Spring 2015
Group 4
Michael Amaral
John Brushwood
Reginald Fergerson
Zachary Kirby

The D*rink*W*izard*

Senior Design II Documentation

8/01/2015

D*rink*W*izard* Table of Contents

DrinkWizard List of Tables

D*rinkW**izard* List of Figures

1.0 Executive Summary

There is a drinking problem taking over the backyard barbeques of America. Americans are having backyard barbeques in which they are inviting friends and family over for food and drinks, however the ones hosting this event are not able to drink and socialize. The hosts of these barbeques are spending all of their time cooking and playing bartender for their guests. The purpose of the D*rink*W*izard* is to provide our customers with more quality time with their friends and family. While the hosts are busy cooking and socializing with their guests the D*rink*W*izard* can provide the guest with cocktails, thus eliminating the need for the hosts to play bartender.

The D*rink*W*izard* brings automation to the making of mixed beverages through the use of a Bluetooth capable Android phone; Apple and Windows devices are not supported at this time. The D*rink*W*izard* was designed with four main subsystems; the first and most interfaced subsystem is the user application, the second most interfaced subsystem is the vending subsystem, the third subsystem is the Bluetooth communication subsystem, and the fourth subsystem is the power subsystem. The user application contains the software capability to communicate by Bluetooth with the D*rink*W*izard* and helps simplify the creation of the mixed beverages. The vending system contains the vending assembly, which contains a cup detecting ultrasonic sensor, nine fluid containers to hold the mixers and alcohol, and nine fluid pumps to dispense the liquid. The Bluetooth system consists of the Bluetooth module that sends and receive commands from the user interface running on the user's Android device. Lastly, the power system consists of a standard three-prong 120-volt connection and was also going to consist of rechargeable batteries to aid in the portability of the D*rink*W*izard.*

The first design objective is for the D*rink*W*izard* to be a dependable and consistent dispensing system. The system focuses on producing the correct amount of alcohol to mixer ratio for each drink made. Making the system consistent ensures that each user who experiences the D*rink*W*izard* has the same experience every time they use the system. Having a dependable and consistent vending system opens the possibility of a future commercial application.

The second design objective is for the D*rink*W*izard* to run on a rechargeable battery. The traditional three-prong 120-volt connection provides power to the system, while inside the home or wherever a 120-volt outlet is provided, and provides the power to recharge the rechargeable battery. This was left off due to time constraints and budget.

2.0 Project Description

2.1 Motivation and Goals

The sun is high in the sky with a cool breeze blowing across your skin, its 85 but with the breeze it feels like a relaxing 78. The sun is beaming down on your grill as you are trying to entertain your best friends for Memorial Day. Your best friend's girlfriend ask you to make her one of your famous mixed drinks. The simple answer of no, which would disappoint your best friend, is not feasible and you don't want to step away from the grill. This is a classic dilemma and with the D*rink*W*izard* you would never have to be put in this dilemma again. When you have the D*rink*W*izard* you could pull out your Android smartphone and simply tap on your famous cocktail and the D*rink*W*izard* goes to work making your best friends girlfriend her requested drink. This scenario right here is the main motivation behind designing and building the D*rink*W*izard,* no more upset friends.

Another motivation behind this project is the fact that we are living in the sunshine state; warm weather and beaches are all around us. The D*rink*W*izard* allows you to take all things alcohol related and transport it to the beach. The D*rink*W*izard* allows you to go from being the cool guy with a drink-making machine to being the most awesome dude, who is the life of the party, on the beach. Girls and party goers flock to you to check out this awesome contraption that you call the D*rink*W*izard,* you may even meet your future ex-wife.

One major goal of this project is to create a user inspired mobile application. This application has the most used interface for the D*rink*W*izard* so it needs to flow and be easily navigated by the user. The application also needs to be very responsive and have the bugs worked out so it is not prone to crashing. This application is the lifeblood of the D*rink*W*izard* and could destroy the user experience. The D*rink*W*izard* also has a positive secondary goal, sociability. This socializing aspect allows the users to continue the activities they are doing without having to stop to grab the ingredients and make the adult beverage of their choosing. The last goal is the portability aspect of the D*rink*W*izard*. The D*rink*W*izard* needs to be portable so that the user can attend events such as the beach, tailgating parties, backyard barbeques, pool parties, or even the dreaded family dinner nights. Having a machine that is portable allows our user to attend the events of their choosing and have cocktails available at their fingertips; it takes them from being a zero to a hero.

2.2 Objectives

The main objective of this project is to create a dependable and consistent adult beverage dispenser. We are determined to produce a quality product at a

reasonable price. The D*rink*W*izard* is made to be as portable as possible, the objective is for the user to want to take this system with them to the various events they will attend. Making the system as portable as possible, helps encourage the user to take this system with them. The D*rink*W*izard* user interface has been designed with the user in mind. This user application is intuitive and promote repeated use of the system. When the D*rink*W*izard* was first envisioned it was envisioned as a system that would allow the user to have more of a social life. This system holds true to that vision and allow the user to spend more time socializing with friends and family and less time making the adult beverages for those guests.

The warm and sunny Florida summer is upon us, our second objective is to have a working prototype completed by mid-summer. Florida is famous for its sunny days and beautiful weather. We want to take advantage of this great weather and maybe get a little socializing done of our own. Having a working prototype allows ourselves time to use, test, evaluate and fix any potential flaws in the system. Having this stringent deadline allowed us to deliver a finished top-notch, version 2.0, system by the end of summer.

2.3 Project Requirements and Specifications

As this project was being designed a key aspect that always came up is portability. This system is designed with the user in mind. The system was designed be a rectangular box with dimensions large enough to encase nine liquid containers, the nine fluid pumps, the rechargeable batteries, the control unit and the power system components. This vending unit box was designed to have a removable platform that keeps the vending unit box at an idea drink dispensing height; idea height would be bar top height. This platform sits atop four heavy-duty rolling casters to aid in the moving of the system. The box was going to also have handles on two opposing sides so as to aid in the transportation of the system, but the overall system was light enough that the handles were excluded and the extra savings were much preferred in order to keep the overall cost down.

The project requirement for the dispensing subsystem is to first be able to detect when a typical solo cup is placed in the unit. The second requirement for the dispensing unit is for it to be able to dispense liquids from nine storage containers. The third requirement for the system is that the drinks created are consistently made to that specific drinks specific standard. An example of a specific drink standard would be if the user ordered a Hurricane from the system it would dispense 0.5 oz. rum, 0.75 oz. dark rum, 3 oz. fruit juice, 3 oz. mango juice, and 2 oz. sweet and sour mix into the users cup. The consistent factor would be that the quantities listed for the Hurricane are reproduced exactly the same every time the Hurricane beverage is made.

The first project requirement, and main requirement, for the user application is for it to have a crash free application for the Android platform. Having a crash free application is important to giving the user a quality experience as well as giving them a reason to continue to use our product. If the application continually crashes or has an excessive number of bugs then the user will quit using our D*rink*W*izard.* The second requirement for the user application is for the application to be able to display a list, with images, of preprogrammed drink recipes. A possible upgrade for the version 2.0 will be the ability for the user to control how strong the specific drink is, example of this would be a “I’m tired” selection, “Bartender mode” selection, and then a “Rock out” selection. These selections in order would create a weak drink, normally made drink, and, for those long days at work where you would like to forget about the day, extra strong mode. The third requirement for the user application is for the application to be able to transmit and receive data by Bluetooth. The drink selection is sent via Bluetooth to the main box and at that point the drink will be made. A possible add on for version 2.0 is a Breathalyzer and fingerprint scanner on the main system box. This Breathalyzer would take a users BAC, blood alcohol content, and send this data back to the user application for it to analyze and then inform the user on the effects of more alcohol consumption. The fingerprint scanner would be used to identify which user is at the main vending unit and dispense that users drink. This would account for multiple users submitting drinks at the same time as well as ensure that D*rink*W*izard* is serving a beverage to a user that is 21 and up. The fourth requirement for the user application is the application has an age verification system. Users under the age of 21 should not be able to send drinks to the main vending unit dispensing system.

The project requirements for the power subsystem are to have output voltages of 3.3, 5 and 12 volts to power the fluid pumps and the main control unit. The next requirement is that the power subsystems have the voltage output necessary to charge a rechargeable battery. The third requirement for the power subsystem is for the design of the system to be as efficient and space saving as possible.

The last subsystem is the Bluetooth subsystem. The project requirements for the Bluetooth subsystem is that the system be able to transmit and receive data via Bluetooth with an Android device running our user interface application. The data that the Bluetooth module could potentially transmit back to the user application is order received, please wait, order not received, or for version 2.0 it could send back Breathalyzer data for the user application to analyze and display for the user.

3.0 Related Research to Project Definition

There are multiple automated drink mixers on the market today. Some of them have features, which are beneficial for designing the *DrinkWizard*. Five related projects have been researched by the *DrinkWizard* developers and will be outlined in this section. The five that have been chosen are Under the Sun Drink Mixer, BaR2D2, Smartender, Somabar, and the Inebriator.

Several different types of technologies have also been researched prior to making a decision for each system of the *DrinkWizard*. First, different microcontrollers were researched. Among these are the MSP430, Arduino, and PIC. Other components researched include different modules for each type of communication (Wi-Fi, Bluetooth, and near-field communication), liquid crystal display (LCD) modules, and sensors to determine how much liquid is in a glass, and various power supply components.

One system, which required a large amount of research, was the fluid delivery system. There are many options for transporting fluid from its container to the cup. As seen in some of mixers reviewed below, pumps can pull the liquid from containers or mixers can take advantage of gravity and let the desired amount of liquid simply fall into the cup. Each method provides not only advantages but drawbacks as well. The specific advantages and disadvantages of each method will be discussed in the Fluid Delivery Systems subsection of the Relevant Technologies section.

Different methods of wireless communication have also been researched; however, they will be covered in depth in the communication section. Wireless communication makes the *DrinkWizard* much more user friendly by providing the ability to order drinks without having to be next to the mixer as long as a cup is under the liquid spout. All the previous technologies will be discussed in the following subsections.

Finally, in this section different strategic components will be discussed, features to make the *DrinkWizard* more sophisticated. One of the components is a ultrasonic sensor to detect how much liquid is left in each container in the mixer. The second module is a Breathalyzer that will allow the user to regulate safety of guests using the mixer. A few uses for the Breathalyzer have been discussed including prohibiting the order of beverages if the value seen by the Breathalyzer is too high, forcing the user to pass a test to use the system, and even just making it available as an option for the novelty of it. Unfortunately, the Breathalyzer ended up unused due to complications and time constraints.

3.1 Related Projects and Products

This section covers products similar to the D*rink*W*izard*, which are on the market or already being used today. Although there are more than those in this section, five have been chosen due to one or more interesting related features. The five mixers are Under the Sun Drink Mixer, BaR2D2, Smartender, Somabar, and the Inebriator. They are in this order in the following subsections.

3.1-A: Under the Sun Drink Mixer

Under the Sun D*rink*W*izard*, is a product created by students at the University of Central Florida. Like the D*rink*W*izard*, it was designed to facilitate the creation of mixed beverages. A smart phone, similar to the design of the Drink Wizard, also controls the Under the Sun Drink Mixer. A key difference between Under the Sun Drink Mixer and the D*rink*W*izard* is that Under the Sun Drink Mixer draws power from a solar panel, making it ideal for outside usage on a sunny day at a social gathering [9].

Another major difference between the Under the Sun Drink Mixer and the D*rink*W*izard* is that Under the Sun Drink Mixer generates a personalized code for each drink. The user then brings the code to the machine and the mixer scans this code and creates the drink [9]. This design has been considered for the D*rink*W*izard* but as it is being designed for personal use where not many people have access to the drink itself, the designers have decided to allow the drink to begin being made as soon as an order is received from the Android device. There are other minor differences in the two drink mixers but those outlined above are the main ones. Under the Sun Drink Mixer and the D*rink*W*izard* are similar products designed for similar purposes.

3.1-B: BaR2D2

A hobbyist with no formal education created Bar2D2. The concept is the same as that the D*rink*W*izard* but with added storage for ice and canned beverages. A focus has also put on looks and style and BaR2D2 has been equipped with sound-activated neon lights [10]. Both mixers, BaR2D2 and the D*rink*W*izard*, include the ability to order a drink remotely and have it dispensed via pumps instead of having a person craft the drink so they are focused on the same concept.

A major difference between the two mixers is their method of ordering a drink. BaR2D2 uses a laptop computer to send drink selections from a database of known drinks. The D*rink*W*izard* uses an Android device with a custom application to send drink selections. While the device used to make selections is different,

both drink mixers rely on Bluetooth technology to allow the user's device and the mixer to communicate.

Both projects are similar and attempt to achieve the same goal. The main difference is that BaR2D2 is radio controlled, mobile, and communicates with a laptop. The D*rink*W*izard* is made to be stationary, hold more bottles for a wider selection, and communicates with an Android device instead of a laptop. At the time of writing, BaR2D2 is also being upgraded.

