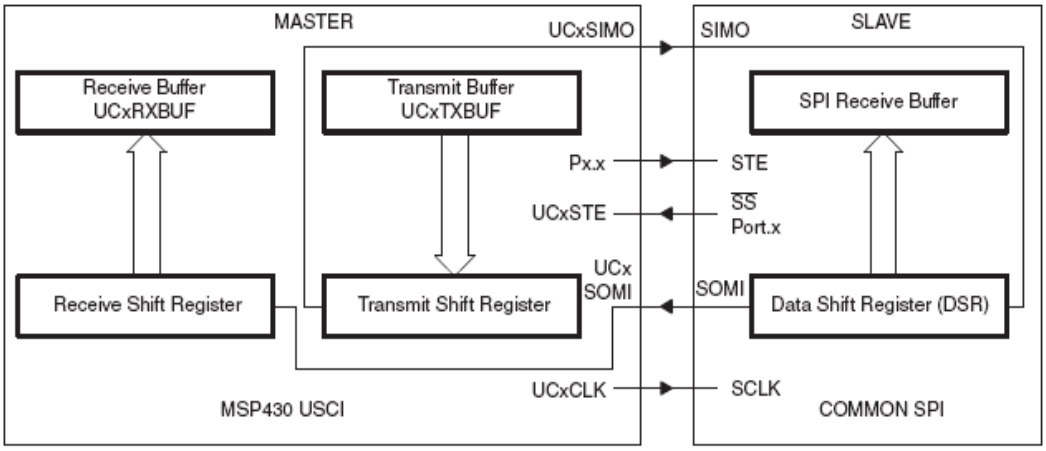

Figure 2.5.2a- SPI Interface Master Mode [L8]
Connections from Master to Slave in 3 or 4 wire mode.
(Permission Granted by Texas Instruments)

Operation of the SPI communication module is similar to the UART communication. In transmit mode a character is written into the transmit buffer. This automatically clears the transmit flag, starts the bit rate clock, moves the data into the transmit shift register, and shifts the data out one bit at a time at a speed according to the bit rate clock. This is the same bit rate clock that the UART module used to generate the baud rate. In SPI mode, modulation of the bit rate clock is not possible or necessary. Since the transmitting and receiving devices are guaranteed to have synchronous clocks, a precisely accurate clock rate is not necessary. The start and stop bits are not necessary either because the number of bits shifted in or out is directly related to the bit clock rate which is synchronous to both the transmitting and receiving devices. In SPI mode transmitting and receiving can occur at the same time because of the independent transmit and receive shift registers and buffers.

As with UART, a large number of interrupts can be generated in SPI mode. All of the interrupts are prioritized and combined so that one interrupt service routine can be used to quickly decode which interrupt flag triggered the interrupt. When an interrupt occurs a number corresponding to a particular interrupt flag is written into an interrupt vector register (UCA0IV). If interrupts are enabled, the interrupt service routine will be entered and the number that was stored in the interrupt vector register can be extracted and used to call different functions that are written to handle the different interrupt types.

I^2C: The third type of serial communication available in the F5435 microcontroller is called I2C, which stands for Inter-Integrated circuit. This is a two-wire synchronous serial communication scheme that can have multiple masters and slaves and operates in full duplex mode. Figure 2.5.2b illustrates the connections and the pull up resistors required for I2C communication.

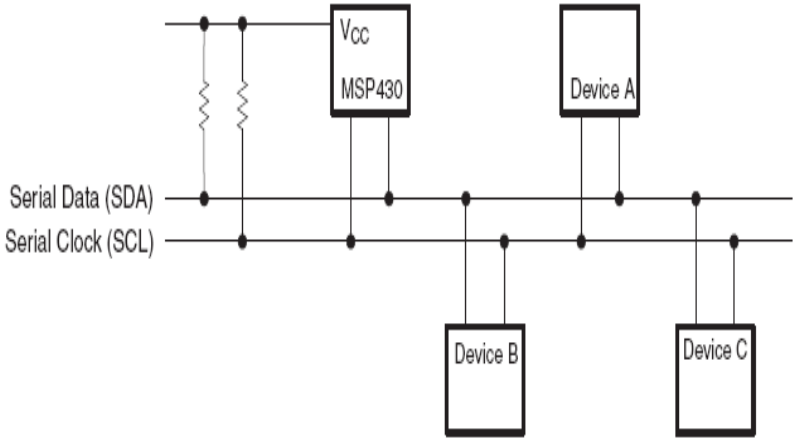

Figure 2.5.2b - I^2C Communication [L8]
This Figure shows the I2C communication bus. Many devices can be used as master or slave.
(Permission Granted by Texas Instruments)

In a sense it is a combination of UART and SPI. Each clock pulse of the bit rate clock causes one bit of data to be transmitted or received. Data is transferred by bytes and always operates with MSB first. The two pins that are used are called SDA and SCL. SDA is the serial data line and SCL is the serial clock line. Both of these lines must be pulled high with pull up resistors. The clock line is used to synchronize all of the devices and the data line is bidirectional for transmitting and receiving. Each device is assigned a unique address. Data is transmitted by initiating a start condition followed by a 7-bit address corresponding to one of the devices along with a *R*/*W* bit. The *R/W* bit indicates whether the master is receiving or transmitting data to or from the slave.

2.5.3: Real Time Clock

The F5435 microcontroller is equipped with a real time clock that has the ability to keep track of seconds, minutes, hours, day of week, day of month, month, and year and even adjusts for leap years. It has interrupt capabilities programmable alarm settings, and it can be calibrated for accuracy with software. The real time clock module of the microcontroller can also be configured for use as a general-purpose counter. When the real time clock is used in counter mode it provides a 32-bit counter by cascading four 8-bit counters together. This allows for 8, 16, 24, or 32 bit overflow conditions which is slightly more flexible than the other built in timers of the F5435 which have a maximum of 16 bit overflow. The alarm system in the real time clock module is very flexible, but it can only be used in calendar mode, not counter mode. It can be programmed to trigger based on minutes, hours, day of week, day of month, or some combination. As an example, the user can set the alarm to trigger at exactly 2:30 pm on the fourth day of every month.

Like nearly every module of this microcontroller the real time clock module must be configured with software before it is of any use at all. This includes the clock that it will be sourced by. For maximum accuracy it is best to source the clock from an external crystal. When this is done, the calendar clock is not synchronized with the processor clock. This can cause a problem when reading the real time clock registers. The registers are updated with new time data once every second. If the user software happens to be trying to read the minutes from the minutes register at exactly the same time that the minutes register is being updated, then erroneous data may be obtained. The real time clock module avoids this problem by creating a “keep out” window of time that any reads during that time should be ignored. When a register is being updated a “keep out flag” is reset (goes to zero). An interrupt service routine can be written so that when the keep out flag is set, the program will enter the interrupt service routine and it has nearly a full second to read valid data. Even in a relatively modest case where the processor clock is only running at 1 MHz, each cycle is a microsecond, and reading a register value should only take 10 to 20 clock cycles. About 50,000 valid reads could occur during the one second that the data is valid meaning that one second is more than enough time to read all the registers associated with the real time clock module.

2.5.4: Built in Timer Operation

The MSP430F5435 has three independent timer modules built into it. Like all of the built in modules, each timer must be sourced by one of the clock sources mentioned in the Unified Clock System part of this paper. Each timer can be sourced by any combination of the different clock sources and dividers that create a wide variety of options for timer speeds. The timers can be used in two main modes of operation. They can be used in capture mode or compare mode.

In addition the timer can generate outputs to the port pins. The specific timer modules on this microcontroller are:

- Timer A0 (has 5 Capture/Compare registers)
- Timer A1 (has 3 Capture/Compare registers)
- Timer B0 (has 7 Capture/Compare registers)

Compare Mode: The compare mode is used when using the timer as a *counter*. To add another level of complexity, the *counter* has four different modes of operation. It is really only three modes of operation because one of the modes is OFF. The three counting modes are UP mode, CONTINUOUS mode, and UP/DOWN mode. Table 2.5.4a summarizes the four operating modes.

MCx	Mode	Description
00	Stop	The timer is halted.
01	Up	The timer repeatedly counts from zero to the value of TAxCCR0.
10	Continuous	The timer repeatedly counts from zero to 0FFFFh.
11	Up/Down	The timer repeatedly counts from zero up to the value of TAxCCR0 and back down to zero.

Table 2.5.4a- Counter Operating Modes [L8]

These are the four modes operation. MCx is the value that must be written into the control register for the desired mode of operation.

The up mode works by having the counter count up to a software-defined value and then start over from zero. This way the user has complete control over the timer period. The software-defined value for the timer period must be written into the CCR0 (Capture/Compare register 0). When the timer counts up to that value an interrupt flag is set. If the CCR0 interrupt is enabled then an interrupt service routine will be entered and the software in that service routine will determine what action is taken. One use of this mode of timer operation is for stepping the stepper motors in our project. The following code segment is used to initialize Timer A1:

```
TA1CCTL0 = CCIE;          // CCR0 interrupt enabled
TA1CCR0 = 327;            // Capture/Compare register 0 value
TA1CTL = TASSEL__ACLK + MC__UP + TACLR;      // ACLK, up-mode, clear
```

This sets up TimerA1 to be sourced by the auxiliary clock, which runs at 32768 Hz. If the timer is to be used to step the motor every 10 (ms), then a value of 327 must be written into the CCR0 register. When it counts to 327 an interrupt service routine will be entered and the code used to step the motor one step will be executed. Timer A1 has two other Capture/Compare registers, which can also be used to generate interrupts at different timer periods. However they will not cause the timer to start over. Since it can be difficult to keep track of multiple interrupts from multiple sources, all of the interrupts are combined and prioritized

to source a single interrupt vector. The interrupt priority goes down with each higher value of Capture/Compare register.

The continuous mode works by continuously counting up to 0xFFFF before starting over. In this mode, the timer can generate interrupts along the way, but doesn't restart its count until it reaches 0xFFFF. When the timer counts to the value in one of the capture compare registers, an interrupt service routine is entered. Any value can then be added to the current value of the counter. This allows the user more flexibility in the generation of timer periods because each period can be different from the previous one.

The up/down mode works by having the counter count up to a certain value then count back down to zero. Again, it can generate interrupts along the way but always continues on the path of counting up and down without any abrupt restarts. This mode of operation is useful when multiple outputs are being generated for an application where dead times are needed between the output signals. The example TI uses is for an H-bridge where it is never desirable to have both of the outputs driving the H-bridge to be high simultaneously. This would create a situation where the power supply is directly shorted to ground. A direct application of this feature could be used in our project.

Capture Mode: The timer can also be used in capture mode. This mode is used for time measurements and can be used to compute speeds. When a capture is initiated the value of the timer is automatically copied into the CCR0 register and the interrupt flag is immediately set. If the CCR0 interrupt is enabled, the interrupt service routine is entered. The value of the CCR0 register can then be read and used in some type of computation. As an example, if the timer value is read during two different interrupts, the two values can be subtracted to determine the time between interrupts. Since the captures can be initiated by external sources, this provides a very useful way to take very accurate and fast time measurements when a hand-operated stopwatch won't suffice.

Output Mode: One extremely useful feature of the timer modules is their output modes. Each timer has up eight different output modes. The output mode of the timer allows it to generate an output automatically without the use of the processor. When the timer overflows in counter mode, an interrupt flag is set and an action must be initiated by the processor in an interrupt service routine. This means that whatever the processor was doing, it must stop and go service the interrupt routine. When the timer is used in output mode this is not the case. The timer module will generate an appropriate output without the processor ever knowing about it. Table 2.5.4b summarizes the eight different output modes.

OUTMODx	Mode	Description
000	Output	The output signal OUTn is defined by the OUT bit. The OUTn signal updates immediately when OUT is updated
001	Set	The output is set when the timer counts to the TAxCCRn value. It remains set until a reset of the timer, or until another output mode is selected and affects the output.
010	Set/Reset	The output is set when the timer counts to the TAxCCRn value. It is reset when the timer counts to the TAxCCR0 value.
011	Toggle/Reset	The output is toggled when the timer counts to the TAxCCRn value. It is reset when the timer counts to the TAxCCR0 value.
100	Toggle	The output is toggled when the timer counts to the TAxCCRn value. The output period is double the timer period.
101	Reset	The output is reset when the timer counts to the TAxCCRn value. It remains reset until another output mode is selected and affects the output.
110	Toggle/Set	The output is toggled when the timer counts to the TAxCCRn value. It is set when the timer counts to the TAxCCR0 value.
111	Reset/Set	The output is reset when the timer counts to the TAxCCRn value. It is set when the timer counts to the TAxCCR0 value.

Table 2.5.4b - Timer Module Output Modes [L8]

2.6: Power Supply

One of the largest problems that the Storm Water Management Department was having with their existing system was that the power supply for the control circuitry was intermittent. Thus, we were tasked with providing a reliable power supply in our design. As an integral part of our design, the power supply needed to be capable of providing power to the stepper motors as well as the control circuitry and the user interface. At first we considered designing a single unit that could provide the appropriate levels of power to each part of the control system. However, we soon determined that this aspiration was impractical because of the drastic differences in power levels required by the stepper motors and the controlling circuitry. From the existing rainfall simulator system we saw that the control circuitry had its own power supply and that each of the Applied Motion motor drivers had their own supplies enclosed in a black box, literally. Thus, we decided to design two different modules for the respective applications. For the stepper motors we needed to design a high voltage supply, while we designed a separate low voltage supply for the control circuitry and the user interface.

2.6.1: Regulated vs. Unregulated Design

When considering what type of power supplies we should design for the project, we considered the necessity of having a regulated power supply for each the stepper motors and the control circuitry. The benefit of the regulated power

supply is that the voltage level of output of the circuit is maintained at a desired level, regardless of the current draw of the load, with a small margin of error. However, the tradeoff is that the circuitry is more complicated and expensive, in most cases. The un-regulated power supply is beneficial in that the circuitry is very simple, thus the cost is minimal. Also,

> An unregulated linear supply is less expensive and more resilient to current surges; however, voltage decreases with increasing current draw. This can cause serious problems if the voltage drops below the working range of the driver. Also of concern are the fluctuations in line voltage. This can cause the unregulated linear supply to be above or below the anticipated voltage [D1].

Therefore, determination of the type of supply depends on how sensitive the load is to fluctuations in its power input. For the control circuitry and user interface, which are composed of highly sensitive components like the microcontroller, LCD and over-travel sensors, we determined that a low voltage regulated power supply was necessary. For the stepper motor power supplies we determined that a high voltage, high current, un-regulated power supply would be optimal and even necessary, for reasons explained in the appropriate section below.

2.6.2: Linear vs. Switching Power Supply

Because some of the components that this power supply supports are highly sensitive to fluctuations in their input voltage, we determined that we must design a regulated power supply. This choice left us with two basic options for the configuration, a linear regulated or a switch mode regulated power supply.

The benefits of using a linear regulated power supply are mainly that the circuitry for the supply is simple and the components are relatively cheap. However, its downfall is that it is not electrically efficient; the regulator essentially throws away any excessive current to regulate the output. Basically, a linear regulator has current flowing through it, continuously.

Depending on how much power the linear supply has to regulate, the circuit can generate relatively large amounts of heat. Because the plan is for the end product to be enclosed in a water resistant case, the heat from the linear supply could greatly affect the ambient temperature for all of the components of the control circuitry.

As a contrast, the switch mode power supply is much more electrically efficient; the basic concept for the switching supply is that it takes the unregulated power and passes it through a switching transistor, typically a MOSFET, before it is passed on to the load. To regulate the output, the transistor is switched on and off, which essentially creates a square wave for the current applied to the load. A divider network monitors the output voltage and the duty cycle of the applied the switching transistor regulates current. This is in order to maintain the appropriate

average current to maintain the load voltage at a constant level. Thus, we see that instead of sinking all of the excess current into ground, the switching supply simply restricts the current from flowing through the device. Hence, the efficiency of a switching regulator is much greater than a linear device, and it generates much less heat.

However, the downfall of the switching regulator is that the circuitry can be more complicated. And, because the regulator modulates the switching transistor at relatively high frequencies, the switching supply can generate a significant amount of electromagnetic interference.

We found that the motor driving circuits constrained the power supplies in our design. The datasheet for our drivers states that the 12V inputs to the drivers are very sensitive, and that a linear regulated supply should be used [A1]. We would prefer to use a switching regulator because the heat dissipated is much less than that of the linear regulator. However, because of the constraint of the sensitivity of the driver circuits a linear regulated power supply was the only option for this design.

2.7: Spray Nozzles [D11]

The characteristics of the rain nozzles that are used to deliver the simulated rainfall to the test bed were an important factor in the design of the control system. The nozzles are attached to a swing arm that is controlled by the stepper motors. The swing arm defaults the position of the nozzles to emit water into a runoff trough, such that it does not reach the test bed. When rainfall to the test bed is desired, the swing arm pans the nozzles across an exposure window, allowing the rain to fall on the test bed. The intensity of the rainfall can be determined by how long the nozzles are exposed to the window. So, for higher intensities of rain, the nozzles must be exposed to the window for longer amounts of time, visa-versa for lower intensities. Thus, the stepping rate of the motors is dependent upon the desired rainfall intensity. The present simulator boasts the capability of providing rainfall intensities from 0 to 12 inches per hour. The nozzles that are used are Veejet 80100 nozzles. The significance of these nozzles is that they can deliver water droplets that resemble the size of natural raindrops, if the water pressure on the nozzle is maintained at 6 PSI. The flow rate of the water dispersed from one nozzle is approximately 15.0 $inches^3$ per second, at 6 PSI. The rainfall simulator has 20 nozzles that deliver the simulated rain. Thus, the maximum continuous rainfall rate can be determined for the 30’ by 8’ test bed, using the equation below.

$$Rainfall\ Rate = \frac{20 * 15.0\ inches^2/sec}{30 * 8 * 144\ inches^2} \approx .00868 \frac{inches}{sec}$$

Thus, the simulator will provide approximately 8.7 milli-inches of rainfall to the test bed for every second that the nozzles are exposed to the windows. Therefore, to maintain a resolution of 0.1 inches, the stepper motors must be capable panning the nozzles across the exposure windows with only 10 seconds of exposure time error during any test. The nozzle swing arm is attached to a disk that is mounted on the shaft of a 10:1 ratio gear head that is mounted to the shaft of the stepper motor. As the disk rotates within a 90 degree region the nozzles pan across the exposure window. Therefore, the stepper motors must be capable of repeatedly rotating 900 degrees at high step rates while maintaining sweep-time error of less than 10 seconds. In the experimentation with the small-scale model it was determined that the stepper motors are more than capable of performing the necessary rotation rate.

2.8: Transmission Lines

An important aspect of the operation of the control system is the interface for the control signals to be carried from the control system to the end items. The transmission lines between the control system enclosure and the stepper motor enclosure provide this interface. The Storm Water Management Department expressed concern that the transmission lines may be a factor in the poor response of the present system. They expressed that a nice perk for our design would be to have longer transmission lines, to keep the control system away from the rainfall. However, the designer of the present system expressed that the lines are as long as they can be, to maintain signal integrity. Figure 2.8a shows the end items of the transmission lines, one at the plug end and the others in the stepper motor enclosure.

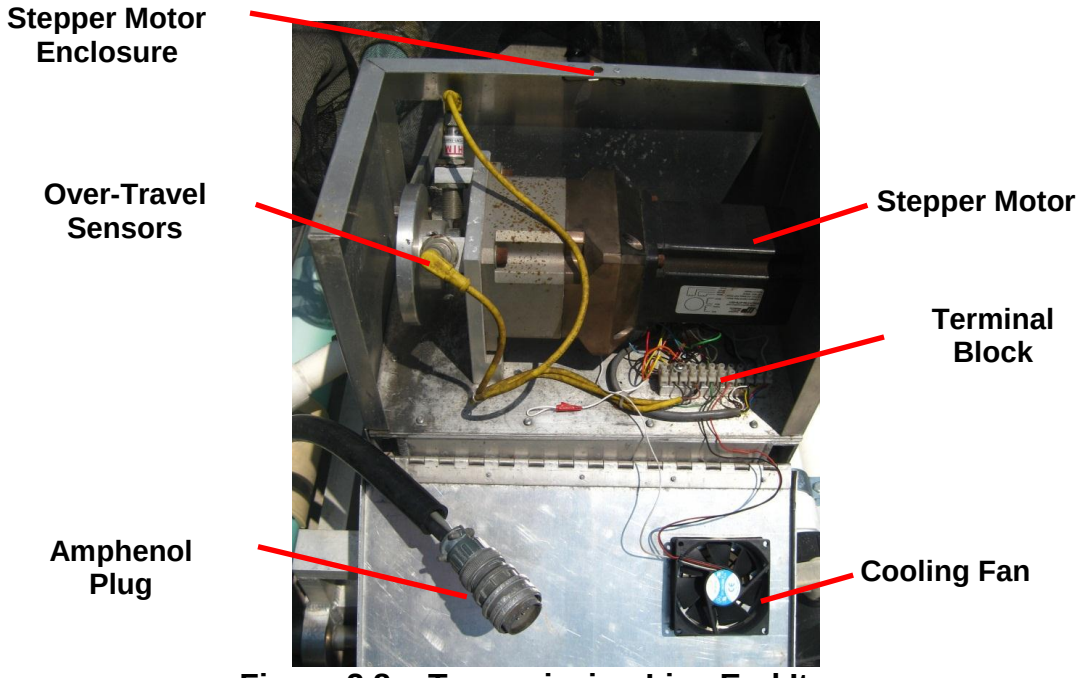

Figure 2.8a: Transmission Line End Items

The most important aspect of the transmission lines is how well they conduct the signals, or conversely, how much they attenuate the signals. If the control system sends a signal to the stepper motors and the signal is attenuated too much the motors will not operate in conjunction with the control system. Thus, the overall accuracy of the control system is dependent upon the transmission lines. Because there is no position feedback, other than the fault condition over-travel sensors, the quality of the signal is of great importance. Thus, it became necessary to determine whether new transmission lines were required.

Each of the two transmission lines on the present system are composed of two cables that originate at terminal blocks, in the stepper motor enclosures, and are threaded through a plastic tube, for protection against the elements. The cables terminate to an Amphenol 97-3102A-22-19 plugs that can be connected to the control system enclosure. The part numbers of the two inner cables cannot be determined because of the plastic tubing. However, after inspection, it was determined that one of the inner cables is composed of four 16-18 AWG conductors, while the other is composed of seven 20-22 AWG conductors. The 16-18 AWG cable is used for the control signals to the stepper motors, while the 20-22 AWG cable is used for the signals to and from the over-travel sensors and the cooling fan. To determine the attenuation of the voltage signal across the 50-foot conductors the following equation was used.

$$V_{drop} = \frac{I * (1.68 * 10^{-8}\Omega m) * (50ft)3.28\ ft/m}{\pi * \left(\frac{1}{2} * \frac{.005 * 92^{\frac{36-AWG}{39}}}{12}\right)^2}$$

Calculations were made for all of the probable wire gauges to determine the integrity of the signals to each of the end items in the stepper motor enclosure. Table 2.8b: depicts the percentage drop of the voltage signal for each end item.

Component	Power Rating	AWG	Voltage Drop	% Drop
OT Sensor	12V @ 0.2A	20	0.098V	0.82%
		22	0.157V	1.31%
Cooling Fan	12V @ 0.25A	20	0.123V	1.03%
		22	0.196V	1.63%
Stepper Motor	38V @ 5.6A	16	1.10V	2.89%
		18	1.74V	4.57%

Table 2.8b: Attenuation Levels for Conductor Possibilities

To maintain a 5.0% standard for signal integrity to the stepper motors the transmission lines cannot be extended if 18 AWG is used. Since the current draw is relatively small for the OT sensors and the cooling fan the length of the transmission lines is not constrained by their conductors.

Upon further inspection, it was found that the Amphenol plug end items are not properly insulated from the weather. There is a cavity that can accumulate water and dirt at the interface from the cable to the plug. When the simulator is not in use the cables are simply left outside. Thus, the plugs are susceptible to exposure to water, which can cause corrosion or electrical shorts that can damage the system or degrade the quality of the signals. To decrease expenses for our design it was decided that the present transmission lines will be kept. However, we deemed it necessary to properly insulate the plugs with an epoxy sealant at the interface from the cable to the plug.

2.9: Optional Wireless Integration

One of the tools that the rain simulation laboratory relies on heavily is the use of rain gauges. They have been obliged to manually measure the amount of rainfall coming from the simulator by putting on their raincoats and boots, taking their rain gauges, and manually measuring the amount of rainfall output by the machine. When there is only one simulation to be done for the day, this process may not seem to bother the lab workers. However, the reality is that the lab may have multiple materials for a variety of customers that they will need to test in one day. This renders the stone-age method of measuring the rainfall rather troublesome and time-consuming for the researchers.

One of our ideas was to have the rainfall automatically measured by rain gauges and have the information sent back to the microcontroller and ultimately to the user through a reasonably simple wireless system. This could allow the lab workers to stay dry and also eliminate or at least reduce the possibility of making errors in the manual rain measurement process. We planned to have the microcontroller accept the feedback from the rain gauges and readjust its speed and pause time, to make the test more accurate. Also, the data can be used to track the past performance of the simulator and to alert the lab team if and when the simulator will be due for maintenance, or in this case, re-calibrated. This data log would allow the team to see at which point in time during their simulation the rainfall intensity got off track. In total, there are three different wireless technologies considered: Wi-Fi, Zigbee, and Bluetooth. Of course, there is always the possibility of simply going for the wired version. The main differences between the different wireless communication systems are basically the frequencies at which they are transmitted, the way in which the signals are encoded for transfer, the range of transfer, price, and the uses for which they were developed.

We wanted to have a minimum of six rain gauges evenly distributed on the 8ft x 30ft rain bed. Two of the rain gauges would go at one end, two in the middle, and two at the other end. Because the Stormwater Lab typically tilts the rain bed in order to see the chemical run-off at a specific incline, the rain gauges reading will have to adjust itself in order to account for this incline. This is why we have chosen to place the rain gauges evenly dispersed on the entire surface of the rain bed. The wide displacement of the rain gauges will also help the research team decipher whether one of the spray nozzles is functioning correctly and is not clogged with something. It will be easy to tell if one of the rain gauges shows a drastic difference in rain measurement. Similarly, if there are high winds that may carry the rain away from the rain gauges, this will also be evident through the comparison of the individual rain gauge readings. Below in Figure 2.9a is the layout that we would like to have for the rain gauges. Currently we only show six rain gauges and we would like to add more in order to increase accuracy but the amount of rain gauges will ultimately depend on the funding we will receive from the Stormwater Lab.

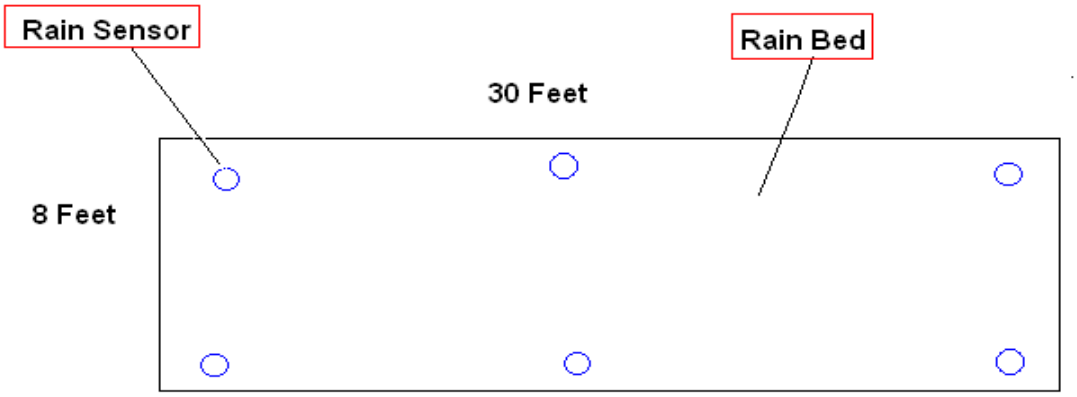

Figure 2.9a- Current Layout for 6 rain gauges

2.9.1: The IEEE 802.11 Wi-Fi Standard

The first wireless technology considered was Wi-Fi. Wi-Fi is perhaps the most popular wireless WPAN (Wireless personal area network) among people today. Due to the fact that Wi-Fi is used widely in homes, universities, and businesses there is a large source of help available especially when it comes to troubleshooting any issues that may arise. Below in Figure 2.9.1a is a typical Wi-Fi home network.

