Spring 5-16-2014

SCADA Test Bed - Water Tank System

Allison Linn
Western Kentucky University, allison.linn236@topper.wku.edu

SCADA TEST BED – WATER TANK SYSTEM

A Capstone Experience/Thesis Project

Presented in Partial Fulfillment of the Requirements for

the Degree Bachelor of Science with

Honors College Graduate Distinction at Western Kentucky University

By

Allison Linn

*****

Western Kentucky University
2014

CE/T Committee:

Dr. Stacy Wilson, Advisor

Dr. Mark Cambron

Ami Carter

Approved by

_____
Advisor
Department of Engineering

ABSTRACT

A SCADA, or Supervisory Control And Data Acquisition, system is a computer-controlled system that monitors and controls industrial processes remotely. Electric grids, water supplies, and pipelines are all significant SCADA systems. These systems can be controlled and monitored remotely and are typically composed of multiple programmable logic controllers (PLCs). However, SCADA systems are inherently vulnerable to cyber attacks.

During the previous academic year, an electrical engineering senior project team was assigned the task of designing and constructing a SCADA test bed for WKU's Engineering-Manufacturing-Commercialization Center (EMCC). The test bed was to have three systems simulating SCADA systems: a traffic light system, a water tank system, and a power grid system. Because there were three systems desired, the three-person project was separated into three parts. Each person was charged with the hardware design, programming, and implementation of a system. The focus of this research project was the water tank system. This paper details the hardware and software design process that was eventually built and implemented and discusses the problems that were encountered and how they were solved.

Keywords: SCADA, SCADA Test Bed, Water Tank Simulation

I dedicate this thesis, my degree, and all things to my mom and dad.
I would not be here if the two of you hadn't pushed me to reach the potential
only you saw that I had.

ACKNOWLEDGEMENTS

To start, I must thank Dr. Stacy Wilson for her consistent help in getting me through this project. From putting up with all of my text messages to not getting frustrated when I ordered parts 6 different times because I could never decide what my hardware should look like, thank you.

To Dr. Mark Cambron, thank you for the unbiased opinion you consistently offered the SCADA Test Bed senior project team this year. Thank you also for being there when I needed to chat with someone these past four years. Above all else, thank you so much for fussing at me sophomore year when I lost sight of the end goal and stopped caring about school. I was too good to give up on engineering, but I didn't know it until you told me so. I wouldn't be writing this today had you not given me the push I needed in 2011.

To Ron Rizzo, Alonzo Alexander, Daniel Cotton, Tim Bucklew, and Amer Salihovic, thanks for your consistent help in the EMCC. I couldn't have completed my water tank system without your insight on the mechanical world.

To Ryan Peach, I would not have remembered ladder logic without you, thanks so much for helping me relearn it!

To Jon Rogers, I didn't think it was possible for engineers to be so sweet before I met you. I have never felt inferior when asking you for answers I should probably know

by now, and I am so glad to have had the opportunity to call you my machinist. Thank you for spending 6 hours of your free time on a Friday and Saturday night helping me build onto my wood foundation, thank you for making my thesis writing process a little less lonely, and thank you most of all for being a great listener and friend.

To my Papa, thank you for giving me 'the engineering speech' ten years ago and for giving it to my dad so many years before that. As of May 17, you will have officially continued an engineering legacy in the Linn family, and I am so happy to be a part of that legacy. In regards to this project specifically, thank you for being there to trash talk with me and hunting down every electrician in West Virginia to find out about the possibility of controlling a single-phase output with a three-phase VFD. I may not have needed it in the end, but I am glad to know I had your support anyway. Nanny, too, thank you for being there to chat and encourage me to keep at it all semester. I'm glad that I've been able to talk to the two of you so much more this year than any year before.

Most of all, Mom and Dad, thank you for everything. Thank you most of all for the two best gifts I have ever received: Audrey and Amy. I may not have always appreciated you four, but I can assure you I appreciate nothing more than you now and forevermore.

Last but not least, Kitty, I never knew something so tiny could make my life so much better every day. Thanks for staying up with me into the wee hours of the morning, holding my hand when I'm stressed, and letting me nuzzle your soft little belly even though you hate it.

VITA

July 15, 1992	Born-Fort Leonard Wood, Missouri
2010	William Mason High School, Mason, Ohio
2013	Summer Internship, Oak Ridge National Laboratory
May 17, 2014	Western Kentucky University, Bowling Green, KY

FIELDS OF STUDY

Major Field: Electrical Engineering

Minor Field: Mathematics

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Industrial processes require a huge amount of machinery supervision. As long as a machine is running, there is certain data that needs to be acquired from it to ensure that the process it contributes to is running correctly and efficiently. If people were in charge this, there would need to be 24/7 patrolling of the machine with constant, tedious data readings and equipment checks. Not only is this is a waste of technician expertise but it also is costly for a company to pay for a person to constantly monitor a piece of machinery. It also introduces the risk of human error to a process with inherent risks of its own. As a result of this, control systems like SCADA, or Supervisory Control and Data Acquisition, were created.

Supervisory Control and Data Acquisition

Supervisory control first came about in the 1940s when electric utility companies wanted to be able to operate remote substation equipment without needing to send personnel to the site. Control programs like this began to be implemented across several industries, reaching a peak in popularity during 1950 and 1965. Around this time, telemeter devices were developed to assist in monitoring. Telemeters record readings from a distant or inaccessible point and transmit them by radio. Telemetry was soon after replaced by computer software, which was eventually replaced by microprocessors.

Computers offered the flexibility of data acquisition over wireless communication. This was the birth of SCADA.

SCADA monitors and controls processes or machinery and communicates with the supervising personnel from a centralized location. It is made up of a SCADA central host computer, many scattered data units such as sensors, and PLCs, which are Programmable Logic Controllers. If a system begins to run abnormally, for instance a pressure gauge reads a lower pressure than it should, a SCADA system would inform the system's supervisor of the problem and give exact, real-time measurements and readings from the process. The data acquired can vary from the very simple, like reading the temperature and humidity of a climate-controlled building, to the very complex, like monitoring the flow of traffic in a traffic light intersection.