3.1-C: Smartender

The Smartender automated beverage system is a professional, portable touch-screen bar. It allows mixed drinks to be ordered using a touch-screen [11]. This is the first fundamental difference between Smartender and the D*rink*W*izard*. The D*rink*W*izard* communicates wirelessly with the Android application to accept orders from the user. Having a touch-screen for orders may limit the D*rink*W*izard,* as users would need to be in close proximity to the mixer.

Smartender is capable of making a wide variety of drinks. The creators claim that over six hundred different types of drinks can be made [11]. The D*rink*W*izard* is able to compete with this feature by allowing for the creation of custom drinks. These drinks can be saved to the D*rink*W*izard*'s database that can hold many more than six hundred drinks.

Both drink mixers achieve the same purpose. The major difference comes from the way drinks are ordered. Whereas the Smartender automated beverage system uses a touch-screen to allow users to place orders, the D*rink*W*izard* uses an Android application to keep traffic around the mixer to a minimum and allow users to simply pick up a drink when it is ready.

3.1-D: Somabar: Robotic Bartender

Somabar acts as a robot bartender, which uses a Wi-Fi connected application to receive orders. This Kickstarter funded project has an application for which there are Android and iOS variants. It works by selecting a drink through the application and creating it by pulling the ingredients from the Soma Pods. The Somabar also has an extremely compact design and is capable of fitting on most kitchen counters. Another feature the Somabar presents is the ability to monitor liquid levels visually by storing the liquid in plain sight [12]. The D*rink*W*izard* is designed to use light-emitting diodes to show the user when the liquid level drops below a predetermined level. Three hundred different mixed drinks can be made with the Somabar's customizable Soma Pods. No such limitation is imposed upon the D*rink*W*izard*. An interesting feature of the Somabar is that the individual drinks enter a reservoir before being added to the cup. The reservoir is then

flushed with water after the all contents of the drink have been dispensed to the cup below [12]. While designing the D*rink*W*izard*, the choice of dedicated liquid lines was made to avoid this idea. Each liquid is dispensed from its own container and tube that eliminates the need for a flushing system.

The D*rink*W*izard* and the Somabar are similar products. Both are designed to communicate with and utilize a mobile device for easy ordering. The D*rink*W*izard* will not begin dispensing liquid unless a cup has been placed under the spout to avoid spills and waste while the Somabar contains a tray to catch unwanted liquid from falling while a cup is not present. Both products use a simple but sophisticated mobile application creates mixed drinks for users to enjoy.

3.1-E: The Inebriator

The Inebriator is a cocktail maker that caters to a lively crowd by striving to attain the modern appearance. It is controlled via an Arduino microcontroller and uses a separate console to select drinks. The drink is also moved to each bottle of liquid as it is being made and is able to be accelerated and decelerated to create the drink as fast as possible without spilling any liquid [13]. A ring of light-emitting diodes signifies the state of the drink. It turns red when a selection has been entered and the drink is being created and flashes blue when the drink is complete and ready for the user to take.

This mixer is vastly different from the design intended for the D*rink*W*izard*. First of all, an Arduino board does not control the DrinkWizard. Secondly, images show users selecting from a menu that has been printed off and placed with the Inebriator's mixer [13]. The D*rink*W*izard*'s Android application eliminates the need for a physical menu. Users are able to browse the entire selection available and even create their own drinks to add to the menu. Finally, the designers of the D*rink*W*izard* agreed that the beverage should not be moved as it is being made to avoid unnecessary spills and waste liquid. The Android application is also responsible for telling users when their drink is ready to be picked up.

3.1-F: Conclusion

All the drink mixers reviewed in this section have been successfully created and seen use at least as a prototype. Each of them offers features that the D*rink*W*izard* could benefit from utilizing. A common factor in most of these mixers is a predetermined list of drinks, which limits the user to a set number of drinks. For this reason, it has been decided that users are allowed to create custom beverages then add them to the master list so that they do not have to recreate it every time they want the drink. The option to delete beverages is also included so that drinks can be deleted if the user finds that too many have been entered and having the custom list is more of an inconvenience than a welcomed feature.

3.1-F-1: Portability

Portability is another feature seen in multiple mixers reviewed above. Based on current plans for the *DrinkWizard*, portability was not highly prioritized; however, it is small enough to be moved by one person or two people, at most. Since the mixer was put on wheels, one person can now transport it easily. The majority of the weight comes from bottles and liquid. Therefore when the mixer is not loaded, it should be light enough to pick up and carry without the use of a dolly, hand trunk, or any other piece of equipment. If a second version of the *DrinkWizard* is created, it is likely that portability will be a major priority and the mixer itself will be smaller. In this case, the more portable products above will be reviewed again.

3.1-F-2: Ordering

The order process varies widely among different mixers. Some use devices with well-established operating systems while others use more specialized means. The designers of the D*rink*W*izard* have decided to take the approach of the Somabar and use an Android application. While an iOS version is not planned at this time, it could be implemented in the future. The reason for choosing an Android application stems from the usage the user will obtain from the device and the functionality provided by the Android software development kit. All necessary capability is readily available to Android and needs to be implemented by the developers. The user also has access to the rest of the Android device and can use it at their convenience.

3.1-F-3: Beverage Creation

The final process of interest to the designers is what occurs with the beverage while it is being created. It is possible to move the cup to each bottle as it is being created (as is the case in the Inebriator) but the D*rink*W*izard* must have the cup remain stationary and deliver the liquid to it via dedicated lines. Designers opted to keep lines separate to eliminate the need for a flushing system. The Somabar uses this approach. All the projects reviewed above provide valuable information for the design of the D*rink*W*izard*.

3.2-A: MCU

There are many different microcontrollers. That is why the microcontroller is one of the hardest parts to choose. The microcontroller must be picked based on the design specifications and the budget. Some microcontrollers that our group looked at for our D*rink*W*izard* project was: the TI MSP430G2553, the Atmel Atmega16 and the Microchip Pic24 microprocessors. The D*rink*W*izard*'s microcontrollers have the task of controlling the pumps and determining if the

bottles of liquor or fluid are empty. The microcontrollers also have the duty of communication, via Bluetooth with the Android device, to take drink orders. The microcontrollers also have to store and recall all of the different drink combinations. This means that the microcontroller must have enough memory to be able to store all of the information necessary for production of over 30 different drinks. Also, the microcontroller must have the ability to communicate via UART to the Bluetooth module. All three microcontrollers are compared in *table 3.2-A*.

	Texas Instruments MSP430G2553IN20	**Atmel Atmega16**	**Microchip Pic24FJ16MC101**
Operating Voltage	1.6 – 3.6 V	4.5 – 5.5 V	3.0 – 3.6 V
Current Draw	230 μA	1.1 mA	1 mA
I/O Pins	16	32	15
Flash Memory (KB)	16	16	16
RAM (B)	512	1024	1024
Architecture	16-bit RISC	8-bit RISC	16-bit RISC
Number of General Purpose Registers	12	32	16
Operating Temperature (C)	-40 to 85	-55 to 125	-40 to 125
Max Voltage at I/O pin	Vcc – 0.3 V	6 V	5.6 V
Max Clock Rate	16 MHz	16 MHz	16 MHz
Socket Type	PDIP,QFN, TSSOP	PDIP, TQFP, QFN, MLF	PDIP, SOIC, SSOP
Pin Count	20	40	20
Price Per Unit	$2.80	$7.16	$2.50

Table 3.2-A: Microcontroller Comparison

3.2-A-1: Atmel Atmega16

The Atmel Atmega16 is a very popular and versatile microcontroller. This chipset is also the most expensive of the three chips that were compared. This microcontroller uses an 8-bit advanced RISC architecture. It has a throughput of up to 16 MIPS at 16 MHz and 131 total instructions, most of which are single clock cycle execution. The operating voltage has a narrow range of 4.5 V to 5 V. This microcontroller has 1024 bytes of ram and 16 KB of flash memory. The

Atmel Atmega16 meets all of the needs of our D*rink*W*izard* project with ease, but is over budget for the MCU portion costing over seven dollars per unit. For that reason it was not used in the D*rink*W*izard*.

3.2-A-2: Microchip Pic24FJ16MC101

The Microchip Pic24 microcontroller is very similar to the Atmel Atmega16. It also has a CPU speed of 16 MIPS and 16 KB of flash memory. Unlike the Atmega16, the Pic24 has a 16-bit architecture, which could be needed if we were to ever expand our system to incorporate more drinks or drink ingredients. The Pic24 also has a narrow range of operating voltage of 3 V to 3.6 V. This chip also has a maximum input voltage on an I/O pin of 5V. This is ideal as most of the sensors used have an output of 5V. This chipset does support capacitive touch sensing if a touch screen where to be added to the D*rink*W*izard* in the future. There are three comparators on board and each can have up to four inputs. This is good for comparing the inputs of the liquid level sensors. This is an ideal microcontroller for our project, but due to the unfamiliarity of this chipset and the added cost of purchasing a programmer for this chipset, this chipset was not the one selected as the control for our D*rink*W*izard project*.

3.2-A-3: MSP430G2553

The last microprocessor that we looked at was the familiar TI MSP430G2553 microcontroller chipset. With only 51 instructions it has the lowest number of instructions in the group, making it the least complicated to code for me. This is a very versatile chip that has ultralow power consumption. This chip has several I/O pins, which would be suitable for our D*rink*W*izard* project. It also has 16 KB of built in storage that can be used for our code and recipe list. This chipset also only requires a low operating voltage ranging from 1.6 volts to 3.6 volts, the widest range of the group. The MSP430G2553 also has a standby mode that reduces power consumption down to less than 0.5 micro amps and has a very fast wake up time from standby of less than one microsecond. This is ideal for a battery-operated system, as it will allow for an extended operating time. Unfortunately, the MSP430 chipset has a maximum input voltage on the I/O pins equal to or less than the operating voltage. This microcontroller also has a powerful 16-bit RISC CPU and 16-bit registers. All of these things combined make this a very powerful microcontroller and the choice for our D*rink*W*izard* project.

There are some drawbacks to the MSP430G2553. One major disadvantage is that the maximum voltage that the microcontroller can safely handle at an I/O pin is Vcc to 0.3 V. When using sensors that output a maximum voltage of 5 V, a problem will arise. External hardware must be added to be able to use these

sensors safely. Another disadvantage to using this chip is that it has the least amount of RAM. When using a lot of registers this could be a problem. However this does not affect our project.

The MSP430 line of microcontrollers is very familiar to us. We have written several programs for this line of microcontrollers for one of our undergraduate classes, Embedded Systems. This class has prepared us for using such a chip inside of an embedded system like the D*rink*W*izard*. We also already own several of these microprocessors, therefore reducing out of pocket cost for our group. Also along those lines the msp430 chips can be programmed with the Launchpad, which every group member also owns. Because of all of these reasons, the Texas Instruments MSP430G2553 microcontrollers are used in our system.

3.2-C: Communication Modules

The D*rink*W*izard* is going to require some form of communication between the android tablet, or an android phone, and the main housing of the D*rink*W*izard*. During our research in communication modules we came across quite a few forms of relevant technologies for wireless communication. The 5 that we found most applicable are listed below

- Bluetooth
- Wi-Fi
- Near Field Communication
- ZigBee
- IR Communication

The first and probably most popular form of small device communication technology that we researched about is Bluetooth. Bluetooth is a global wireless standard that enables convenient and secure connectivity for devices and services. It was created by Ericsson in 1994, and was originally conceived as an alternative to wired RS-232 cables. The technology behind Bluetooth is radio transmissions. It operates in the unlicensed industrial band between the ranges of 2.4GHz to 2.485GHz and spectrum spread frequency hopping full duplex signal at a rate of 1600 hops per second.

The main advantages of Bluetooth are that when devices are paired together, other Bluetooth devices cannot interfere with the connected devices. It acts as an extremely strong link between two devices only. Another very advantageous feature of Bluetooth is its extremely low power consumption during idle times. This feature is important for long lasting battery life on our Android tablet as well as the D*rink*W*izard* while on battery. The final important feature of Bluetooth is its fairly low cost for integration components.

There are a few small disadvantage of using Bluetooth. The major one is its operating range. While it does use radio waves and can penetrate most building materials, the usable advertised range is normally about 50 feet. Another disadvantage is its relatively small bandwidth of around 800Kbps.

The next type of communication that we found relevant to research for the D*rink*W*izard* is Wi-Fi. Wi-Fi uses radio waves just like Bluetooth to send information through the air. These transmission frequencies are either 2.4GHz or 5 GHz. These higher frequencies allow the signal to carry more data. The current standard for Wi-Fi communication is IEEE802.11n. It is reported to be able to achieve speeds as high as 140Mbps.

The major advantages that Wi-Fi has over most other communication methods are its speed and range. Wi-Fi has a bandwidth of about 11Mbps and a usable range of about 300 feet. The two things that do not appeal to its use in the D*rink*W*izard* are its security and overall cost for components.