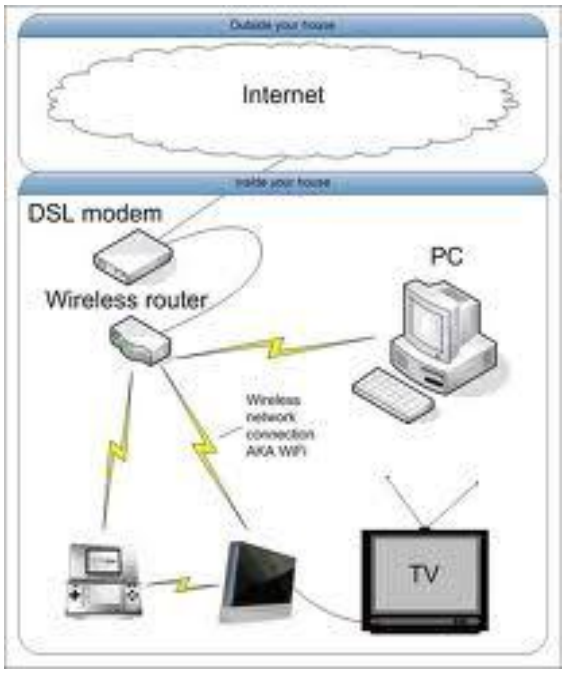

Figure 2.9.1a – Typical Wi-Fi configured network for home use. [B5]

Wi-Fi, just like the other wireless technologies researched, is governed by the IEEE 802.11 standard. Wi-Fi comes in 4 main types: a, b, g, and n. The fact that there are multiple varieties makes this appealing since our design plans for the small prototype built were very different from the final integration to the rain simulator. Although its flexibility was appealing, other variables still needed to be assessed. One of those variables was its power consumption. Out of the three wireless options, Wi-Fi by far consumes the most power. Since we have been working with the low-power MSP430 family of microcontrollers from TI, integrating a high power Wi-Fi wireless communication network would be more of a con than a benefit to our project. This is because our power supply will be designed with the use of low-power devices in mind. We already know that we will need a 3.3v and 5v output from our power supply to power our main micro controller so it would be much easier to be able to use the same voltage to power the wireless port. We also have to think about the fact that could have no more than 10 rain gauges reporting their results back to our main micro controller. Each of the rain gauges will require a power source and in this application, preferably a battery or wireless power source. With that being said, Wi-Fi does not have the capability to run off of a battery. Additionally, out of the three wireless implementations researched, Wi-Fi is by far the most expensive. It was difficult to find Wi-Fi transceiver modules that are easily integrated into the MSP430 microcontroller family, which is one of the main groups of microcontrollers that we are considering. Also, the implementation of a Wi-Fi network used simply to retrieve data from rain gauges that sit less than 20 meters away would be illogical since Wi-Fi has a range of up to 100 meters and

we would thus be using a high power, long range wireless solution for a low power, short range problem.

Additionally, Wi-Fi is one of the personal wireless area networks that transmit data in the ISM (Industrial, scientific, and medical) bands at the 2.4GHz or 2.5GHz frequency. This is where Wi-Fi's popularity, which we initially saw as a benefit, now becomes a problem. Due to the fact that there is an abundance of Wi-Fi networks with a much higher density at and around UCF and specifically around the rain simulation lab, there is the risk of us having to deal with the noise produced by other Wi-Fi networks set up to communicate at this same frequency. Fortunately, Wi-Fi has 11 smaller channels within its 2.4GHz bandwidth that can be used. If Wi-Fi is used, the Wi-Fi channel used will have to be carefully considered and chosen so that nearby Wi-Fi networks are not using the same channel. Lastly, Wi-Fi networks need around 3 to 5 seconds to connect which ranks it in second place when compared to its Bluetooth and Zigbee competitors. When it comes to speed, Wi-Fi has got the other two options beat with a transfer rate of anywhere from 11Mbit/sec to 54Mbit/sec. Although its high speed is impressive, it is not imperative for our application since we may only be sending a maximum of 15 signals from rain gauges to our microcontroller.

Although there is a lot of information on Wi-Fi and troubleshooting support, the vast majority of it concentrates on Wi-Fi being used as a communication network for computers to connect to the Internet. There are not many resources for interfacing a Wi-Fi transceiver to a microcontroller, and even less resources for integrating it with low-power family of microcontrollers that we are considering since their power needs are on opposite sides of the spectrum. For our specific application, it would make much more sense to go with a low-power wireless option that is also economical.

2.9.2 Bluetooth Wireless Technology [B11]

The second wireless option researched was Bluetooth. Bluetooth's power consumption lies in between that of Wi-Fi and Zigbee. Bluetooth was designed as a solution to short-range wireless applications. It has gained its popularity by being used for simple applications that were previously addressed with a short wire alternative. Bluetooth is really all about providing convenience to the end user. Examples of Bluetooth use include connectivity to PDAs, cell phones, and mice to computers. There are some Bluetooth modules that can extend its signal to reach up to 100 meters but that is not what Bluetooth was designed for nor is it very budget friendly. The extension of the Bluetooth signal also comes at a much higher rate of power consumption. The transfer rate for Bluetooth is between that of Wi-Fi's and Zigbee's at 1Mbit/sec with a connection time per device of up to 10 seconds. Because baud rate and connection time are not critical to our application, the transfer rate acts as a plus while the connection time just makes it a little less appealing. Bluetooth, like Wi-Fi and Zigbee, also uses the ISM 2.4GHz band to transfer its data and is also susceptible to noise

infiltration from surrounding networks. However, the risk for noise interference is not as great compared to other wireless technologies because Bluetooth does frequency hopping. Other wireless technologies like Wi-Fi only ends up occupying about one third of its total bandwidth while Bluetooth is constantly "hopping" from frequency to frequency.

Bluetooth uses up a much wider range of its bandwidth so in the cases where there is interference between signals, it only lasts for a fraction of a second since Bluetooth will be moving to another frequency channel very quickly. Bluetooth was also designed to transfer large packets of data at a time hence the reason why it is used to transfer signals from cell phone and PDAs. Along with the transfer of larger data packets comes a larger consumption of power. Relative to Zigbee, Bluetooth will need to have its battery replaced or recharged much more often. Bluetooth, like Zigbee and Wi-Fi, can also support multiple end devices being connected to the central device, or host, at one time. Bluetooth's devices have a master-slave configuration in which the main host of the network connects to each slave or end device. The slaves cannot talk to one another without going through the master device but a master and slave device can switch roles almost instantaneously. Even though one Bluetooth master can host up to 285 end devices, it can only actively communicate with one piconet, or 7 end devices, at one time. We had not yet decided on whether we wanted to read from all of the rain gauges at one time or if we were going to read the data one at a time. If we chose to read the water measurement one at a time, this would not be an issue. However, if we, along with the Stormwater lab, decided that we would like to read the data from all of the rain gauges at the same time, this could have become an issue.

2.9.3 Zigbee Wireless Technology [B4], [B9]

The third wireless option researched was Zigbee. As opposed to Wi-Fi, Zigbee is a very low powered wireless choice. This is a device that was designed with low power consumption as a goal. The low power consumption will make it easier for us to integrate it with the MSP430 family of micro controllers, which were designed with the same initiative. They will be easier to integrate since they both require a similar amount of power to operate thus we will not have to create any additional outputs from our power supply to feed the Zigbee transceiver's power needs. Additionally, the Zigbee transceivers that will be placed at each rain gauge will be able to run off of a battery. This will make it much easier since we will not have to route wires to each rain gauge. Depending on the type of battery used, they will be able to power our rain gauge signal transceivers from months to even years. For example, a single sensor can run up to 5 years with a single AAA battery. In order to achieve this low-power consumption, the network will have to be configured to run in beacon mode. While in this mode, the devices will only consume power when they are being used. In beacon mode, the coordinator will send out a signal to the end devices on the rain gauges. The end devices will then decide if any action is needed on their part. If no action is

needed, the devices go back to sleep until the coordinator sends out another beacon. The opposite of this mode is non-beacon mode which allows all end devices to communicate to the coordinator at any time. This mode is susceptible to interference due to all of the devices trying to talk at the same time. It also consumes more power since the coordinator will have to awake and ready to receive a signal from an end device at any time.

Zigbee also operates at a frequency of 900MHz to 928MHz in North America or at 2.4GHz permitted worldwide. Texas Instruments has provided us with free samples of the CF2520 Zigbee transceiver, which operate at the 2.4 GHz band so we will more than likely be using this transceiver model. Again, the issue of network noise arises. Since there will be many Wi-Fi networks around the rain simulator and both signals travel at the same frequency it is very likely that we will encounter some noise. However, there are 16 different 5MHz channels within the 2.4GHz band that Zigbee can access. Many of these 16 different channels do not overlap with the Wi-Fi frequency thus eliminating our concern for noise infiltrating our network. There are also three different rates that Zigbee can operate at: 20kbit/sec, 40kbit/sec, and 250kbit/sec. The faster the speed of the Zigbee transceivers, the more expensive they are as well. Having the option between 3 speeds and prices will give us the opportunity to find a well-suited balance between speed and price. Surprisingly, Zigbee has a pretty wide range of reach spanning anywhere from 10 to 100 meters away from the transceiver. It was surprising to know that its range extended up to 100 meters simply because it consumed such little power. At the rain simulation lab, the rain bed sits directly outside of the building. The range for these rain gauges will never lie outside of 100m so the Zigbee's range of reach is more than what is needed. However, the idea that one day in the future, the rain bed may be moved to a new location where the range would be larger than 100 meters was considered. Thankfully, there are a wide variety of Zigbee routers all of which are very budget friendly. Zigbee also makes the existence of network sizes up to several thousand devices possible and practical. We are not going to have several thousand ports for the rain simulator but if more gauges need to be installed in the future or last minute, this will make it very easy. The simple fact that Zigbee can handle receiving and transmitting multiple signals from and to many devices is what really caught our attention. In the case of the rain simulator, we would have to set up what is referred to as the "Star Topology Network". This network essentially has multiple nodes corresponding to multiple devices which all report to one central PAN (Personal Area Network) coordinator. The PAN coordinator is the Zigbee device that is responsible for creating and managing the Zigbee network. The PAN coordinator would be what gives our network its unique 16-bit PAN ID when it's initially set-up. The multiple devices reporting to the PAN coordinator would use the unique PAN ID to decide whether or not to join the network and transmit their data. Another common topology is the mesh network. The main difference between the mesh and the star is the mesh network's redundancy and ability to self-heal. The mesh network has multiple paths to get to the same destination. In a mesh network, the signal is still able to get to its

destination even if one Zigbee device goes down because it can easily find another path to take.

In our case, we do not need the intricacy of a mesh network because we only need to read signals from a limited number of devices. The mesh network is also more tedious to set up and it did offer us any benefits that are necessary to our application. The connection time for devices to join a Zigbee network takes around 30 milliseconds so the data would be able to be retrieved almost instantaneously. As mentioned before, a high data transfer rate is not critical for our application so in this case it is an additional benefit. In the end, Zigbee's low-power, low transfer rate, and intended use with small packet devices seemed to be a great choice according to the need of our project.

2.9.4: Siteplayer & Wi-Fi Joint Integration

Another consideration for the project was to relay the data logged from the rain gauges to the Internet. We considered integrating a Siteplayer module, the world's smallest Ethernet web server. Siteplayer would enable us to take the data retrieved from the rain gauges and publish them to a designated website by storing the information on the small server on the Siteplayer chip. Siteplayer could also be linked in the opposite direction so that the user would be able to every time the Siteplayer device received data, it is stored in its RAM until a webmaster can go online and manually publish the information to the web. The issue that arises here is that if the data log for a specific simulation is not published to the web before the device is powered, the information will be lost. Additionally, the Siteplayer development kit does not come cheap at $99 just for the development kit. We also had to consider that this would need to be hard wired to an Ethernet port or wirelessly connected through a Wi-Fi network. If we did not have access to a direct Ethernet port, then we would have to set up a new Wi-Fi network or use a nearby Wi-Fi network to transport the information to publish to the web. At this point, it begins to get complicated and more expensive. Because this is only an added feature to the control system, our team along with the Stormwater management team would have to discuss and decide whether the benefits of web access for the data will justify the extra cost. We eventually decided that this option was not necessary.

2.9.5: Rain Gauge Data Logging

Another issue that needed to be addressed was how we were planning on communicating the data that was collected via the rain gauges to the user. Once the water level was read from the rain gauges we could either send the data to the user interface screen and give the user a live report on the water levels or we could log the data in a separate memory device for the user to access and evaluate later.

If the data were to be sent to the user in real time as the simulation is taking place, we would have to send the data from each individual rain gauge or take and average reading of the total number of rain gauges and relay that to the user. Since the data would be presented to the user in real time continuously at a specific sampling rate, we would have to take into consideration the role the microcontroller would play in this relay of information. Ideally, the microcontroller will be focusing on the stepper motor control, which is first priority. If we then have to use the microcontroller to also transfer the data from all of the rain gauges at the sampling rate we specify to the LCD screen then we would need to make sure that this would result in minimal work being done by the microcontroller so as to not compromise our first priority of controlling the stepper motors. We would also need to consider whether we want to display all of the rain gauge readings on the LCD screen or if we want to take and average of all of them. Ideally, it would be best if we could display both measurements: the readings from each individual rain gauge as well as the average between them all. Displaying the individual readings from each rain gauge would greatly help in confirming that the entire rain bed receives an even amount of rainfall. This would also help in judging whether external factors, such as wind, are messing up the rainfall intensity readings and consequently the rainfall intensity as perceived by the rain bed. Displaying the average rainfall intensity collected by all of the rain gauges would be a nice quick reading for the researches that can in turn be easily communicated to their customers. The main tradeoff we ran into here is that if we had too many rain gauges, there would not be enough room on the LCD screen to display them all but simultaneously more rain gauges would result in better surveillance of the live simulation. Because the number of rain gauges will not be limited by the Zigbee wireless technology we have chosen to use, the limiting factor would then turn out to be the funding that the Stormwater lab ultimately wants to contribute to the team. Depending on the number of rain gauges, we would either be able to show all of them or at least a good sample of them. Since we could not display the readings for all of the individual rain gauges then we could only show as many as we could fit on the screen while still ensuring that they are evenly distributed from one another on the rain bed.

The other option would be to completely avoid using the microcontroller and instead log the data retrieved through the rain gauges to an external memory device. Because we wanted this to be user friendly, the ideal memory device would be a flash drive or an SD card. This way, as the simulation runs, the Zigbee receiver would only need to receive a signal from the microcontroller specifying the sampling rate at which to retrieve the water level readings from all of the rain gauges. The Zigbee receiver would then take the data and instead of having to go through the microcontroller to be displayed on the LCD user interface screen, it would write the data to the memory device. For clarification, we would include a small LED that would remain lit while data was being written to the flash drive or SD card to alert the user not to remove the memory device while it is being written to and avoid potential write errors. The user could then be able to take this memory device and read it from any computer with USB or

SD card reading capabilities. We wanted the data to be written to the memory device in a rather simple format such as a .txt file so that the user could open and analyze the data from virtually any computer without the need for specific software. This method also makes the data more portable for the end user so that they could take the memory device and go wherever they need to be. Lastly, having this external memory device would make it easy to fix if there were ever to be any issues with the memory. Eventually, we scrapped this idea.

2.9.6: Rain Gauge Design

For the actual rain gauges, we had the option of either designing and building our own rain gauges or buying wireless rain gauges and modifying them to transfer the information wirelessly via Zigbee technology. After exploring the different designs for rain gauges, it seemed that the best design was one that was self-emptying once it was full. The problem with this kind of rain gauge though was that it requires a level surface to accurately function and empty itself when it needs to. Due to the fact that the rain gauges need to be attached to the rain bed and the rain bed as often elevated for simulations, this kind of rain gauge would not function properly. An alternative sort of rain gauge that is very commonly used is a rain gauge that greatly resembles a glass tube with markings to display the measurement of rainfall. The one issue that we had to resolve was how to compensate for the higher reading of rainfall due to the slanted rain bed. The full development of the gauge took place in our final semester and can be seen in our conference paper.

2.10: Programming Interface

2.10.1: JTAG

Joint Test Action Group or JTAG was originally formed to develop a method to test circuit boards after manufacturing for faulty solder joints and other imperfections that may have been acquired during manufacturing. Eventually JTAG became the IEEE 1149.1 standard and since then quickly grew in processor development and programming as well as allowing external inspection of the inner details of digital processors. Texas Instruments was the inventor of the JTAG scan emulation feature. Before JTAG scan emulation was developed, the preferred method of testing and emulation was called “in-circuit emulation.” In-circuit emulation required that the target processor or device be replaced with another device that acted or emulated the original target device but had extra pins available that made the internal structure of the device visible externally. The biggest downside to in-circuit emulation is cost; as the speed of the device increased, so did the cost of in-circuit emulation. The JTAG standard avoids this added cost because JTAG communicates directly with the target device instead of utilizing a replacement emulation device. As well as avoiding the use of an

extra emulation device, JTAG allows data to be read and written to the device without interrupting execution.

Texas Instruments has added some capabilities to their JTAG devices since then, including the ability to access registers and cache. As stated before, JTAG was originally used for testing solder points and PCB traces but has grown and evolved into a standard for testing and debugging embedded systems. JTAG can access the processor from the very first instruction after processor reset which allows a developer or software to access the processor memory and software before execution of code. The JTAG interface has also allowed developers to step instruction by instruction through the code for debugging in real time. Another capability of JTAG is used for the programming of software into flash memory. By using these two fundamental functions of the JTAG interface, a developer can write software to memory, and actively debug the code in real time. Being able to debug and reprogram the hardware in real time can greatly improve the development time. The JTAG interface is a four or five pin interface with parallel device capability. This allows for multiple devices to be interfaced with only one JTAG port on the circuit board. Figure 2.10.1a below shows a series of devices connected to one JTAG port using the parallel device capability. The five JTAG pins are: Test Data Interface (TDI), Test Data Clock (TDC), Test Clock (TCK), Test Mode Select (TMS), and Test Reset. The Test Reset pin can be left off in some instances but it is normally used. The data is sent serially and one bit is sent per clock pulse at a typical frequency between 10 and 100 MHz. There is no standard speed because it is dependent on the chip being programmed or debugged. There is also no standard JTAG connection. Some board manufacturers will provide a header pin while other manufacturers will only leave test points.

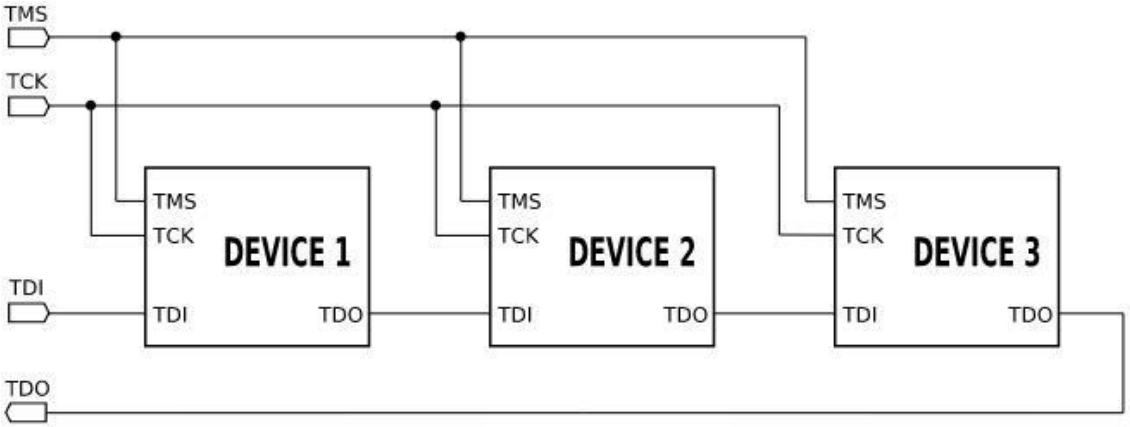

Figure 2.10.1a - Parallel Device JTAG Connection[A3]
This diagram shows a properly connected three-device network to one JTAG port for diagnostics and programming. This particular schematic is using the four-wire JTAG interface.
(Permission for this image is granted under the GNU Free Documentation License.)

Another critical aspect of JTAG is that while the pin out is standard, the clock frequency and instruction set is not necessarily the same between all parts or manufacturers. But, in order to be compliant with the IEEE standard, the manufacturer must release the Boundary Scan Description Language (BSDL) file. In the file must be all the information needed that describes how to access a device using the JTAG interface. Some of this information includes a description of the device, its functionality, and a generic parameter such as a package type. The file must also include a port description that describes the pins on the device and a map of the physical pins to the logical pins. The BSDL file must also include the information for the boundary scan cells and the description for register access.

2.10.2: Spy-Bi-Wire

Spy-Bi-Wire is a technology invented by Texas Instruments that is based off of the JTAG protocol. It was designed for Texas Instrument devices that have a limited number of pins. The MSP430 line of microcontrollers has some chips that are limited in pin count and the standard JTAG interface would use up too many pins on the device. The Spy-Bi-Wire technology only uses two pins instead of the standard four or five pins of JTAG. One of the pins is the bidirectional data pin, which replaces the two data pins of the standard JTAG. The other pin for Spy-Bi-Wire is the clock pin, which is used to replace the Test Data Interface, Test Clock, and Test Mode Select pins. While the Spy-Bi-Wire technology reduces the number of pins necessary to perform programming and debugging it does have some downsides. Because the data pin is now bi-directional, it is impossible to send and receive data from the device simultaneously. This can become problematic when debugging the device. It is also much slower than JTAG during programming. In the JTAG protocol, there are three separate clock pins while in Spy-Bi-Wire there is only one. This reduction in clock pins reduces the maximum programming speed of the device.

In our project, there is no specific requirement for speed in programming or debugging. Therefore, either interface would be appropriate for our use. Initially, because of the development board (MSP-EXP430G2) that was used, the Spy-Bi-Wire programming interface was used because it was the only interface supported by the board. As our project evolved, and with the help of Texas Instruments free sample program, the interface was changed to JTAG. This decision was made because of the ability to program the chip faster and to become more comfortable with the JTAG interface. Because of the proprietary nature of Spy-Bi-Wire, it was decided that it would be in the best interest of the group members to use the industry standard of JTAG over the Spy-Bi-Wire interface.

2.10.3: Bootstrap Loader

The bootstrap loader in Texas Instrument devices is another propriety method of programming. In the MSP430 line of microcontrollers from Texas Instruments, the bootstrap loader allows for direct programming of the flash memory of the device. It is activated sending commands via serial UART to the device. The bootstrap loader (BSL) enables the control of the device and allows the user to store data to the flash memory. In order to use the BSL, a bootstrap loader entry sequence must be applied to dedicated pins, followed by a synchronization character, then the data of the command. As well as following this specific sequence to activate the BSL, the user must send a password along with the command. The proper sequence to start the BSL is dependent on which method of communicating with the device, UART, USB, or JTAG. In the MSP430 line of devices, Texas Instruments has built in a JTAG fuse, which if blown, disables the JTAG ability of the device. This prevents anyone from reading or changing the flash memory of the device. However, the original manufacturer or designer may have a need to access the memory of the device after the final product has been delivered to a customer. In this case, the manufacturer will have to use the BSL to access the device. The ability to password protect the BSL allows the manufacturer protect their design and software while still being able to perform maintenance and diagnostics after the delivery of the product. If one incorrect password entry is made for the bootstrap loader, the entire flash memory of the device is erased. This feature prevents most attempts of circumventing the protection.

The BSL loader has a minimal number of “Core Commands”

- RX Data Block – This command can write data to an address
- RX Data Block Fast – Identical to RX Data Block except it does not verify that the data was correctly written
- TX BSL Version – The BSL will transmit its version
- TX Buffer Size – The BSL will transmit the bytes available in the data buffer
- RX Password – Unlocks the BSL if the correct password was sent. If an incorrect password was sent, it will erase the flash memory.
- Erase Segment – Erases an address
- Unlock/Lock Info – Toggles the INFO_A lock
- Erase Block – Erases a block of flash memory at an address
- Mass Erase – Erases all the flash memory
- CRC Check – Performs a CRC check using the CCITT standard
- Load PC – Makes the BSL begin execution at a given address

There are a few more commands but they are not critical to the operation of the BSL. As was stated previously, the primary purpose of Texas Instruments bootstrap loader is to allow a manufacturer or developer to still have access to the flash memory of the MSP430 device after the JTAG fuse has been blown.

This creates a secure device to protect potentially proprietary software code. As the software in our project did not need to be secured from others, the JTAG fuse was left intact and the BSL was not needed for accessing the flash memory of the device.

2.11: Protection from the Elements

Given the fact that we designed the control system for a rain simulator, there is bound to be lots of water in many different forms surrounding the machine and valuable electronics at all times. We needed to be sure to protect any and all sensitive devices from various types of negative environmental impact. Certain items, like the wireless rain gauges, would need to be protected from a direct stream of water whereas other components that do not assume their permanent position on the rain bed, like the microcontroller, only needed to be shielded from light rain drizzle from the rain simulator that can be blown by the wind and humidity. The team needed to take the appropriate precautions to ensure that the high-heat temperature of Florida does not work against us.

2.11.1: Enclosure

For the United States, the National Electrical Manufacturers Association or NEMA is the association that publishes the standards for electrical enclosures. NEMA ratings cover every aspect of protection that could be required for enclosures. A few of the different NEMA ratings include NEMA Type 1; a general purpose enclosure which provides some protection against dust and light, indirect splashing but is mainly designed to protect against contact with live parts. NEMA Type 2 is a drip tight enclosure but does not offer any additional protection over type 1. NEMA Type 3 enclosures are weather resistant and offer protection against rain, sleet, etc. Type 4 enclosures are watertight enclosures. The remaining NEMA ratings are used for hazardous installations and corrosion protection. Because of the high amounts of moisture and potential for direct water spray onto our circuit board, we have decided to use either a NEMA Type 4 or 4X enclosure, if possible, because of the protection offered from hose directed water. The NEMA 4X enclosures offer the added protection of corrosion protection and while not necessarily a requirement, it would be an added benefit. One of the biggest issues with using a NEMA rated enclosure is that rated enclosures are very expensive. In the process of researching difference waterproof cases, there were not any enclosures that seemed to meet the requirements of waterproof and low cost. However, we found a business that manufactures custom stainless steel enclosures for refrigeration equipment. They have agreed to donate the materials and labor necessary to build any enclosure necessary for the project. We planned on using a simple steel box, with a vent system on the bottom of the box. The vent needed to have an extended tunnel to prevent water from being able to come in contact with the vent fan blades. Because of the high amount of heat that we expected the motor

drivers to produce, the enclosure needed ventilation to remove the heat and could not be sealed watertight. We hoped that the combination of the tunnel and conformal coating would provide adequate protection from moisture for the circuit board and components.

In the end Sunshine Metals graciously donated time and materials to create the enclosure we needed. The stainless steel enclosure was welded together and contains different features that help shield the sensitive electrical components from the elements. The enclosure has 2 circular openings on the back panel where two fans are attached. One of the fans is located higher than the other and directs airflow outward and the lower fan directs airflow inward. In order to protect the user from the high voltages, the interior has been divided into two levels. To divide the two levels, we used polycarbonate as an intermediate shelf. The heavier components were assembled on the bottom level and the lighter ones on the top level. This was done to keep the center of gravity of the enclosure low to the ground thus greatly reducing the probability of the enclosure from falling over if it were to be bumped into. This also separates the high voltage power supply from the user preventing inadvertent contact. The polycarbonate allowed us to have a sturdy shelf suitable for holding the components. It was also transparent and made it easier for light to reach the electrical components towards the front of the enclosure. This will come in handy if any maintenance needs to be done to the system. Additionally, the polycarbonate was also easy to drill into. This made the securing of many components to it quite simple.

2.11.2: Conformal Coating

Because of the environment that our design is subjected to, it is important for the electronics to be protected against heat and humidity. The methods for controlling heat are discussing in the cooling section of the research. The humidity in the environment can be very high, perhaps 100 percent. The enclosure for the electronics has been designed to hinder any liquid water from entering the device, but water vapor in the air will still be present. The typical method of protecting printed circuit boards from damage due to moisture is to apply a layer of conformal coating. The conformal coating offers protection from dust, moisture, condensation, and chemicals. This is a critical aspect of the design of our circuit boards due to the extremely high levels of moisture likely to be present in the air where our device operates. There are a few different methods of applying the conformal coating. The easiest method for small production runs is brush application. As the name suggests, the coating is just brushed onto the finished PCB but it is subject to many defects and is highly subjective method of application with the need for a skilled and experienced technician. The skills of the person applying the brushed conformal coating play a large role in how well the board and components are protected. The next type of application method is dipping application. This method is more useful in high production runs due to the high repeatability of the application. When dipping the board in the conformal coating, the coating will even penetrate under devices.