There are many examples of SCADA systems in the real world. Any industrial process in the real world can be run with SCADA systems, but they are typically found in complex industrial processes where human control is too difficult or cumbersome. A few very common SCADA systems are traffic light systems, water tank systems, and power grid systems. Traffic light systems monitor the flow of traffic in order to efficiently control a road intersection. In a water tank system, reservoir level and pipe pressure can be monitored while water flow is controlled. SCADA is good for power grid systems because there are so many different power substations and transmission towers that must all work in order for power to run efficiently. When a transmission line goes down, it is communicated to the operators who can then go to the right location to fix it.

A SCADA system is made up of two elements. The first is the SCADA Master Station Computer System (MSCS). This part of a SCADA system collects constant, real-

time data from the remote terminal units connected to it. There are two software needs for these computer systems. The first necessary software capability is to be able to repeatedly check the units. The other software must be able to retrieve that data, store it, and process it. The processing part of this would just be cataloguing the information into a user-friendly fashion, like organizing information into tables or unit conversion.

The second element of a SCADA system is the Human-Machine Interface (HMI). This is where the processed information from the host computer is organized and displayed to the human operator. This could range in complexity from simply displaying diagnostic information to producing detailed schematics and animations representing the machines being controlled. On a much smaller scale, like in our SCADA Test Bed, it could be as simple as an LED light on a small circuit indicating when there is a problem.

The supervisory control aspect of SCADA is accomplished in either of the two main SCADA elements. When a SCADA MSCS is sophisticated enough, it can read when a system's data, for example the temperature, is out of functional range and complete a task, like change the thermostat setting, to fix the problem without human intervention. If the system is not sufficiently sophisticated, the HMI can be used to control the system. In the same way that the SCADA HMI displays communication from a machine to a user, the HMI allows a user to communicate with the machine in order to control a part of the system.

The way that information gets from the system to the user is through wireless communication over a network. When SCADA systems first emerged, all communication was made in isolated environments that rarely shared information outside the process' own environment. Now it is far more common for Internet-based communication to be

the choice of companies that utilize SCADA systems. Larger corporations especially need to share valuable data across larger distances. Because a SCADA system's information is transported using telemetry techniques such as telephone lines, cable, radio, and fiber, it is possible for unauthorized people to access these mediums. Sharing information over such a large and accessible network presents the risk for someone to hack into a SCADA system in order to steal information or maliciously control an industrial process. This type of malicious hacking would be referred to as a cyber attack.

SCADA Cyber Attacks

A cyber attack is defined as "the deliberate actions (perhaps over an extended period of time) to alter, disrupt, deceive, degrade, and/or destroy computer systems or networks or the information and/or resident in or transiting these systems or networks" (Tsang 2). There are three types of cyber attacks: intentional targeted attacks, unintentional external attacks, and unintentional internal attacks. Intentional targeted attacks occur when someone purposely gains unauthorized access to the computers within the SCADA system. The goal of this kind of attack usually occurs to gain sensitive information from the system or to harm people utilizing the system. Most of these attacks are caused by people with inside information because performing such an attack requires in-depth knowledge of the system and line of communication. Unintentional external attacks are consequences that occur as a bi-product of another virus or control system failure. Unintentional internal attacks are consequences of internal personnel's unauthorized actions. Detailed below is an example of each.

The most well-known SCADA system attack occurred in January 2000. It was an attack on the Maroochy Shire Council's sewage control system in Queensland, Australia. When the control system was installed, pumps refused to start and stop, alarms would go off and not be reported, and the control center lost communication with the pumping stations several times. This went on for a while under the operators' assumptions that there was a leak somewhere in the pipes. Even when valves were opening without commands to do so, they did not question what was going on.

The unauthorized opening of the valves caused around 264,000 gallons of raw sewage to be flooded into the ground of a hotel, park, and river (Tsang 6). Several months of logging the problem went by before it was discovered that the valves were activated by controllers that were not authorized by the system operators. It was discovered much later that a disgruntled ex-employee had hacked into the system in order to cause problems and get his job back under the pretense that he could fix the problem.

An example of an unintentional external cyber attack occurred on August 14, 2003 (Tsang 6). An electric power blackout swept through significant portions of Midwest and Northeast United States and Ontario, Canada. In Ontario, rolling blackouts happened for an entire week. In some places in the United States, it took 4 days for power to be restored, costing somewhere between 4 billion and 10 billion dollars. The blackouts were a result of several environmental interferences with important power lines. The operators were not being alerted to the location of these interferences because of an alarm failure in a processor that was designed to alert operators of critical operational changes in the system.

An example of an unintentional internal cyber attack happened in June of 1999 (Tsang 13). 237,000 gallons of gasoline leaked from a pipeline into a creek flowing through Bellingham, Washington. The gasoline eventually ignited and burned 1.5 miles along the creek. The fire killed 3 people, injured 8 people, consumed a home, and damaged the city's water treatment plant. The total cost of damages was $45 million. The pipeline failed due to poor communication to the controllers because operators would run tests on the SCADA system while it was operating, leading it to be unresponsive during the pipeline failure.

Because SCADA systems are becoming larger and more complex, they are being connected to corporate networks and the Internet. This opens up the ability for systems to be hacked through uncontrolled connections like mobile devices or dial-up connections. The systems are also almost always run with microprocessors. Microprocessors allow for digital communication, but introducing a more complex digital component to a system allows for a system with a higher risk of flaws. If these flaws were found, those attempting to gain access to a SCADA system could exploit them.