The next type of wireless communication that sounded interesting was NFC or Near Field Communication. It is generally used for ticketing or tag like data transactions. The two devices that are communication must be very close to each other to operate correctly, usually less than 3.9 inches. NFC employs electromagnetic induction between two loop antennae. The operating radio frequency band is 13.56MHz and the data rate range from 106kbps to 424kbps. The major downfall for this technology application for the D*rink*W*izard* is very limited range of operation. Since the entire idea of the D*rink*W*izard* is to have the unit remotely mix a drink from a distance, the 3.9-inch range of Near Field Communication does not allow our D*rink*W*izard* concept to function as intended.

The next communication standard that we researched was ZigBee 802.15.4. Their main applications are for the control and monitoring of sensors, lighting controls, and security equipment where very low levels of throughput data are needed. The ZigBee protocol operates on three license free bands, 2.4GHz, 915MHz in North America and 868MHz for Europe. The 2.4GHz band can support up to 16 channels and a maximum data rate of 250kbps. The 915MHz band supports up to ten channels and have a maximum data rate of 40kbps and the 868MHz band supports a single channel with a maximum data rate of 20kbps.

The data rates and frequencies used by the ZigBee standard were very appealing for use in the D*rink*W*izard*. The major problem that we encountered with ZigBee is that there is relatively low support as it is an emerging and somewhat underground technology as of now.

The final technology explored for communicating with the D*rink*W*izard* was Infrared communication. IR transmits data signals through LEDs or lasers. The frequencies used in this type of communication are between 100GHz to 1000THz, which is extremely high in the electromagnetic spectrum. The high frequencies used can relate to fairly high-speed ranges up to 16Mbps. The disadvantage of IR transmission is that is usually operates on a point-to-point system when a line of sight is required for reliable transmission.

Since the D*rink*W*izard* communication is going to be extremely low data rates, the advantage of IR communication could not be justified because of the extreme attenuations in the wireless signals when line of sight is lost. The D*rink*W*izard* basic concept is projected to be used indoors and possibly have the unit remote mounted in another room. Based on the research done, we chose to use Bluetooth communication platform to achieve sufficient, reliable and secure transmission of drink orders from the Android platform to our main pumping system.

3.2-D: Serial LCD Display Modules

An added feature that our team would like to integrate into the D*rink*W*izard* is the ability to have an on-board display that notifies the user when a certain fluid container is empty or running very low. In order to do this our D*rink*W*izard* needs to have an LCD display located somewhere on the outside of the frame. This display should ideally be able to show the status of all empty bottles at time. To do this efficiently our LCD display needs to be at least two lines long.

During our research, we found 3 companies that offer LCD displays that offer full swing RS-232 serial communication. The first company is Crystalfontz that offer a 16X2; the second display found is made by Parallax Inc, and Newhaven Display International makes the last display that we found to meet our requirements.

The first display made be Crystalfontz is the CFA632-YFH-KS. It is a full swing RS-232 16X2 LCD module. The display measures 4.25 inches long x 1.65 inches high x 0.78 inches deep. The character read out size is 0.15inches x 0.28 inches. It is RoHS/REACH compliant, has a yellow-green LED Backlight, and has a -20C to 70C temperature operation. The D*rink*W*izard* is to be operated primarily indoors and should be well within these temperature requirements. The *figure 3.2-D-1* shows the LCD display in yellow backlight mode.

Figure 3.2-D-1: CFA632-YFH-KS Display

Image courtesy of Crystalfontz

This display offers USB, RS-232, Logic Level Serial, Logic Level Serial Inverted, I2C and SPI. The six interface types can used by changing the interface selection in a command and using the appropriate jumper configurations. The serial interface of choice is RS-232 as our team has become very familiar with its initialization and character handling techniques. It has a 6 o'clock viewing angle, which should be adequate for the purpose of our project. The Crystalfontz display also has an adjustable contrast; it can be adjusted by using the command control. The Crystalfontz also has an EEPROM memory to customize a "power-on" display message. The price for a single display unit is $39.35

The second comparable unit is the Parallax 2x16 Serial LCD backlit display. This display is 3.15 inches long x 1.42 inches high. It offers clear 40 pixel characters and supports ASCII DEC 32-127. It also has a yellow-green LED backlit display. The Parallax display module has an adjustable knob for contrast. This feature is slightly different than the command control coded version by Crystalfontz. A sample display can be seen in *figure 3.2-D-2* below.

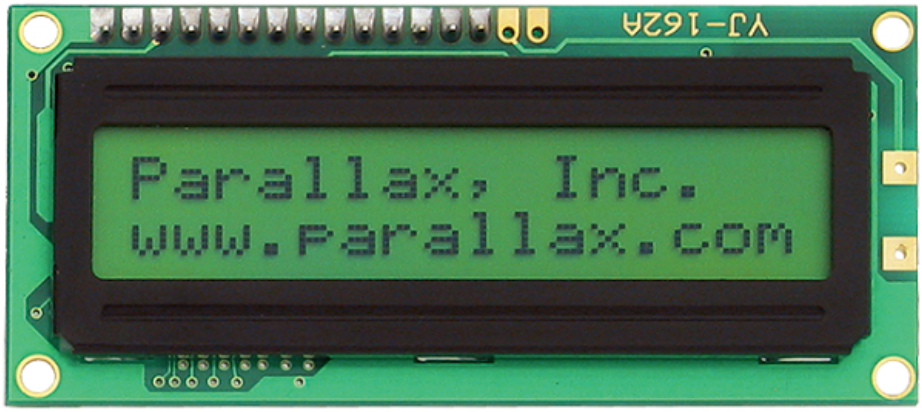

Figure 3.2-D-2: Parallax 2x16 Display

Image courtesy of Parallax

This unit from Parallax also has a selectable asynchronous serial baud rate of 2400, 9600 and 19200. Our code is written to initialize the UART to a baud rate of 9600 for the Drink Wizard. This unit also has eight user definable custom characters. The price for the smaller display unit from Parallax is $29.99.

The last display module that we found relevant to our project is the NHD-0216K3Z-NSW-V3. It is also a 2x16 character display that is RS-232, SPI, and I2C compatible. It measures 3.14 inches long x 1.41 inches high, which is almost the exact same size as the Parallax module. The Newhaven display is a blue-white LED backlight versus the other two units that were yellow-green. This display also has a 6 o'clock viewing angle like the others and has a wide operating temperature range of -20C to 70C. An example of the unit with the white LED backlight display can be found in *figure 3.2-D-3*.

Figure 3.2-D-3: NHD-0216K3Z-NSW-V3 Display

Image courtesy of Newhaven Display

Our team thought that the blue and white backlit display looked quite a bit better than the standard yellow-green found in the other two units. The New Haven unit has only 3 pins required for operation. It also can display normal ASCII text in the 0x20 to 0x7F range just like the other two units. All of the adjustability of the Newhaven module is done through a command control just like the Crystalfontz unit. Some of them are turning the display on and off, resetting the cursor to home, blinking cursor, setting the backlight brightness and contrast, change and display the RS-232 BAUD rate. The price for a single Newhaven display is $19.85. Our team found this extremely reasonable and competitively priced for all of the features and support offered for this product.

3.2-E: Fluid Delivery Systems

There are several different ways for liquids to be transferred from one container to another. A conventional method is to use a pump. This pump can be either mechanical or electrically driven. Another method is to use a gravity fed system, which is a system that uses gravity to control the flow of the liquid. Regardless of

the system used, the liquid must travel through a tubing element to reach the final destination. Of course, there are also many different types and sizes of tubing that have different characteristics, which need to be used in the correct manner based on the system design requirements.

Pumps come in three major varieties: direct lift, gravity, and displacement. A direct lift pump uses a method of moving a liquid mechanically by using an actual container like a well with a bucket or by moving the liquid up with a tool like a water wheel. A direct lift pump has a very low to medium efficiency rating as well as a low to medium output rating. This type of pump can vary from a couple of inches to a number of meters. Given the drawbacks of this kind of pump, it would be an unsuitable choice for the D*rink*W*izard* because of the size constraints and inefficiency in this type of application.

The second type of pump, gravity pumps, is the least common way for transferring a liquid. These pumps rely entirely on gravity to do all of the work. This is the type of pump that is used in a gravity fed system. The pump is actually not a pump at all; it uses gravity to create a force on the liquid that is to be transferred. A siphon is a good example of this, as well as a heron fountain. These pumps are generally more efficient than the direct lift pumps but less efficient than the displacement pump. They also cannot create any amount of liquid pressure by themselves, another pump would have to added to create pressure in the system. Another drawback to this category of pumps is that they must be primed, meaning they do not create a suction to draw the liquid on their own.

The third type of pump is a displacement pump. There are several different types of displacement pumps. Generally a displacement pump, does what its name infers, it displaces a liquid in order to transfer that liquid. An example of a displacement pump is a centrifugal pump. This pump uses an impeller that is spinning at a rate of speed that is high enough to expel the liquid using the vanes of the impeller. This can be seen in *figure 3.2-E-1*.

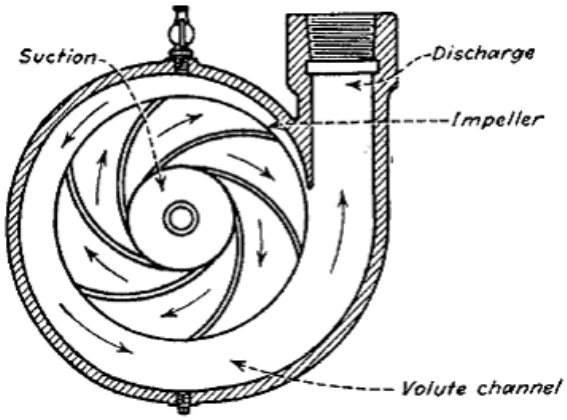

Figure 3.2-E-1: Centrifugal Pump

Image courtesy of Empowering Pumps

The down side to most of these pumps is that they have to have the fluid touch the internal elements of the pump. With certain fluids this could be a problem. The pump may result in a catastrophic failure if the fluid inside the pump causes a problem. For example if a high sugary liquid is to be pumped then the pump may become clogged with sugary syrup like substance that causes the pump to fail. However, there is a displacement pump that does not touch the liquid. This pump is a peristaltic pump.

The peristaltic pump works much like squeezing a tube of toothpaste. There is a tube that houses the liquid and a rotor with lobes or rollers. The tube is held in the housing of the pump with the rotor and rollers located in the center of the pump. The rollers run along the length of the tube pressing it against the housing of the pump and thus creating suction on one side and displacement on the other. The roller continues to roll along the tube and will pump the fluid through the tubing. A peristaltic pump can be seen in *figure 3.2-E-2*. The blue section is the tubing and the gray section is the rotor with the rollers.

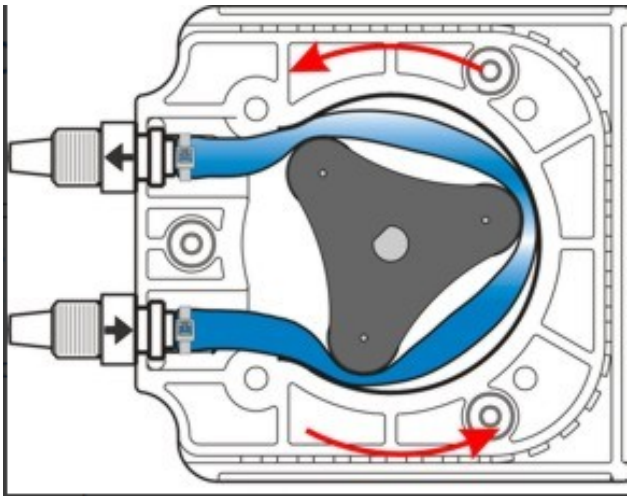

Figure 3.2-E-2: Peristaltic Pump

Image courtesy of Blue-White Industries

The peristaltic pump is also very accurate at dosing. The accuracy is due to the fixed distance between rollers. Once this distance is known and the inside diameter is also known, the volume of fluid dispensed can easily be calculated by using equation in *figure 3.2-E-3*.

$$Volume\ dispensed\ = 2\pi r^2 l * number\ of\ rollers * number\ of\ rotations$$

Figure 3.2-E-3: Volume Calculation

This allows for a uniform dispersion of a liquid. Because of this accuracy and repeatable performance of this style of pump, these pumps are used in medical and laboratory applications where dosing accuracy is a necessity. These pumps also allow for easy maintenance and changing of the tubing in which the fluid flows.

There are some disadvantages of using a peristaltic pump. The prefix peri- means wave. The peristaltic pump does create an oscillated pumping action, not unlike a wave. Also the peristaltic pump does not generally have a high output. However the size of the tube is proportional to the diameter of the tubing. When the tubing size has a requirement to be increased, the pump will also need to become larger to accommodate the larger tubing.

Regardless of the style of pump that has been used some sort of bleeder system has been implemented into our system. This will allow air into the reservoir so that the pump will not have to work against the vacuum that would be created by pulling a liquid out of a container and not replacing the volume with air. The simplest way is to place a bleeder valve in the bottle. The bleeder valve is a one-

way check valve that only allows a flow to be generated in one direction. The one-way check valve will allow air to flow into the system as the liquid is pumped out, equalizing the pressure inside the container. This will allow a consistent even flow of the liquid being pumped.