However, this does cause some issues due to the complete coverage of the board if the PCB masking is not perfect. A third method of application is by using a sprayed on application and this method typically results in an excellent coverage without the requirement of a perfect mask that the dipping method requires. There are also a few different application techniques that are accomplished using robots and of course, these methods are not available to us without significant costs.

There are multiple chemicals that can and are used in conformal coatings. Silicone is one of the more common chemicals used in conformal coatings and offers an all-around protection. It has good resistance to high temperatures, moisture, shock, and thermal shock. The flexibility of the silicone coating gives it the ability to withstand the shock and thermal shock. Another common chemical used is an epoxy coating which allows for a very hard coating. Epoxy's benefits include ease of application, excellent chemical resistance, and excellent abrasion resistance. The downside to the hardness of epoxy is that it is nearly impossible to remove from the board once applied and it reduces the ability of the components to resist physical shock. Acrylic conformal coatings are common in some situations because of the ease of application. Acrylic coatings are easily applied, removed, repaired and are low cost. Acrylic has good moisture resistance and dries quickly but does not provide the same protection to abrasion as epoxy. Urethane coatings also have good moisture resistance but during application it tends to stress components while curing and is nearly impossible to remove once cured. This inability to be removed along with the stress placed on components during application ultimately excludes urethane as a choice in conformal coating. The final common conformal coating is Paraxylene. Paraxylene provides excellent coverage but is very expensive and quite difficult to apply. Between all of these different types of conformal coatings, silicone appeared to be the best choice for our uses because of the ease of application and great moisture protection. During testing we discovered that the need for conformal coating was minimized. This was due to the length of the cables allowing us to location the enclosure far enough away from the simulator that it should not receive any direct rain fall.

2.11.3: Heat Dissipation

The decision to use a device which may require a heat-sink comes with its own complications. The heat-sink requires a thermal interface compound which helps to conduct heat between the IC and the heat-sink. Arctic Silver 5 is a commonly used high performance thermal interface material and one group member already has some of this material on hand. This product has a thermal resistance of less than .0045 $^{o}C\text{-}in^2/W$. Other than the extremely low thermal resistance, there is nothing that sets Arctic Silver apart from other thermal interface materials. There are a few different types of heat-sinks, as well as two main materials that heat-sinks can be made of. The most common material to be used for heat-sink construction is aluminum alloy. Aluminum has excellent thermal

conductivity, is cheap, and can be extruded. Extruded heat-sinks are very easy to manufacture and this greatly reduces cost. The second common material for heat-sinks to be manufactured from is copper. Copper has around twice the thermal conductivity of aluminum alloy but is around three times heavier and around 5 times more expensive than aluminum, depending on the current market price. The other downside of copper is that it cannot be extruded but must be machined. This adds a lot of time to the manufacturing process and along with the increased cost of the raw copper, these heat-sinks tend to be considerably more expensive than aluminum alloy heat-sinks. The third material that heat sinks are made out of is diamond. Diamond heat-sinks do not dissipate heat in the same methods as aluminum or copper heat-sinks but they are much more effective at removing heat. Of course, diamond is much more expensive than aluminum or copper but it is very useful in some products. Aluminum heat-sinks are our preferred choice because of the low cost while still maintaining high thermal conductivity.

Stamping, casting, extrusion, bonded, and folded fin are the main types of heat-sinks. Stamped heat-sinks are typically low cost solutions for low heat situations. Cast heat-sinks are typically higher density than the stamped heat-sinks and allow for unique shapes to be manufactured that cannot be made using other techniques but the cast heat-sinks have slightly lower thermal conductivity than extruded heat-sinks and cost more to manufacture. This method of construction is used for applications where a difficult or unique shape of heat-sink is needed. Extruded heat-sinks are very easy to manufacture, have high thermal conductivity, and allow for very complex two dimensional shapes that can dissipate large heat loads. The only issue with extruded heat-sinks is that they are restricted to the manufacturing limitations of extruded aluminum. Bonded heat-sinks can increase the heat dissipating capacity of extruded heat-sinks due to the possibility of greater fin height to gap ratios. Folded fin heat-sinks are made by folding sheet metal into fins and then attaching those fins to a base plate via epoxy or brazing. This type of heat-sink is typically used when extruded or bonded fin heat-sinks cannot be used.

There are many different fin geometries and designs or shapes of heat-sinks and each shape has its benefits. Rectangular fins have better performance than square fins and round fins but they do have higher pressure drop. Rounds fins are used when the direction of airflow is unknown. When using an aluminum heat-sink the finish on the heat-sink can affect the radiated heat performance. The typical finish that is applied for increased performance is flat black anodized as a flat black finish increases the radiated heat. Of course, anodizing the heat-sinks adds cost to the manufacturing of the heat-sink. For the purposes of our project, the additional cost of anodizing the heat-sink is not necessary.

Thermoelectric cooling uses the Peltier effect to transfer heat against the temperature gradient. This type of cooling is extremely inefficient and can be quite costly. The main competitor to thermoelectric coolers is vapor-compression

refrigeration. The only real advantage that thermoelectric cooling has over vapor-compression refrigeration is that thermoelectric coolers have no moving parts and are therefore more reliable. The other benefit is that thermoelectric coolers are smaller and flexible. With a thermoelectric cooler, one side gets cold and one side gets hot; the hotter the hot side gets, the colder the cold side gets and therefore to use one effectively, it still requires a heat-sink. Peltiers are quite expensive and will not be used in this project; the benefits of thermoelectric coolers will not be realized if they were to be used in our project.

Another common type of cooling method is liquid cooling. This type of cooling uses a liquid, most often distilled water, in a typically closed loop system to remove heat from components. This type of cooling is most often used in high performance computer systems where a simple heat-sink and fan combination does not provide adequate cooling. The ability of water to quickly carry the heat away from the processor is what makes liquid cooling very useful in high heat applications. Carrying the heat away from the processor can allow the use of a large radiator and therefore increases the cooling capacity of the system. Of course, it is almost always a bad idea to combine water and electronics so the system must be completely leak free. If the device is going to be portable or handled roughly, water-cooling is not a good choice for cooling. Water-cooling is not a viable method of cooling in our project. The DRV8432 chips do not produce enough heat to necessitate water-cooling.

3.0: Small-Scale Prototype

When deciding on a design approach for this project, many factors had to be considered. The rainfall simulator is currently being used by the Storm Water Academy five days a week during regular business hours. The only time the machine would be available to us is during nights and weekends. This was a problem because not all of the group members live near UCF where the Storm Water Academy is located. The time was not the only issue. The strategic components of this machine are quite costly. The two stepper motor drivers cost $1200.00 each and the two stepper motors cost $400.00 each. Along with the other components of this system, replacement costs for these items far exceeded the funding that the group was going to receive. It was decided that experimenting on this existing equipment was too risky. Since none of the group members had any experience in controlling stepper motors, the likelihood of a catastrophic failure was high. A low cost small-scale model of the system was needed for us to use as a learning/experimenting tool.

The components needed to build a small-scale model consisted of only the components that are directly related to our final project design. The actual rain simulator consists of a large water pump, a holding tank, two large gantry cranes, two stepper motors, two stepper motor drivers, test beds, and many mechanical components that tie the whole system together. However, since our control system will only be controlling the two stepper motors it is not necessary to

include all of those components in our small-scale implementation. The components that were chosen to be used in the small-scale implementation are: two small stepper motors with similar characteristics to the ones used in the actual rainfall simulator, two stepper motor drivers, a user interface that includes a keypad for input and a small LCD screen to display the input settings, a multi-output voltage power supply, electrical hardware and connectors that will be of similar quality to the final design hardware, at least one microcontroller that will interface all of the system components together, and a project box to contain all of the components.

It was decided that production of the small-scale model should be the highest priority during the first half of the Senior Design 1 semester. Obviously some research was needed to gain a fundamental understanding of the operation of a stepper motor, but by building a crude working model first it would become very evident which components would need the most research for the final design. The components used in the small-scale model needed to have similar characteristics to the components used in the final design in order for us to gain the maximum amount of knowledge from building it. The model needed to be kept low cost and the procurement of the parts needed to be easy and fast. This meant that most of the parts would need to be purchased locally to keep online ordering times down and the circuit boards would need to be made entirely using perforated boards to save on cost and development time.

The software developed for the small-scale model needed to be effective, but fully optimizing it was not a priority. Since the goal of the model was to learn as much as possible, the code would not need to be rigorously tested for errors yet. However, some error checking algorithms would need to be included as part of the learning process. To maintain the goal of being low cost the Integrated Development Environment (IDE) for software building and testing needed to be inexpensive or free. Also the microcontroller itself needed be inexpensive. With all these things in mind a basic idea of the small-scale model was formulated. The sections that follow describe all of the components that were used in the model.

3.1: Small-Scale Model Microcontroller

All four of our group members have very limited experience with embedded systems programming and it quickly became evident that a large portion of our project would be embedded systems software development. With only the limited knowledge of the Motorola MC68HC11, we had to decide on a suitable microcontroller. The MC68HC11 was ruled out due to the fact that it is obsolete and no longer being manufactured. Due almost entirely to the fantastic ability of the TI-89 calculator we turned to Texas Instruments to see what they had to offer. They are one of the largest manufacturers of integrated circuit chips, and any company that can build such a great calculator certainly must have other great products. TI recently released a new microcontroller development platform

called the “LaunchPad.” This development board supports all of their “Value Line Series” of microcontrollers. They are called Value Line because they are very inexpensive. For only $4.30 a complete development kit including two microcontrollers can be purchased, and as long as TI has them in stock they will overnight ship them for no additional cost. This fit our requirements of low cost and fast procurement. TI offers an easy to use Integrated Development Environment called Code Composer Studio that can be used to program any of their microcontrollers in the high level languages of C or C++. Since all of the group members have experience programming in C, this was a major plus. Another benefit of this development kit is the extensive amount of resources that are available for it. Many examples of open source C code are available for download. These code examples cover nearly all of the basic features of every microcontroller in the Value Line series. With the vast amount of available help resources, the extremely low cost, and the fast procurement time, we figured that we could not go wrong, so a LaunchPad Development kit was ordered. Figure 3.1a shows the Launchpad.

The LaunchPad development board supports all of TI’s MSP430 14 or 20 pin PDIP (Plastic Dual Inline Package) microcontrollers. The MSP430 microcontrollers feature a 16-bit architecture and do not allow for external addressing. This means that all of the memory space is contained on chip, and this puts a hard limit on the available space that can be used to store the program. All of these features had to be considered when deciding if any of the Value Line microcontrollers would be suitable for use in the small-scale model.

Figure 3.1a - LaunchPad Development Board

A microcontroller can easily be placed in the socket at the center of the board. A USB cable interfaces this board to a computer. TI’s Integrated Development Environment makes C level code generation simple, and it is very easy to download the code onto the microcontroller. The microcontroller can then be removed from the development board and placed in the actual circuit.

Texas Instruments offers a wide range of microcontrollers for an equally wide range of applications. The MSP430 microcontrollers are marketed for use in measurement, metering, sensing, consumer electronics, and general purpose. A step up from the MSP430 is the Stellaris® ARM® Cortex™-M3-based microcontrollers. These microcontrollers are marketed for use in connectivity, security, motor control, human machine interface (HMI), and industrial automation. The fact that they are intended for motor control, industrial automation, and human machine interface was very appealing since these are three of the main features of our control system for the rain simulator machine. However, the cost of a development board and computer interface for these microcontrollers was significantly higher than the $4.30 LaunchPad development board. The Stellaris® ARM® Cortex™-M3-based microcontrollers feature a 32-bit architecture and have significantly more internal memory. These features seemed like overkill for our relatively simple small-scale model, and the additional cost of a development kit made us stick to the original plan of using the LaunchPad development kit with a Value Line microcontroller.

The first microcontroller that was experimented with was the MSP430G2231. This is one of the microcontrollers that is included with the LaunchPad development kit.

The main features of the G2231 are [L1]:

- Low Supply-Voltage Range: 1.8 V to 3.6 V
- Ultra-Low Power Consumption

- Basic Clock Module Configurations
  - Internal Frequencies up to 16 MHz With One Calibrated Frequency
  - Internal Very Low Power Low-Frequency (LF) Oscillator
  - 32-kHz External Crystal
  - External Digital Clock Source
- 16-Bit Timer_A With Two Capture/Compare Registers
- Pulse Width Modulation capability with Timer_A
- Universal Serial Interface (USI) Supporting SPI and I2C
- 10-Bit 200-ksps A/D Converter With Internal Reference, Sample-and-Hold, and Autoscan
- Spy-Bi-Wire Interface
- Available 14-Pin Plastic Dual Inline Package (PDIP)
- 2 kB of internal Flash memory for program storage
- Built in Temperature Sensor

It did not take long before our software exceeded 2 kB, and it became clear that the G2231 would not work for the small-scale model. Also it only has 10 general purpose I/O pins which we used up in a hurry and were in need of more. It did, however, serve as an excellent learning tool. Many of the features previously listed could be utilized, and by following the available tutorials for this

microcontroller we learned how to implement them. This brought about ideas such as using the internal temperature sensor to monitor the temperature of all the components enclosed in the project box. This could be used as an over temperature warning system. The 32 kHz crystal could be used to source the internal Timer_A so that very accurate timing could be achieved over a broad range of operating conditions. The pulse width modulation capabilities might be useful if the need arose to micro-step the stepper motors. Even though the G2231 had reached the end of its useful life for this project, many good things came out of it.

The search was on for another of TI's microcontrollers, but one with more programming space and more I/O pins. After discovering that Texas Instruments offers free samples of many of their microcontrollers, we ordered a few different ones. We decided to try out the best Value Line microcontroller they had to offer. We figured that if the best one they had was not capable enough for our project then we could abandon the Value Line Series. We still liked the idea of the Value Line microcontrollers because of their ease of use, low cost and the fact that we already had a development board for them. The best one they offer is the MSP430G2452. It has all of the features previously listed for the G2231 and several more including [L2]:

- 16 General Purpose I/O pins
- 8 kB of memory
- Analog Comparator
- Capacitive Touch Capable I/O ports

We determined that this microcontroller would definitely work for our small-scale model. We were dismayed to find out after receiving it that the development board was going to need a firmware update before it would be able to program the G2452. Since this microcontroller had just been released, Texas Instruments did not even know about this problem yet (or at least they claimed not to know about it).

This problem demonstrated the benefits of the design approach we took for the small-scale model. Since everything that can go wrong will go wrong, it is better to experiment with the design components early to gain valuable hands on experience and learn about the problems before it becomes too late.

Since it was unknown when the firmware update would be released we decided to experiment with another of the free microcontrollers we ordered from TI's sample program. The one we chose was the MSP430F5435. While still in the MSP family, this one was not a member of the Value Line Series. Although much more capable, this microcontroller still presented several problems. It was not a DIP package microcontroller, so it could not be placed in the LaunchPad development board for programming. Since the spacing between the pins is less than half a millimeter, we would need a breakout board for it and a way to solder

it on. We ordered a cheap breakout board and when it arrived we soldered it on in the Senior Design lab using the very capable MetCal soldering station.

Many very useful features of the MSP430F5435 were applicable to our project. Some of the most useful ones include [L3]:

- 67 General Purpose I/O pins
- 192 kB of internal Flash memory for program storage
- Optimized for end equipment that uses Zigbee/RF communication
- Real time clock
- Low Frequency internal reference oscillator trimmed to 32768 Hz
- Enhanced UART support
- Ability to store data to memory without using the processor

Texas Instruments states in the data sheet for the MSP430F5435 that one of the typical applications of this microcontroller is digital motor control. This seemed like an ideal choice for our small-scale model and possibly even for the final product. The 67 I/O ports would offer great flexibility in the number of external peripherals that could be connected. It was plenty for what we had planned and if we decided to add anything extra, it would not be a problem. The 192 kB of internal flash memory was nearly one hundred times as much as the microcontroller we originally tried. This would allow us to quickly generate a working prototype without having to spend excessive time trying to optimize the code for space. The fact that it was optimized for end equipment that utilized Zigbee/RF communication was seen as a plus although we have not determined the exact type of wireless interface we will use for the wireless rain gauges we plan to have in our final design. The Real Time Clock might be very useful because one of the parameters our control system must accurately control is timing. The low frequency internal reference oscillator is a useful feature because it can be used to stabilize the frequency of the internal clocks without needing an external crystal. The Enhanced UART (Universal Asynchronous Receiver Transmitter) serial communication system will simplify the communication between the microcontroller and the LCD screen. If more than one microcontroller is used the Enhanced UART might make it easy to set up a master-slave configuration if necessary. The ability to store data to memory without using the processor is another potentially useful feature. If the processor is busy controlling the motors, it might be possible to store rainfall data to memory without bothering the processor. Figure 3.1b shows our F5435 makeshift development board and Table 3.1c shows the parts and prices used to build it.

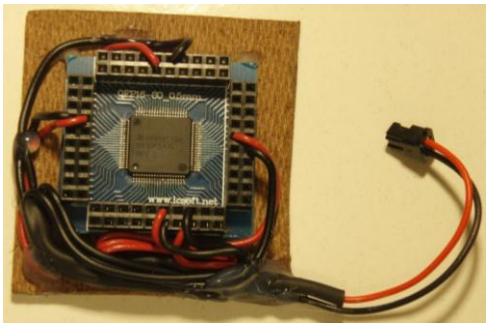

Figure 3.1b - Development Board
This is the MSP430F5435 soldered onto a breakout board. All of the pins are now easily accessible. The black and red wires are power and ground wires that are needed to power this microcontroller. They are all soldered onto a two-pin connector for ease of use.

Qty	Part Name	Part Number	Unit Price	Total Price	Supplier
1	Launchpad development board	MSP-EXP430G2	$4.30	$4.30	Texas Instruments
1	Microcontroller	MSP430F5435	Free	Free	ti.com
1	80QFP Breakout Board	SOC013	$4.50	$4.50	iteadstudio.com
4	2x10 pin female headers	-	$0.50	$2.00	Skycraft
			Total	$10.80	

Table 3.1c - Parts List for Microcontroller Development Board

One more problem still persisted. How were we going to program the MSP430F5435 if we could not put it in the LaunchPad Development board? All of the Texas Instruments MSP430 microcontrollers can be programmed using the standard JTAG four or five wire interface. However, in an effort to reduce the number of pins required for programming, TI developed a two wire interface they called Spy-Bi-Wire, and the LaunchPad development board supports this type of programming. This method results in a smaller footprint of pins required, but it is also a slower way of programming a chip. The slower programming speed would be an easy sacrifice to make if it meant we could use our existing development board. Figure 3.1d shows the connections necessary for Spy-Bi-wire programming

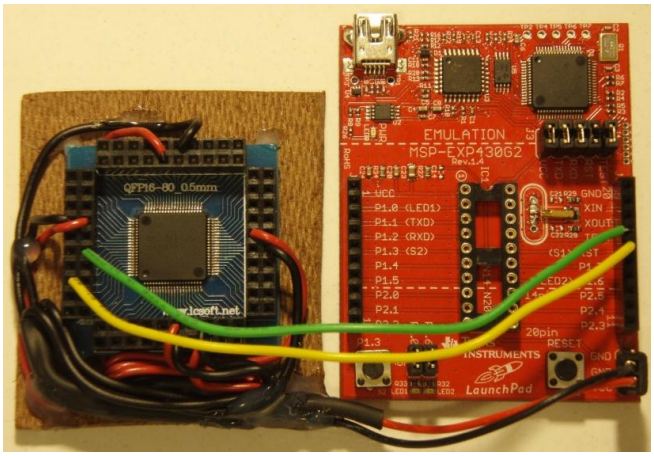

Figure 3.1d - Spy-Bi-Wire

This is the simple two-wire Spy-Bi-Wire interface between the MSP430F5435 and the LaunchPad Development board. The development board provides the power, the interface to the computer and the two ports required for programming (Test and Reset).

3.2: Stepper Motors

The rainfall simulator uses two stepper motors that our control system will need to control, so it was decided that our small-scale model should also have two stepper motors. The two stepper motors that were chosen are built by the manufacturer - Applied Motion. The specifications for the chosen motors are [L4]:

- M/N: 5023-100
- Bipolar motor windings
- 2 phase
- Phase current: 2 amps/phase
- Phase voltage: 3.2 Volts DC
- 1.8 degrees per step – 200 steps per revolution
- Available for $7.50 each

This motor was chosen for several reasons. The price was within our budget and we found a local supplier with plenty of them. The motor characteristics closely matched those of the motors that are in the actual rainfall simulator. Most importantly the step resolution and the bipolar configuration matched. The goal of building the small-scale model is to learn how to control this type of motor. The main difference between these motors and the rainfall simulator motors is that the phase current and phase voltage is lower. This motor can be controlled by either full stepping, half stepping, or micro-stepping.

The following table 3.2a and accompanying Figure 3.2b show the two motor windings and the winding energizing sequence for full stepping and half stepping the motor. These are the two methods that were being considered for driving the motors in the small-scale model.

Full Stepping CCW Shaft End				
	Motor Winding Polarity			
Step #	Red	Blue	Yellow	White
1	-	+	-	+
2	+	-	-	+
3	+	-	+	-
4	-	+	+	-
5	-	+	-	+

Half Stepping CCW Shaft End				
	Motor Winding Polarity			
Step #	Red	Blue	Yellow	White
1	-	+	-	+
2	off	off	-	+
3	+	-	-	+
4	+	-	off	off
5	+	-	+	-
6	off	off	+	-
7	-	+	+	-
8	-	+	off	off
9	-	+	-	+

Table 3.2a - Winding energizing sequence for two different step methods

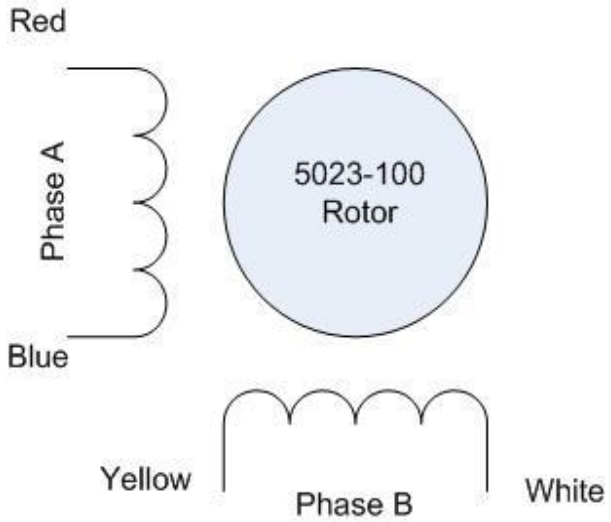

Figure 3.2b - Stepper Motor Winding Configuration

Phase A and Phase B are the two motor windings. Red, Blue, Yellow, and White are the colors of the external wires. The accompanying table shows the energizing sequence to cause the rotor to rotate in the counter-clockwise direction as viewed from the shaft end of the motor.

Table 3.2a shows that in the full stepping configuration both motor windings are fully energized at all times. This results in a higher motor torque but a lower step resolution. This configuration is also easier to implement in software. The lower step resolution (200 steps per revolution) is not considered a problem for the small-scale model since the final product will likely use the full stepping method. Micro-stepping is a variation of half stepping that allows for a nearly infinite number steps which is highly unnecessary for this model.

3.3: Stepper Motor Drivers

When we started developing a plan for the small-scale model, we were new to stepper motor control. After a few internet searches we found an abundance of stepper motor driver IC chips that were supposed to be easy to use and require few external components to control a motor. This was very appealing. The alternative would be to design our own circuit from scratch. Since we had no idea where to even start with our own design it was decided to use a motor driver IC.

The motors used for the small-scale model are bi-polar motors, so the motor winding current needs to be reversible. Once again we turned to Texas Instruments. They offer many stepper motor driver IC chips for bipolar motors, but most are for very small motors with currents of less than 750 mA per phase. The motors we are using for the model are rated at 2 amps per phase. Ideally we were looking for a DIP package driver chip so it could be easily tested on a breadboard. Due to the fact that all of the circuits for the model needed to be built using perforated boards, we did not want to have to build them more than once in the event of errors. Two driver chips seemed particularly well suited for driving our motors. The first was the Texas Instruments DRV8412. This chip is a dual full bridge PWM motor driver and has the following characteristics [L5]:

- High Efficiency – Up to 97%
- Low $R_{DS(on)}$ (Drain to Source Resistance) MOSFETs (80 mΩ at T_J = 25°C)
- Up to 50 volt motor supply voltage
- 2 x 3 amp continuous output current
- Power Pad Down (PCB is used as the heat sink)
- PWM Switching frequency up to 500 kHz
- Programmable current limit protection
- No external voltage clamping diodes needed
- 44 pin HTSSOP package
- Free Samples available from Texas Instruments

The second chip that was looked at was the UC3770BN. This is a high performance stepper motor drive chip that has the following characteristics [L6]:

- Full Step, Half Step, or Microstep
- 1 x 2 amp peak output current
- 10-50 volt motor supply voltage range
- 5 ma – 2 amp current limit control
- Thermal protection
- 16 PDIP package
- Requires external high side voltage clamp diodes
- Free Samples are available from Texas Instruments

The first motor driver, the DRV8412, had all of the features we were looking for. It had enough outputs to drive both motor windings of a single motor with one chip. The continuous output current capability of (3) amps per winding was sufficient for the small motors we will be driving in our model. The high efficiency is desirable since all of the electrical components of our system will be in a relatively small box and internal heating is a concern. However, the chip is only available in HTSSOP package and the heat sink must be placed under the chip. Actually the PCB (printed circuit board) itself becomes the heat sink, which means special care must be taken when designing a circuit for this chip to go in. We were unable to find a pre-made breakout board that would work for this application. The second motor driver, the UC3770BN, is better suited for our model. Figure 3.3a shows a block diagram of the internal workings of this chip.

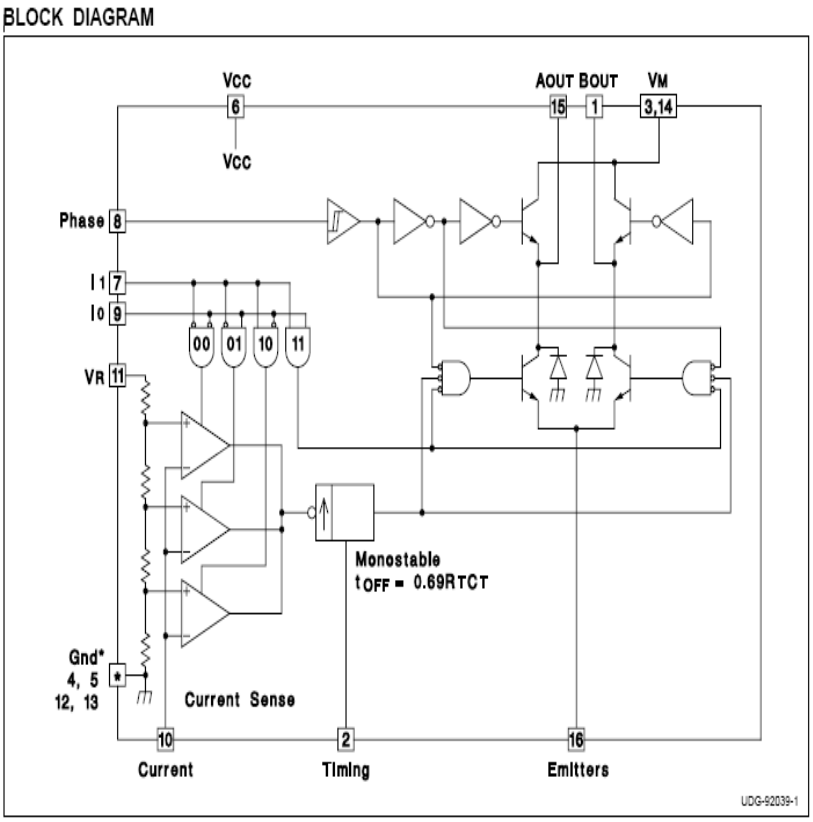

Figure 3.3a - UC3770BN Block Diagram [L6]
Block diagram of the UC3770BN motor driver chip showing the external connections. No external circuit components are shown in the data sheet for this chip. (Permission Granted by Texas Instruments)

The UC3770BN is available in a DIP package, which makes it easier to test before building it into a final circuit. It is capable of delivering up 2 amps of current and driving a motor at up to 50 volts. This is within the range of the small motors needed to control. This motor driver does have some downfalls too. The documentation available for it is sparse unlike that of the DRV8412. The needed external circuit components are not well defined, particularly the external clamping diodes. Each chip is only capable of driving one motor winding; so two chips would be needed to drive each motor. Despite the downfalls, we decided to use these to build the motor driver for the small-scale model.