Possible attacks could come from "backdoors and holes in network perimeter, vulnerabilities in common protocol, database attacks, and man-in-the-middle attacks" (Tsang 13). Due to SCADA system modems, network perimeters can be accessed easily by those trying to troubleshoot or repair a system. Without proper implementation of security, it would not be difficult for an unauthorized person to access the system through a backdoor entry and gain control of a system's controls. SCADA systems use protocols to communicate between the control devices. Vulnerabilities exist in these protocols because they were designed with no security and no authentication requirements to make

commands to the system remotely. A person gaining access to one of the control devices would not need to authenticate their authorization to control the system. A very well known type of attack is referred to as a man-in-the-middle-attack. A hacker will gain control of the communications in a system and talk directly from one half of the system to the next, making each believe that they are talking in a secure channel, when they are in fact reading a conversation designed by the hacker (HarGrove).

These attacks could obviously be very detrimental to a large area of people if a terrorist gained access to a water supply and contaminated it or caused a gasoline pipeline to leak intentionally. There are a few reasons why large-scale attacks like this have not occurred. First of all, it would take a highly-trained coder with insider information to gain access to SCADA control systems because detailed knowledge of the control system is absolutely necessary. Secondly, it takes a long period of time to be able to perform a remote attack. It would take a highly skilled coder at least 6 months to reverse engineer a single SCADA control center network because they are all so different and control so many different things. Finally, it would take a large tolerance for failure. Without trying it, there is no way to know if an attack will succeed or not. If it doesn't succeed, all the time would have been spent coding and hacking the system only for the victim to set up a stronger security system. Given a proper amount of skill and time devoted to an attack, a terrorist could cause large-scale devastation with a single attack, so it is extremely important to study unauthorized infiltration possibilities and adequately secure SCADA systems.

CHAPTER 2

DESIGN PROCESS

In order to properly study SCADA system vulnerabilities and attempt to secure them, it is often beneficial to create a SCADA test bed where new security systems and potential attacks can be designed and simulated. It was the desire of the WKU Engineering-Manufacturing-Commercialization Center to set up such a test bed. The SCADA test bed has three separate systems that simulate SCADA systems in the real world: a traffic light system, a water tank system, and a power grid system. The focus of this research is the water tank system.

Problem Statement

A basic water tank system shown in Figure 1 had been created with two main hardware components, a pump and a tank, prior to the beginning of this research project. The minor hardware components include barb fittings, tubing, and a four-legged aluminum base to raise the tank above the pump. The system is gravity-fed so that the water pressure and gravity are the only forces that pull water from the tank and through the pump. When connected and plugged

Figure 1: Hardware components of original water tank system

in, the pump cycles the water directly from the bottom of the tank and pushes it back to an entrance in the top of the tank. Several changes were needed in order for the water tank system to simulate a SCADA system.

In order for the system to simulate a SCADA system, there had to be a variable that was monitored in the system and a variable that was controlled. Minimally, for the water tank system to resemble a real water tank system, the water level must be monitored, and the flow of water from the tank must be controlled. It was determined that a float sensor would be best for monitoring water level on a small scale. The monitoring and controlling capabilities of the system had to be designed and introduced.

Hardware Design

The hardware was the first aspect of the system to be designed. There needed to be at least four main additions to the main hardware components. The first was a PLC. A PLC allows monitoring of the inputs from the system and, based on their readings, control outputs. Another main component needed for the system was a reservoir. The reservoir represents the water source for a water tank, which in a real water tank system is a water treatment facility. The reservoir did not need to be complicated or built like the water tank, but it did need to be airtight and sturdy. A Sterilite Ultra-Seal 16.2 cup storage container was chosen (Space Saver). The container is airtight, had latches to keep it closed, and had a steam release vent that conveniently will serve to relieve unwanted pressure from the system.

The float sensor was then included to monitor the input. Originally, the plan was to have a single float sensor maintaining ‘closed’ position when the water tank was

considered full. After a lot of thought, it was decided against that one float sensor was insufficient. If the tank had only one float sensor, the water tank would drain when the float sensor was full based on a timer and refill after the timer was completed. This introduces the risk of an inconsistent filling or draining time due to varying water flow rates. In order to keep the system from relying on a timer, a second float sensor was incorporated into the design.

To control the water flow from the water tank, two pieces of hardware are required. The first is a valve, located at the bottom of the tank, which can be closed when it is necessary to fill the tank with water and open when it is necessary to drain it. The type of valve needed to be able to maintain electrical control in a 24V DC system is a solenoid valve. The other output control unit is the water pump, which was in the original

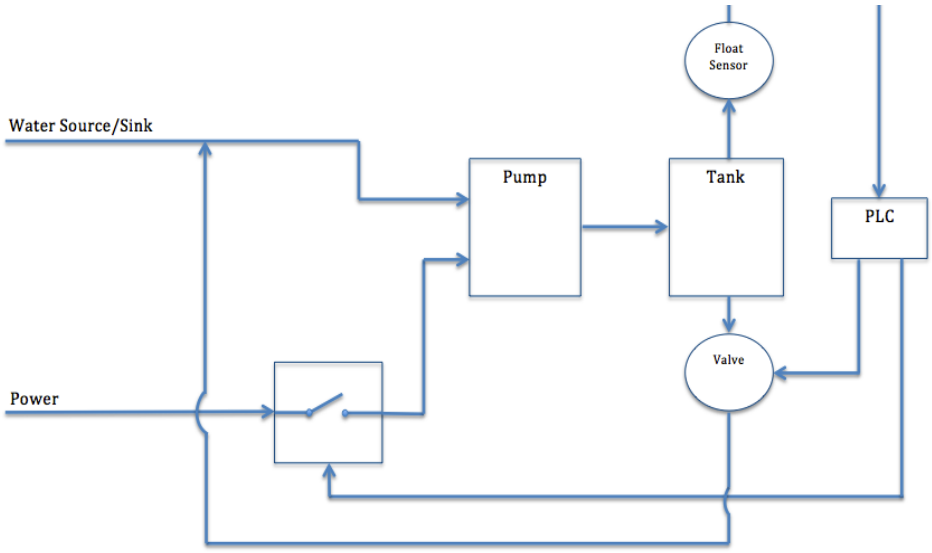

Figure 2: Hardware flow chart

system. The water pump will fill the tank with water from the reservoir when the valve is closed.