There are also a lot of different tubing materials that can be used. There are advantages and disadvantages to using different materials. Some materials can withstand high temperatures and some materials can withstand harsh temperatures. The design specifications need to be considered when selecting the tubing material.

Once the D*rink*W*izard* is to be pumping alcohol, we needed a material that is food safe and also can withstand the pumping of alcohol and high sugary fluids. There are several materials that would meet these specifications. The top choice for this product is tubing made out of a special type of silicon-based material, called Tygon. Tygon is a food safe material that can withstand the abuse that will be thrown at it by our machine. It is also clear as glass so we the flowing of the liquid can be viewed to determine if a problem has arisen. This type of tubing is also lightweight and very flexible. This is ideal because of the use of a peristaltic pump needs a flexible tubing to work efficiently. Of course with modern day concerns about plastics, the Tygon tubing contains no BPA.

Tygon tubing meets several different safety standards. Some of these standards include but are not limited to standards set by NSF, FDA and 3-A criteria. The Tygon B-44-3 tubing is non-toxic, taste-free and odor-free. It meets the NSF Standard 51 safety criteria for use in handling foods and beverages. This product is also very lightweight and flexible.

Being so lightweight and flexible means that we can use less tubing and connectors than a rigid tubing system. The lightweight plays a large factor when trying to make this a portable device, as the ability to lift it is important. This specific line of Tygon tubing is also not very expensive, costing around 22 cents per foot. This specific tubing has a decent lifespan in a peristaltic pump, which can be seen in *figure 3.2-E-4*. Since any pump will only be on for a minute or two at a time the tubing will have a very long lifespan on the D*rink*W*izard*. Being inexpensive and so versatile are the main reasons why it was used in the D*rink*W*izard*.

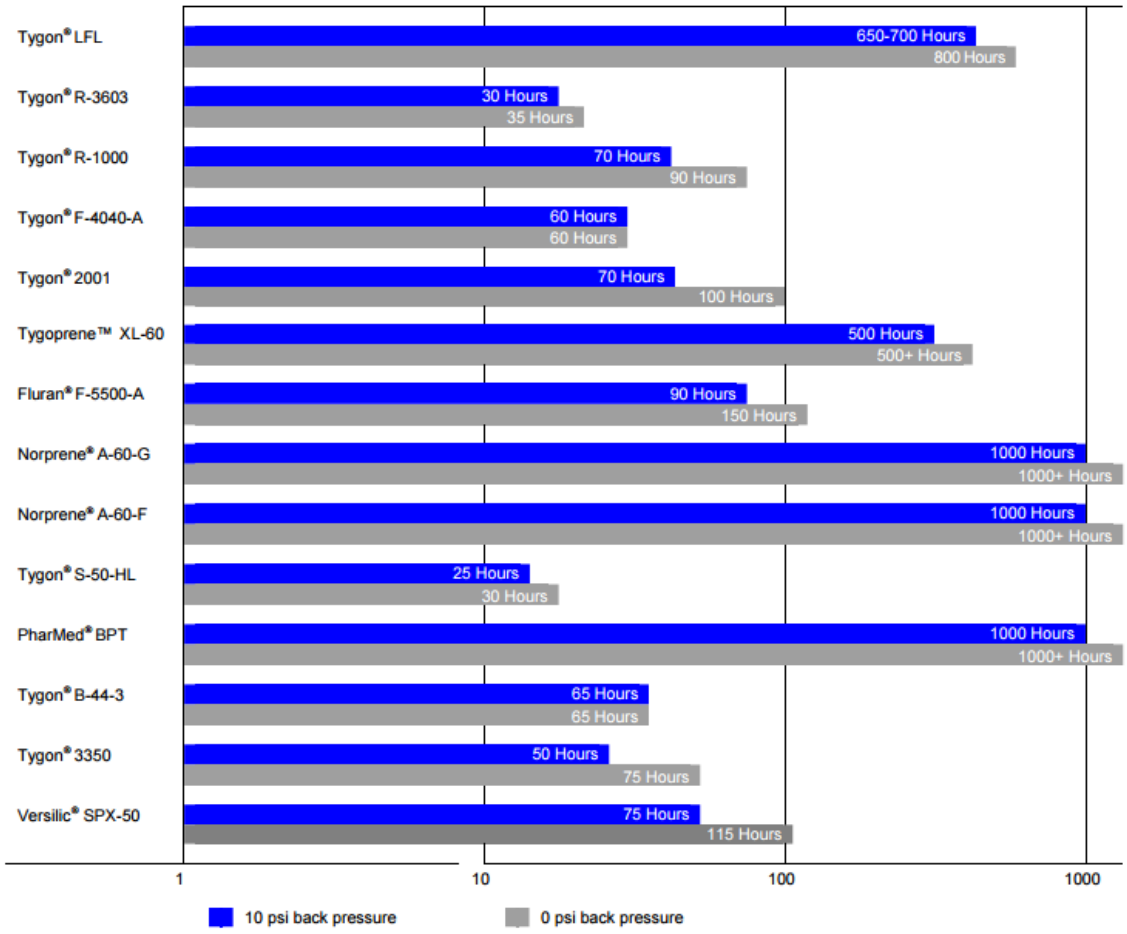

Figure 3.2-E-4: Lifespan of Tygon Tubing

Image courtesy of Saint-Gobain

However, there needs to be a method of connecting all of these things together. Hose adapters were used between the pump and the tubing. These connectors are made of a food safe plastic like material and will be held together with the friction of the hose on the barbed connectors and with the added safety measure of zip ties to add an additional tension.

The pump system consists of nine individual pumps. These nine pumps are peristaltic pumps. These pumps were picked for the D*rink*W*izard* because they are very reliable, easy to use, easy to repair and very accurate. If a portion of tubing breaks then the head of the peristaltic pump is removed. Then the tubing is replaced and the head put back on to the peristaltic pump.

The peristaltic pump is also very easy to use. This pump has a flow rate of 1 - 100 ml. This is not the fastest pump available but will be able to do the task that is given to it in a timely manner. If an eight-ounce drink is pumped from only one

of these peristaltic pumps, it would take roughly two and a half minutes from start to finish. This is not the best solution but meets our cost vs. performance requirements.

3.2-F: Liquid Level Sensors

Our project scope for the liquid level sensing was very basic; we needed to design a circuit that would tell us when a particular bottle was below a desired level. While searching different liquid measuring and sensor applications, there were many different types of electronics being used. There are 2 major categories for measurements, one being obtrusive and the other being unobtrusive.

- Float Valve Measuring (Obtrusive)
- Electro-Optic Measuring (Obtrusive)
- Ultrasonic (Unobtrusive)
- Capacitive Level Detection (Unobtrusive)
- Load Cell Weight Measuring (Unobtrusive)

The first and simple form of obtrusive measuring being done is by using a simple single pole switch attached to a float valve inside each of our liquid containers. The idea behind this method would be to use the switch in series with a pull-up resistor to act as a logic level “1” or “0” being sent to an input pin of an MSP430 microcontroller. Refer to *figure 3.2-1*; the changing level of liquid would cause the valve to disconnect.

Figure 3.2-1: Float Valve Switches

These types of simple float switches range in price from $5-$20. They are all primarily made out of nylon or polypropylene and are tolerant of alcohol dispersion such as ethyl. The normal size thread for these types of switches is 1/2” NPT

Another form of obtrusive sensing that would be relevant in our design was an electro-optic liquid level sensor. These operate on the principle of IR refraction. An LED IR beam is emitted and bent internally within the sensor. The sensor's output status would be dependent on the amount of IR light received at the receiver. The refraction caused by having a liquid at the sensor would lower the received IR past a threshold amount to change its output status. Refer to the *figure 3.2-2* below for a diagram of the electro-optic sensors operating theory.

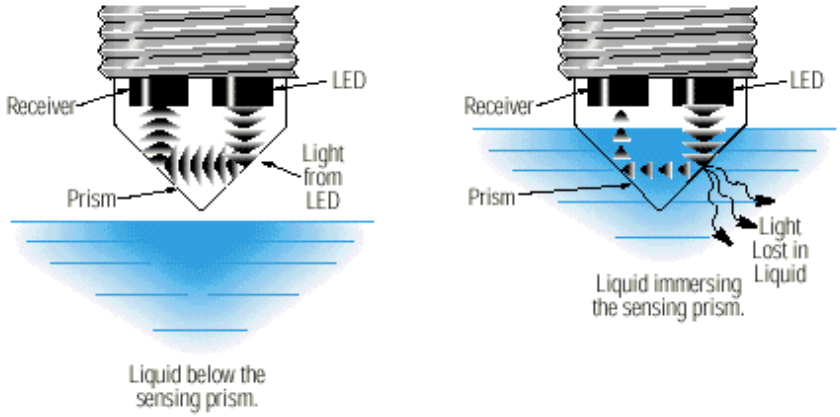

Figure 3.2-2: Electro-Optic Sensors

The advantage of the electro-optic sensors is their repeatability and switching speed. Using solid-state components, these types of switches are capable of 1mm accuracy of liquid levels. These types of electro-optic sensors range in price from $115-$250 per unit depending of material type.

The other types of liquid level sensors are known as unobtrusive. The idea behind these sensors is to use some other form of technology to measure liquid levels without physically touching or disturbing the liquid itself. This category of unobtrusive sensors caught our team's attention immediately as it had a few major advantages for our project. Because our project dealt with food and beverage, we would not have to worry about bacteria growth or the possibility of contamination of liquids by having various electronic sensors submerged into various liquid that were set for human consumption. We could easily change out fluids within our system without having to worry about sensors still containing trace amounts of other liquids inside the containers as well. Some of the types of unobtrusive sensors researched were ultrasonic, capacitive and load sensing.

Ultrasonic liquid sensors rely on the acoustic properties of liquid versus free air space to properly detect the presence of a liquid. *Figure 3.2-3* shows an example of an unobtrusive ultrasonic sensor application. The figure demonstrates the ease of use. These types of sensors prices begin at around $160 each.

Figure 3.2-3: Ultrasonic Sensor

Another similar technology to ultrasonic level detection is capacitive level detection. These sensors basic operating principle is similar to that of the ultrasonic sensor. These capacitive sensors are mounted onto the outside of plastic or fiberglass containers and measure the capacitance of various fluids inside. They can be calibrated to detect free air space capacitance versus a liquid's capacitance and deliver a correct corresponding output signal. These sensors are usually mounted to the sides of containers with adhesive or straps, examples of these are demonstrated with *figure 3.2-4*. These sensors are fairly expensive and costs per unit are similar to the ultra-sonic sensors.

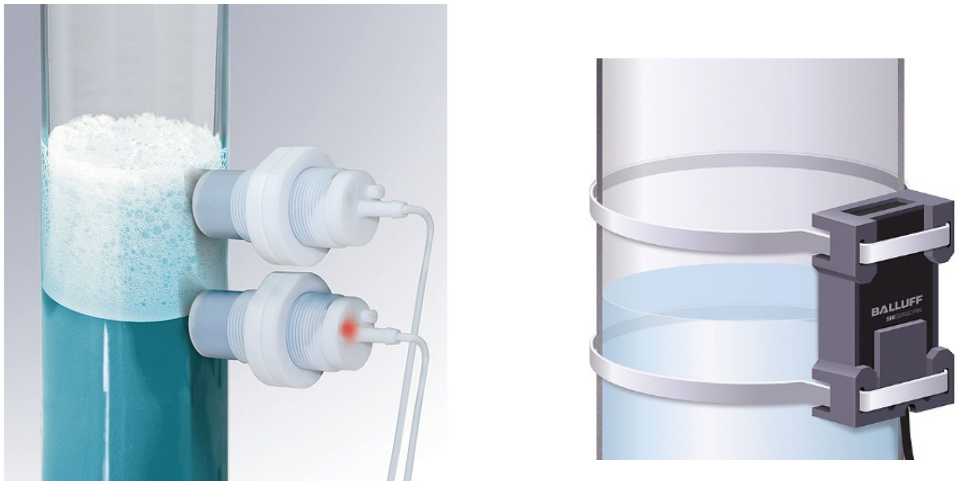

Figure 3.2-4: Ultrasonic Level Detection Mounts

The final and probably most simplistic version of unobtrusive sensors that we researched is load cell. The idea is to use a half bridge strain gauge load cell to monitor the weight of each liquid container. This method of measurement doesn't

rely on measuring the actual level of the liquid, but to monitor the weight of the remaining liquid inside each container. Load cells come in various weight ranges and sizes. Our drink wizard container volume capacities are to be limited to amount of the average size alcohol bottle which when full should weigh less than 5 pounds. The load cell could be actively monitoring the weight until a lower threshold is reached. A microcontroller subsystem would have the entire load cell inputs and trigger a desired system fault and display a corresponding error code to an LCD display. Pictures of various load cells can be seen in *figure 3.2-5*.