Overview of the UC3770BN operation [L6]: The purpose of the UC3770BN motor driver chip is to drive one winding of a bipolar stepper motor. Pins 1 and 15 are the connection pins for the motor winding. Pins 3 and 14 are the motor supply voltage pins. As stated in the data sheet [L6] the maximum supply voltage for the motor winding is 50 volts. This is many times higher than the nameplate rating on the motor, which is why it is important to be able to limit the current that flows out of pins 1 and 15. Pin 6 is the supply voltage for the chip itself and 4, 5, 12, and 13 are the ground connections. The purpose of the rest of the pins is not

immediately obvious. The following paragraphs give a detailed explanation of the remaining pins.

Pin 11 is the voltage reference pin. Any voltage between 0 and 15 volts can be applied to this pin. As the block diagram in Figure 3.3a shows, pin 11 is connected to a voltage divider network. The voltage divider is being tapped by three analog comparators, and the output of each of the comparators is fed into a timer. The three comparators represent three current thresholds. The type of motor being controlled and the application of the motor determine what the current thresholds should be set to. The current thresholds are user defined by the reference voltage. They can be three discrete values or if pulse width modulation is used they can be a constantly changing range of values. The latter is how micro-stepping is implemented with this motor driver.

Pins 7 and 9 are the digital inputs that determine which one of the three comparators is activated. The digital inputs corresponding to each of the analog comparators are shown in the block diagram. Pin 7 and pin 9 can be connected to a microcontroller. By doing this, software will be able to select any one of the four options. Notice there are only three comparators, but there are four selectable settings. A setting of (1,1) will turn off the two lower transistors of the H-bridge. This is the only way for software to turn off the motor winding current. The data sheet for the UC3770BN states that the three current thresholds are 100%, 71%, and 50% and they are tailored for half-stepping applications. Of course these percentages are relative to the user defined voltage reference. The voltage reference needs to be set so that the 100% current threshold does not exceed the nameplate rating on the motor or the maximum value of two amps that the chip can handle.

Pin 10 is the current sensing pin. It requires some external circuitry that will allow it to convert the motor winding current into a voltage. This voltage is then fed into the three comparators where it is compared to a software selectable scaled version of the reference voltage. Connecting a 1 ohm resistor between the Emitter pin (pin 16) a ground will force all of the motor winding current to travel through the resistor and the voltage drop across the resistor can be read by pin 10, the current sensing pin. The use of a 1-ohm resistor will not significantly impede the flow of motor winding current and it provides a very simple way to determine the current. Pin 16 is called the Emitter pin because it is connected to the common emitters of the H-bridge.

Pin 8 is the phase selection pin. It is controlled by the microcontroller. The UC3770BN was designed so that only two of the transistors in the H-bridge can be on at any one time. This is a convenient safety feature because it means that the user never has to worry about sending a phase select signal that will unintentionally connect the motor supply voltage directly to ground. This situation would exist if the two right side transistors were conducting or the two left side transistors or all four of them. Instead the phase signal can only activate two

diagonal transistors. For example, when the top left transistor is conducting and the bottom right transistor is conducting, the motor winding current will flow from pin 15 to pin 1. The only other option is for the top right and bottom left transistors to be conducting which will cause current to flow from pin 1 to pin 15. By simply pulsing pin 8 with the microcontroller, the motor winding current is reversed.

Pin 2 is the timing pin. This is connected to an external circuit consisting of a capacitor and resistor in parallel. The capacitor and resistor combination form an RC time constant. This time constant is what determines the amount of time that the internal timer will be activated. This is the pivotal component of the current limiting capabilities of the UC3770BN. If 36 volts is applied to the motor voltage pin, the motor winding current will rise exponentially. Since 36 volts is over ten times the nameplate rating of the 5023-100 motor we are controlling, the current would quickly exceed the 2-amp rating of the motor. However, when the current sensing pin sees that the voltage drop on the external 1-ohm resistor has risen above the selected threshold, the timer is activated. When the timer activates, it immediately turns off the both lower transistors in the H-bridge and stops motor winding current. It remains off for a very short period of time (~ 30 μs). The off time is directly proportional to the time constant of the external RC timing circuit. After the short period of time the timer turns the motor winding current back on and waits for it to go above the threshold value. The cycle repeats for as long as the motor is on.

Figure 3.3b is the step response generated in MATLAB using the actual motor winding parameters (resistance and inductance) of the stepper motor we will be using to build the small-scale model. By using a 24-volt supply the motor winding current will reach the 2-amp peak value in less than half of a millisecond vs. nearly 10 milliseconds if the nameplate voltage was applied. The UC3770BN can handle up to a 50-volt input for the motor supply voltage, but the power supply used in the small-scale model can only supply a maximum 24 volts.

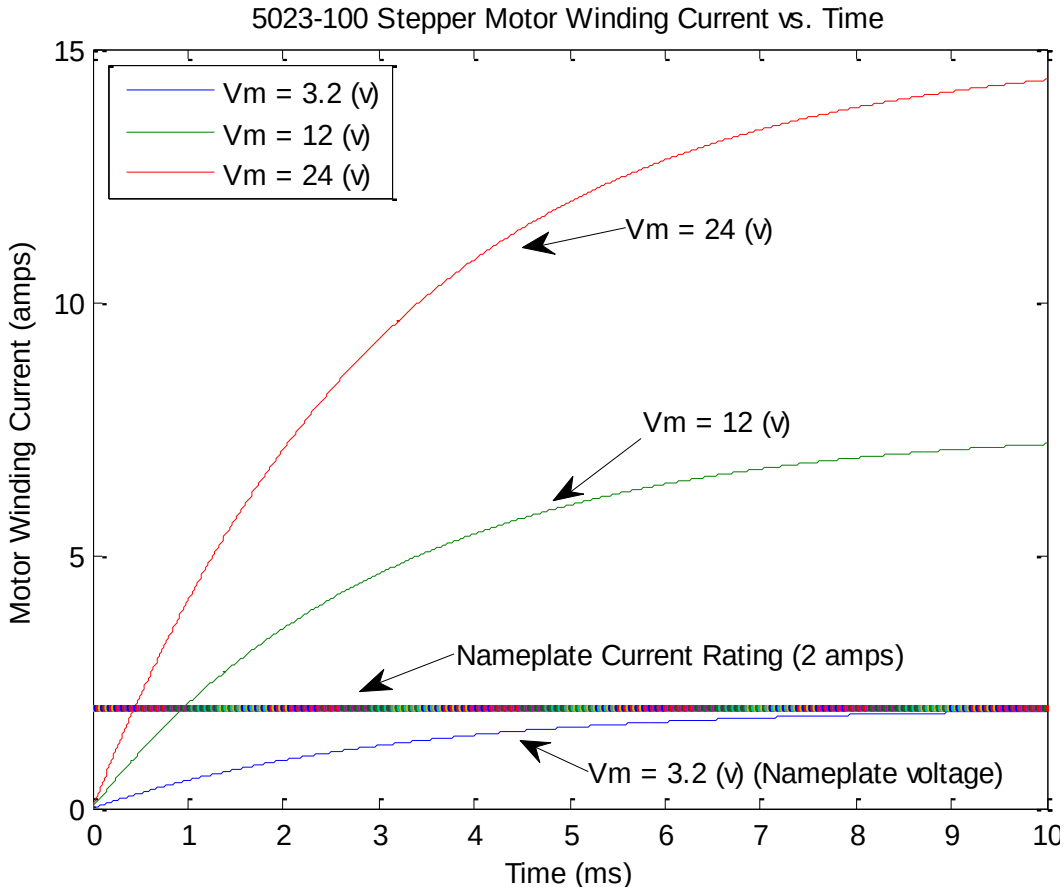

Figure 3.3b - Step Response

This Figure demonstrates the benefit of driving a motor at a voltage that is higher than the nameplate voltage rating. The motor winding current reaches its final value of 2 amps much faster. It also shows that when over voltage is applied the current must be limited to the safe operating range of the motor.

After studying the way the UC3770BN driver chips operate, a motor driver circuit utilizing these ICs was built on a perforated board. Table 3.3c lists all the parts and prices for the motor driver that was built for the small-scale model. Figure 3.3d shows the completed circuits.

Qty	Part Name	Part Number	Unit Price	Total Price	Supplier
2	Motor Driver	UC3770BN	Free	Free	ti.com
2	Terminal Block	-	$0.40	$0.80	Skycraft
2	16 PDIP socket	-	$0.80	$1.60	Skycraft
2	Current Sense Resistors	64K9380	$0.505	$1.01	newark.com
4	Diode	GI9544	$0.50	$2.00	Skycraft
2	Fuse Holders	-	$0.50	$1.00	Skycraft
2	2A 250V Fuses	-	$0.50	$1.00	Skycraft
1	Potentiometer	22C2W103	$0.50	$0.50	Skycraft
4	Capacitors	102	$0.22	$0.88	Spark Fun
2	Capacitors	104	$0.22	$0.44	Spark Fun
6	Resistors	10 kΩ	$0.03	$0.18	Spark Fun
2	Resistors	45 kΩ	$0.03	$0.06	Spark Fun
1	3 ¼" x 2 ½" Perforated Board	-	$2.00	$2.00	Skycraft
3	2 pin connector cables	-	$0.50	$1.50	Skycraft
			Total	$12.97	

Table 3.3c - Parts List for Motor Driver Circuit

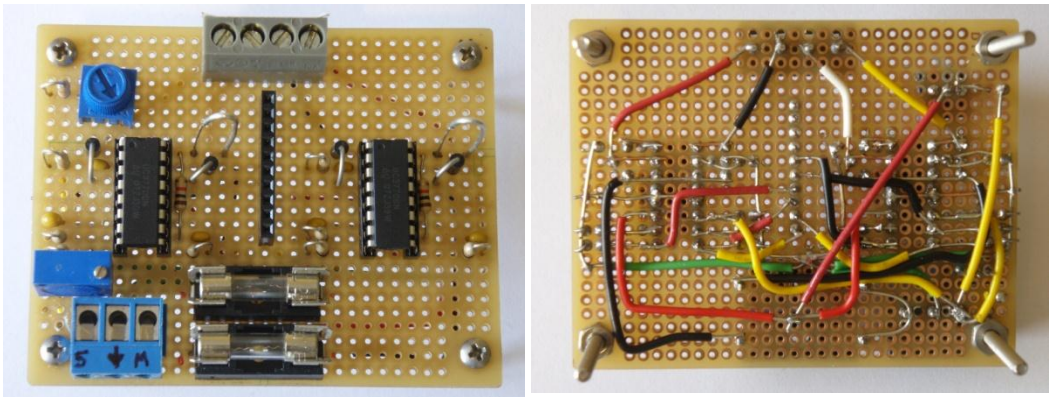

Figure 3.3d - Top and Bottom View of the Motor Driver Circuit

3.4: Power Supply

A power supply capable of supplying at least 30 watts output at multiple voltages was needed. The microcontroller requires a well-regulated 3.3 volts, the motor driver chips require a well regulated 5 volts, and depending on the liquid crystal display that would be used, it required 3.3 volts or 5 volts. The voltage required for the motors was much more flexible. As long as it was in the range of 10 to 50 volts, it would drive the motors. However, as Figure 3.3b shows, as the motor drive voltage increases the performance also increases. The regulation of this supply was not as important as the others. Ideally, the power supply needs to be built as efficient as possible to reduce unwanted heat, and it should be built in one small package. In an effort to get the small-scale model up and running as soon as possible, it was decided to build a power supply with only 3.3 volt and 5 volt outputs. We already had a 12/24 volt regulated power supply capable of outputting enough power to drive the motors. This supply came from the Stormwater Management Academy. It was the original power supply that the rain simulator used, but it was replaced due to its intermittent failures. After some testing, we determined that it worked well enough to use in the model, but would not be reliable enough to use in the final product. The input power rating of this power supply is up to 336 watts. The 24 volt output is rated at 48 watts. Each of the small stepper motors chosen for the model requires a maximum of 12.8 watts, so two driving two motors with this power supply should not be a problem.

To build the dual 3.3 and 5 volt regulated power supply we scavenged a local electronics surplus store to find the basic components. This was not going to be a switching power supply, so the first component needed would be a transformer with the proper turns ratio to get step the voltage down to between 8 and 10 volts. Then a full wave bridge rectifier would rectify the signal and a large capacitor would be used to filter out the ripple. The output of the capacitor should then be a smooth DC signal that will be fuse limited and connected in parallel to two adjustable voltage regulators. The voltage regulators require some external components such as resistors, potentiometers, and capacitors. Figure 3.4a shows a block diagram of the voltage regulator circuit.

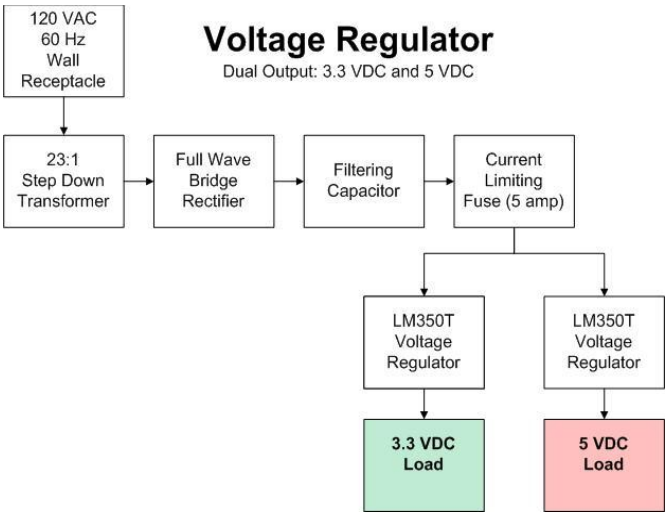

Figure 3.4a - Dual Voltage Regulator

This block diagram shows the general layout of the proposed low voltage power supply. The outputs will be used to power the motor driver chips, the microcontroller, and the LCD. These three loads combined will draw less than one amp.

The voltage regulators that were chosen are Motorola LM350T adjustable regulators. They have the following specifications [L7]:

- 1.2 – 33 volt adjustable output
- Up to 3 amp output current (Max power dissipation ≤ 30 watts)
- Internal thermal overload protection
- Internal current limiting
- Up to 80 dB ripple rejection

The output voltage range was well within the range required for our power supply. The internal thermal overload and current limiting features make this voltage regulator nearly indestructible. The nice features coupled with the fact that these regulators were locally available for only $0.75 each made them an ideal choice for our power supply. The design of a full wave rectifier is very straightforward. It merely consists of four diodes connected in a bridge circuit. Rectifier diodes are widely available as discrete components, and full wave bridge rectifiers are available fully assembled in convenient to use packages. Even though these components are relatively inexpensive and locally available, we had a better idea for procuring the parts. One of the group members had a large stash of unregulated plug in power supplies. After taking several of them apart to study their inner workings, it was decided to use the pre-built bridge rectifier circuit out of one them for our power supply. It already contained the four diodes (1N5391), the filtering capacitor (2200 µF, 16 volt), and a fuse (250 VAC, 5 amp). Due to the simplicity of the circuit, no learning experience was missed by using this pre-built assembly, and it had the benefit of being entirely free. Table 3.4b lists the names and prices of the parts used to build the low voltage power supply. The images that follow shown in Figure 3.4c is the finished part.

Qty	Part Name	Part Number	Unit Price	Total Price	Procurement
4	Rectifier Diode	1N5391	$0.00	-	Stock
1	Fuse	5 amp, 250 VAC	$0.00	-	Stock
1	Capacitor	2200 µF, 16v	$0.00	-	Stock
1	Transformer	3FD-520	$4.50	$4.50	Skycraft
2	Regulator	LM350T	$0.75	$1.50	Skycraft
3	Resistor	120 Ω 5%	$0.03	$0.09	Spark Fun
1	Resistor	330 Ω 5%	$0.03	$0.03	Spark Fun
2	LEDs	Red,Green	$0.17	$0.34	Skycraft
2	Potentiometer	271-342	$2.99	$5.98	Radio Shack
2	Capacitor	P975-ND	$0.22	$0.44	Spark Fun
3	Terminal Blocks	276-1388	$0.50	$1.50	Radio Shack
1	Perforated Board	-	$2.00	$2.00	Skycraft
			Total	$16.38	

Table 3.4b - Power Supply Parts Breakdown

The table shows a list of all the parts that were used in constructing the low voltage power supply for the small-scale model.

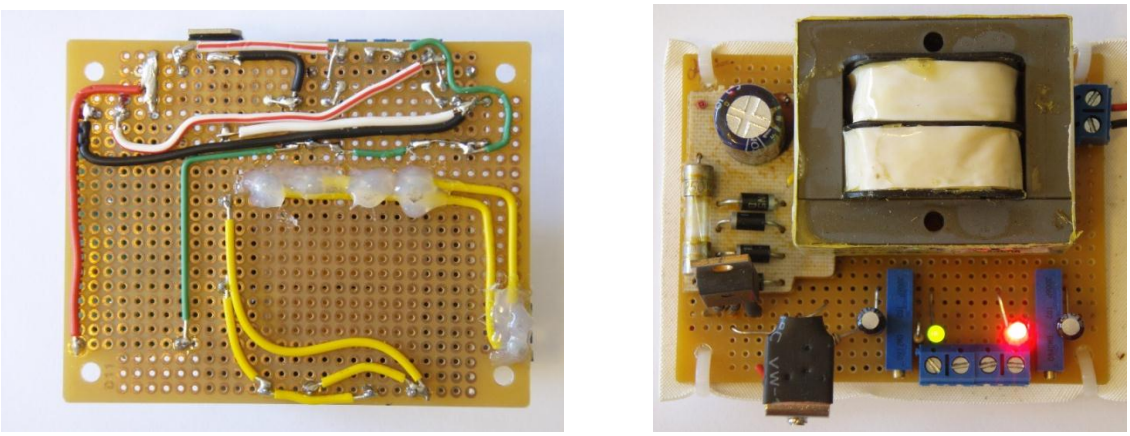

Figure 3.4c - **Dual Voltage Power Supply (3.3v and 5v)**

The images show the top and bottom views of the completed dual voltage regulator. The red LED and adjacent terminal is the 5-volt output. The green LED and the adjacent terminal is the 3.3-volt output. The LEDs were chosen to indicate power is available and also to serve as a warning so the 5-volt source does not get inadvertently connected to the components that require 3.3-volts. Next to each LED is a potentiometer that can be used to “fine tune” the voltage to exactly the desired level.

3.5: User Interface

The user interface was one aspect of the small-scale model that needed to be identical to the final design. By using the same interface on the model, we could rigorously test it and perfect its operation before building it into the final design. One user interface that was considered was to use a PC as the main interface tool. With this setup a graphical user interface would need to be built in a language such a Visual Basic or Java. The user would input the desired settings into the computer where they would be displayed on the screen. The computer would then need to communicate that information to the microcontroller that controls the stepper motor drivers. This is the type of interface that the rain simulator is currently using, but it is plagued with problems. The problems with the PC interface to the control system were one of the underlying reasons we were commissioned to do this project. We decided not to use this type of interface for several reasons. At the start of the project none of the group members were familiar with RS-232 communication with a computer. This was an added level of complexity that we were not ready to deal with yet. A lot of time would be needed to develop a graphical user interface and again, none of the group members had experience with that. The computer that the rain simulator is currently using is old and in need of replacement. Several computer failures have already caused unnecessary down time for the rain simulator. We did not want our final product to be negatively affected by a component that we had no control over, so we decided to eliminate the PC from the control system altogether. This would not only simplify our design, but it would make it more robust. The interface that we decided to use for our model is a twelve-digit keypad and a small liquid crystal display. Two different liquid crystal displays were tested. One was a 2-line x 16-character display, and the other was a 4 line x 20 character display. Both of these components are inexpensive and easy to obtain. The size of the liquid crystal display did not need to be very large because only two parameters need to be displayed. The two settings that the rain simulator needs are rainfall intensity and rainfall duration. Figure 3.5a shows an image of the user interface prototype and the parts list is shown in Table 3.5b.

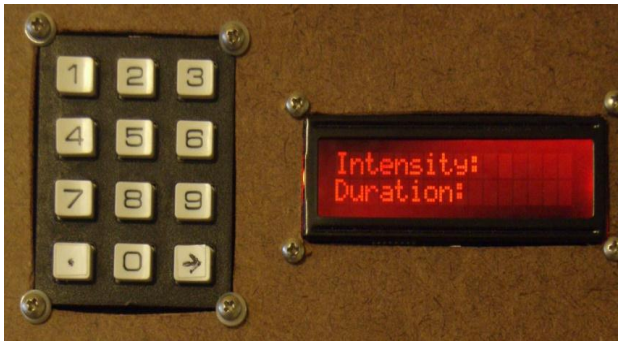

Figure 3.5a - User Interface

This is the user interface that was built into the model. The LCD is the 2-line x 16-character display which easily displays the Intensity and Duration settings.

Qty	Part Name	Part Number	Unit Price	Total Price	Supplier
1	2x16 LCD	LCD-09068	$24.95	$24.95	Spark Fun
1	3x4 keypad	COM-08653	$3.95	$3.95	Spark Fun
1	1 x 10 Pin Female Header	-	-	-	Included w/ Launchpad
1	1 ½" x 1 ½" PerforatedBoard	-	$0.95	$0.95	Skycraft
3	Resistors	10 kΩ	$0.03	$0.09	Spark Fun
			Total	$29.94	

Table 3.5b - Parts List for the User Interface of the Small-scale Model

3.6: Software for the Small-Scale Model

Two different software approaches were taken when developing the rain simulator model. The idea was that two of the group members would independently develop the user input software. When they were both completed, they would be compared, and the best features of both could be combined into the final software. The software for the small-scale model is a continuous work in progress. Since the software for the model and the final design are very closely related, the model provides an excellent testing device. This project is heavily dependent on software, so having this model to use for testing is very important. The model was completed early in the semester and throughout the remainder of the semester, code for different parts of the project was written and thoroughly tested. Many hours have been spent writing code that takes input from the keypad, converts it into intensity and duration, then converts those numbers so they can be used to generate step speeds for the motors and pause times. This is all a direct result of the research that was documented in the microcontroller research and development section. Interrupt service routines have been written and tested. The use of the timers has been mastered to generate interrupts at the step rate of the motor. All of the ports and pins have been assigned according to their capabilities. The software was all written in blocks according to its function. One C source file was written to read and decode the keypad buttons pressed. A separate C source file was written to control the motors. This made it easier to develop specific areas of the software and determine errors. In the end all of the individual C source files can be combined in one project for the final design.

3.7: Final Build of the Small-Scale Model

Throughout the course of building the small-scale model, many lessons were learned. As Neils Bohr said: “An expert is someone who has made all the mistakes that can be made in a narrow field.” We may not have become experts in the field of stepper motor control but the many mistakes that were made along the way are now much less likely to show up in the final design. Many hours were wasted unsoldering components and replacing them with different ones. It was not a total loss though as many new soldering techniques were developed. The dual voltage regulator is still usable despite the 5-volt supply having a ripple when a load is connected. This problem was not entirely due to negligence though. When the transformer was being tested, it was discovered that the listed turns ratio was incorrect and what was supposed to be a 10-volt output was actually 8 volts. However it was a dual tap transformer and by rearranging the wire connections a higher output voltage of 18 volts could be obtained. When testing with this higher output voltage, the linear regulators were forced to step down from 18 volts to 3.3 and 5 volts. The regulators became hot quickly and it was decided to use the 8-volt output of the transformer instead. After this decision was made the circuit was built on a perforated board only to find out that 8 volts is not high enough, and we would have been better off to use heat sinks and deal with hot regulators.

The motor drivers were tested extensively on a breadboard. Several chips were burned out in the process so fuses were placed in series with the motor windings to try to avoid this. The reference documentation for the chosen motor driver chips was severely lacking in information about the external parts needed. This resulted in a “try it and see what happens” approach. This approach could have been avoided with further research, but the intent of the small-scale model was to get it working as soon as possible and learn as much as we could about how to control a stepper motor in the process. Perhaps a better approach would have been to purchase a pre-built motor driver assembly that could be used for testing our software. This is the classic chicken and egg scenario. If research comes first, then the final design will be better, but by starting with absolutely no knowledge of stepper motors and no hands on exposure to the parts, we didn’t know what to research. The decision to use a perforated board to build the motor driver circuit lead to a plethora of problems. Having over one hundred hand soldered connections in very close proximity to each other on something as imprecise as a perforated board leads to many parasitic conductive bridges between solder points. The problem is amplified when multiple parts must be replaced multiple times. Many hours were spent repairing the board after it was built due to these problems. Because of this only one motor driver was built instead of the planned two. Overall, the design and build of the small-scale model was highly successful. We learned about the specific areas that will require the most research, and we now have a working model that can be used for testing software and for demonstrating our ideas to our sponsors. The following table,

Table 3.7a, summarizes the costs and materials used in the small-scale model and the images Figure 3.7b show the completed small-scale model.

Qty	Part Name	Part Number	Unit Price	Total Price	Supplier
1	Dual Voltage Regulator	See Section 3.4	$16.38	$16.38	See Section 3.4
60	Stainless Nuts, Bolts, Washers	-	$0.05	$3.00	Skycraft
1	5/12/24 volt Power supply	T-120D	Free	Free	Stormwater Academy
1	Stepper Motor	100-5023	$7.50	$7.50	Skycraft
1	3/14 appliance cord	-	$2.25	$2.25	Skycraft
1	Microcontroller	See Section 3.1	$10.80	$10.80	See Section 3.1
1	Motor Driver	See Section 3.3	$13.03	$12.97	See Section 3.3
1	User Interface	See Section 3.5	$29.94	$29.94	See Section 3.5
			Total	$82.84	

Table 3.7a - Parts List for the Final Build of the Model

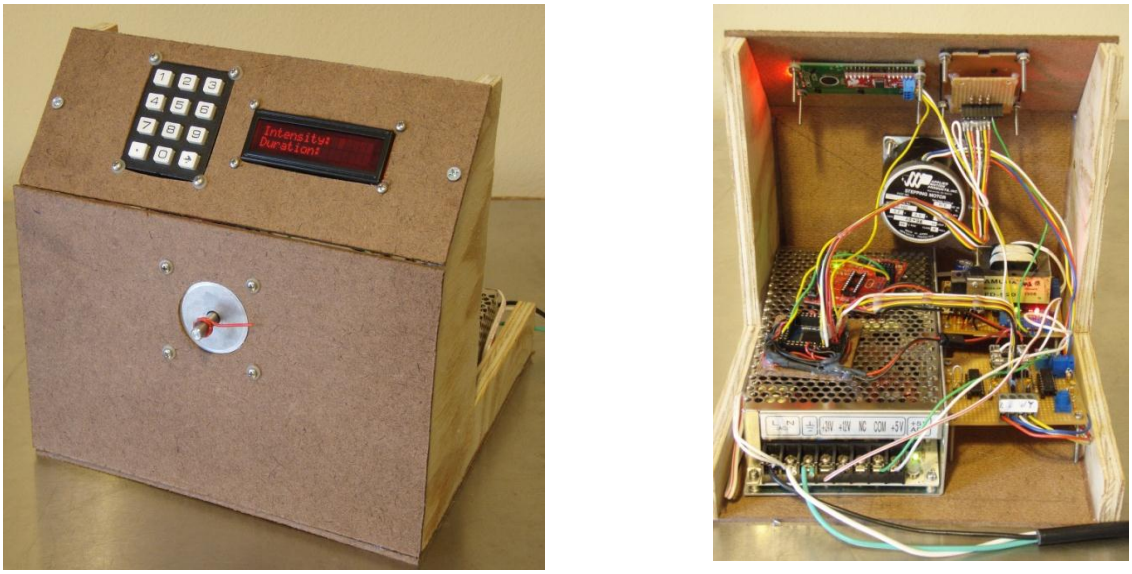

Figure 3.7b - Small-scale Model Final Build

The upper two images show the back and front views of the completed model. The user enters the desired intensity and duration and the simulation starts. We will gradually build more complexity into the software using this model.

4.0: Final Design

After considering the many options available, we were able to make our final design decisions. The stepper motors, motor drivers, user interface, microcontroller, power supply, and transmission lines part numbers were chosen and the circuit schematics were put together.