The hardware flow chart in Figure 2 shows the interaction between the hardware components of the water tank. The Source/Sink Reservoir (SSR) feeds water to the pump, which feeds water to the tank. The water will then fill to Float Sensor 1, which is located at the top of the tank where it would be considered full, or drain to below Float Sensor 2, which is located at the bottom of the tank where the tank would be empty. The PLC will then read the float sensors and control the valve. When the valve is opened and the pump turned off by the PLC, it will drain water from the tank to the SSR. When the valve is closed, the pump will be turned on and fill the tank with water.

Because a reservoir was added to the system, it was necessary to make significant changes to the hardware as shown in Figure 3. The first action item was to get holes installed in the SSR. An input hole was needed to hold the barb fitting so the tank could drain into it. An output hole was needed to hold a barb fitting so the tank could supply water to the pump. The first hole was not very precise however the only criterion was that the hole be in the lid and away from the steam vent. The second hole was a little trickier. In order to maintain a gravity-fed system, the hole needed to be taller than or at the same height as the input of the pump. The pump input is 2.5" off the ground, so the hole in the SSR needed to be

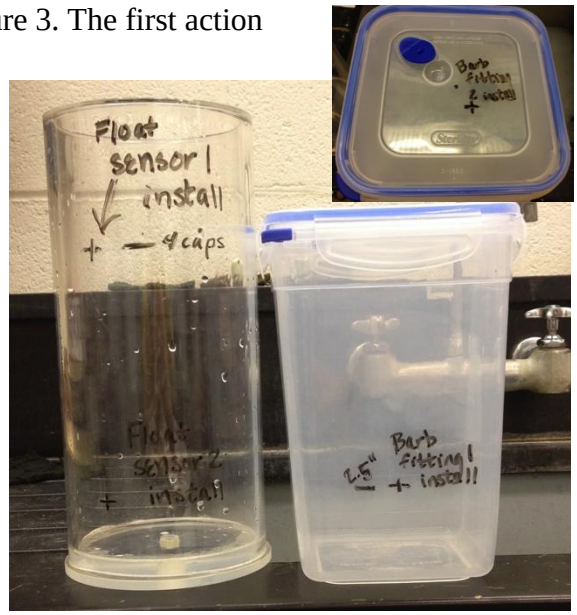

Figure 3: Hole installation sites in water tank and SSR

installed at least 2.5” from the bottom. Because the SSR is the water tank’s water supply, it was important not to install the barb fitting any higher than necessary as to have as much water going through the system as possible. For this reason, the hole was installed exactly 2.5” from the bottom and in the center of the container.

Once the holes were installed for the barb fitting in the SSR, the holes were also installed in the water tank where my float sensors would be installed. Because the SSR minimum water level was 2.5” from the bottom, the SSR was filled with water to the 2.5-inch line, and then that water was poured into the water tank. Then the height of that water line was measured and marked from the top of the tank down. On this line was where the hole for the Float Sensor 1 installation was marked. Therefore Float Sensor 1 could be installed at the maximum water level line because no more water should be removed from the SSR than that. The Float Sensor 2 hole was marked just high enough that the sensor would be able to drop its indicator when the tank was empty.

The next design step was to figure out how to attach the SSR to the wood foundation. It was decided that to hold the SSR in place, only four wooden lips would be needed as shown in Figure 4. By making each of the wooden pieces an inch longer than the SSR’s width, the four could be arranged into a square that would wedge the SSR into place. The SSR now stays in place, but it can be removed if necessary.

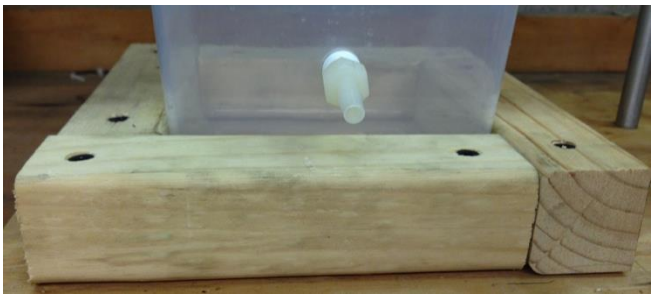

Figure 4: Wooden lip holding SSR into place

With the addition of the SSR, the height of my water tank needed to be raised in order to still be conducive to a gravity-fed system. Aluminum rods of the same width as

the original rods (0.5”) were added to the pre-existing aluminum base plate. The four aluminum rods that were ordered were 24” each. The SSR is 9.25” tall; the barb fitting coming from the top of the SSR adds 1”. The solenoid valve and all of its components made a combined length of 5.75”. The total height for those pieces was 16”, which means that the water tank must be suspended at least 16” to maintain a gravity-fed system. In order to make up extra space for tubing, an extra 2” was added to the measurement and the aluminum rods were cut to 18” tall.

In order to attach the aluminum rods to the wood foundation, I had the rods drilled and threaded vertically about 2”. The underside of the wooden base was then marked by tracing the outside of the rods and marking the center. Countersink holes were added to the base of the wood foundation where rod center locations were marked to make screws flush to the base as seen in Figure 5. The new rods were then screwed into the new wood foundation. After that, the rods were attached to the aluminum plate. From there, it was necessary to attach the solenoid valve to the water tank and install the piece that will hold up the valve. The originally plan was to design and have built a U-shaped bracket in order to suspend the valve from the aluminum baseplate, but after significant research, a ‘nonconductive routing ring’ was found, which measured the perfect length and could be used for suspending the valve. Routing rings are usually used to suspend or contain large amounts of wires. Because the valve is relatively lightweight, a routing ring worked for this design purpose.