Figure 3.2-5: Load Cells

Image courtesy of Honeywell

3.2-G: Power Supply Components

The power supply requirements for the D*rink*W*izard* are as follows, the input voltage will range from 10 volt to 14 volt, the maximum input current will be 3.5 amps, and output voltages of 12 volts, 5 volts and 3.3 volts. The goal was to achieve at least 90 percent converter efficiency. The power supply components are reviewed below.

3.2-G-1: Step-Down (Buck) Converter - 3.3-volt

The input voltage for the D*rink*W*izard* will be in the range of 10 volts to 14 volts, however the MCU and Bluetooth module need lower voltages. These devices that require lower voltages need a step down converter to achieve the appropriate voltage. The simplest way to convert the voltage would be to use a linear regulator. The problem with the linear regulators is that energy is wasted as heat during their operation. The extra energy is just wasted and the whole assembly board heats up due to energy being converted to heat. The goal of achieving at least 90 percent converter efficiency could not be realized with this linear regulator. This is where the step-down buck converter outclasses linear regulators. Buck converters can be remarkably efficient, some are 95% or higher, making the buck converter an excellent choice for our integrated circuit. The first group of buck converters reviewed was the buck converters for the 3.3 volts need. The requirement for these buck converters was to have 3.3 volts output

and at least 0.5 amps output for the current. The buck converters reviewed for the 3.3 volts need are as follows, TPS62177, LMZ21700, TPS62150A and TPS54226.

The first 3.3-volt step-down buck converter reviewed was the Texas Instruments TPS62177. The TPS62177 is a synchronous step-down DC-DC converter with high efficiency. The operating range for the TPS62177 is 4.75 volts to 28 volts and has a maximum of 0.5 amps output for the current.

The second 3.3-volt step-down buck converter reviewed was the Texas Instruments LMZ21700. The LMZ21700 is a Nano module that produces step-down DC-DC solution for space-constrained applications. The operating range for the LMZ21700 is 3 volts to 17 volts and has a maximum of 0.65 amps output for the current.

The third 3.3-volt step-down buck converter reviewed was the Texas Instruments TPS62150A. The TPS62150A is a synchronous step-down DC-DC converter that is adjusted for applications with high power density. The operating range for the TPS62150A is 3 volts to 17 volts and has a maximum of 1 amp output for the current.

The last 3.3-volt step-down buck converter reviewed was the Texas Instruments TPS54226. The TPS54226 is an adaptive synchronous buck converter. The operating range for the TPS54226 is 4.5 volts to 18 volts and has a maximum of 2 amps output for the current.

These buck converters at first glance meet the requirements of 3.3 volts output as well as 0.5 amp output for the current. However there are other aspects to consider such as efficiency, cost, and size. These aspects are compiled into an easy to read table, *table 3.2-G-1*. The table for the 3.3-volt buck converters is shown below.

Converter	Price	Size (mm2)	Efficiency	Vout (MAX)	Iout (MAX)
TPS62150A	$1.63	81	88%	6.3 V	1 A
TPS54226	$1.54	226	93.6%	5.5 V	2 A
TPS62177	$1.17	60	85%	3.3 V	0.5 A
LMZ21700	$1.76	39	82%	6 V	0.65 A

Table 3.2-G-1: 3.3-Volt Buck Converters

One of the goals of this power supply design was to achieve at least 90% efficiency for the converters. This goal alone rules out most of the buck converters reviewed. Another parameter that could be a problem if there are space constraints is the size of the converter. The D*rink*W*izard* does not have space constraints so this parameter is not a concern for this design. The price

parameter does concern the group members. Since the group is paying for this out of our pockets, the group wants the best bang for their buck. The group wants the most cost effective part that meets the design restraints. The converter that does this for the 3.3-volt need is the Texas Instruments TPS54226.

3.2-G-2: Step-Down (Buck) Converter - 5-volt

The second group of buck converters reviewed is the buck converters for the 5-volt need. The requirements for this group of buck converters are to have 5 volts output and at least 1 amp of output for the current. The buck converters reviewed below are the TPS563200, TPS562209, and LMZ21701.

The first 5-volt step-down buck converter reviewed was the Texas Instruments TPS563200. The TPS563200 is a simple synchronous step-down converter. The operating range for the TPS563200 is 4.5 volts to 17 volts and a max output current of 3 amps.

The second 5-volt step-down buck converter reviewed was the Texas Instruments TPS562209. The TPS562209 is another simple and easy to use synchronous step-down converter. The operating range for the TPS562209 is 4.5 volts to 17 volts and a max current output of 2 amps.

The last 5-volt step-down buck converter reviewed was the Texas Instruments LMZ21701. The LMZ21701 is a simple Nano module that is an easy to use step down DC-DC converter for space-constrained applications. The operating ranges for the LMZ21701 is 3 volts to 17 volts and has a max output current of 6 amps.

These buck converters at first glance meet the requirements of 5 volts output as well as 1 amp output for the current. However there are other aspects to consider such as efficiency, cost, and size. These aspects are compiled into an easy to read table, *table 3.2-G-2*. The table for the 5-volt buck converters is shown below.

Converter	Price	Size (mm2)	Efficiency	Vout (MAX)	Iout (MAX)
TPS562209	$0.95	76	90%	7 V	2 A
TPS563200	$1.05	149	92%	7 V	3 A
LMZ21701	$1.96	39	85%	6 V	6 A

Table 3.2-G-2: 5-Volt Buck Converters

One of the goals of this power supply design was to achieve at least 90% efficiency for the converters. Another parameter that could be a problem if there are space constraints is the size of the converter. The D*rink*W*izard* does not have space constraints so this parameter is not a concern for this design. The price parameter does concern the group members. Since the group is paying for this

out of our pockets, the group wants the best bang for their buck. The group wants the most cost effective part that meets the design restraints. The converter that does this for 5-volt need is the Texas Instruments TPS563200. The group could have gone with TPS562209 because it met the goal of at least 90 percent efficient, however for the small increase in cost the group decided to go with the most efficient converter reviewed.

3.2-G-3: Buck-Boost Controller – 12-volt

The third group of buck-boost controllers reviewed is the buck-boost controllers for the 12-volt need. The requirements for this group of buck-boost controllers are to have 12 volts output and at least 2 amps of output for the current. The buck-boost controllers reviewed below are the LM5175, LM5022, LM3481, and LM3478.

The first 12-volt buck-boost controller reviewed was the Texas Instruments LM5175. The LM5175 is a synchronous four-switch buck-boost DC/DC controller. The LM5175 is capable of regulating the output voltage at, above, or below the input voltage. The operating range of the LM5175 is 3.5 volts to 42 volts and has a max current output of 4 amps.

The second 12-volt buck-boost controller reviewed was the Texas Instruments LM5022. The LM5022 is a high voltage MOSFET controller for use in boost regulators. The operating range for the LM5022 is 6 volts to 60 volts and max current output of 20 amps.

The third 12-volt buck-boost controller reviewed was the Texas Instruments LM3481. The LM3481 is a performance controller for switching regulators. The LM3481 can be operated at extremely high switching frequency. The operating range for the LM5022 is 2.97 volts to 48 volts and a maximum current output of 20 amps.

The last 12-volt buck-boost controller reviewed was the Texas Instruments LM3478. The LM3478 is a low-side MOSFET controller for switching regulators. The LM3478 can be operated at extremely high switching frequency. The operating range of the LM3478 is 2.97 volts to 40 volts and a maximum current output of 20 amps.

These buck-boost controllers at first glance meet the requirements of 12 volts output as well as 2 amps output for the current. However there are other aspects to consider such as efficiency, cost, and size. These aspects are compiled into an easy to read table, *table 3.2-G-3*. The table for the 12-volt buck-boost controller is shown below.

Converter	Price	Size (mm2)	Efficiency	Vout (MAX)	Iout (MAX)
LM5175	$8.51	552	98%	55 V	4 A
LM5022	$3.71	460	86%	500 V	1 A
LM3478	$3.81	665	88%	500 V	20 A
LM3481	$3.80	683	88%	500 V	20 A

Table 3.2-G-3: 12-Volt Buck-Boost Controller

One of the goals of this power supply design was to achieve at least 90 percent efficiency for the converters. Another parameter that could be a problem if there are space constraints is the size of the converter. The D*rink*W*izard* does not have space constraints so this parameter is not a concern for this design. The price parameter does concern the group members. Since the group is paying for this out of our pockets, the group wants the best bang for their buck. The group wants the most cost effective part that meets the design restraints. The buck-boost controller that does this for 12-volt need is the Texas Instruments LM5175. The LM5175 jumps up the cost however it is 98 percent efficient which is extremely impressive. The LM5175 is the only controller that meets our group requirement of 90 percent efficiency so the cost is worth it.

After reviewing the different step-down buck converters and buck-boost controllers the final parts came out to be the TPS54226 for the 3.3-volt output, the TPS563200 for the 5-volt output, and the LM5175 for the 12-volt output. Once all the different parts are put together the power supply has an overall efficiency of 97.09%, an overall footprint of 744 mm2 and a total cost of $11.93. Achieving 97 percent efficiency for only $11.93 is impressive and our group is satisfied with our overall design.

3.2-G-4: Power Supply Batteries

The D*rink*W*izard* is intended to be used outside or out at tailgating events. In order to produce this portability the D*rink*W*izard* will need to have battery power capabilities. The battery topology that is selected will need to meet the power requirements presented to it. The battery topology that is chosen will have to run, under a worst-case condition, 9 1.3-volt pumps that will be used to pump the fluids for the D*rink*W*izard*. The selected battery topology must also power the D*rink*W*izard's* microcontroller unit and the Bluetooth module. The following will consider the most common battery topologies and review them for the purpose of use in the D*rink*W*izard* project.

3.2-G-4.1: Nickel Cadmium (NiCad)

The first battery topology investigated is the nickel cadmium (NiCad) topology. The NiCad is a strong workhorse; it is the only battery that performs well under demanding working conditions. NiCad batteries do not perform well if they

remain sitting in chargers for days and only used for brief periods. A periodic full discharge is imperative to the lifespan of a NiCad battery, without the periodic full discharge large crystals will form on the cell plates, most commonly referred to as memory, and the performance of the NiCad battery gradually deteriorates. Some advantages of the NiCad battery are fast charging times, even after prolonged storage. NiCad batteries can take a high number of charging and discharging cycles, proper maintenance will provide over 1000 charging and discharging cycles.

The NiCad battery has a long shelf life and can be stored in any state of charge. An important factor of the NiCad battery is that if abused it is very forgiving; this makes it one of the most rugged rechargeable batteries out there. NiCad batteries are economically prices and hold the lowest cost battery in terms of cost per cycle. A few disadvantages of the NiCad battery is a relatively low energy density, especially when compared to newer topologies. The memory effect that NiCad batteries have, thus they must be periodically drained to prevent this memory effect.

A concern for the D*rink*W*izard* project is that NiCad batteries are environmentally unsafe; they contain toxic metals and are even being limited in some countries. Another disadvantage of NiCad batteries is they need recharging after storage, so if the D*rink*W*izard* has not been used in a while the customer will have to charge the D*rink*W*izard* batteries before going to a tailgate or outdoor cookout that has no power supply. After reviewing the advantages and disadvantages the conclusion was this battery topology will not work for the D*rink*W*izard* project; while the price point, long shelf life, and rugged performance were all appealing aspects, the disadvantage of not being ready to use after storage and periodic needs for discharge ruled this topology out.

3.2-G-4.2: Nickel-Metal Hydride (NiMH)

The next battery reviewed was the nickel-metal hydride, NiMH, battery. NiMH batteries date back to the 1980's where new alloys were developed that was stable enough to use in a cell. The success of NiMH batteries has been driven by the high energy density as well as the use of environmentally friendly metals. Compared to the nickel cadmium battery, NiMH batteries contain up to 40 percent higher energy density. Another advantage over the nickel cadmium battery is NiMH batteries are less prone to memory effect. The periodic deep cycles are not mandatory as often as they are with the nickel cadmium batteries. Some disadvantages of the NiMH battery are limited service life, continually deep cycling this battery drastically drops the performance. The NiMH battery also requires a longer charge time compared to the nickel cadmium battery and the NiMH generates more heat during charging. A trickle charge is essential and must be controlled vigilantly.

NiMH batteries also have a high self-discharge rate, about 50 percent higher than nickel cadmium batteries. A major concern for the D*rink*W*izard* project is the performance degradation if stored at elevated temperatures. NiMH batteries should be stored in a cool place and be on a constant charging state. Like the nickel cadmium battery the NiMH battery is high maintenance and requires routine full discharge to prevent crystalline formation. While the nickel-metal hydride battery has the advantage of being environmentally friendly, there is way more disadvantages for the use of this battery in this project. The customer would have to continually monitor the charge and discharge condition as well as continually have the vending unit plugged, thus keeping a constant charge on the NiMH battery. The cost versus advantages is not there as well; all together this is not a good choice for this design.

3.2-G-4.3: Lead Acid

The third battery topology reviewed was the lead acid battery. The lead acid battery was the first rechargeable battery for commercial use, invented back in 1859. During the 1970's researchers were able to develop a maintenance-free lead acid battery. This maintenance-free battery was sealed and allowed the battery to be operated in any position; it no longer has to sit vertically. To ensure consumer safety, valves were added to allow venting of gas during charge and discharge.