4.1: Microcontroller

The microcontroller that was selected for use in the final project was the MSP430F5438a. The reason for switching to this microcontroller is that Texas Instruments mistakenly sent a target board for that particular model when we asked for an MSP430F5435 target board. The F5438a microcontroller has exactly the same functionality as the F5435, but it contains more I/O pins. All of the information written about the F5435 peripheral modules, setup, and operation still applies to the F5438a. No additional features were used. All of the software previously developed for the F5435 microcontroller was easily ported over to the new one and no setbacks were incurred. The new target board was much easier to program and the JTAG Field Emulation Tool allowed for faster programming times and more efficient in-circuit debugging.

4.2: Motor Drivers

The design for the motor drivers in the final design was mostly software design to operate the motor driver IC. However, there were some requirements that needed to be met and most of those were provided by Texas Instruments in the DRV8432 Data Sheet. In order to implement the Cycle-by-Cycle current limiting, a resistor needs to be placed between the OC_ADJ pin and Ground on the motor driver. The stepper motors that are already in place at the Rain Simulator are rated for 5.6 amps and therefore we planned on using a resistor value of 50,000 Ohms. If was discovered during testing that this resistor value needed to be changed, then that change could occur relatively easily. In Figure 4.2a, all of the necessary connections and components for the DRV8432 Motor Driver are shown. All of the parts on this schematic are called out except for the Roc_adj and Loc. As stated above, the Roc_adj value is 50,000 ohms and the Loc value is calculated using the equation:

$$Loc_{minimum} = \frac{(PVDD * Toc_{delay})}{Ipeak - Iavg}$$

Where Toc_{delay} = 250nS and Ipeak = 15 amps

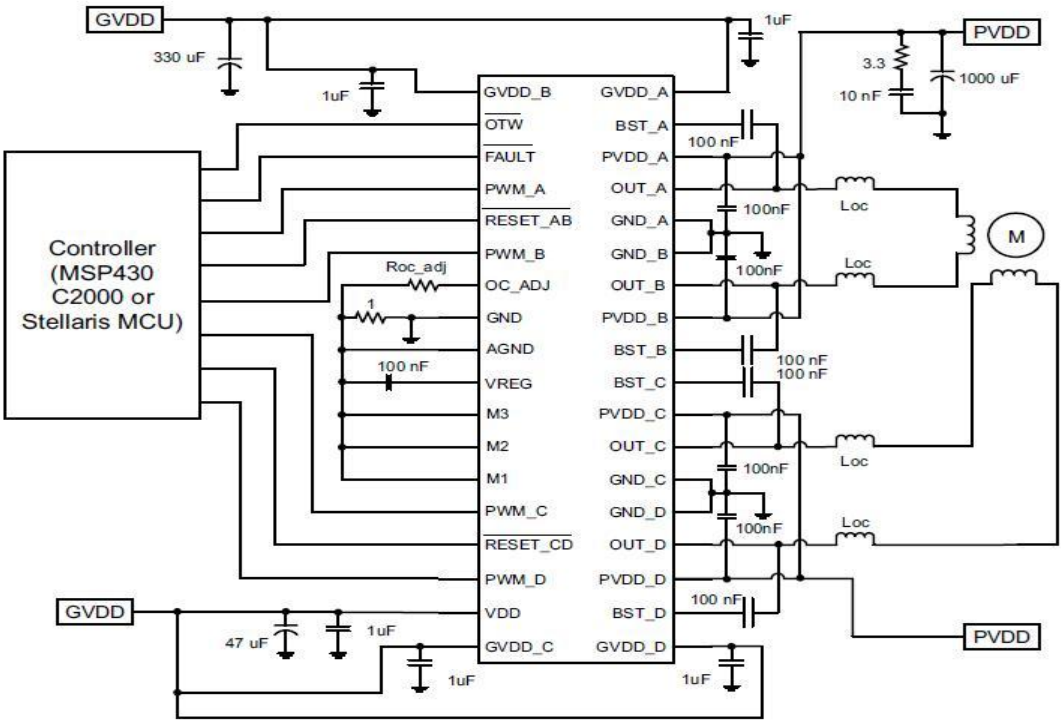

Figure 4.2a – Motor driver Schematic [A1]
This schematic shows the proper method of connection for the DRV8432 Motor Drivers.
(Permission Granted by Texas Instruments)

The DRV8432 only requires four inputs from the micro-controller. The software will need to apply a PWM signal to these inputs. The frequency and order of the pulses determines the speed and direction of the stepper motor. The generation of the PWM signals is discussed in the micro-controller design section of the project. The components that are needed for operation of the motor driver do not have strict requirements; thus, we planned on using generic parts from a local supplier or Internet distributor.

4.3: Power Supply

When we were presented with this project, we were not given a list of specifications that needed to be met for all parts of the control system. Thus, it was left to us to determine what the power supply requirement is for the operation of the stepper motors. From the specifications of the motors we found that the motors required 5.6 Amps of current for each phase on each motor to run at full torque capacity, while the nameplate voltage is 2.7 Volts. [D6] If the current supplied to the windings is not the nameplate current, the torque output of the motors reduces. Thus, we need the power supply to supply as much current as possible as is cost efficient, approaching 11.2 Amps per motor.

At first this seems like a simple high current, low voltage power supply design. However, we found that in order to allow the stepper motor to perform more steps per second, have a higher stepping rate, the motors can be supplied with a voltage that is commonly up to 50 times greater than the nameplate voltage [D2], so long as the current rating of the motor is not exceeded, which is what the chopping motor driver circuit in our design limits. We found that stepping rate of the motors may have a high requirement for lower rainfall intensities, depending on the gear head ratio on the motor attachments, which we found to be 10:1. But, the choice of our motor driver chip limits the voltage that can be supplied to the motors at 50 V_{DC}, nominally. Thus, the requirements for the high voltage supply are as follows:

- Be able to supply as close to 11.6 Amps per motor as possible
- Achieve the most cost effective voltage level output, approaching 50 VDC

To design this supply circuit we had to choose between a regulated and an unregulated power supply. The initial perception is that the regulated supply would be the best solution because the motors draw a high amount of current thus the voltage level would need to be highly regulated. However, because the driver circuit for the motor contains a freewheeling circuit, that supplies current back to the power supply, the regulated power supply would back fight the freewheeling current that attempts to return to the supply. So, this back fighting would cause a large voltage to accumulate that could possibly damage the motor driver or regulated power supply circuitry [D1]. Furthermore, as is discussed in section 2.6.2, if we used a switching power supply, the high power PWM switching current could cause a large amount of EMI, which we felt could potentially interfere with the control circuitry and driver circuits. Therefore, we decided that the potentially cheaper and simpler un-regulated power supply is the best option for the high voltage supply.

The basic setup for an un-regulated power supply is a simple one indeed. The purpose of the supply is to step down the main voltage input to the desired voltage level, and keep the output voltage ripple within limits. To perform the step down, a power transformer is implemented. Thus, the secondary voltage is a smaller RMS voltage that needs to be rectified and then smoothed out with a filtering capacitor, and then current limited by a fuse. There are many circuits that can be used to accomplish this rectification and filtration; for our design we chose to simply use the full wave bridge rectifier and capacitor. The flowchart diagram, in Figure 4.3a, is shown below to illustrate this process.

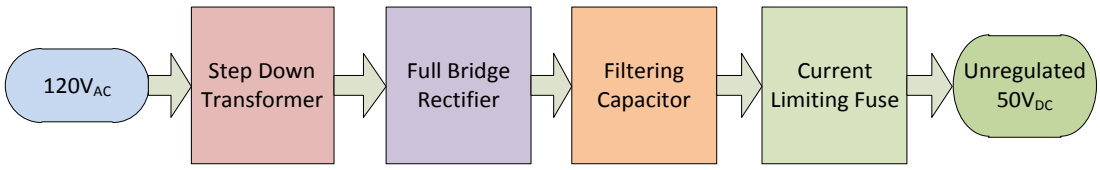

Figure 4.3a - Flowchart Diagram of Un-regulated Power Supply Design

4.3.1: Transformer

Perhaps the most crucial component of the unregulated power supply is the power transformer that is implemented to convert the high level power down to the desired output level. Since each motor requires 11.6 Amps when running at full torque capacity and we ideally want to have 50 V_{DC} applied to the motors, we need to determine the minimal voltage and current ratings that are acceptable for the transformer used in our design.

We found that having a power supply that is capable of providing at least ¾ the amount of nominal current to the stepper motors, while in full-step mode, would be adequate [D1]. Furthermore,

> Bipolar chopping steppers are very current efficient as far as the power supply is concerned. Once the motor has charged one or both windings of the motor, all the power supply has to do is replace losses in the system. The charged winding acts as an energy storage in that the current will re-circulate within the bridge, and in and out of each phase reservoir. While one phase is in the decaying stage of the chopper, the other phase is in the charging stage, this results in a less than expected current draw on the supply[D1].

We even found this to be true in our small scale implementation, where rated current for the combination of the two phases in the motor is 4.0 Amps, we used a transformer that was only rated for 2.0 amps, and the operation of the circuit was not impeded.

Voltage levels on a transformer are typically given in V_{RMS}. Because the filtering capacitor will hold the output voltage close to the peak value, we need to convert the desired voltage of the secondary winding on the transformer from the rectified DC output levels. Since the desired output voltage is 50V_{DC} and the voltage drop across a full-bridge rectifier is typically around 1.4 V_{DC} we can calculate the desired secondary voltage, as shown in the equation below.

$$V_{secondary} = \frac{(50 + 1.4)}{\sqrt{2}} V_{DC} \approx 36.34\, V_{RMS}$$

And since we found that the current rating on the transformer needs only to be ¾ the nominal current draw of the stepper motors, we find that the actual necessary minimal power rating for the transformer that provides the maximum allowable voltage is as shown by the equation below.

$$Power\ Rating = 11.6 * .75A * 36.34\, V_{RMS} \approx 316\, Watts$$

Thus, our design required a transformer that was capable of performing the RMS voltage level step down from 120 V_{RMS} to approximately 36 V_{RMS} at most, and the current rating on the secondary winding must be at least 8.7 Amps.

Since we needed to assemble two high voltage power supplies, one for each stepper motor driver circuit, we found that we needed to purchase at least two transformers that were capable of a voltage up to 36 V_{RMS} and 8.7 Amps on their secondary windings or one transformer that is capable of at least 17.4 Amps. Furthermore, to maintain cost effectiveness, the price of the transformer weighed greatly on the choice in the design. We searched Mouser, Digikey and Newark for a product that is capable of our requirements. The comparison between three of the options is displayed in Table 4.3.1a below.

Distributer	Manufacturer	Part Number	Power Rating	Price
Mouser	Triad Magnetics	VPT48-20830	2x24V @ 20.83A	$168.78
Newark	Multicomp	MCFE500/25	2x25V @ 10.00A	$99.84
Digikey	Hammond	165S30	1x30V @ 10.00A	$85.05

Table 4.3.1a - A summary of transformer options
[D12] [D13] [D14]

4.3.2: Full-Wave Bridge Rectifier

A small, yet extremely important, component of this supply is the full wave bridge rectifier. Without this component, the supply would be incapable of providing the DC power that is required by the motors. Since all of the current flows through the full-wave bridge we needed to have a component that is rated for at least 11.6 Amps per power circuit for a motor driver.

Furthermore, we needed to find a component that was voltage rated for at least twice the peak voltage value of the secondary winding of the transformer, because the current flowing back to the power supply from the bipolar driver circuit can cause large momentary voltage increases. We determined that a voltage rating of at least 85 Volts is necessary. Also, to improve efficiency of the system, we would like to find a component with the least forward voltage drop across the diodes, so that a higher voltage could transmitted to the capacitor and then the motor driving circuit.

Finally, we considered the actual mounting configuration of the full wave bridge, how the component would be mounted to the power supply. There are many configurations that the full wave bridge can assume, including through-hole, surface mount, panel, and chassis quick-connect. Because this high voltage power supply is composed of very few, yet very large components, we found it impractical to use a through-hole or surface mounting part, because it is unlikely that we would use a printed circuit board. Thus, we decided that the bridge should be one that we could bolt onto a fastening plate and have quick-connect connectors. Again, we searched and found many components that are capable of

meeting the requirements of the design. Table 4.3.2a displays some of the leading choices for our full wave bridge.

Distributer	Manufacturer	Part Number	Power Rating	V_f	Price
Mouser	Taiwan Semiconductor	GBPC4002	200V @ 40A	1.1V	$ 3.92
	Taiwan Semiconductor	GBPC4004	400V @ 40A	1.1V	$ 4.16
Newark	Multicomp	CM3502	200V @ 35A	1.2V	$ 2.25
	Multicomp	MCCM2502-RH	200V @ 25A	1.2V	$ 1.83
Digikey	Diodes, Inc	GBPC2502	200V @ 25A	…	$ 2.79
	Diodes, Inc	GBPC2504	400V @ 25A	…	$ 2.79

Table 4.3.2a - A summary of Full Wave Bridge Options [D15] [D16] [D17]

4.3.3: Capacitor

The final integral part of the stepper motor power supply is the filtering capacitor. The purpose of the capacitor is to store charge from the rectified input from the transformer while input waveform reaches its peak value. While the input waveform decays from the peak value, the capacitor releases the stored charge to the load and maintains the output voltage at a relatively stable level until the input waveform recovers and charges the capacitor again. The ripple voltage is determined by the capacitance and the average voltage for the output of the supply is approximately the maximum voltage minus half of the ripple voltage.

We found that to maintain the voltage of the output, within a 10% ripple from the maximum value, the minimum capacitor value can be calculated as shown in the equation below [D4].

$$C = \frac{5 * I_o}{V_s * f}$$

Where, Vs is the maximum value that the input waveform reaches, f is the frequency of the input and Io is the load current. Thus, we calculated the capacitance using a load current of 11.2 Amps, a frequency of 60 Hz and a maximum voltage of 41 V_{DC}. Therefore, we found the value of the filtering capacitor should be in the range of 22mF as shown by the equation below.

$$C = \frac{5 * 11.2}{41 * 60} \approx 21{,}749\mu F$$

Because a 0.022 Farad capacitor can be expensive, thus hindering the cost effectiveness of our design, we decided to allow a 20% ripple voltage. This allowance is acceptable because the driver circuits and stepper motors are not

sensitive to voltage levels. Thus, we needed only find a cheaper capacitor in the range of 0.011 Farads. This allowance generates an average voltage of about 37 Volts and does not hinder the performance of our system.

Again, since we did want to use a printed circuit board in the design of this power supply we needed to find a capacitor that could be mounted to a back plate and be easily connected to the quick connect full wave bridge. Furthermore, since the capacitor could be subject to voltages higher than the nominal value, because of back feeding current from the bipolar drivers and irregular voltage levels from the main source, we need a capacitor that has a voltage rating that is close to twice as much as the nominal value at 41 $V_{DC.}$ Table 4.3.3a reveals some options that met our specifications.

Distributer	Manufacturer	Part Number	Rating	C	Price	HRS @ C°
Mouser	Mallory	CGS103U075V3C	75V	10,000 uF	$23.90	… @ 85
	Kemet	ALS30A103KE100	100V	10,000 uF	$25.65	… @ 85
	United Chemi-Con	36DA183F100CC2A	100V	18,000 uF	$28.53	… @ 95
Newark	EPCOS	B41456B9229M	100V	22,000 uF	$51.29	… @ 85
	Cornell Dubilier	59K0139	75V	14,000 uF	$23.19	4000 @ 85
	Rubycon	38M0408	100V	10,000 uF	$20.37	3000 @ 85
Digikey	EPCOS	B41560A9109M	100V	10,000 uF	$19.86	3000 @ 105
	EPCOS	B41456B8229M	63V	22,000 uF	$17.81	12000 @ 85
	Cornell Dubilier	338-1265-ND	75V	10,000 uF	$28.68	2000 @ 85

Table 4.3.3a - Summary of Capacitor Options [D18] [D19] [D20]

4.3.4: Stepper Motor Supply Design Summary

Now, having considered each of the components of the supply, all of them must be integrated into the overall design. The main goal of this power supply design is to meet the power specifications with the most cost effective combination of components. Therefore, the price and even the availability of the components weighed in on the decision process, as well as their power characteristics. When choosing which transformer to use, we decided that it would be more advantageous to use the two transformers at $30V_{RMS}$ instead of the lower 24 or 25 V_{RMS}. This choice resulted in a voltage output of 42 VDC. By keeping the voltage higher, we increase the torque performance of the motors at higher stepping rates.

Also, after some testing, we found the power consumption of each motor to be significantly less than what we calculated. Each motor consumes about 30 VA. So, we were able to use much smaller, much less expensive transformers and capacitors. We were able to procure several 3,900 uF capacitors that were rated

at 450 VDC and had very high ripple current ratings. These capacitors were donated to our project for free. That significantly helped to keep the cost down for this part of the project. We choose to implement the each leg of this power supply with a bank of 4 capacitors, 1 full-wave bridge and 1 transformer. For safety considerations we choose to integrate some 100k bleed resistors across the capacitors. This was to assure that the voltage would slowly die down after the supply was turned off. Figure 4.3.4a is the schematic of the final power supply that was used in the project.

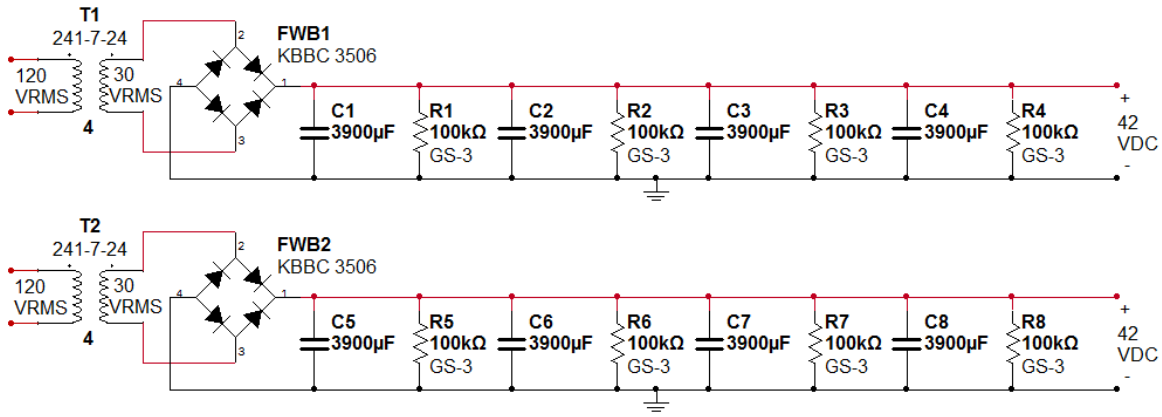

Figure 4.3.4a - Schematic of Design for High Voltage Power Supply

4.3.5: Control Voltage Power Supply

The second power supply provides the low voltage specifically needed by the control circuitry. This supply needed to be capable of providing the appropriate levels of power to the user interface, the microcontroller, condition indicating lights, cooling fans and over-travel sensors. Each of these components has a specified operating voltage.

After considering the voltage level requirements of each of the components we determined that the low voltage supply would need to be capable of providing 3.3VDC for the microcontroller and indication lights, 5.0VDC for the LCD and 12VDC for the over-travel sensors, cooling fans, and motor drivers. We know that the microcontroller, LCD and LED lights have a relatively small current draw. However, upon inspection of the cooling fans and over-travel sensors we found on their name plates that the fans require 0.25 Amps each and the over-travel sensors require 0.2 Amps each. Thus, the 12VDC portion of the supply must be capable of providing 1.3 to 2.0 Amps, nominally.

However, since the control system is enclosed we needed to implement fans as cooling devices. The 12VDC supply needed to be capable of providing enough current for the cooling system as well as the motor drivers and indication lights.

Anticipating about four fans for cooling, each running at 0.25A, means that the supply should be capable of providing 4A at 12 VDC. In the final design we ended up using only two cooling fans reducing the demand for the 12VDC supply to 2.5A.

In the small-scale model for the motor control system we had to construct a small power supply that could provide 3.3 and 5.0 VDC for the microcontroller and LCD. In the model a linear regulated power supply was constructed, tested and verified capable of being used in the end product. However, the main project requires a 12VDC supply, also. The transformer in the model was not rated to handle the necessary 12VDC and 4 Amps. Thus, another transformer needed to be purchased for the full-scale design.

Since the portions of the supply that provide the 3.3VDC and 5.0VDC have already been designed and tested, the only further need is to find a voltage regulator device capable of providing 4A at 12VDC. The LT1083 linear regulator by Linear Technology is a device that is aptly capable of performing this task. It is an adjustable regulator that is rated for supplying up to 8.0 Amps of current with a 5.0 Volt drop across the regulator; and it can operate with a dropout voltage as low as 1.0 Volt, depending on the load current requirement [D21]. The 5VDC regulator that was used in the final design was a MC7805CTG from On Semiconductor and the 3.3VDC regulator that was used in the final design was a UA78M33DYC from Texas Instruments.

To design the multi-output power supply three regulator circuits have to be realized. All of the regulator circuits require the main 120 Volts to be transformed down, and then rectified, to a level within the range of their inputs. Then the input must be filtered by a capacitor to maintain a relatively stable DC level on the inputs to the regulators. All of the regulators can tap from the main rectified DC source. Thus, all of the input capacitors will be in parallel and are effectively one capacitor. To compute the necessary capacitance for a 20% ripple on the input to the regulators the following equation was used.

$$C = \frac{2.5 * (5.0A + .25A + .25A)}{12V_{AC} * 60Hz * \sqrt{2}} = 2{,}455\mu F$$

The output of each regulator will have a .1 uF capacitor attached to it, to help reduce high frequency noise and output ripple. Also, the circuit diagram for the low voltage power supply is shown in Figure 4.3.5a. This design includes three LEDs to indicate that the supply for each voltage is on.

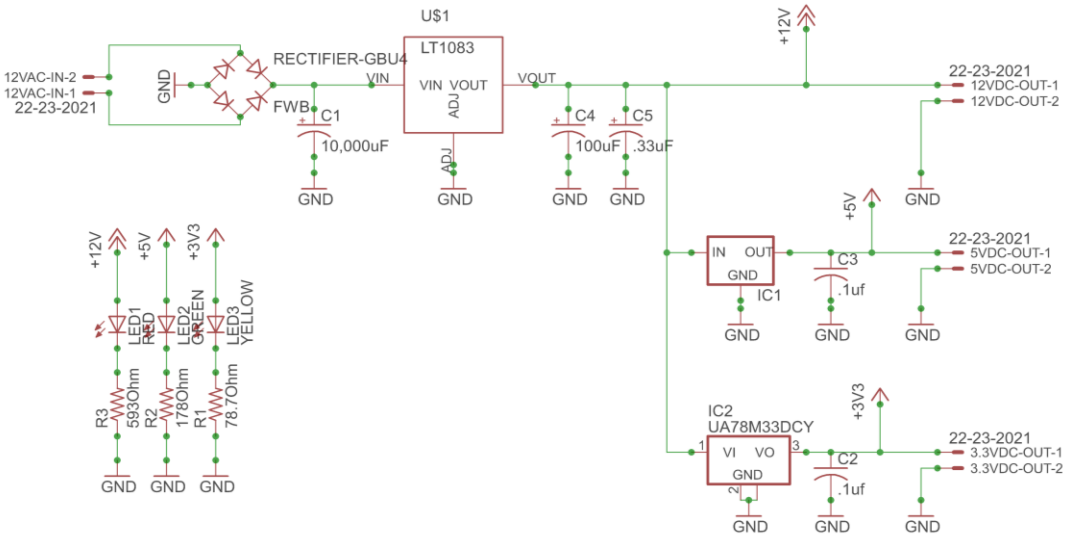

Figure 4.3.5a - Circuit Diagram for the Low Voltage Power Supply

4.4: Over-Travel Sensors

Since the control system for the stepper motors is an open loop system there is no feedback that relates to the position of the rotor. If the stepper motor operated ideally this would not be a problem. However, depending on the step rate and load conditions the stepper motor can skip steps; this is known as slipping. If the motor slips the control system assumes the rotor is in one position while the rotor is actually offset from that position by how many steps it has skipped.

This error can cause the rain simulator to expose the rain nozzles to the window for an incorrect amount of time; and it can cause the motor shafts to over-rotate, leaving the nozzles in an undefined area. To prevent the motor from over-rotating, in either direction, two HTM FCM1-1805N-ARU4 proximity sensors are attached to the casing, as can be seen in Figure 2.8a. A ferromagnetic lug is embedded in the disc on the motor shaft; its position corresponds to the defined region of rotation for the rain nozzle angle. If the lug ever reaches the over-travel sensor, the sensor is triggered and it sends a fault condition signal to the control system.

The over-travel sensors are NPN switches with open collector outputs, and they can be powered by the same 12 VDC signal that had to be supplied to the stepper motor cooling fans. The OT sensors were easy to interface to our system because they could be directly connected to the ports of the microcontroller with the use of pull-up resistors. When any of the sensors were triggered, they would drive the signal low and cause the microcontroller to generate an interrupt on the falling edge of the signal.

4.5: Wireless Rain Gauge

Table 4.5a below clearly shows the varying characteristics among the three wireless technologies researched.

Device	Frequency	Range	Power Consumption	Baud Rate	Data Packet Size
Wi-Fi	2.4GHZ	<100m	High	<56Mb/s	Large
Bluetooth	2.4GHZ	<10m	Medium	1Mb/s	Large
Zigbee	2.4GHZ	10-100m	Low	<100Kb/s	Small

Table 4.5a- Comparison Chart of Wireless Technologies

Wi-Fi has proven to be the technology with the highest transfer rate, the highest power consumption, and comes standard with the longest range that the signal can reach. Although Wi-Fi is a great and widely used technology, it does not seem to fit in well with our specific application to the rain simulator. Wi-Fi offers us with more power than we really need for our application. The deal breaker with Wi-Fi is definitely the higher power consumption. Ideally, we would like to put a transceiver on each rain gauge located on the bed of the rain simulator. If we need higher power, we will have to find a way to connect a power supply via wires to each transceiver or use a battery that will have to be changed out often or at least more often than it would need to be with Zigbee or Bluetooth. The frequent changing of batteries will induce a higher cost and decrease convenience for the end users of the rain simulation lab. And if we wire a power supply to each transceiver we will then be introducing another problem: acquiring wires that can perform adequately in the presence of lots of water and humidity. Additionally, we would have to track the wires back along the edges of the rain bed. This would put more wires in the way than we need there to be. The lab workers are constantly walking around the rain bed and having more wires present would only provide a bigger inconvenience and safety hazard. Thus, Wi-Fi was not the technology for this application.

Bluetooth proved itself to be a relatively medium power-consuming product with a mid-level transfer rate for short-range applications. Even though there are ways to make Bluetooth carry a signal to a larger range, this would require much more research and what seems like endless testing on our end. Bluetooth was *not* designed for long range and it would be much easier to work with a technology that was designed for this only because we could expect to have less problems and less headaches. Additionally, Bluetooth was also designed to carry large packets of information at a time. Again, this exceeds the requirement for this specific application. And along with exceeding requirements, comes higher cost as well.

After careful consideration of the three wireless options, Zigbee seemed to meet our every critical need. The most important factors for our application are ease of use and integration, price, and low-power consumption. With Zigbee, we are able to power the transceivers on the rain gauges with a simple AAA battery and they will need to be changed no more than once a year. Additionally, we are able to integrate virtually any number of rain gauges needed. This benefit will extend much farther than the duration of this project as well. In addition to creating a successful control system for the rain simulator by the end of this year, we want this system to continue being dependable for many years thereafter. Also, we need to consider the fact that the team may or may not be available to assist with any future issues there may be with our control system. Whatever we integrate not only needs to be reliable but also convenient and easy for the lab technicians to fix any small issues and provide the appropriate maintenance to assure the performance of the machine over time. The plan was to move ahead with the integration of a Zigbee including 1 chip on system working jointly with our main MSP430 class micro-controller and up to 10 end devices each located on a rain gauge. Please reference our conference paper for the final design of the wireless rain gauge.

The final product for the wireless rain sensor is best described as a sophisticated integration between a manually read rain gauge, electronics, and a wireless antenna to convert it into a fully-automated rain gauge. This rain gauge has been designed to hang from the main horizontal beam of the nozzle support system. Due to the variation in the incline that the rain bed will be subjected to, we initially considered constructing a self-balancing base to hold the rain gauge. However, due to the large variation in materials tested on the rain bed, the rain gauge would have had to be able to attach to all of the different types of materials ranging from hard concrete to porous soil. The group would have had to construct different mounts for the base of the rain gauge that would be changed depending on the type of material being tested. To avoid this additional effort, the rain gauge will be hung very low from the horizontal beam to the point where it almost reaches the rain bed but far enough to not be touching the rain bed when put at an incline. This design does not need a base support system and will automatically always be held in the appropriate direction to catch and measure the incoming rainfall.