Figure 5: Counter sink holes plan

After the water tank, valve, SSR, lip pieces, and tank base were installed; the pump was screwed down to the wood foundation in front of the SSR. All hardware components were then connected with 0.375” tubing. To the left of the pump, a previously designed switch-housing unit for the control switches was installed and is shown in Figure 6.

When all other pieces were taken care of, a relay was installed to the pump. A relay was necessary in order to be able to manually turn on and off the pump with switch 1. If a relay were not used, the pump would be running as long as it is plugged in. Because the pump runs on AC power, the relay needed to go from AC power to 24V DC power, which is the type of power the PLC runs on. AC power has a lot of regulations because it is dangerous to leave AC power hardware unprotected. Because the system contains water, it was important to protect the relay. For this reason, a NEMA 12 rated housing for my relay was purchased and installed. When mounted onto my foundation, the housing protects the relay from water splatter and keeps the wires contained and away from user interaction.

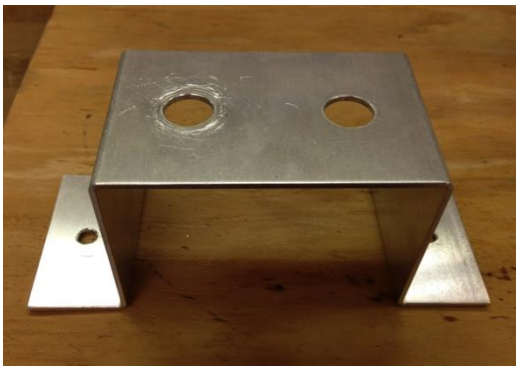
Figure 6: Switch housing unit

Software Design

PLCs are controlled by ladder logic. RSLogix 5 was used to write the code for the PLC being used, which is a MicroLogix 1000. The code simply needed to read the two inputs, the float sensors, and control the two outputs, the pump and the valve. Two

control switches were also added to the system to be able to control when the system was on and which mode it ran in. This is a change from the original system that ran the whole system in a cycle as long as the pump was plugged in.

As is visible in the programming flow chart shown in Figure 7, the PLC would first read the combination of switches open. The first switch (S1) is the system on/off switch. When Switch 1 is off, the PLC will do nothing with the outputs and just keep checking the switches. When Switch 1 is flipped on, the PLC will read Switch 2 (S2) to determine which action is to be made. If the second switch is open (in the off position), the system will run in a constant cycle of draining and filling. It would look for the closing of Float Sensor 1 (FS1) or Float Sensor 2 (FS2) to determine the current water level to decide whether to begin draining or filling the tank. If FS1 is closed, the system is drained by opening the valve and turning off the pump until FS2 is open, which means the tank is empty. It will then close the valve and turn on the pump to fill the tank and repeat the cycle. If the second switch is flipped, the tank will be drained and stop.

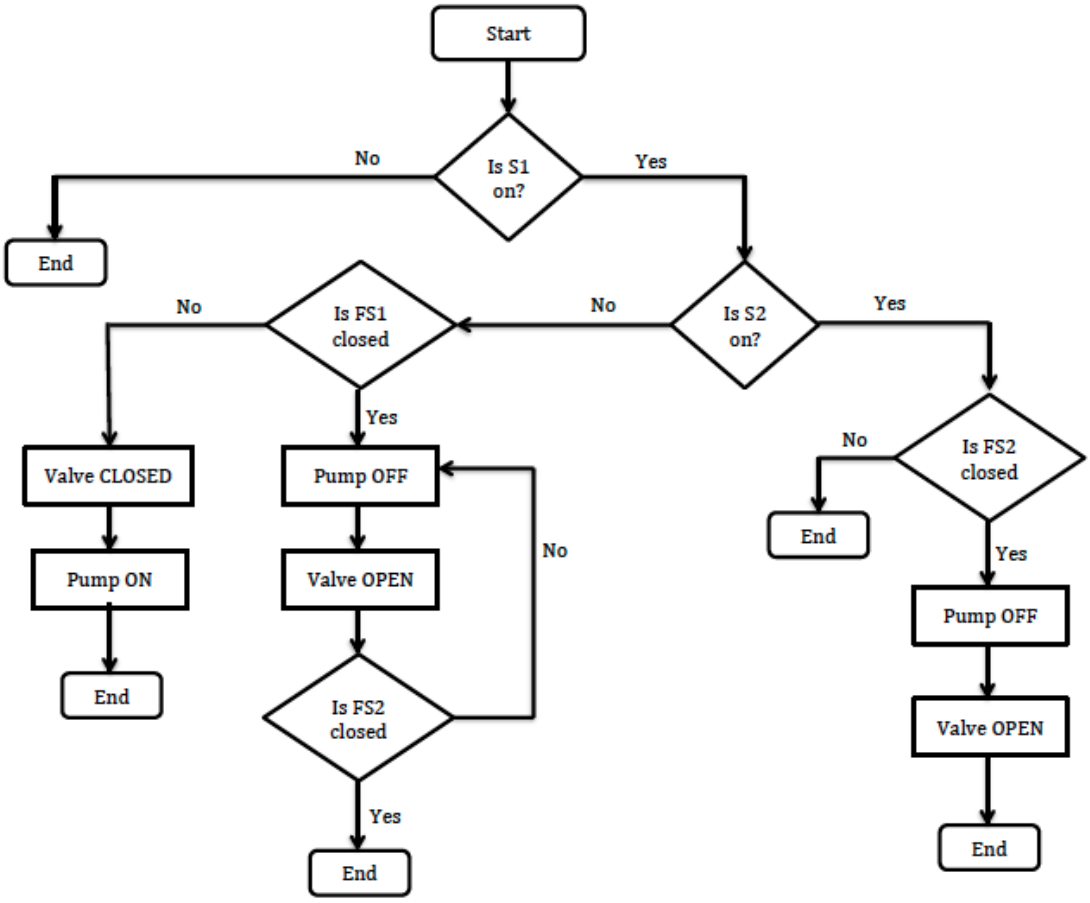

Figure 7: Programming flow chart

CHAPTER 3

CHALLENGES AND SOLUTIONS

Most of the design challenges came from the hardware portion of my project, which will be discussed below.