The biggest advantage of the lead acid battery compared to the nickel cadmium battery is that lead acid batteries are not subject to the memory effect. Lead acid batteries also have the best charge retention among rechargeable batteries; lead acid batteries take nearly a year to discharge 40 percent of its stored energy. One major disadvantage of the lead acid battery is that full discharges causes extra burden on the battery and each deep cycle drains the battery of a small amount of the overall capacity. Another disadvantage is that lead acid batteries have a longer charge time; the average charge times are 8 to 16 hours. This means that lead acid batteries must be stored in a charged state to not only drop the charge time but also because storing lead acid batteries in a discharged state can cause battery sulfating. Battery sulfating is a condition that makes the battery difficult, more like impossible, to recharge.

A few advantages of the lead acid battery are that they are inexpensive; in terms of cost per watt-hour they are the lowest cost. The lead acid is a mature and well-understood technology; if used correctly the lead acid battery is durable and will provide dependable service. Lead acid batteries also have low maintenance requirements, no memory effects. The lead acid battery is also capable of high discharge rates. Some downfalls of the lead acid battery are that is must be stored in a charged condition. The energy to weight ratio is low, this limits the battery to stationary or wheeled applications. The lead acid battery only allows a limited number of full discharge cycles. The lead acid battery is also

environmentally unfriendly. Another concern is thermal runaway that can occur with improper charging. The main advantage of this battery is the cost. This is a very common battery topology and these batteries can be obtained at a reasonable cost. If the lead acid battery topology were the one that is chosen for the D*rink*W*izard* project, then the design of the vending unit would possibly have to change. The lead acid batteries are relatively heavy in a portability sense, this would require the project to investigate a wheeled designed to make transportation easier.

3.2-G-4.4: Lithium Ion

The next battery reviewed was the lithium ion battery. Rechargeable lithium ion batteries were developed around the 1980's but were not safely implemented till 1991. Lithium ion, Li-ion, is the fastest growing as well as most promising battery topology. The energy capacity of Li-ion is usually twice the capacity of the standard nickel cadmium battery. Li-ion batteries are also low maintenance and require no scheduled cycling as well as there are no memory effect. Li-ion batteries are fragile and require protection circuits to maintain safe conditions. Another major concern with Li-ion batteries is the aging concern. Capacity deterioration has been reported after one year. This deterioration happens if the battery is used or not. After three years Li-ion batteries frequently fail. Another drawback is that manufactures recommend storing the Li-ion battery in a cool place to slow down the aging process. This project being leveraged to be an outdoor companion goes against this cool storage recommendation. The most notable advantages of Li-ion batteries are the high-energy capacity, with a potential for even higher capacities.

Relatively low self-discharge, compared to nickel cadmium and nickel-metal hydride batteries the self-discharge rate is less than half. The most important factor of Li-ion batteries is the low maintenance; this is the lack of memory effect as well as the lack of need of periodic discharges. A few of the disadvantages are the requirement of a protection circuit; this circuit limits the voltage and current. The major disadvantage is the aging of the battery, even when not in use. Another major disadvantage of Li-ion batteries is the cost, cost are usually 40 percent higher than the cost to manufacture nickel cadmium batteries. Cost is an important factor the group is looking at and with cost being 40 percent higher and a nearly 100 percent guarantee the battery will fail after three years, this Li-ion topology is ruled out for this project.

3.2-G-4.5: Lithium Polymer

The fifth, and last battery, topology reviewed was the lithium polymer battery. The original design of Li-polymer dates back to the 1970's and it uses a dry solid polymer electrolyte. This polymer resembles a plastic-like film and offers ruggedness and safety. The safety aspect comes from the lack of danger from

flammability. Since no liquid or gelled electrolyte is used there is no danger of flammability. The downfall of a completely dry lithium polymer is that the dry polymer suffers from poor conductivity. The resistance is too high and the current demands for most modern devices are not feasible. To make the lithium polymer more conductive, some manufactures have added some gelled electrolyte. This hybrid combination gave rise to lithium ion polymer and is the most common topology found in most mobile devices today. The major advantage of the Li-polymer topology is the form factor; the batteries are wafer-thin, about the size of a credit card. The Li-polymer topology is also lightweight; at the size of a credit card they save space and weight. The safety aspect of Li-polymer is also improved; the design is more resistant to overcharging and there is less chance of electrolyte leaking out. Some major disadvantages of the Li-polymer topology are the low energy density. The major disadvantage that this group is concerned with is the expensive manufacturing cost. The advantages of this battery topology do not add any benefit to the D*rink*W*izard* project; the only advantage we would get from this topology is the lightweight and space saving characteristics. We are not that concerned with the weight and have room for batteries. The cost is the biggest deterrent and is the reason we will look at another topology for the D*rink*W*izard* project.

3.2-G-4.6: Battery Final Plan

After reviewing all the advantages and disadvantages of the different battery topologies, one topology fits the needs of the Dr*ink*W*izard* project. The topology chosen for this project is going to be the lead acid battery. This battery is the most common and dependable battery around. The cost is of the most important factor to this group and we can obtain these batteries for free. One of the group members already has a pair of these batteries and will provide these batteries for the project. The project will be modified and adjusted to account for the disadvantages that the lead acid topology had.

3.2-G-5: Power Supply Schematics

The power supply is a major part of the D*rink*W*izard* project, without power there is no D*rink*W*izard.* With the power supply being such a vital part to the success of this project the group wanted to make sure it was designed correctly. A vital tool for designing a power supply is Texas Instruments WEBENCH Designer. The WEBENCH Designer helps determine parts as well as helps produce schematics and PCB board layouts. The D*rink*W*izard* will have an operating range of 10-14 volts and output voltages of 12, 5 and 3.3 volts. The current needs from the power supply will be 0.5 amps, 1 amp and 2 amps. Taking these needs and inputting them into the WEBENCH Designer we can obtain the following schematics. The first schematic that can be seen in *figure 3.2-G-5.1* is the 3.3-volt design with a current rating of 0.5 amps.

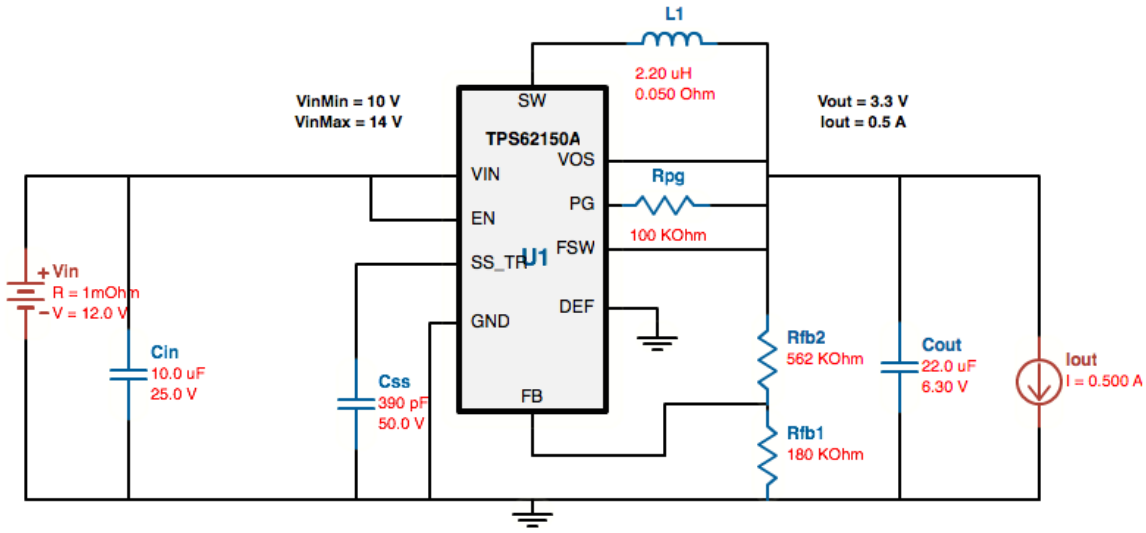

Figure 3.2-G-5.1: 3.3-Volt Power Supply Schematic

Image courtesy of Texas Instruments

From this schematic, that were created using Texas Instruments WEBENCH Designer, we can see all the resistor, capacitor and inductor values needed to produce the output voltage of 3.3 volts and an output current of 0.5 amps. The step-down buck converter that was used is the Texas Instruments TPS62150A.

The next schematic that needed to be designed in the WEBENCH Designer is the 5-volt output and the 1 amp current output. The schematic that was created using the WEBENCH Designer can be seen below in *figure 3.2-G-5.2*.

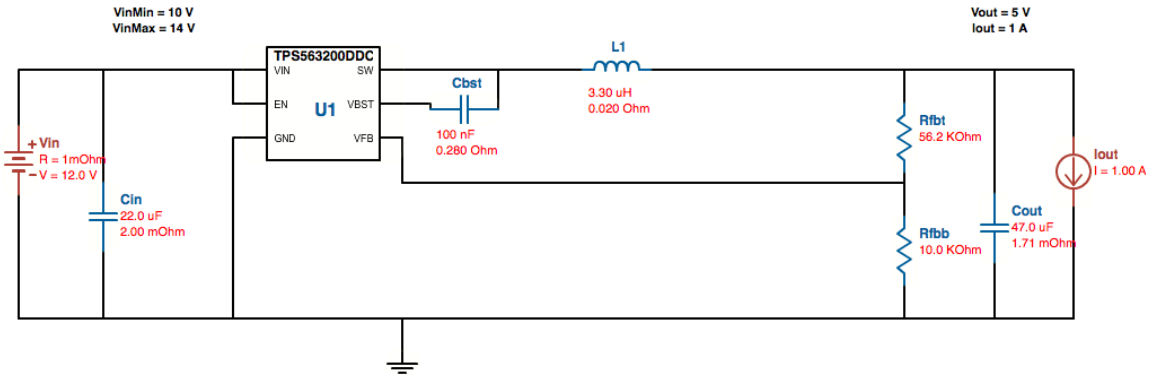

Figure 3.2-G-5.2: 5-Volt Power Supply Schematic

Image courtesy of Texas Instruments

From this schematic, that were created using Texas Instruments WEBENCH Designer, we can see all the resistor, capacitor and inductor values needed to

produce the output voltage of 5 volts and an output current of 1 amp. The step-down buck converter that was used is the Texas Instruments TPS563200DDC.

The last schematic that needed to be designed in the WEBENCH Designer is the 12-volt output with the 2 amps of current output. The schematic that was created using the WEBENCH Designer can be seen below in the following three figures, *figure 3.2-G-5.3*, *figure 3.2-G-5.4*, and *figure 3.2-G-5.5*.