The rain gauge was also designed to surpass the rainfall measurement accuracy of 0.25 inches. From 0 to 1.4 inches, the gauge will measure with a 0.1-inch accuracy and anything above 1.4 inches will be measured with an accuracy of 0.2 inches. The water level sensing system consists of 32 bare conductors that are submerged into the water with an open circuit in between that is later bridged by the rising water. Each conductor resides at a specific water level and will only conduct current when the water reaches that specific level.

Thus by applying a small voltage of 3.3VDC to one side of the conductors, we will only see this small voltage when the water has reached this discrete level.

Through the use of two 16:1 multiplexers, we were able to connect and control the voltage through all 32 conductors. A small Texas Instruments MSP430-G2553 microcontroller has been programmed to scan all 32 conductors and determine how many are under water. For every high conductor that the microcontroller sees, it will increment a counter. Once all 32 conductors have been cycled through, the final number on the counter will denote the amount of high lines read. This amount of high lines will be the amount of water levels reached and from this we can convert to the reading of the water level in inches. This final reading in inches is then sent to the XBEE antenna which in turn sends the signal to another XBEE module located on the MCU PCB. This receiving XBEE then sends the signal to the MCU where it is then communicated to the end user via the LCD screen on the outside of the enclosure.

Due to the fact that rain gauge is a stand-alone system, it needs its own power supply. To power the rain gauge we are using two AA batteries each of which are 1.5VDC. The two multiplexers need a minimum of 3.1 volts to function and the microcontroller will only take a maximum of 3.3 volts so we also integrated a 3.3VDC boost converter to ensure that our voltage level is always at the necessary magnitude to power these two crucial components. Additionally, an LED was integrated to the system to give a visual notification of when the system transmits data. All of the electronics will be stored in a small water-resistant box attached to the water-gauge. The researchers will be responsible for keeping track of the battery life of the rain gauge and changing them when needed. All in all, the rain gauge will be a stand-alone system, which will take a water level and convert it to a signal that will ultimately be communicated to the researcher through the LCD on the main enclosure. Figure 4.5b below shows the block diagram for the rain sensor.

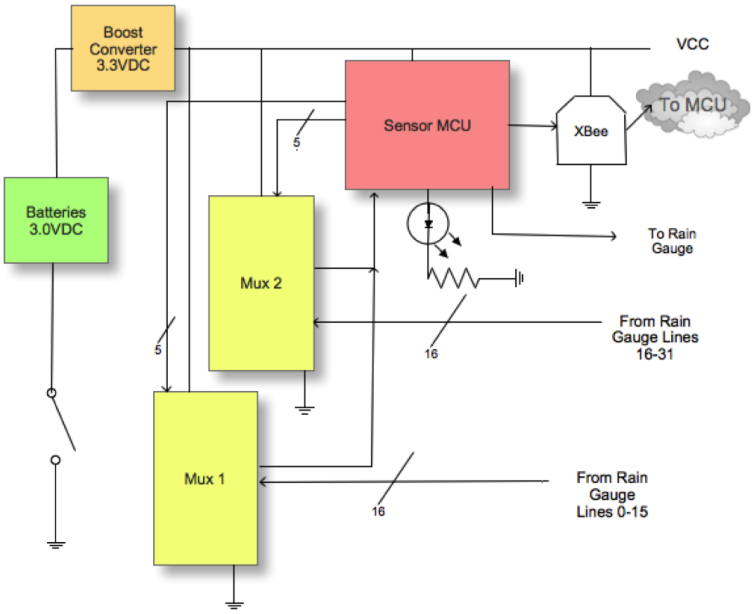

Figure 4.5b – Rain Sensor Block Diagram
This block diagram demonstrates the major features of our wireless rain sensor.

4.6: Software Design

4.6.1: Software State-Machine

State machine software implementation is an industry standard for many software applications and especially embedded systems design. It is an organized way of implementing a routine and allows for easy changes to be made if software requirements are unexpectedly updated. Figure 4.6.1a shows the main states of operation of the rain simulator machine.

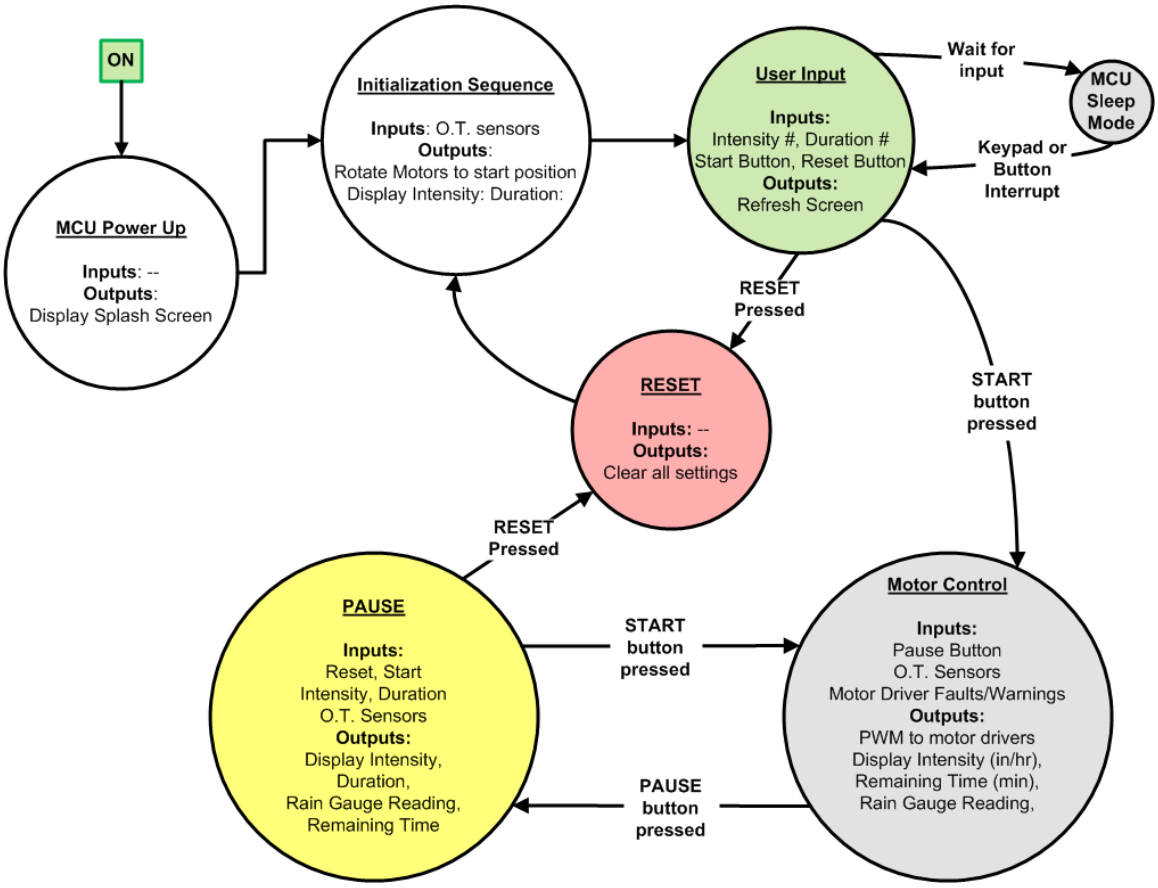

Figure 4.6.1a - Rain Simulator Software State Machine

This Figure shows an overview of the primary operating states. The inputs and outputs of each state are defined and the transitions between states are shown.

4.6.2: Reading User Input

The Figure shown below shows the flow chart for the user input routine. This is another example of a state machine. The primary state is for the microcontroller

to be in sleep mode waiting for a button to be pushed. The entire routine is interrupt driven. When a key is pressed the state changes to the “Determine Key” routine. Here the inputs determine the outputs. If an invalid key is pressed, there will be no output and the program will revert back to the waiting state. This is how error checking is handled. The rain simulator is incapable of producing 1000 inches of rain per hour, so the software will not allow that input to be entered. This is handled with Case Statements and If statements. After the inputs are complete, the software changes to the motor control state. Figure 4.6.2a shows a block diagram of the software flow for reading user input upon startup.

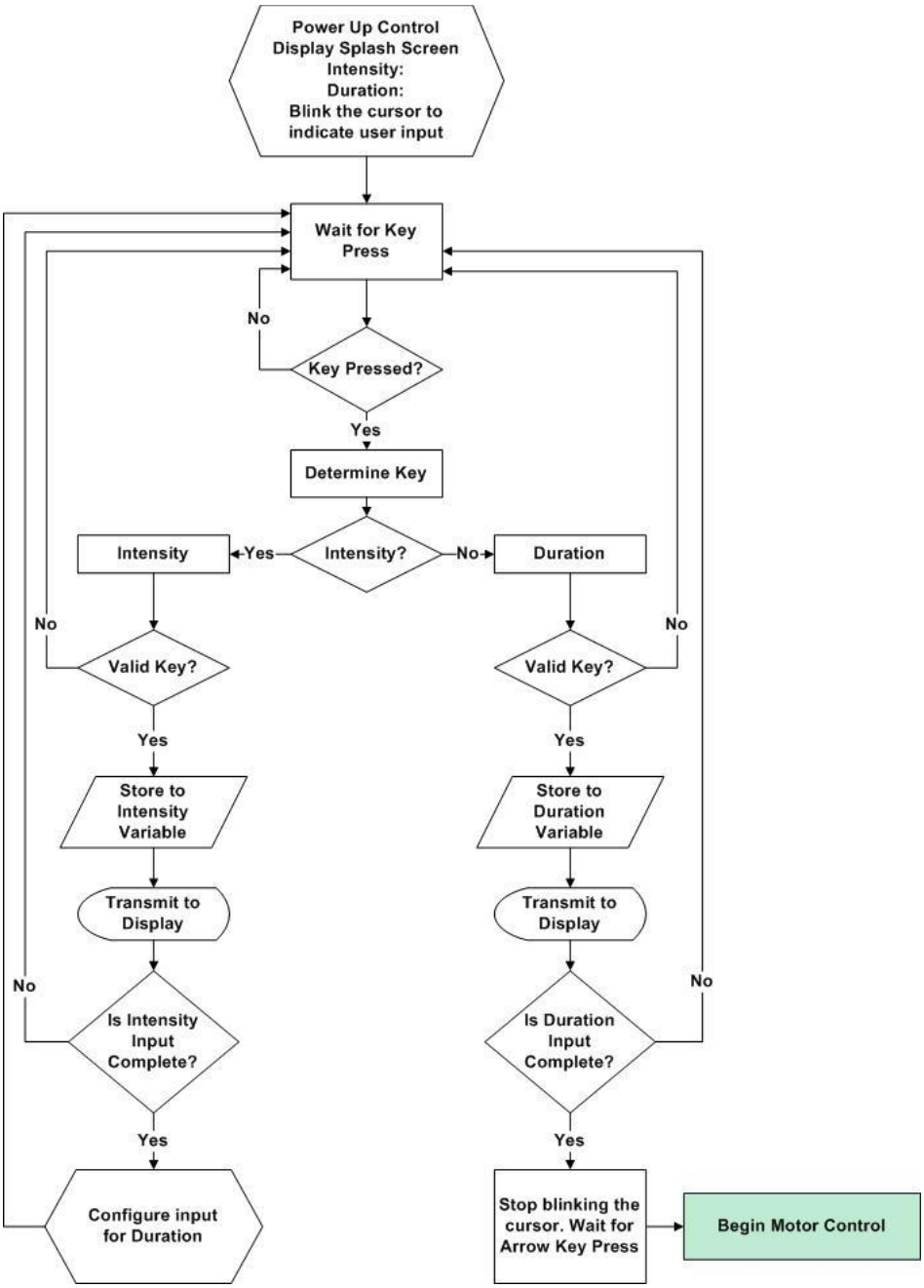

Figure 4.6.2a - User Interface Software Flowchart

4.6.3: Port Assignments

Careful consideration needed to be made in assigning the port pins to the peripheral modules. Port 1 and Port 2 are the only ports that can generate interrupts with digital input. That means that only 16 inputs can generate interrupts. Other ports have specific functions, so if that function is needed those ports must be reserved and should not be used for anything else. For example, Port 8 is the output port for one of the timers. Port 6 and Port 7 are used for the Analog/Digital Converter. Port 3 is used for the communication modules. This limits the number of general-purpose I/O ports that can be used to turn on indicator lights or other trivial things. Figure 4.6.3a shown below illustrates the Port connections that will be made for this project.

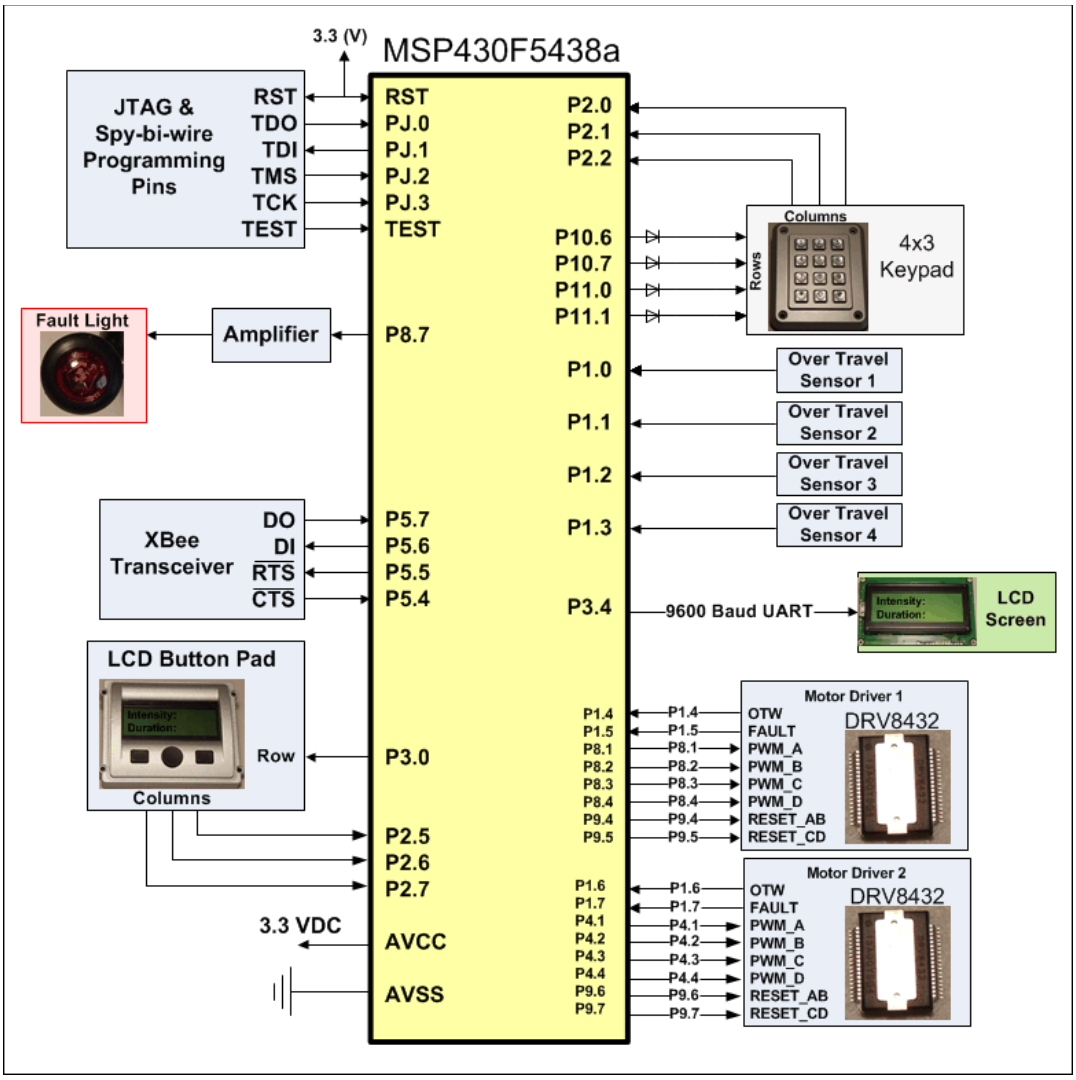

Figure 4.6.3a - Port and Pin Assignments for the Final Project

4.6.4: Motor Controller – PWM Signals

The PWM signals for the motor controllers are generated with the Timer module that was previously discussed in the Built In Timer section of this paper. The three timers in the F5435 microcontroller can generate a combined 15 outputs with their output modes. Any of these can be used to generate PWM signals. The processor only needs to write a value into one or more of the capture/compare registers of the timer, select the output mode, and start the timer. After that, the timer generates the PWM signals with no intervention from the processor. The following segment of code shows how a timer is set up to generate a PWM signal on two of the output pins. With this simple code segment, two PWM signals are generated simultaneously with the same period but different duty cycles. The timer is sourced by the 32 kHz ACLK. This is the same way the PWM signals are generated in the final project.

```
P2DIR |= BIT2 + BIT3;                  // P2.2 and P2.3 output
P2SEL |= BIT2 + BIT3;                  // P2.2 and P2.3 options select
TA1CCR0 = 128;                         // PWM Period/2
TA1CCTL1 = OUTMOD_6;                   // CCR1 toggle/set
TA1CCR1 = 32;                          // CCR1 PWM duty cycle
TA1CCTL2 = OUTMOD_6;                   // CCR2 toggle/set
TA1CCR2 = 96;                          // CCR2 PWM duty cycle
TA1CTL = TASSEL_1 + MC_3;              // ACLK, up-down mode
```

Figure 4.6.4a – Timer setup for PWM signal generation

This code segment is an example of PWM output setup.

4.6.5: Real-Time Clock

When the rain simulation is running, the processor needs to be able to keep track of the duration and update the LCD display with the time remaining in the simulation. The screen needs to be updated every minute. One way to achieve this is to set up one of the timers to generate an interrupt every second so a counter can be incremented to keep track of time. When the counter reaches 60, one minute will have elapsed and the screen can then be updated with the proper remaining time. The problem with this method is that if the processor is doing something important, the timer interrupt may cause undesired behavior. If the interrupt is ignored until a later time, then small errors may accrue in the time measurement. Another perhaps more efficient method of keeping track of the simulation time is to use the built in Real Time Clock. Since the real time clock updates its own registers with seconds, minutes, and hours, no complicated timer interrupt routine would be needed. When the simulation starts, the minutes register of the real time clock can be read and that value saved as the starting time reference. Then during the simulation, if the processor is not busy, the minutes register can be read, the new value compared to the starting value, and

the screen can be updated accordingly. This way if a few seconds goes by without reading the time, it will have no effect on the accuracy.

4.6.6: Interrupt Handling

The TI microcontrollers have a very clean way of handling interrupts. Since the microcontrollers have a countless number of ways of generating interrupts, the need arises to organize and prioritize them. Some interrupt events may be more important than others. For example, all of the pins on ports 1 and 2 can individually generate interrupts, but only one interrupt can be serviced at a time. If two inputs generate interrupts simultaneously then one of them must take precedence over the other. The way this is handled is with interrupt vectors. When an interrupt flag is set, an even number is written into the interrupt vector register. If more than one flag is set, the one with the highest priority goes into the interrupt vector register. When the interrupt service routine is entered, the interrupt vector is used with case statements to determine which action needs to take place. The following code segment illustrates the way this works with the Universal Serial Communication Interface module.

```
#pragma vector=USCI_A0_VECTOR
__interrupt void USCI_A0_ISR(void)
{
        switch(__even_in_range(UCA0IV,4))
                {
                case 0: *code*                  // Vector 0 – No Interrupt
                case 2: *code*                  // Vector 2 – RXIFG (Receive flag)
                case 4: *code*                  // Vector 4 – TXIFG (Transmit flag)
                default: *code*
                }
 }
```

Figure 4.6.6a – Interrupt Service Routine

This code example demonstrates the use of interrupt vectors.

The USCI_A0_ISR(void) statement is the name of the interrupt service routine. The __even_in_range(UCA0IV,4) command tells the interrupt service routine to use the value that is in the UCA0IV vector register only if it is even and in the range of 0 to 4. This is how TI recommends using the interrupt vector in an interrupt service routine. The case statements can either execute code directly or call a separate function. When the UCA0IV register is read it automatically resets the flag of the highest priority pending interrupt. All of the I/O pins on ports one and two and the other built in modules of this microcontroller handle their interrupts in the same way. It is a very fast and easy way to decode and prioritize interrupts.

4.6.7: Serial Communication

The LCD screen that was used in the small-scale model was built so that one pin from the microcontroller could send serial data to the LCD using the UART communication module. UART communication was previously discussed in the Universal Serial Communication Interface section of this paper. We determined that this same setup would likely to be used in the final project. The following segment of code shows how the USCI module is configured in C language for the MSP430F5435. The baud rate generator is being sourced by a 1 MHz clock signal (Sub-master clock) and the end result is a baud rate of 9600 baud.

```
UCA0CTL0 = 0x0000;              // 8 bits of data, LSB first, no parity, 1 Stop Bit
UCA0CTL1 |= UCSWRST;            // **Put state machine in reset**
UCA0CTL1 |= UCSSEL_2;           // Baud rate clock is sourced by 1 MHz SMCLK
UCA0BR0 = 6;                    // Lower Byte - 1MHz - 9600 baud
UCA0BR1 = 0;                    // Upper Byte - 1MHz - 9600 baud
UCA0MCTL = UCBRS_0 + UCBRF_13 + UCOS16;    // Oversampling and
                                           // modulation
UCA0CTL1 &= ~UCSWRST;           // **Initialize USCI state machine**
```

Figure 4.6.7a – Serial Communication
This code segment demonstrates the setup of 9600 baud UART communication.

Communication to the Zigbee transceiver that will be used for reading the remote wireless rain sensors is accomplished with the SPI (Serial Peripheral Interface). This method of communication was also discussed in the Universal Serial Communication Interface section of this paper.

4.6.8: JTAG – MSP430 Devices

All of Texas Instruments MSP430 microcontrollers have some JTAG functionality built into them for debugging, program development, and flash memory programming. The JTAG interface on the MSP430 line of devices in not IEEE 1149.1 compliant because none of the devices have boundary scan cells. Because there are no boundary scan cells, there is no BSDL file for this line of chips and the device cannot be connected in parallel as shown in Figure 2.10.1a. This parallel programming capability is not necessary for our project as we will only be using one microcontroller. Texas Instruments provides the two schematics shown in Figures 4.6.8a & 4.6.8b that describe the recommended connection for the JTAG interface for the MSP430 line of devices as well as the recommended connection for Spy-Bi-Wire for these devices. On the MSP430 devices with more pins, the JTAG interface has dedicated pins on the microcontroller while on the devices with lower pin counts, the device multiplexes the JTAG pins with other general purpose pins. On higher pin devices, the standard 4 wire JTAG interface is used with the TMS, TCK, TDI, and TDO lines.

The TMS line controls the JTAG state machine, the TCK line controls the JTAG clock, the TDI is the JTAG data input and TCLK input, and the TDO line is the JTAG data output line. On the lower pin count devices, the TEST line is set high in order to enable the JTAG function on the shared pins of the device. For the MSP430 devices that support the Spy-Bi-Wire interface, the JTAG logic internal to the device is exactly the same as the non Spy-Bi-Wire devices. The difference between JTAG and Spy-Bi-Wire is that there is additional logic that is used to convert the two wire interface into the four wire interface internally. The Spy-Bi-Wire Test Clock (SBWTCK) is the test clock line and the Spy-Bi-Wire Test Data Input/Output (SBWTDIO) is the test data input and output. To further reduce the number of pins necessary, the SBWTDIO pin is shared with the RESET pin on these devices.

The major JTAG and Spy-Bi-Wire functions available on the MSP430 devices are:

- IR_SHIFT – Shifts an 8-bit instruction into the JTAG instruction register. At the same time an 8 bit value is shifted out through the TDO
- DR_SHIFT – Shifts 16 bits of data into a JTAG register and simultaneously a 16-bit value is shifted out of the TDO.
- DR_SHIFT20 – Shifts a 20-bit address into the JTAG memory address bus register and simultaneously shifts a value out of the TDO. This function is only available on the MSP430X architecture devices.
- MsDelay – Waits for a specified time in milliseconds
- SetTCLK – Sets the TCLK to 1
- ClrTCLK – Sets the TCLK to 0
- TDOvalue – The variable containing the last value shifted out on TDO

IR_SHIFT is a macro that loads an 8 bit instruction into the JTAG instruction register least significant bit (LSB) first and at the same time, an 8 bit value containing the JTAG version identifier is shifted out of the TDO. This same value is always shifted out during an IR_SHIFT command. The MSP430 device captures each bit of the instruction register on the rising edge of the TCLK. DR_SHIFT16 loads a 16-bit word into the JTAG data register most significant bit (MSB) first and each bit is captured on the rising edge of the TCLK. At the same time, the TDO shifts out the last value in the data register on the falling edge of the TCLK. DR_SHIFT20 is only used on the MSP430X architecture devices because these devices have a 20-bit address register to address up to 1MB of memory. MsDelay makes the programming software wait for a specified time. SetTCLK sets the input clock high and ClrTCLK sets the input clock low.

The available JTAG instructions are used to program the flash memory of a device and are all transmitted LSB first:

- IR_ADDR_16BIT – Enables setting of the Memory Address Bus (MAB) to a value with the next 16-bit command using DR_SHIFT16. The MAB is set to the value written to the JTAG MAB register.
- IR_ADDR_CAPTURE – Enables the readout of the MAB.

- IR_DATA_TO_ADDR – Enables setting of the Memory Data Bus (MDB) to a value with the next 16-bit command using DR_SHIFT16. The MDB is set to the value written to the JTAG MDB register.
- IR_DATA_16BIT – Enables the setting of the MDB to the specified 16-bit value shifted in with the JTAG data access.
- IR_DATA_QUICK – Enables the setting of the MDB to a specific value shifted in the next JTAG data access. The MAB is automatically set by the Program Counter and is increment by two automatically on the falling edge of the TCLK. This is used to quickly write data to memory.
- IR_BYPASS – Delivers the input to TDI as an output on TDO.

The next list of JTAG instructions are used for accessing the microcontrollers CPU and will not be necessary for programming the device.

- IR_CNTRL_SIG_16BIT
- IR_CNTRL_SIG_CAPTURE
- IR_CNTRL_SIG_BYPASS
- IR_DATA_PSA
- IR_SHIFT_OUT_PSA

The following two commands are used for blowing the JTAG fuse and preventing further access to the device via JTAG. These commands will not be used in this project.

- IR_Prepare_Blow
- IR_Ex_Blow

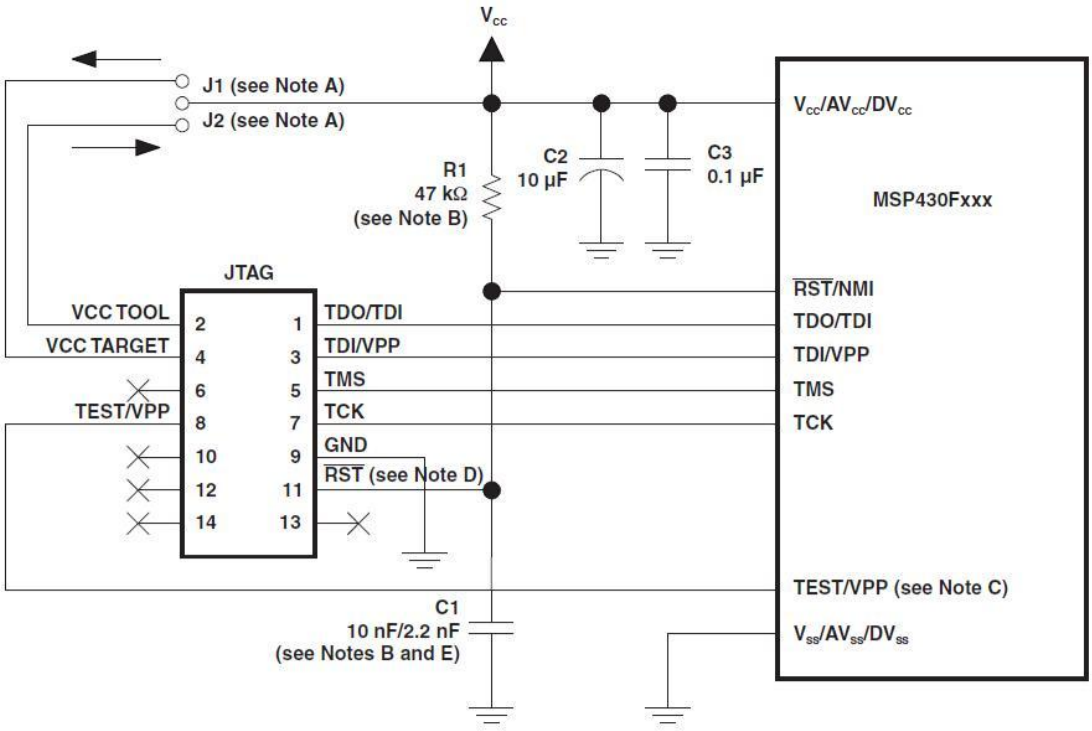

Figure 4.6.8a- Recommended JTAG connection.[A4]

This image is the recommended JTAG connection for the MSP430 line of microcontrollers.