Solenoid Valve Choice

From the beginning of this project, it was well known that a valve would be needed in the system that can be controlled by the PLC. When a valve was finally found that was 24V DC, the valve size that matches the tubing size was ordered. The tubing has an outside diameter of 0.375", so a 3/8" solenoid valve was ordered. When the solenoid valve arrived, it was realized that the valve shouldn't have been ordered based on the tubing size but instead based on the barb fitting size. The barb fitting size was 0.25". The reason that a 1/4" valve should have been chosen is because one side of the valve screws into the base of the water tank and the other side of the valve has a barb fitting screwed into it that will be connected by tubing to the top of the SSR. Had the correct size valve been chosen, the valve would have been much smaller and not nearly as heavy.

Because the wrong size valve was ordered, reducers were needed for both sides of the valve to get the piping from 0.375" to 0.25" to fit a barb fitting in one side and connect to the tank with the other. Figure 8 shows the solenoid value with the extensions and reducers. On the side that connects to the tank, a reducer was found that did just that

(B). It added a bit of length, which originally was worried a concern, but when the nipple pipe was added, it was too short to allow for space for the valve wires under the aluminum baseplate. In order to make it long enough to compensate for the wires and long enough to be suspended by the routing ring, a 2” nipple pipe (A) was used.

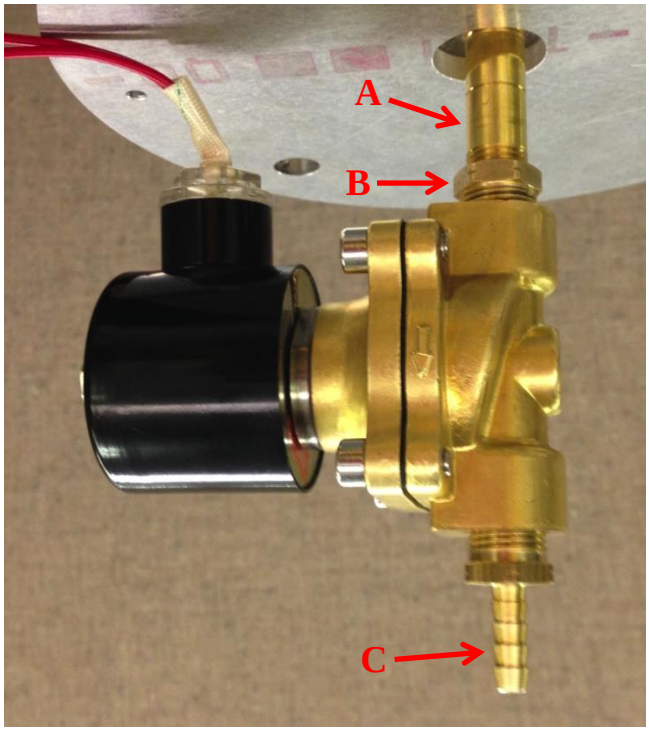

Figure 8: Solenoid valve with extensions/reducers

One of the goals of the water tank base was for it to be as short as possible. The concern of a tank system that was too tall was that it introduces a risk of unbalanced weight in the system that would make is susceptible to tipping over. Because of the necessary reducer and nipple pipe length, length was being added to the already lengthy valve. To compensate for that, the new goal was to find a reducer that would take up as little space as possible. When the reducer was added to the plastic barb fitting from the original system, it added an extra inch to the valve, and meaning that the aluminum rods would need to be 19” tall. A reducing piece that connects a 0.375” pipe to a 0.25” barb fitting (C) was found and connected to the valve. With all of the pieces connected, the valve totals 5.75”. This saved an inch from the aluminum rods and caused them to be 18” long instead of 19”.

Float Sensor Installation

After the holes were installed in the water tank, the float sensors needed to be installed. The problem was, though, that the water tank is cylindrical and the float sensor installation parts were for a tank with a flat wall. When I attempted to install the float sensors, the rubber washer that came with the sensors would not lay flush to the tank wall because there was no pressure from the circular sensor piece to keep it sealed as shown in Figure 8. This caused it to leak water from the installation site. In order to solve this problem, waterproof caulk was used as shown in Figure 9. After curing for two days, the tank was filled with water, and the installation sites had no sign of water leakage.

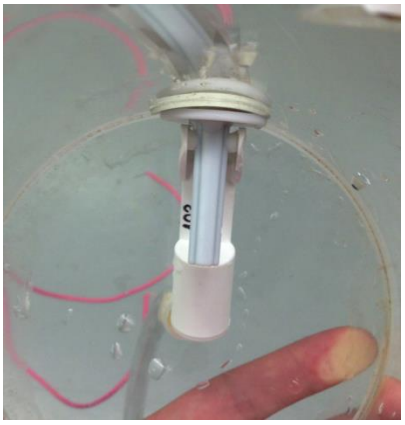

Figure 9: Float sensor installed with included installation parts

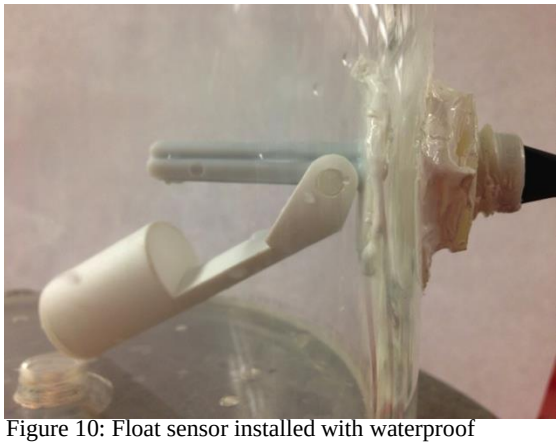

Figure 10: Float sensor installed with waterproof caulking

Aluminum Baseplate Eats Allen Wrench

The aluminum rods holding up the tank are attached to the tank's aluminum baseplate with four tiny screws like the one shown in Figure 10. When an aluminum rod is inserted vertically through the 0.5" hole in the baseplate, it is held into place by a screw installed horizontally that, when tightened, will put pressure on the rod and keep it in place. When the aluminum rods were cut to size, polished, and ready to install, it was