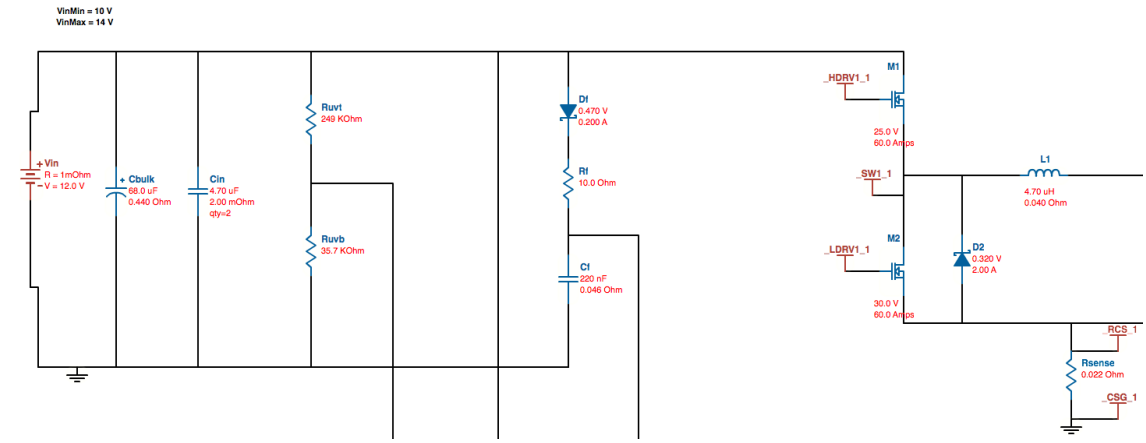

Figure 3.2-G-5.3: 12-Volt Power Supply Schematic 1

Image courtesy of Texas Instruments

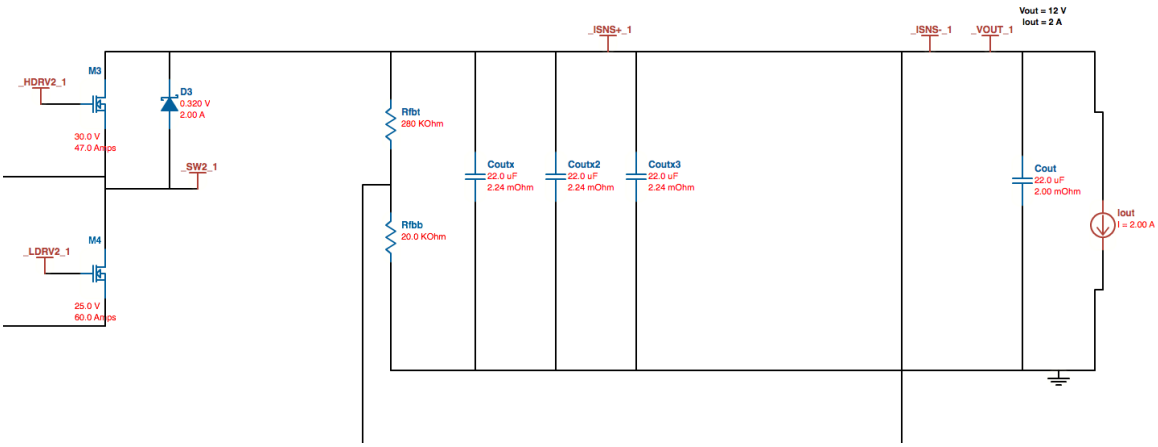

Figure 3.2-G-5.4: 12-Volt Power Supply Schematic 2

Image courtesy of Texas Instruments

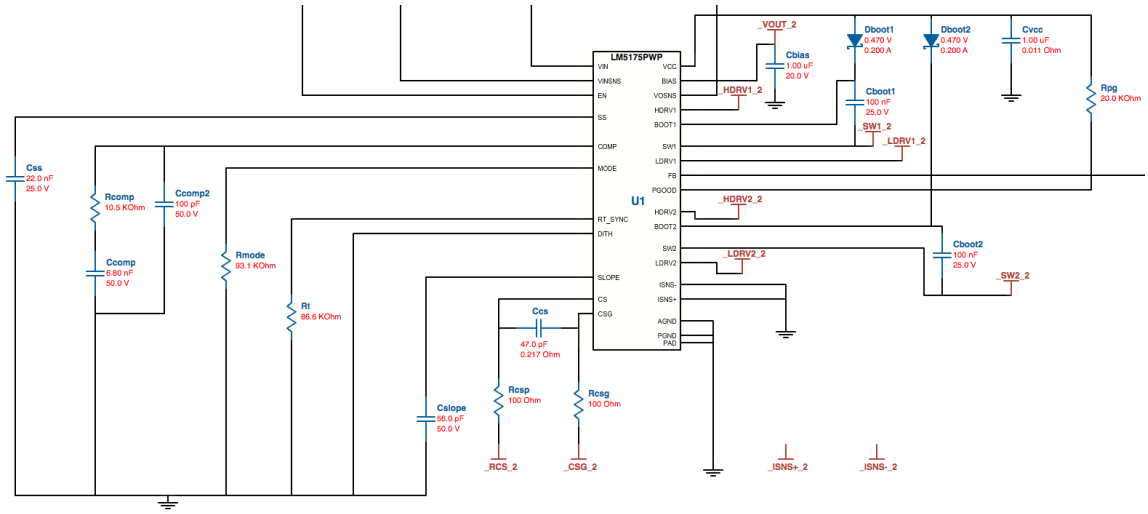

Figure 3.2-G-5.5: 12-Volt Power Supply Schematic 3

Image courtesy of Texas Instruments

From these schematics, that were created using Texas Instruments WEBENCH Designer, we can see all the resistor, capacitor and inductor values needed to produce the output voltage of 12 volts and an output current of 2 amps. The buck-boost controller that was used is the Texas Instruments LM5175PWP.

3.2-G-6: Power Supply PCB

The power supply components are extremely small. The components will have to be installed on a PCB board, printed circuit board. Using the PCB board allows our group to design a power supply circuit that is cheaper and speedier than other traditional wiring methods. Another advantage of the PCB board is that components are mounted and wired to one single part, the traditional wiring errors are thus eliminated. The PCB board will have to be designed in a board construction program, such as CadSoft Eagle version 7.2. From this software our PCB board specs can then be sent off to a company to be produced for us. This outsourced building can also mount some of the smaller components for us. Having a third party mount some of the smaller components eliminates the chances of group members damaging the traces on the board. To ensure a proper PCB board configuration was picked, the Texas Instruments WEBENCH Designer was used once again. The WEBENCH Designer had a lot of helpful features such as a thermal simulation. A simulation could be ran on our PCB board layout to determine where the heat would build up on normal operating conditions. This simulation was very effective to routing components that are sensitive to heat away from those sources that produced the most heat.

The first PCB board that was designed in the Texas Instruments WEBENCH Designer was the board containing the Texas Instruments TPS62150A step-down buck converter. This board can be seen in *Figure 3.2-G-6.1* below.

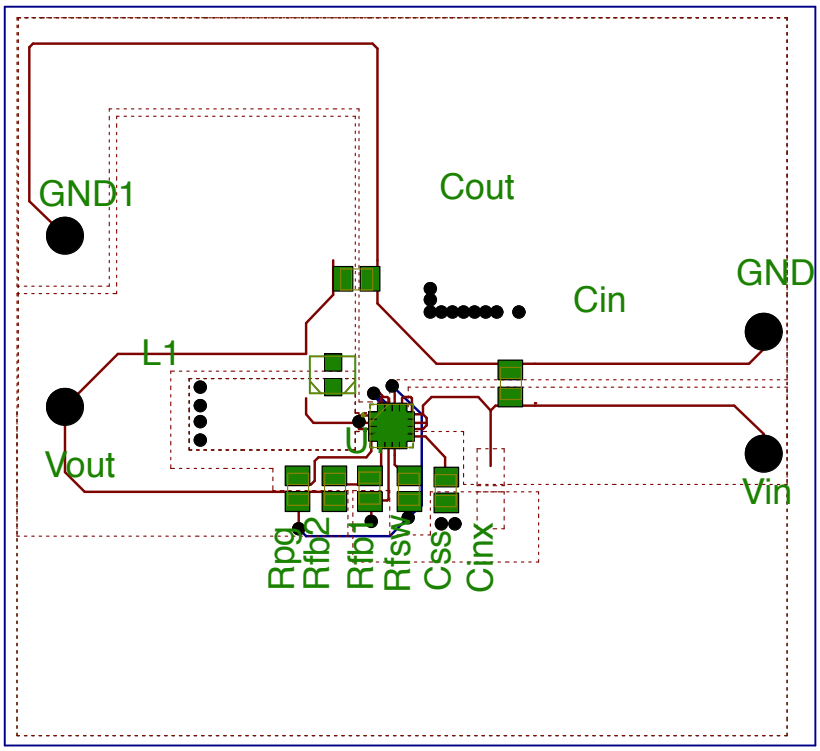

Figure 3.2-G-6.1: 3.3-Volt Power Supply PCB Board

Image courtesy of Texas Instruments

The second PCB board that was designed in the Texas Instruments WEBENCH Designer was the board containing the Texas Instruments TPS563200DDC step-down buck converter. This board can be seen in *Figure 3.2-G-6.2* below.

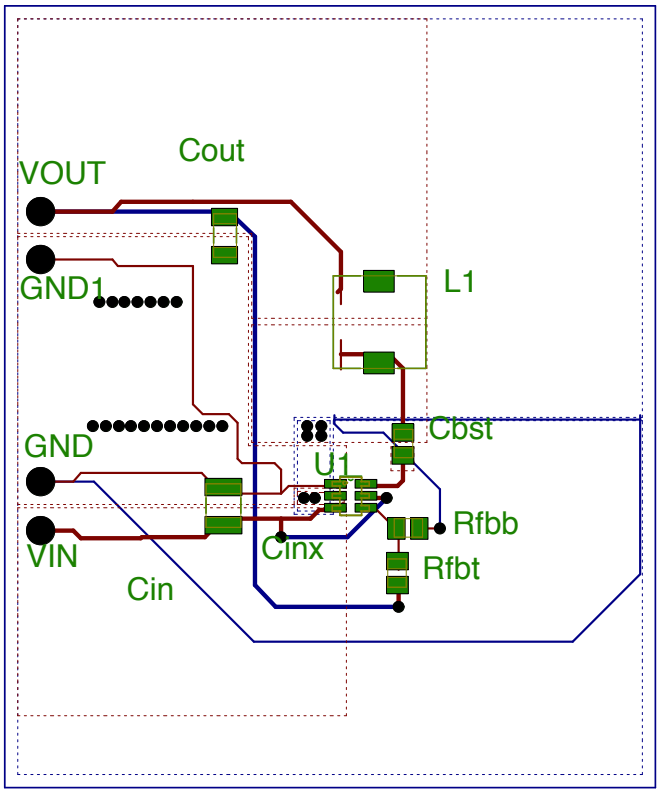

Figure 3.2-G-6.2: 5-Volt Power Supply PCB Board

Image courtesy of Texas Instruments

The last PCB board that was designed in the Texas Instruments WEBENCH Designer was the board containing the Texas Instruments LM5175PWP buck-boost controller. This board can be seen in *Figure 3.2-G-6.3* below.

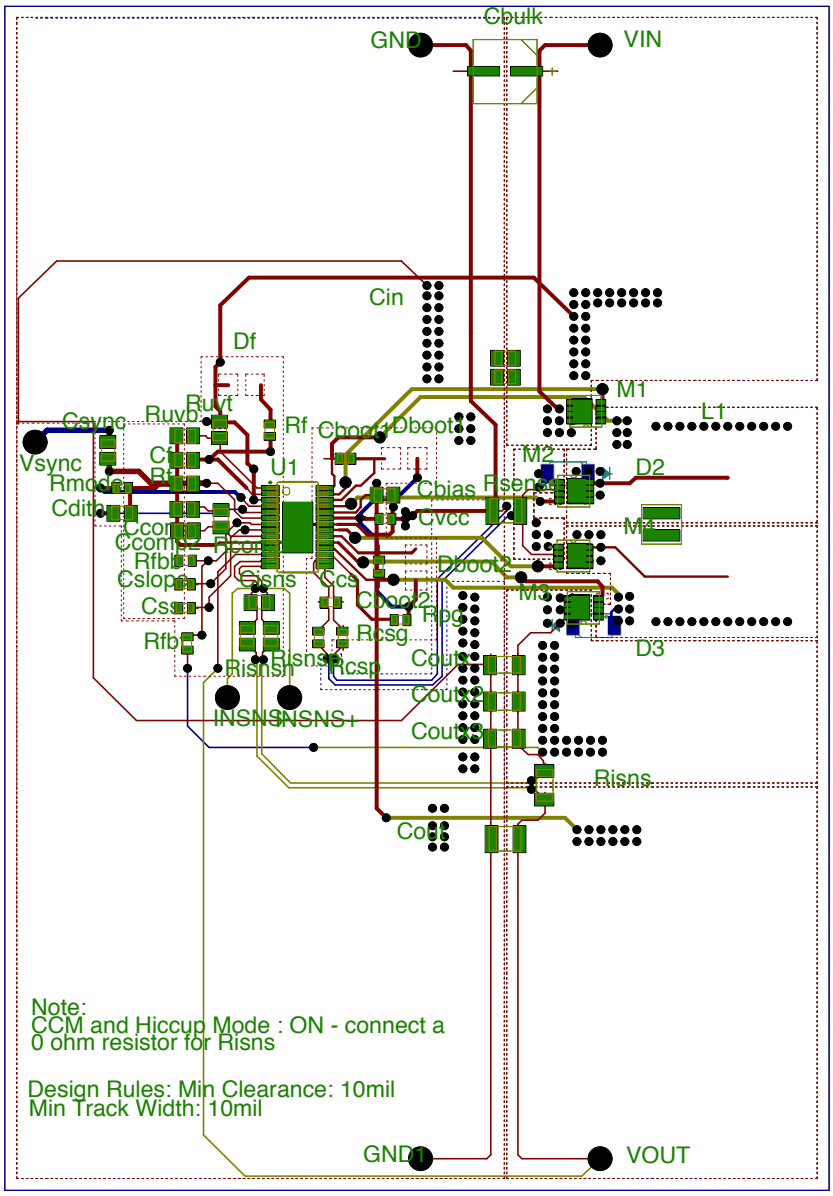

Figure 3.2-G-6.3: 12-Volt Power Supply PCB Board

Image courtesy of Texas Instruments

3.3-A: Ultrasonic Sensors

Ultra-sonic sensors are widely used in many different applications. There are also many different types of ultra-sonic sensors available on the market today that can meet a variety of design specifications that may be needed. The four basic types of ultrasonic sensors are proximity sensors, ultrasonic through beam sensors, ultrasonic retro-reflexive sensors and ultrasonic two point proximity switches. The only one relevant to our project is the ultrasonic proximity sensor.

Ultrasonic sensors are devices that can produce and transmit ultrahigh frequency waves. These waves are well above the audible frequency of human hearing, which is within the range of 1 Hz and 20 kHz. These waves are transmitted from the device and reverberate off of an object that is within the range of the ultrasonic sensor.