(Permission given by Texas Instruments)

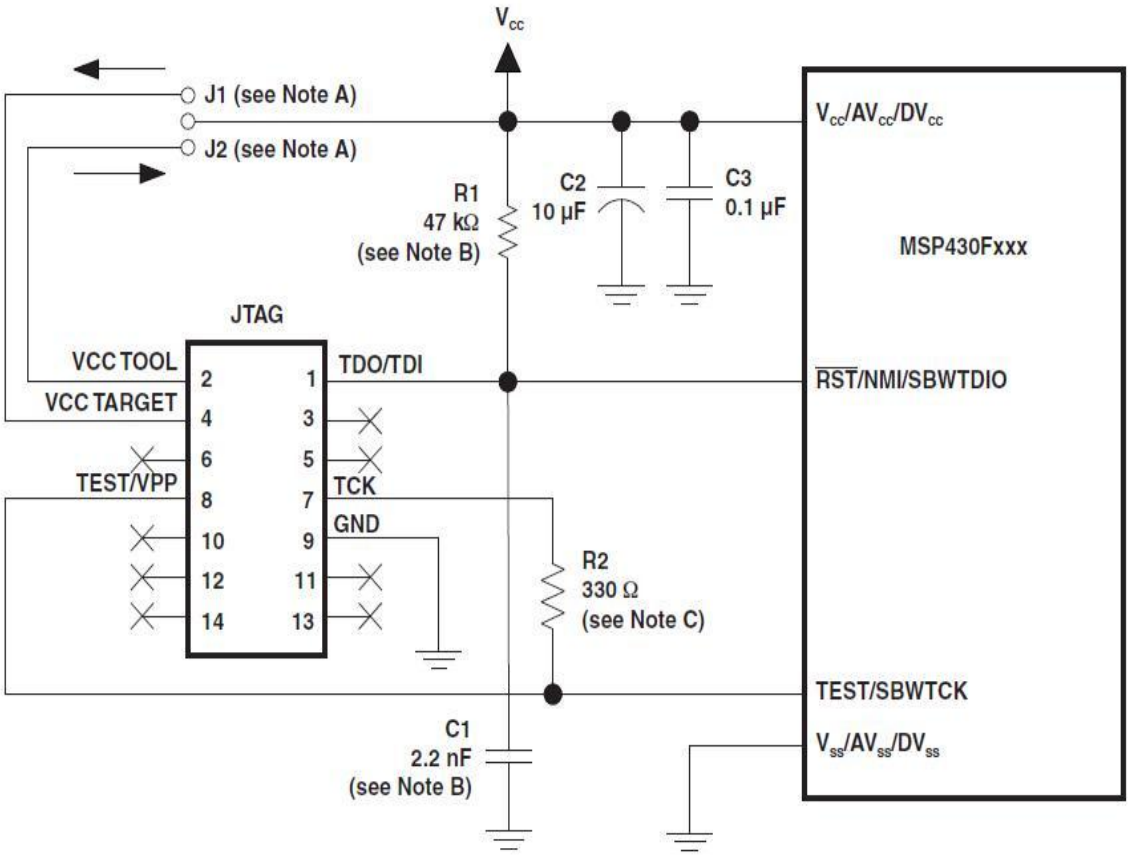

Figure 4.6.8b - Recommended Spy-Bi-Wire Connection. [A4]

This image is the recommended connection for Spy-Bi-Wire for the MSP430 line of microcontrollers. (Permission given by Texas Instruments)

The MSP4305xx devices have some additional capabilities above and beyond the other MSP430 families; however, these functions typically just remove the need for an external clock signal during certain flash memory programming. This function is automatically taken care of when using a flash programmer tool such as the MSP-FET430UIF.

4.6.9: JTAG – MSP-FET430UIF

The MSP-FET430UIF is a Texas Instruments USB flash emulation tool that is designed for the Texas Instruments MSP430 line of microcontrollers. It is a combined JTAG and Spy-Bi-Wire debugging and programming tool. The MSP-FET430UIF supports all of the MSP430 devices and is supported by a number of integrated development programs. The image, Figure 4.6.9a, shows the MSP-FET430UIF. This tool is capable of providing the necessary supply voltage to the MSP430 device so that an external power supply is not necessary. The tool also

supports the JTAG Security Fuse blow to protect the flash code inside of the MSP430 device. This particular flash emulation tool is more capable than any other Texas Instruments flash emulation tool because it has a variable supply voltage, while others are fixed, as well as being the only tool that has the JTAG Security Fuse blow capability. This particular tool is also the only one that supports both JTAG and Spy-Bi-Wire debugging and programming. The only feature that this particular flash emulation tool does not have is the Application UART. This feature is not used in the process of debugging or programming the devices used in our project. However, the ability to use both Spy-Bi-Wire and JTAG could be a very useful feature in future projects because of the potential need to minimize the use of pins on the device.

Figure 4.6.9a - MSP-FET430UIF
As shown in this image, the MSP-FET430UIF is a very simple device with a USB connection for the computer and a JTAG connection.

Using the MSP-FET430UIF to program a MSP430 device requires that the TDO, TDI, TMS, TCK, and RST lines be connected between the MSP-FET430UIF and the MSP430 device to use the programmer in JTAG mode. To use the programmer in Spy-Bi-Wire mode, only the TDO/TDI and TCK lines need to be connected. These two configurations can be seen in Figure 4.6.8a and Figure 4.6.8b. Also, as seen in the figures, the MSP-FET430UIF is capable of providing all of the power necessary during programming, or the device can be powered externally from some other source. This tool is supported by the Code Composer Studio software, which is the IDE used for this project.

4.7: User Interface Design

4.7.1: LCD – PC2004-A [A5]

The LCD that was chosen for use as the display in the project is a LCD with a 4 line by 20-character display. The viewing area of the screen is 76mm wide by 25.2mm high. Each pixel is a 5x8 pixel array allowing for alphanumeric characters as well as some symbols. This display is not capable of graphics other than what is possible with a 5x8 pixel array, but that is not necessary for the project. With the need to display up to four pieces of information this display should work perfectly for our uses. We planned on displaying the two pieces of input information needed to run the simulation, the rain intensity and the rain duration. Along with these two pieces of input, the users have requested that we also display the time remaining in the simulation. They have also requested that the display shows the number of sweeps, or if wireless rain sensors are implemented, then the measured intensity. With these four pieces of information, the users should be able to determine anything necessary about the current simulation. The PC2004-A display is back lit with a LED and requires a nominal 5-volt supply and has adjustable contrast. The operating temperature of the device is from 0°C to 50°C which is within the environmental limits of our project. The expected lifetime of the device is over 50,000 hours, which with an average run time of three hours a week is over 320 years.

This display has a narrow viewing angle both vertically and horizontally. Because of this low viewing angle, it is necessary to mount the display in an orientation that makes it easy for the user to read. Typical response time for the LCD is unknown but similar products have a typical response time of 150 milliseconds. There is no need for a fast display and a response time in this range is sufficient. The LED back light draws up to 260 milliamps at an operating voltage of 4.2 volts. The LED has an operating temperature of -20°C to 70°C which is a greater range than the screen. The nominal power supply voltage for the device is 5 volts but the data sheet recommends the voltage be adjusted depending on the ambient temperature. The LCD draws a typical current of 2.5 milliamps; well below the current draw of the LED back light. With this LCD, it is possible to read a value from the screen but this function is unnecessary for our uses. The 16 pins on the LCD and their purpose are shown in Table 4.7.1a, on the next page.

Pin Number	Symbol	External connection	Function
1	V_{SS}	Power Supply	Signal Ground for LCD
2	V_{DD}	Power Supply	+5V supply for LCD
3	V_O	Power Supply	Contrast Adjust
4	RS	Microcontroller	Register Select signal
5	R/W	Microcontroller	Read/Write select
6	E	Microcontroller	Operation enable signal
7-10	DB0-DB3	Microcontroller	For data transfer between microcontroller and LCD
11-14	DB4-DB7	Microcontroller	For data transfer between microcontroller and LCD
15	LED Positive Supply	Power Supply	+5V for LED backlight
16	LED Negative Supply	Power Supply	Ground for LED backlight

Table 4.7.1a – LCD Connections
Table showing the 16 pins on the PC2004-A, the functions of each input, and where the pins need to be connected.

4.7.2: Display Driver – KS0066U [A6]

The GMD2004D has a HD44780 compatible integrated display driver already mounted to the LCD PCB. The display driver is a dot matrix LCD driver and controller capable of displaying 1 to 2 lines of 5x8 characters. It is designed to be interfaced with a 4-bit or 8-bit microcontroller. We planned on using this code with our microcontroller to display information to the screen. www.Sparkfun.com also offers a backpack that will do a serial to parallel conversion for this LCD. However, the MSP430F5435 should have enough pins available to send the characters and commands to the display driver directly without using the backpack. The LCD driver is capable of running with a reduced number of pins by only using half of the data lines; however, this decreases the speed of the

display. There is no requirement for display speed for the project other than a reasonable speed to provide for updating the duration of the simulation every minute. Because of this we attempted to only use half of the data lines in order to reduce the number of pins used on the microcontroller.

The display driver performs the control of the display and this driver has the following instruction set available to it:

- Clear display
- Return home
- Entry mode set
- Display ON/OFF control
- Cursor or display shift
- Function set
- Set CGRAM address
- Set DDRAM address
- Read busy flag and address
- Write data to RAM
- Read data from RAM

The HD44780 display driver is originally designed by Hitachi as an LCD driver. The HD47780 has become a sort of standard for LCD drivers because of the ease of use for small displays. For simple displays, it can display information with a simple six-line connection to a microcontroller. The lines needed are four data lines, a register select line, and an enable data read line. The HD44780 compatible display drivers are very common. This was another reason for choosing this LCD screen for our project. Because of the wide use of HD44780 LCD drivers, there is a lot of free code and examples available. This enables us to quickly learn how to interface the device with our microcontroller as well and enable us to become familiar with this commonly used display adapter.

The HD44780 has two 8-bit registers, one is an instruction register and the other is a data register. The instruction register stores the instructions and the address information for display data RAM or character generator RAM. Data written into the data register is automatically written into the display data RAM or the character generator RAM. The HD44780 has a busy flag which, when set to 1, the HD44780 is in an internal operation mode and any instructions sent to it will not be accepted. To read the busy flag, the microcontroller needs to set the register select flag to 0 and the read/write line to 1. When this is done, the busy flag is output on data line 7. The address counter is used for both the display data RAM and the character generator RAM. If an address of an instruction is written to the instruction register, the address is automatically sent to the address counter and whether this address is used as a display data RAM address or a character generator RAM address is determined by the instruction. After writing into the RAM, the address counter is automatically incremented by 1 and the contents of the address counter are output to the data bus 0 to 6 lines when the register select line equals 0 and read/write line equals 1. The display data RAM

(DDRAM) stores the display data in 8-bit character codes up to 80 characters. The character generator ROM (CGROM) generates a 5x8 dot character pattern from an 8-bit character code up to 208 different patterns. It is possible for the user to define his/her own characters if needed. We do not think that we will not need this functionality of the device.

As stated before, the HD44780 is capable of receiving data in either 4 or 8 bit modes. To use the 4-bit mode, the four higher data lines (4-7) are disabled and the four low data lines are used with the four higher bits being transferred on the low lines followed by the four low bits transferred next.

The LCD that we are using is shown in the image below, Figure 4.7.2a. As seen in the image, the screen and backlight are integrated on one PCB and the LCD is controllable using the pins at the top of the device. The screen will be displaying a prompt for input of rain intensity and simulation duration. Along with these two pieces of information, the screen will display the measured rain intensity. The duration time will count down during the simulation and upon reaching 0, the simulation will end and the display will show the actual simulation run time, the average measured rain intensity, and the number of sweeps of the rain nozzles during the simulation.

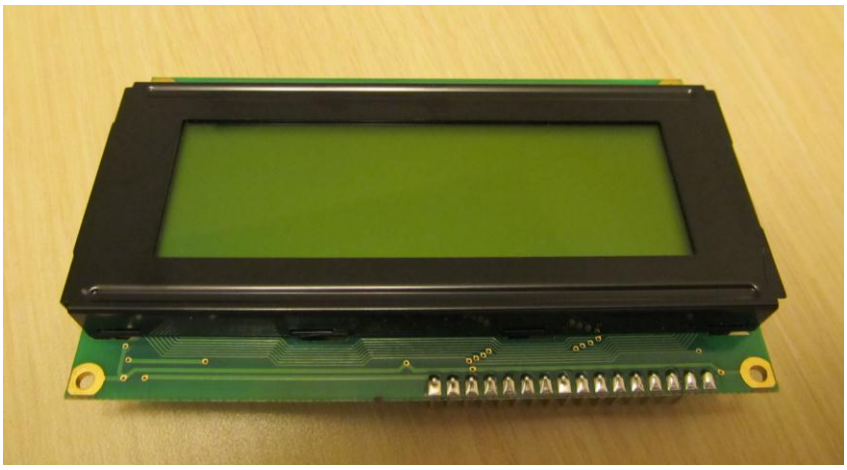

Figure 4.7.2a - LCD
This is the 4x20 character LCD that we are planning on using in the final project. As shown, it is a very simple display.

4.7.3: Water Resistant Keypad - 1K12T103 [A7]

The keypad that is needed for the input on the final product needs to be waterproof or water resistant and needs to be fairly rugged. The need for waterproofing is fairly obvious and it needs to be rugged because the users may not be gentle with it. In looking at Digikey's website, there are a few acceptable

keypads from two main manufacturers. The two manufacturers are Storm Interface and Grayhill Incorporated. The keypads from Storm Interface are typically lower priced than the ones from Grayhill Inc. while still having the same features. The one that we have chosen to be suitable for our uses and within our budget is the 1K12T103. The 1K12T103 can be seen in Figure 4.7.3a. The data sheet for this keypad says that it is designed for rugged environments and is vandal resistant. The vandal resistance is hopefully not necessary but it is an added feature that should allow for a long lifetime of the device. The 1K12T103 is a 3x4 matrix keypad with an Ingress Protection 65 rating. This rating means that the device is completely protected from dust intrusion and contact. The water protection part of the IP rating shows that the device is protected from a water jet sprayed from any direction for at least 3 minutes. This protection level from water should be well above sufficient to prevent any faults from occurring during use of the simulator at the Storm Water Academy. The casing and buttons of the keypad are manufactured with chromed zinc metal and are rated to withstand an impact of 20 Joules. The keypad buttons are manufactured with permanently engraved numbers and letters on the buttons, which should also allow for a long useable lifetime. Along with the protection from the elements, this device is a matrix style keypad, which works on the same principles as the keypad used in the small-scale model, allowing for easy transition from the small-scale model. As well as being very similar to the keypad used in the small-scale model, the 1K12T103 is very reliable with an operational lifetime of four million cycles minimum per button. This should allow for a long lifetime of the product in the harsh conditions that may be encountered at the Stormwater Academy simulation site. The buttons are guaranteed to have a maximum contact bounce of 5 milliseconds and a maximum contact resistance of 100 ohms. These are both quite low values and should simplify the software de-bouncing code. The keypad has an operating voltage of 24 VDC and a max operating current of 50 milliamps.

Figure 4.7.3a - 1K12T103 Storm-Interface Keypad
This is the 3x4 Telephone Matrix Keypad from Storm-Interface.

The method for attaching the device to the enclosure had not yet been determined but the keypad from Storm Interface can be attached to a flat service or it can be flush mounted under a panel. The benefits to mounting the keypad to the surface is that the size of the hole in the panel is much smaller, just to pass through the connection to the keypad while the flush mount requires a hole the size of the keypad itself. While the surface mounting requires a smaller hole, it may not be as aesthetically pleasing as the flush mount. Functionality is key, but appearance is always important too. In the final design we decided to use the flush mounting method.

Storm Interface has some accessories available for the keypad. The first one is the PC Interface providing an RS232 connection for the keypad. The next accessory is a set of four blank key tops, which may be useful for the auxiliary inputs such as Start and Pause. The other two accessories are the Privacy Shield and the Rear Casing, which are of no use for us.

4.7.4: Display Bezel [A9]

The original plan for protecting the LCD from the elements was to mount it behind a piece of Plexiglas or Lexan so that it is not exposed but still visible. However, Storm-Interface has been very generous and has donated a display bezel for the final product. The 5000 Series display bezel is another impact resistance device similar to the 1K12T103 Keypad from Storm-Interface. Figure 4.7.4a shows an image of the display bezel with the three integrated buttons. The bezel data sheet also states that this device is weather sealed for outdoor use with a scratch resistant, anti-reflective window. As well as being protected from the environment, the bezel is resistant to most cleaning chemicals. The bezel is designed for an operating temperature range of -20°C(dry) to 60°C which is well within the expected environmental conditions of the Rain Simulator. The bezel is designed for a 4x20 character LCD or a 122x32 dot graphic LCD screen and Storm-Interface was kind enough to include a 4x20 character LCD with the bezel. The characteristics of this LCD are discussed in Section 4.7.2.

Another great feature of this display bezel is the three integrated buttons on the face of the bezel. These buttons are in the perfect location to be used as Start, Pause, and Reset buttons. These three buttons, along with the 3x4 keypad will provide all the input necessary for the project. The three buttons on the display bezel are also set up in a matrix style with one row and three columns which will make it very easy to accept the input from these buttons with the software that was already written for the small-scale model.

Figure 4.7.4a - 5000 Series Storm-Interface Display Bezel
This is the LCD Display Bezel from Storm-Interface. This will provide the environmental protections for the LCD.

4.8: Printed Circuit Board

One of the most critical aspects of the final product given to the users is the design of the printed circuit board. The board houses nearly all of the components used in the design and without the PCB, the project would be nearly impossible to build efficiently. The microcontroller, the motor driver chips, and many other parts are only available in a surface mount package and without a PCB, we would have no way of interfacing these devices with each other. There are quite a few programs out there for designing printed circuit boards and deciding which program to use for the design is a complicated one. Just research alone would not be able to determine the best software for design as each person would have their own preferences for how the software should work. So, as part of the process of choosing the software that was going to be used, three of them were going to be evaluated. The first one of these is www.expresspcb.com and their free software. The pricing for ExpressPCB varies on the board service that you require. For the purposes of our project, the Standard Service boards are the most applicable with a small quantity of boards. This board service is very basic without a silkscreen or soldermask. The cost of a PCB with the Standard Service depends on the size of the board but for comparison purposes the price estimates will be performed with a 10in by 10in board size. ExpressPCB's pricing for two 10in by 10in boards comes to a total of $196.85. With ExpressPCB's software, they include schematic layout software and PCB design software that are able to interface with each other. The benefit with this is that you can lay out the hardware in the schematic section and when moving to the PCB design section, the pins of the parts that need to be connected together are highlighted in a different color when selected. This makes it easier to avoid errors when

connecting the different pins of the devices. However, once we began to use the software, we realized that it did not have the components that we will be using for our design in its database. We were sure that this obstacle could be overcome but it was easier for us to try the other available software suites before attempting this.

The second software suite that we tried out is from www.PCB123.com. For two 10in by 10in boards from PCB123, the total cost estimate was $467.00. While more expensive than ExpressPCB, the price from PCB123 includes solder mask and screen print. Again, the design software is available for free to the public. The first step when you open PCD123's design software is to pick a board name, board size, and the number of layers. We chose to use a 2-layer board design for our project because our PCB is not very complex and with a 2-layer board the cost is significantly lower. The next screen allows you to choose whether or not you want a solder mask and silkscreen layer on your board. For our project, we chose to use both of these. The silk screen may not be necessary for our own uses as we will know exactly which components belong in each position, but it could be useful in the future if a repair of the board is necessary. The person who might repair the board would need to be able to find the different parts on the board and a silkscreen is the easiest way of identifying the components. This screen also allows the user to choose between two different board thicknesses and the copper weight. For each of these selections, we went with the default of .062” thickness and 1oz copper weight. The last selection to make is whether or not you want your board to be lead free. We did not use the lead free construction. However, at this point we ran into the same problem as the last software. The components that we use for our design do not exist in the component database of PCB123.

The third PCB manufacturer that we evaluated was from Advanced Circuits and is available at www.4pcb.com. With Advanced Circuits, they have a special service available for prototyping circuit boards 60 square inches or less priced at $33 per board. This price does not include shipping but they do offer a minimum quantity of 1 board for students. We are also attempting to contact them as their website states that they will sponsor student projects. Once again, this manufacturer offers free PCB design software to the public. Upon opening the software, it walks the user through a wizard that allows you to select the options that you want for your board. Once the initial wizard process was completed, the user can begin to build the circuit. The software is fairly user friendly and again it does not include the MSP430F5435 in the database. Even if we decided to not use the 4PCB software, the pricing on PCBs was much lower than any other places that we had found and we planned on using them for the PCBs that are used in our final product.

The next software suite was the Cadsoft Eagle software from Newark. This is a very popular PCB design package in industry as well as with hobbyists. The only downside is that in order to use the full-featured design software, it must be

purchased. There is a Freeware version available and this is what we plan on using. The Eagle software is much more complex than any of the other PCB design and layout software but it is also much more capable. We learned that this software is fully capable of being used for designing the PCB that need. Thus, we ended up using Cadsoft Eagle for the final design of our PCBs. Cadsoft was kind enough to allow us to use the Professional version of Eagle to design our circuit boards as the Freeware version did not allow the board design size to be as large as we needed.

4.8.1: Conformal Coating- TechSpray 2106[A8]

For the purposes of our project, the silicone coating is the most applicable because of the ease of application and moisture protection. The silicone coating has the added benefits of offering physical shock as well as thermal shock. TechSpray manufactures a silicone resin conformal coating that is designed for extreme temperatures. While we may not necessarily need the high temperature protection, it is an added benefit with no added cost. This spray also has the added benefit of a fast dry time of 45 minutes with a curing time of 24 hours. It is also possible to accelerate this curing time by heating the ambient air to 120 degrees Fahrenheit for 15 minutes. While we did not expect to need the shortened curing times during prototyping and construction, it could be very useful if a repair is made to the device after delivery to the customer. This product is available in a small, 12 ounce, aerosol spray can which is great for the needs of this project. This allows the group to easily and rapidly build the circuit boards to withstand the expected high levels of humidity in the environment. Table 4.8.1a below shows a few of the conformal coatings available from TechSpray. The 2106 is the planned coating for the project. During testing we discovered that the need for conformal coating was minimized. This was due to the length of the cables allowing us to location the enclosure far enough away from the simulator that it should not receive any direct rain fall.

Chemical Number	2102	2103	2104	2106
Type of Coating	Silicone	Acrylic	Urethane	Silicone
Dielectric Strength (Volts/Mil)	1100 Dry 976 Wet	2086	380	560
Fungus Resistant	Yes	Yes	Yes	Yes
Dry Time	1 hour	15 minutes	15 minutes	45 minutes
Cure Time	72 hours	24 hours	24 hours	24 hours
Accelerated Cure Time	30 min – 90°F 45 min – 200°F	20 min – 120°F 30 min – 180°F	20 min – 120°F 30 min – 180°F	15 min – 120°F

Table 4.8.1a – Conformal Coating
Different conformal coatings available from TechSpray.

4.9: Thermal Design

4.9.1: Cooling System

During the research of the stepper motor drivers, we discovered that these devices can and will get hot due to the large current through the device. In our design, the motors will be drawing around 5.6 amps continuous which is a lot for such a small device. There are two Texas Instrument motor drivers that would meet our requirements. One device, the DRV8412 can handle motors with a continuous motor winding current up to 6 amps and the DRV8432 can handle motors with a continuous motor winding current up to 14 amps. An important specification needed for heat management is the thermal properties of the motor driver chip. In the case of the DRV8412, the junction to case thermal resistance is 1.1 ºC/W and for the DRV8432 it is 0.9 ºC/W[A1]. There is no functional difference between the two chips and while the DRV8412 meets our needs, the DRV8432 allows for simpler PCB design because it has a thermal pad on the top of the chip, while the DRV8412 has the thermal pad on the bottom of the device. Texas Instruments trademark name for placing the thermal pad on the bottom of the chip is PowerPad™. Locating the thermal pad on the bottom of device has some benefits. With the thermal pad on the bottom of the device, and soldering the thermal pad to the PCB, the PCB becomes the heat-sink and by doing this, it eliminates the need for a large heat-sink on the device which eats up precious real estate on the circuit board. Removing the heat-sink also allows for cost

savings from not needing a heat-sink on the device as well as providing for easier repair of damaged parts. These heat-sinks can be quite expensive, the heat-sink that is used with the Texas Instruments DRV8432 Evaluation Board costs between $23 and $95 per foot. In large production runs, the elimination of the need for a heat-sink could save a large amount of money. The DRV8432 device does not use the PowerPad™ technology and therefore requires a heat-sink to be mounted to the device. For the purposes of our design, there is no restriction on the circuit board size. As shown in Figure 4.9.1a, on the next page, the PCB design for a PowerPad™ device is complicated. Using this technology necessitates multiple vias in the heat-sink area as well some other additional design considerations. Because of the added complexity in design for the circuit board for the PowerPad™ devices, we have decided to avoid using them if possible.

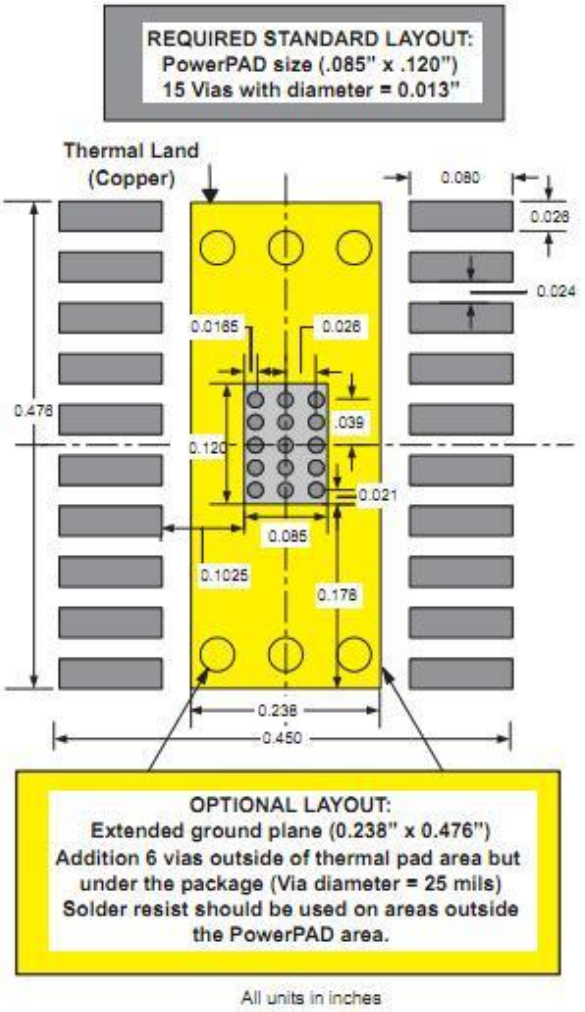

Figure 4.9.1a- PCB Layout for DRV8412[A1]
The required standard PCB layout for a PowerPad™ device. As shown, there are multiple vias required under the device.
(Permission Granted by Texas Instruments)

4.9.2: Heat-Sink

Through a conversation with Texas Instruments inquiring where to source some heat-sinks that are applicable to our device, they provided us with two free samples of the same heat-sinks that they use on their DRV8432 Evaluation Modules. Because of their generosity and the fact that this is the same heat-sink that is used for the development of these motor drivers, we felt confident that they are more than capable for our use. The heat-sink that was provided to us is 32W x 41.91H mm. For a 3 inch length of heat-sink, the thermal resistance is 4.43°C/W. Figure 4.9.2a shows a picture of the heat-sinks that Texas Instruments sent for our use. The data sheet has an important piece of information regarding the thermal resistance of the heat sink as the air flow over the heat sink increases. By forcing air over the heat-sink through the use of a fan, the heat-sink can become far more efficient at dissipating heat. Our plan was to have a fan that can exchange the air in the enclosure but does not directly push the air over the heat-sink. If it is discovered that the natural convection of heat and the removal of that heat through the use of the fan is not sufficient, more fans can be added. The group planned on using used fans from some refrigeration equipment to save on cost. These are 12-volt fans but it is possible to run the fans on 5 volts to reduce the load on the power supply. If the fans do not provide adequate cooling when run on 5 volts, then the voltage can be increased to the maximum level of 12 volts. The use of these inexpensive, commonly available fans should allow for quick replacement in the event of a failure.