time to put the rods into position and tighten the screws. Two of the four rods were at even heights and the third was being tightened with the Allen wrench when the tip of the wrench broke off into the screw. There was absolutely no getting the wrench tip out. The problem was then realized that it had been tightened to a position that was just a bit too tall, and the opposite rod couldn't be lowered without making the rod go below the baseplate surface. In order to compensate for the missing height, a washer was added in between short rod and the wood foundation to add just the right amount of height.

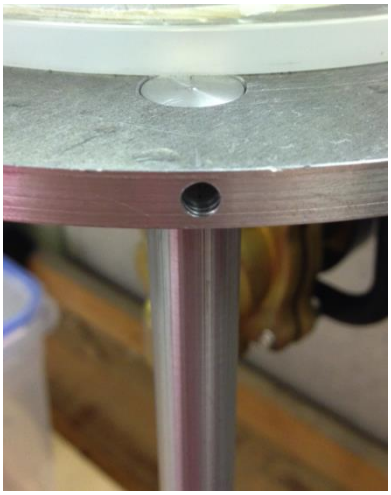

Figure 11: Site of rod-securing horizontal screw

Ladder Logic Looping

Ladder logic is different from other programming languages because there is no explicit way to create a loop inside a loop. PLC code cycles constantly from top left to bottom right while the code is running. The problem, though, is that one cannot program a line of code to complete only after a task inside the line has been completed. For instance, in order to drain the tank, the valve needed to stay open and the pump to turn off after FS1 has been closed and then close the valve and turn the pump on again once both FS1 and FS2 are open to refill the tank. In a single line of code, the PLC would read FS1, open the valve, and turn off the pump when FS1 is closed. The problem then became evident that as soon as the water level drops a few millimeters, FS1 will open again,

causing the pump to turn back on and the valve to close. The water level would then constantly be stuck in the few millimeters above and below FS1.

If a PLC ran on a traditional programming language, it could be programmed to check when FS1 is closed and open the valve until FS2 is open. In order to accomplish this in ladder logic, it is necessary to call a bit and implement a latch. Just like outputs and inputs in PLC code, a programmer can call bits and designate those bits to an action. Telling a bit to latch simply means that the bit will be high and remain high until it is unlatched. In order to accomplish the tank's fill and drain cycle, three bits needed to be created: a valve bit, a pump bit, and a full bit. The code told the PLC to energize the valve (or open it) and to unlatch the pump bit when it reads that the valve bit is high. The pump bit does the opposite by turning on the pump and unlatching the valve (which closes it). The full bit is an imaginary output in that it only reads high or low and doesn't actually complete any action. This bit was necessary because the system can be in two different states when FS1 is open and FS2 is closed because the tank will either be in the process of draining or filling.

To accomplish the tank drain, it takes two lines of code, which are shown in Figure 11. The first line reads when FS1 is closed and latches the valve bit, which keeps the valve open, unlatches the pump, keeping the pump off, and latches the full bit. The second line will tell the PLC to read FS1, FS2, and the full bit. When FS1 is open, meaning the water level has started to fall, and FS2 is closed, meaning the water level is in between full and empty, the valve should be open and the pump off so that the tank can

Figure 12: Lines of latching code in the water tank programming

drain. The full bit has been latched, so the second line also reads for high full bit. It will continue to drain as long as those parameters are true. Then, when FS2 opens, the tank has been emptied, so the fill cycle begins.

The fill cycle is also run on two lines of code. The first line of code tells the PLC to latch the pump, keeping it on, unlatch the valve, keeping it closed, and unlatch the full bit. The second line then watches for the closed FS2, open FS1, and low full bit, and

continues to fill the tank. When FS1 closes, the drain cycle will start again, and it will run in a continuous filling and draining cycle until the user tells it to stop.

Barb Fitting Location in Water Tank Lid

In the original water tank, the barb fitting for the water input of the water tank was in the very center of the water tank lid. Once the float sensors were installed, it was realized that the water flow would interfere with the float sensor reading. The water stream would fall directly on Float Sensor 1 as it was originally. If this stayed the same, the water pressure would likely keep the switch open. If the switch was kept open by the water stream, it would never read that the switch is closed and the PLC would never think the tank was full. It was necessary then that the location of the barb fitting be changed. The hole installation point was marked so that it was as close as it could be to the edge of the lid without compromising the lid's ability to stay closed. Once my hole was installed, the barb fitting was moved over to the new hole. The original hole was not filled because keeping the hole would ensure that too much pressure would not build up in the tank.

CHAPTER 4

CONCLUSION

During the 2013-2014 academic year, an electrical engineering senior project team was assigned the task of designing and constructing a SCADA test bed for WKU's Engineering-Manufacturing-Commercialization Center (EMCC). The test bed was to have three systems simulating SCADA systems: a traffic light system, a water tank system, and a power grid system. Because there were three systems desired, the three-person project was separated into three parts. Each person was charged with the hardware design, programming, and implementation of a system. The focus of this research project was the water tank system.

Completed Water Tank System

The completed water tank system shown in Figure 12 has a water tank (A), a water pump (B), a source/sink reservoir (C), a solenoid valve (D), two float sensors (E, FS1; F, FS2), two switches installed in a switch-housing unit (G), and a relay in a NEMA 12 rated container (H). SCADA systems have elements that are monitored and controlled. This system's water level within the tank is monitored, and the ability to drain or fill the tank is controlled in an automation based on my water level. When Switch 1 is flipped to the 'on' position while Switch 2 remains in the 'off' position, the system will turn on and will run in a continuous draining and filling cycle. When Float Sensor 1 is closed, the

tank is full and the PLC turns off the pump and opens the valve so that it can drain. When the drain cycle is completed, Float Sensor 2 will be open. When the PLC reads that FS2 is open, it closes the valve and turns on the pump to begin the tank filling cycle. When the second switch is flipped to the 'on' position, the system will complete a single draining cycle that simply turns off the pump and opens the valve until FS2 is open.

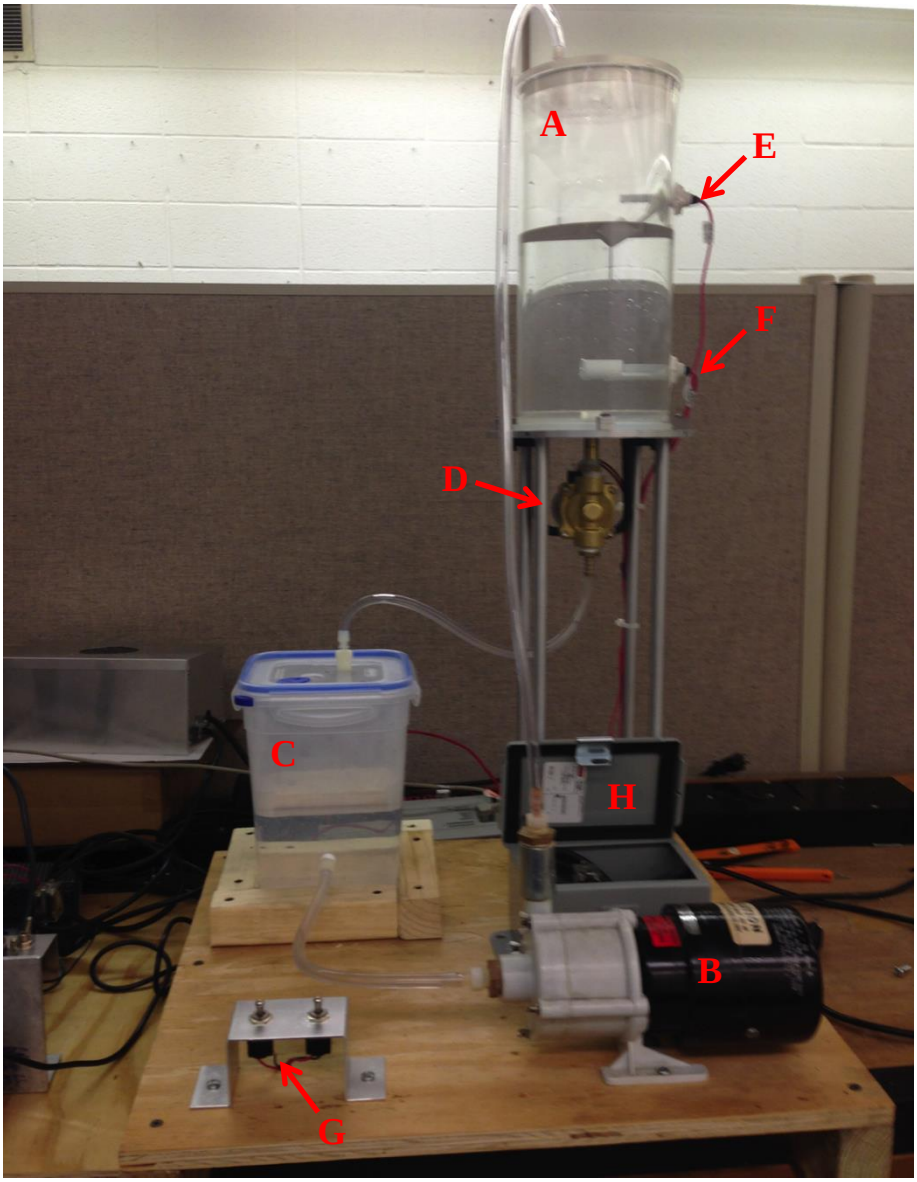

Figure 13: Complete Water Tank System

The final code for the water tank system is shown in the figure below.

Figure 14: Complete Water Tank System Code

Prospective for the Future

Hardware and software improvements could be made in the future to the Water Tank System to be a more realistic SCADA test bed. An adjustable frequency drive would enable the water flow rate to be changed. The system could be made into a true SCADA system by adding remote control capability. It would require changing the PLC

used for the system from a MicroLogix 1000 to a MicroLogix 1100. The reason for this is that the 1100 PLC can be accessed and controlled remotely while the 1000 cannot.

REFERENCES

Berry, Bob. "SCADA Tutorial: A Quick, Easy, Comprehensive Guide." DPS Telecom 8 Aug. 2011: 1-12. Print.

Hargrave, Vic. "What Are Man-in-the-Middle Attacks and How Can I Protect Myself From Them?". Trend Micro - Simply Security, 28 Nov. 2012. Web. 20 Apr. 2014. <http://blog.trendmicro.com/what-are-man-in-the-middle-attacks-and-how-can-i-protect-myself-from-them/>.

"Sterilite Ultra-Seal 16.2 Cup Storage Container." . Space Savers, 1 Jan. 2014. Web. 13 Apr. 2014. <http://www.spacesavers.com/Storage/Airtight-Food-Storage-Containers/Sterilite-Ultra-Seal-16-2-Cup-Storage-Container>.

Thakur, Anshul. "SCADA Systems." . Engineers Garage, 1 Jan. 2012. Web. 20 Apr. 2014. <http://www.engineersgarage.com/articles/scada-systems>.

Tsang, Rose. "Cyberthreats, Vulnerabilities and Attacks on SCADA Networks." Goldman School of Public Policy - University of California Berkeley 1 Jan. 2010: 2,6,13. Print.