The Aavid Thermalloy heat-sink can be seen in Figure 4.9.2c. There is nothing complex about the heat-sink and as shown, it is an aluminum extruded, non-anodized heat-sink.

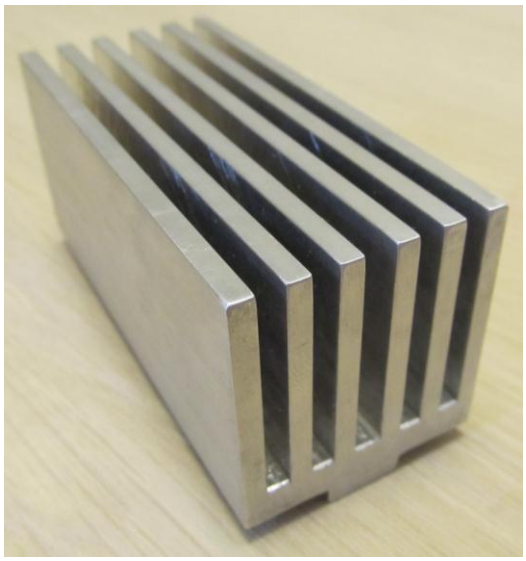

Figure 4.9.2a - Heat-sink
This is the heat-sink manufactured by Aavid Thermalloy.

5.0: Testing Procedures

The following tests provide a general approach to the testing process that will take place once the rain simulator is up and running. Many of the sub-systems are closely inter-related and the testing procedure of one may very well verify the functioning of another system at the same time.

5.1: Rain Nozzle Flow

The actual flow rate of the nozzles needed to be determined. To do this, we planned to place a five gallon bucket directly under the exposure window so that all of the water is collected in the bucket. We planned leave the nozzle pointing vertically down and stationary and run the water pump for one minute. And, then measure the volume of the water in the bucket and calculate the actual flow rate in cubic inches per minute. This value was to be compared with the theoretical value of 15 cubic inches per minute. After consulting with the group and the storm water management academy we decided that this test was unnecessary. The nozzles were already accurately tuned.

5.2: Stepper Motor Position

A problem that was observed in the small scale model was that the stepper motor had a tendency to slip under high acceleration conditions. We needed to test if this same problem occurred in the actual project. Some simple test code was written to step the motors through 90 degrees at stepping rates that correspond to the entire range of rainfall intensities. At each intensity we ran the motors for consecutive 90 degree sweeps and verified that no steps were missed. If steps were missed at any of the intensities, the linear velocity profile was adjusted in software to compensate for this.

The most difficult part of this project was running the stepper motors at high speeds. As the rainfall intensity decreases the speed of the motors increases. Because the rainfall simulator is most often run at the lower intensities, it was important that the motors run properly at relatively high speeds (600-700 steps per second). Figures 5.2a and 5.2b are Matlab simulations that demonstrate the effects of running the motors at low and high speeds.

Figure 5.2a shows a step rate of 10 steps per second with a PWM duty cycle of 6%. The low duty cycle is necessary to limit the current to the motor's rated value of 5.6 amps. Figure 5.2b shows a step rate of 330 steps per second with a duty cycle of 25%. The higher duty cycle is necessary to allow the current to rise to the rated value. The triangle wave causes reduced torque and missed steps and only gets worse as the step rate increases.

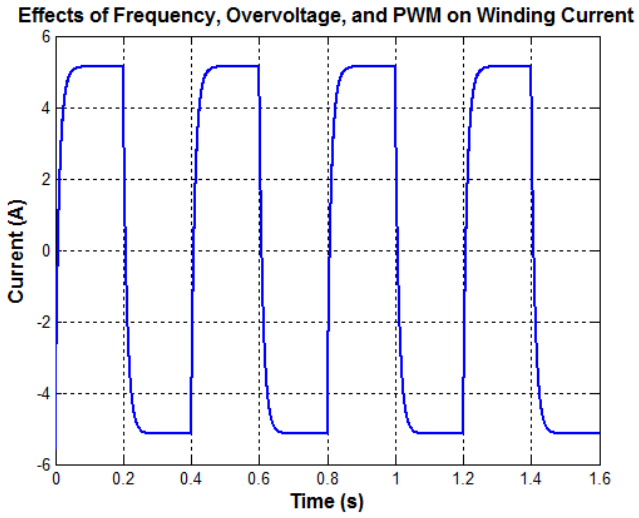

Figure 5.2a – Slow Step Rate
PWM signal with 10 steps per second and duty cycle of 6%

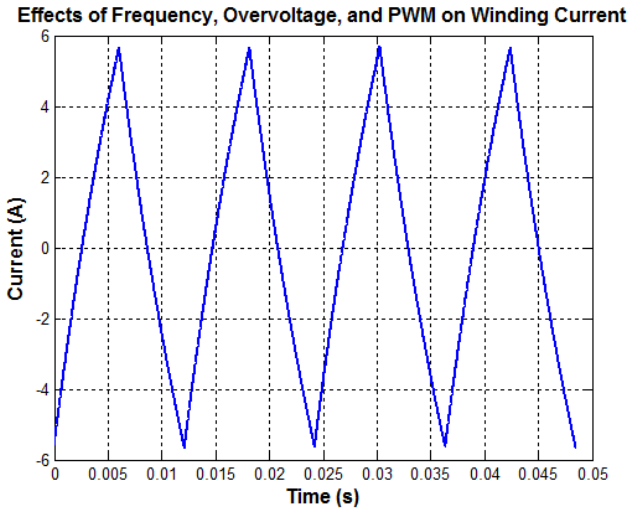

Figure 5.2b – High Step Rate
Step rate of 330 steps per second and duty cycle of 25%

To remedy the effects of the triangle wave a technique was developed to manipulate the duty cycle of the PWM signals so they would turn on at 100% for a period of time just long enough to get the motor winding current up to its rated value. That amount of time turned out to be approximately 1.5 ms. After that amount of time elapsed the duty cycle needed to be changed to 6% to keep the current at the limited value until the next step of the motor. Figure5.2c is a Matlab simulation that demonstrates the effect this would have on the motor winding current.

In reality the duty cycle manipulation technique only works to an extent. While it produces better torque characteristics, it also causes more noise and vibration in the motor. The increased vibration seems to reach a threshold amount and ultimately makes the motor unstable again. Fortunately for this project, the technique worked quite well for the range of speeds needed.

One of the motor driver development kits that Texas Instruments donated contained built in circuits for measuring the current in each of the motor windings. The circuits consisted of a small current sense resistor (0.01 ohm) followed by an RC low pass filter and an op-amp to amplify the voltage dropped across the sense resistor. The output of the op-amp could be read using an oscilloscope. Figure 5.2d is a screen shot of the oscilloscope taken during testing. This demonstrates the duty cycle manipulation technique.

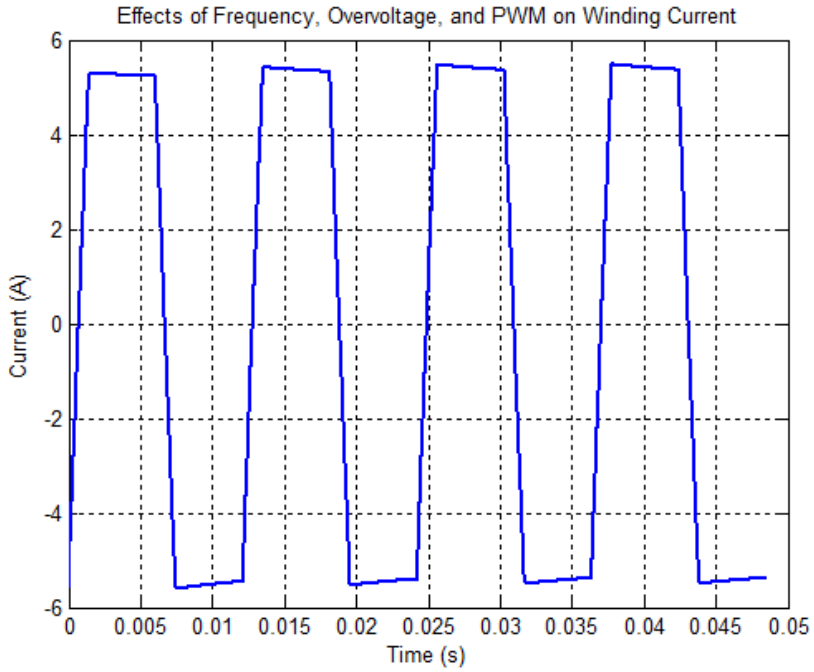

Figure 5.2c – Duty Cycle Manipulation

This is a Matlab simulation of a step rate of 330 steps per second with the duty cycle manipulation technique applied. This technique produces the desired square wave and improved torque.

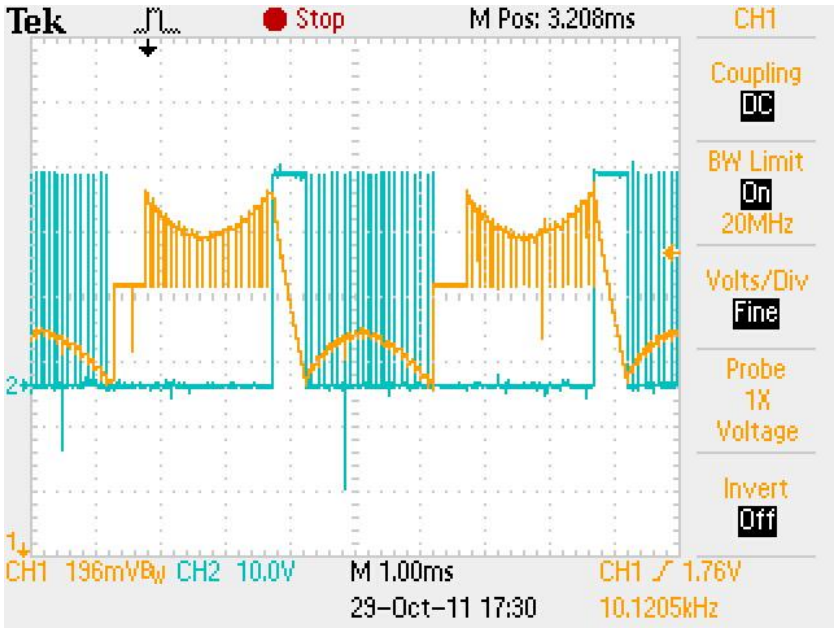

Figure 5.2d – Oscilloscope Screen Shot of Motor Current

This oscilloscope screen shot shows the current in one winding of a stepper motor at a step rate of 800 steps per second with the duty cycle manipulation technique applied. The blue signal is the PWM signal and the orange is the winding current. At the beginning of the step the PWM is set to 100% duty cycle. Then it changes to 6% duty cycle.

5.3: Rain Intensity

An integral part of our design was the wireless rain gauge. If the rain gauge did not accurately read the amount of rain from the simulator it would be useless. Thus, we needed to be sure that the gauge was accurate and consistent. After the sensor was built we put it through many tests to assure that the sensor would consistently relay the accurate level in the gauge. We verified the functionality of the sensor and found it to be consistent and accurate within 0.1 inches. This performance is better than the specification for the design. We found that the gauge would not provide an accurate reading when de-ionized water was used. This makes sense because de-ionized water does not conduct electricity. And, our design is based on the principle that the water will act as the bridging conductor for each discrete water level pin in the sensor. This does not present any problems because all of the water in the rain simulator is heavily ionized.

5.4: Software

Since most of the software is interrupt driven, we will need to verify that the external interrupt sources generate interrupts at the proper time, and the microcontroller handles them as planned. The main external interrupt sources

are the over travel sensors on the stepper motors and the integrated safeties on the motor drivers. The over travel sensors will be easy to test by writing some simple code to step the motors until the over travel sensors trigger the interrupts. The internal safeties will need to be simulated by applying a 3.3 volt source to the interrupt pin of the microcontroller. It would be too dangerous to short out the motor drivers or heat them until an interrupt is generated. The internal interrupts will be continuously tested using the small scale model.

The user input section will be thoroughly tested on the small scale model continuously as the new code is written. For every revision to any area of the code, the user input will be continuously tested and verified to be operational.

If the wireless rain sensors are integrated into the project they will need to be tested for accuracy. They will likely be updating the memory of the microcontroller through the Direct Memory Access module, so it will need to be verified that the microcontroller is reading the proper values. This will be a software test as well as a hardware test. The rain sensors will be placed on the test bed alongside the 14 manual rain sensors. After each simulation the wireless rain sensor values will be compared to the manual rain sensors. If discrepancies exist, we will need to determine if they are caused by software or hardware.

5.5: Power Supply

An important factor in the operation of every component in the control system is whether it is supplied with the proper amount of power. The voltage and current supplied to each component needed to be measured with a multi-meter. These measurements needed to be performed under full operating conditions. Some of the components that needed to be tested were the: control voltage to the microcontrollers, LCD, over travel sensors, motor drivers, wireless transceiver. We put each component of the system under full operating conditions and we measured the voltage level coming from the supply. Even under full load the power supply met the specifications.

5.6: Thermal

Many of the components of the control system have a rated operation temperature region. If the temperature within the control system enclosure exceeds the rated temperature for any of the components failure of the system may occur. The control system will be subject to ambient temperatures up to 40 degrees Celsius. Thus, the system enclosure temperature needed to be monitored with a thermometer during several tests to determine whether enough heat is dissipated. If the temperature in the enclosure reaches a critical value then modifications would need to be made to the cooling system to dissipate the correct amount of heat. Many of the components have rated operating temperatures spanning from 85 to 150 degrees Celsius. We monitored the

ambient temperature of the enclosure and the temperatures of each component while the system was under full operation and verified that none of the components overheat.

5.7: Transmission Line and Connectors

The testing of the transmission lines is another high priority. Because the transmission lines link the mechanical assembly of the stepper to our control system, it will serve as the most important communication component. Due to the fact that the rain simulator is outside, the majority of the transmission lines lie on the ground outside. This makes them very susceptible to environmental effects such as rain and trampling. The insulation on the lines need to be waterproof and hold up to any damage that could be done by people walking on or over them. Additionally, the connectors that link the transmission lines to the stepper motor enclosure and to our main control box need to be up to par. We needed to perform a full inspection on the connector heads. This inspection included taking apart the Amphenol military grade plugs and first of all confirming that there are no foreign objects obstructing the interface between the male and female connectors. We also needed to perform the appropriate cleaning with ethyl alcohol and drying of wire connections. We completed the inspection of the Amphenol plugs and repaired some of the mechanical failures in the connections.

Another important test that needed to be done was the verification that the resistance of the conductors actually equals the theoretical values that we calculate. Because the voltage signals get attenuated with high resistance transmission lines, we needed to make sure that the resistance values are not large enough to compromise the integrity of the signals traveling through them. The resistance measurement also allows us to attain the voltage dropped over the entire conductor and with this information we were able to increase the input voltage appropriately to make up for the dropped voltage on the line. We found the resistance of the transmission lines to be less than 1.0 Ohms per conductor. This was far better than we had anticipated.

6.0: System Operation

One of the main goals of this project was to make the operation of the system very user friendly. This section will describe the operation of the new rain simulator control system. The first step to be taken is to connect the control system to the stepper motors using the Amphenol receptacles located on the right side of the box. Then the box must be plugged into any standard 120 VRMS wall receptacle.

The system can now be turned on by the power switch, located on the right side of the enclosure. On initial power up, the LCD will display “Starting the Rain Simulator” followed by “Initializing Please wait…”. During this time the control

system is setting the initial position of the stepper motors. Once the system is initialized the LCD will display the user input screen. Via the keypad, the user can input the two parameters for the simulation, the intensity and the duration. The intensity input parameter is accurate within 0.1 inches / hour, with a range of 0.5 – 12.0 inches / hour. As an example for an input of 12 inches per hour the user would type ‘1’ then ‘2’ then ‘*’ and lastly ‘0’. The cursor will automatically move to the Duration input. The duration parameter is accurate within 1 minute, with a range of 1-999 minutes. As an example for an input of 15 minutes the user would ‘1’ then ‘5’. Once the user has entered a valid duration the screen will display “Start”.

To start the simulation the user simply has to press the “Start” button, which is located on the LCD bezel. The simulation will start and the LCD will display what the input parameters were for the test and the time that remains in the simulation. Also, during the simulation the LCD will display the rain level that is in the wireless rain gauge.

If the user simply lets the simulation run its course, the control system will perform the simulation for the specified amount of time. During the simulation the measured rainfall will be displayed on the screen. When the time expires the control system will display the user input screen, again. However, if the user wishes to pause the simulation during runtime the “Pause” button can be pressed. The pause button is also located on the LCD bezel. The system will pause the position of the nozzles such that they are not situated over the exposure windows on the simulator, and the test duration counter will halt. From the pause screen, the user can either start the simulation or reset the system. If the user hits the “Start” button the simulation will continue and the test duration will resume counting down. But, if the user decides to reset the system the motors will be set to their initial positions and the user input screen will be displayed one the screen again.

There is a possibility that faults may occur when the simulator is running. If a fault occurs, the fault light, which is located between the LCD and the keypad, will be activated, the simulation will halt and the LCD will display which motor driver the fault occurred in. Examples of such faults may be that the motor driver experiences an over-current condition or an over-temperature condition. If a fault does occur, the fault can be cleared by pressing the “Reset” button or by power cycling the system using the power switch. If the fault does not clear, the system may have incurred serious damage. In that case the design team should be contacted.

After the user is done with a simulation the system may simply be shut off by cycling the power switch to the off position. The system should then be unplugged from both the wall receptacle and the stepper motors. The system should be stored indoors, when not in use.

7.0: Administrative Content

7.1: Financing and Budgeting

The mission of this project is to create a successful control system for the Rain Simulation Lab at the University of Central Florida. With every great engineering project comes the need to effectively create and manage a financial portfolio. Before the project even began, a budget of $200 was decided on for the prototype. The items with the largest cost associated with them were the transformers, LCD screen, and keypad. The prototype costs will not serve as an adequate forecast for the build of the actual rain simulator control system, but it has definitely served the team well for the research and development phase of the project's prototype. From a financial point of view, it is much better to spend more time on learning the behavior of the system at the prototype level. This is because all of the components, with the exception of the main micro-controller, are smaller and cheaper than the components that we had to use for the final product. Additionally, many of the components that we were able to use for the prototype were provided as free samples from gracious companies such as Texas Instruments. This is a large contributing factor to why our prototype was able to be built under budget. The tables below show every expense made in building the small-scale prototype. The tables show how much we paid for each item including whether we were able to get that item as a free sample or donated and it also shows how much the item would have cost if this would have been a true production unit. Also, the tables include all of the components that were used in the prototype as well as all of the items that were burnt out and used up as a result of the development process. Table 7.1a and 7.1b show the expenses for the small scale prototype user interface.

Item	Quantity	Price/Unit	Total Cost	Actual Cost
2x16 LCD Screen	1	$24.95	$24.95	$24.95
3x4 Key Pad	1	$3.95	$3.95	$3.95
1.5"x1.5" PerforatedBoard	1	$0.95	$0.95	$0.95
10kΩ Resistors	3	$0.03	$0.09	$0.09

Table 7.1a: Small Scale User Interface Component Costs

Another very important component of our development was the actual motor driver circuitry. Table 6.1b depicts most of the development costs that went into the small scale prototype motor driver circuitry.

Item	Quantity	Price/Unit	Total Cost	Actual Cost
2x16 LCD Screen	1	$24.95	$24.95	$24.95
3x4 Key Pad	1	$3.95	$3.95	$3.95
1.5"x1.5" PerforatedBoard	1	$0.95	$0.95	$0.95
10kΩ Resistors	3	$0.03	$0.09	$0.09
Motor Drivers	10	$1.00	$10.00	FREE
Terminal Block	2	$0.40	$0.80	$0.80
16 PDIP Socket	1	$0.80	$0.80	$0.80
Current Sense Resistors	2	$0.50	$1.00	$1.00
Diode	4	$0.50	$2.00	$2.00
Fuse Holders	2	$0.50	$1.00	$1.00
2A/250V Fuses	5	$0.50	$2.50	$2.50
Potentiometer	1	$0.50	$0.50	$0.50
Various Capacitors	12	$0.22	$2.64	$2.64
3.25"x2.5" PerforatedBoard	1	$2.00	$2.00	$2.00
2-Pin Connector Cables	3	$0.50	$1.50	$1.50

Table 7.1b: Small Scale Prototype Motor Driver Costs

Furthermore, the development of the microcontroller system contributed to the costs of our development. Table 7.1c shows these costs.

Item	Quantity	Price/Unit	Total Cost	Actual Cost
MSP430 Development Board	4	$4.30	$17.20	$17.20
MSP430F5435 Microcontroller	1	$10.00	$10.00	FREE
80QFP Breakout Board	1	$4.50	$4.50	$4.50
2x10 Pin Female Headers	4	$0.50	$2.00	$2.00

Table 7.1c: Small Scale Prototype Microcontroller Expenses

Also, there were many various components that made up the rest of the small scale prototype. These components also came at a cost. These various components and their costs are shown in table 7.1d; also, the total cost of the entire small scale prototype and what we actually paid for is summarized.

Item	Quantity	Price/Unit	Total Cost	Actual Cost
Wood to Hold Prototype	1	$7.00	$7.00	$7.00
Wire	1	$5.00	$5.00	$5.00
Dual Voltage Regulator	1	$16.38	$16.38	$16.38
Transformer	1	$4.50	$4.50	$4.50
Various Resistors	11	$0.03	$0.33	$0.33
Stepper Motor	1	$7.50	$7.50	$7.50
JTAG Programmer	1	$75.00	$75.00	FREE
Stainless Nuts/Bolts/Washers	60	$0.05	$3.00	$3.00
5/12/24 volt Power Supply	1	$100.00	$100.00	FREE
3/14 appliance cord	1	$2.25	$2.25	$2.25
			$319.25	**$114.25**

Table 7.1d: Various Small Scale Prototype Component Costs

It was better for the team to burn out 10 free sample motor drivers, learn from mistakes made, and apply those lessons learned to the final build. This is all because the majority of the components are much bigger, they have a higher power rating, and they are much more expensive to acquire for the final build than they are for the prototype. Additionally, we found that the transformers that will be needed for the final build are much larger and they could cost at least six times as much as they did for the prototype. Again, it was necessary that we learn the complete behavior of the transformers at the prototype size so we would not exceed the power rating for one of the costly transformers needed for the final build. The biggest and definitely the most expensive component of the final build are the stepper motors. The stepper motors are the junction of where our electrically engineered control system will meet the already established mechanical portion of the rain simulator. The stepper motors currently on the simulator cost around $1200 a piece. We opted to buy a small stepper motor at Skycraft at a price of $7.50 and experiment with that instead of the full-size ones already installed at the lab site. All of the time and efforts that were put into the research and development phase may not reflect in the final build's financial statement as cost savings but they will definitely be evident as cost avoidances. Thanks to the 8 burnt motor driver chips, the 3 burned fuses, the priceless lesson

in soldering, and the over-heated transformer, we were able to learn the ins and outs of the true application and build of a control system. Once the prototype was completed, we had only used up 57% of our original $200 budget. This Figure includes the savings we accrued through using many free samples. Had we not been fortunate enough to have the availability to the free samples, we would have come in at 20% over budget with a final price of $319.25 for the prototype build. Thanks to the free samples graciously provided by Texas Instruments and other donations that we were able to obtain, we were able to build a working prototype at a price of $114.25, a little more than half of our originally budgeted amount.

For the final deliverable project, the costs can be seen in Figure 7.1e below. We were able to remain well below expected costs due to donations from Storm-Interface, Sunshine Metals, and sampled parts from Texas Instruments.

Costs for Rain Simulator				
	Item	**Quantity**	**Projected Cost**	**Actual Cost**
1	Main Board – Including Components	1	$200.00	$177.82
2	**Custom Stainless Steel Enclosure**	**1**	**$900.00**	**FREE**
3	High Voltage Power Supply	1	$200.00	$37.56
4	Wireless Rain Sensor	1	$65.00	$60.73
5	Hardware and Enclosure	1	$100.00	$136.28
6	Misc. Components	1	$100.00	$75.50
7	**Storm-Interface LCD**	**1**	**$92.53**	**FREE**
8	**Storm-Interface Keypad**	**1**	**$64.72**	**FREE**
9	**Eagle Professional License**	**1**	**$1494.00**	**FREE**
10	**Sampled Components**	**1**	**$80.18**	**FREE**
			$3,296.43	**$487.89**
Team Budget			$600.00	

Table 7.1e – Deliverable Cost
This table shows the projected versus actual costs of the final product.

7.2: Donors and Funding

Upon accepting the challenge of the rain simulator for our senior design project, the Stormwater lab did promise to assist us in our journey with some sponsorship money. We have not gotten a fixed amount that they will be giving us but we have been given a rough estimate of around $200. The Stormwater lab previously had a small group of students work on the water filtration system and it had only cost them $200 so they assumed that we would be able to achieve a control system for about the same amount. However, if we refer to the costs that have been invested into the small-scale prototype, it becomes evident that we will not be able to provide a dependable full-scale functioning system with only $200. The team has set up a meeting with the Stormwater lab team in order to present a more accurate estimate of what the final system will cost. We hope to

convince the Stormwater research team that a larger sponsorship will allow us to build a more dependable system, which will help protect their initial investment of $100,000.

The team also tried to reach out to larger companies that may be interested in sponsoring for educational purposes. One of the best sponsors so far has been Texas Instruments. They have gone above and beyond with their free samples and online assistance with specific questions. They have even sent us multiple free parts that would have otherwise cost hundreds of dollars. We have not only used their free samples for the small-scale prototype but we have already selected a few more parts available as free samples that are used in the full-scale system. Actually, the main processor we are using, MSP430F5438, was a free sample and we are using that in the full-scale implementation as well. Storm Interface also sponsored us through their donations of the LCD and Keypad. Lastly, a friend of Luke Falls, Ben Coble from Sunshine Metal Products also graciously donated the stainless steel enclosure for our project which would have otherwise cost around $900.

Our team has also reached out to other more specialized companies in hopes that they may be interested in funding our educational project. Otterbox, a company popular for their shockproof and water-proof cases for electronics, was contacted to see if we would be able to work in conjunction with them to develop a water-proof casing to hold our control system. Also, the two biggest customers of the Stormwater Research lab, the Florida Department of Transportation and Florida Department of Environmental Protection, were also contacted in hopes that they may have an interest in sponsoring the development of the control system which will be used to fulfill their future orders. We have not yet heard back from these companies, but we do hope to work with them in the future. Fortunately, a friend of one of the team members who specializes in making stainless steel casings, has volunteered to help us with our cause and will be making the casing to house our control unit. At this point in time, the sponsorship as a monetary value for the project remains undefined seeing that we have only been able to procure free items and no donations just yet. The team has agreed to divide any and all of the remaining costs evenly among the four of us. Any future sponsorship will only help reimburse the team members initial investments.

7.3: Project Planning

Once the development and research phase was completed, the actual construction of the control system was begun. Table 7.3a shows the schedule that was developed for the time remaining after the Critical Design Review (CDR).

	7-Oct	15-Oct	21-Oct	30-Oct	7-Nov	14-Nov	21-Nov	30-Nov
Replace 5v regulator on LV power supply								
Order motor driver and MCU PCBs								
Order PCB components								
Assemble/Test PCBs								
Build the enclosure								
Determine step speed/pause times								
Build/Test the rain gauge								
Final assembly								
Verify simulations								

Table 7.3a - Timeline for the remainder of the semester after the CDR.

8.0: Executive Conclusion

The main objective for the rain simulation project was to produce a brand new control system for the UCF Stormwater Management Academy. The present control system for the simulator is prone to both hardware and software errors resulting in inaccurate rainfall intensities. The current system also uses a laptop computer in conjunction with motor drivers with excessive capabilities that are not used. Through the development of the small-scale model, we have gained an understanding of stepper motor control systems. Through further research and prototyping, we have established a rugged design that can be easily implemented at a larger level. Our biggest challenge was to cater our design towards an open-loop control system that was capable of not only supplying the necessary power but also accurately positioning the motors. The most important specification in the final product was to control the rainfall intensity to within $\pm$ 0.5 inches per hour. The final design met all specs and requirements while remaining under budget and on time.