The distance from the object is proportional to the time it takes to send and receive the sound waves. Once the speed of sound in air can relatively be considered a constant, around $340\frac{m}{s}$, the time that it takes for the sound waves to return to the device can be used to find the distance from the device to the object in question. The distance could be equated using the formula in *figure 3.3-A-1*.

$$Distance = \frac{1}{2} t_{echo} * v_{sound}$$

Figure 3.3-A-1: Distance Equation

The sending of the signal is called transmitting, while the receiving of the signal is called the echo. A microprocessor can be used to trigger the transmission of the original signal. The microprocessor then begins recording the time it takes to receive the echo back. The microprocessors then process that time and calculate the distance from the object that the ultrasonic sensor is.

Since this device is so versatile, there are many applications that an ultra-sonic sensor can be used. For example, autonomous robots use ultrasonic sensors to be able to accurately avoid obstacles. Once ultrasonic sensors use sound rather than light, they are used in the automotive industry for things such as self-parking cars and UAV navigation. Recently ultrasonic transducers have been used to clean things such as jewelry, guns and teeth. Ultrasonic sensors can also be used to detect high-pressure gas leaks or other hazardous conditions that would produce high frequency sound waves.

Ultrasonic sensors have many advantages over other types of similar sensors. For example ultrasonic sensors do not rely on surface color or the optical reflectivity of an object. So this means that the sensing of a white paper plate or a clear glass plate would result in the same conclusion. Also the ultrasonic sensors with the digital outputs have an excellent repeatable sensing accuracy. Another advantage to using an ultrasonic sensor is that they work regardless of lighting condition, since they rely on sound waves and not light.

Of course there are also downsides to using an ultrasonic sensor. Ultrasonic sensors are only accurate within a certain range. The object has to be a minimum distance away from the ultrasonic sensor to be able to detect the distance of the object. This is due to the timing of the transmission of the signal and being able to receive the echo back. On the flip side, the object has to be

within the range of the ultrasonic sensor. The transmitted signal has a certain amount of distance it can travel before the signal loses too much energy to travel the full distance back to the sensor or the sound waves are absorbed into the object. Another downside of the ultrasonic sensor is that the distance is being measured with sound waves and certain materials or surfaces can cause an error in the measurement.

The ultrasonic sensor that will be used in the D*rink*W*izard* is the HC-SR04. This device has two transducers onboard. One transducer will transmit the ultrasonic frequency sound waves, while the other will receive the ping from the reverberation of the original transmission. The device uses a 5-volt power source to run the onboard circuitry that controls the transmission signal. The device is a four pin device with the pins having one pin for the 5 volt input, one pin for the send command coming from the microcontroller, and one transmit pin for the receiving of the reverberated signal.

These sensors range from inexpensive to very expensive. Our sensor is for detecting objects that are fairly close to the sensor, so a super elaborate sensor would not be necessary. In fact, the sensor we are choosing to use is towards the lower end models costing around 3 dollars per sensor. This will keep this project within the proposed budget and will keep overall cost down, because all of the necessary timing circuitry is already included onboard. The characteristics for this specific ultrasonic sensor can be seen in the *table 3.3-A-2*.

The microcontroller will have to send a timed pulse in order to operate the sensor. The sensor will then respond with a pulsed signal back and then the microprocessor will begin its calculations. This will be discussed later in the cup and liquid level sensors section.

HC-SR04	Minimum	Maximum
Operating Voltage	4.5 V	5.5 V
Operating Current	10 mA	20mA
Idle Current	1.5 mA	2.5 mA
Range Detected	1 inch	13 feet

Table 3.3-A-2: Characteristics for HR-SR04 Ultrasonic Sensors

3.3-B: Breathalyzer

In an effort to make the D*rink*W*izard* more marketable to a potential bar owners, we decided to do some research into incorporating a reusable stand-alone Breathalyzer system into our android tablet ordering platform. We believe that many bar owners would be interested in reducing the liability of over-serving patrons who could potentially drive a motor vehicle after consuming too much alcohol. Our D*rink*W*izard* takes one additional step to protect bar owners by making each patron pass a sobriety test before being able to order an additional alcoholic beverage.

Our research led us to explore the existing systems and possibilities of incorporating a Breathalyzer into our system. The major component of the Breathalyzer is alcohol sensor itself. It measures the presence of ethanol alcohol in a person's breath. The principle concept behind the Breathalyzer is that there is a direct relationship between the amount of alcohol on a person's exhaled air and how much alcohol you could find in that same person's blood. Most states have a driving legal limit of .08grams of alcohol per 100ml of blood. Breathalyzers need to have a calibrated noise floor to operate properly.

Alcohol sensors work by having a certain type of semiconductor to detect and react when in the presence of ethanol gas. A very important part of this semiconductor is to have it heated to an optimal operating temperature to achieve any sort of repeatable accuracy. When the semiconductor comes in contact with the ethanol molecules, the electrical resistance changes and is measured to estimate the Blood Alcohol Content of the given sample. An example of an alcohol sensor can be seen in *figure 3.3-B-1*.

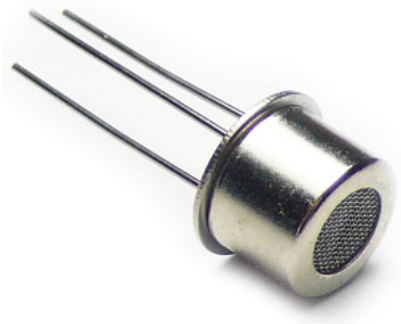

Figure 3.3-B-1: Alcohol Sensor

Image courtesy of Seeedstudio.com

Hand-held pocket Breathalyzers has become very popular in the commercial market today. There are many types and sizes that have already been fabricated and designed. Some units offer stand-alone battery-powered capabilities, while

others offer USB powered options. All of the designs in the market today have some form of LED or LCD display to output the results of each test taken. The concept to integrate this subsystem into the D*rink*W*izard* is to purchase a complete USB powered Breathalyzer device and explore the circuitry to find the proper output signal, and use it to ultimately control a disable mode on our drink wizard. An example of a micro-USB powered Breathalyzer can be seen in *figure 3.3-B-2*.

Figure 3.3-B-2: Micro-USB Breathalyzer

Image courtesy of Ipega

4.0 Related Standards

4.1 Health and Safety

Our group project, the D*rink*W*izard,* has some health and safety standards that the group must review and incorporate in our design. The review and evaluation of these standards is imperative to our design because these standards dictate what materials we can use. The health and safety standards that pertain to the D*rink*W*izard* are shown below.

4.1-A: Vending Machines for Food and Beverages

The first health standard that was reviewed was the NSF/ANSI 25, this standard covers vending machines for food and beverages. The NSF/ANSI 25 standard establishes minimum food protection and sanitation requirements for materials, design, construction, and performance. This standard's stipulations are intended to safeguard beverages from contamination and to ensure that all construction materials resist wear, vermin, and the effects of heat, sanitizers, and other substances that may come into contact with the machines. This standard is especially important to our design if our group decides to eventually commercialize this product. If we decide to go the commercial route, the original design of using wood to house our main vending unit will have to be evaluated and possibly substituted for an alternative material.

4.1-B: Drinking Water System Components

The second health standard that was reviewed was the NSF/ANSI 61, this standard covers drinking water system components. The NSF/ANSI 61 standard establishes minimum health effects requirements for the chemical contaminants and impurities that are indirectly imparted to drinking water from products, components, and materials used in drinking water systems. This standard pertains to our design when the group is looking for fluid storage containers. This will also pertain to the group's selection of fluid tubing. The fluid containers and fluid tubing that the group chooses must be of food grade and contain no chemicals that could pollute the fluids we place in them. The wrong selection in the design could bring up costly and damaging last minute changes in the final product. This standard is important for the group to pick the proper tubing and containers the first time.

4.1-C: Standard Symbols Used for Safety

The first safety standard that was reviewed was the NFPA 170; this standard provides standard symbols used to communicate fire safety, emergency, and

associated hazards information. This standard uses easily understood uniform symbols on labels and signs to provide consistency and eliminate confusion while promoting communication. The D*rink*W*izard* varies from most other vending machines out there; it contains batteries, charging components for those batteries and alcohol containers. Following this safety standard the safety symbols of no smoking, and flammable liquid would need to be incorporated onto our vending unit somewhere. The no smoking symbol is important due to the flammable characteristic of some alcohol. The symbol for flammable liquid would need to be used to inform people, mostly firefighters, of the flammable contents of the vending machine. This safety standard is something for our group to consider, especially if the group would like to commercialize the D*rink*W*izard.*

The last symbol would be used to inform anyone that the vending machine contains an electrical hazard. The entry into the vending machine could cause electrical shock if not handled with proper care.

4.4 Wireless Communication Standards

The Institute of Electrical and Electronics Engineers (IEEE) previously set the standards for wireless communication through Bluetooth using the 2.4-gigahertz frequency band. IEEE 802.11b and IEEE 802.11g set specifications for Wi-Fi and the 2.4-gigahertz band [5]. The IEEE 802.15 standard was created specifically for Bluetooth but Bluetooth Special Interest Group now maintains Bluetooth standards. IEEE 802.11b was first approved in July of 1999 and establishes a maximum data rate of 11 megabits per second but the typical data rate is almost half at 5 megabits per second. In some applications, the range established by this standard is 30 meters. Class 2 Bluetooth technologies have a maximum range of approximately ten meters. Furthermore, IEEE 802.11b uses modulation known as Complementary Code Keying (CCK), which is a modified version of Code Division Multiple Access (CDMA) [5].

The IEEE 802.11b standard regulates chips to have a data transfer rate of 11 megabits per second. This data rate is for ideal conditions. If there is interference or a loss of signal for any reason, the data rate is restricted to 5.5 megabits per second to allow for more error correction [5]. Should the signal deteriorate to an even worse state, the data rate will fall to 2 megabits per second and then the lowest possible of 1 megabit per second. The slower the rate, the more error correction can be done [5]. Practically the maximum possible data rate is about 5.9 megabits per second using Transmission Control Protocol (TCP).

Institute of Electrical and Electronics Engineers 802.11g improved upon IEEE 802.11b and was approved in June 2003. It offers a higher maximum data rate of 54 megabits per second. Although Orthogonal Frequency Division Multiplex (OFDM) achieves this, this rate is not generally reached and instead is capped

around 24 megabits per second [6]. To maintain backwards compatibility, IEEE 802.11g uses several different types of modulation. Extended Rate Physicals (ERPs) are used as a link between the 802.11b and 802.11g exchanges. Direct Sequence Spread Spectrum (DSSS) is also used to match the modulation used by IEEE 802.11b. The four layers used for modulation consist of ERP-DSSS-CCK, ERP-OFDM, ERP-DSSS/PBCC, and DSSS-OFDM [6]. The first ERP-DSSS-CCK is the layer used by 802.11b and is used for communicating with these devices. The second ERP-OFDM deals with data rates introduced by IEEE 802.11a. The third layer is used to further increase data rates to up to 33 megabits per second. Finally, DSSS-OFDM is new to IEEE 802.11g [6]. The IEEE 802.11g standard also outlines a mandatory packet structure.

The current standard is the 2.4-gigahertz frequency band, which is used for Bluetooth communication. More specifically, Bluetooth uses radio waves in the frequency range of 2.4 to 2.485 gigahertz. Under IEEE 802.11b and 802.11g, the width of the frequency channel is set to 20 megahertz.

4.4-A: Bluetooth Standards

There are standards governing each individual aspect of Bluetooth. If all are not followed, a device cannot claim to be a valid Bluetooth device and therefore cannot be sold as one. Those that most directly pertain to the D*rink*W*izard* will be covered in this section. Those standards involve connecting two Bluetooth devices and exchanging data. For the D*rink*W*izard*, an Android device and the drink mixer will be connected. The information sent to the mixer is how much liquid to include when a drink is selected in the Android application. Sent from the mixer to the Android device are alerts to update the user with information about when their beverage is ready to be picked up.

4.4-A-1: Link Loss Service

The Link Loss Service states that should a connection be lost, the user should receive some form of alert. For example, on the HC-06 there is a light-emitting diode (LED), which indicates whether or not, a successful connection has been made. The red LED blinks if a connection has not been made and remains on steadily once a connection has been made [3]. When a connection is lost, the LED will blink again. This satisfies the Link Loss Service.

4.4-A-2: Object Exchange (OBEX)

Generic Object Exchange Profile outlines the requirements for transferring any data between two Bluetooth-enable devices. This is a key specification for file transmission, which is the reason for including Bluetooth in the D*rink*W*izard*. It also provides usage models for the exchange of objects. Some features that

follow these models are Synchronization, File Transfer, and Object Push [1]. File Transfer is the feature the D*rink*W*izard* will utilize.

The File Transfer Profile specification details features required for data transmission. It is an extension of the Generic Object Exchange Profile and depends on the Serial Port Profile and Generic Access Profile [1, 6]. There are three main cases, which are covered in the File Transfer Protocol. The first covers using one Bluetooth-enabled device to browse the file system of another Bluetooth-enabled device. The second deals with the transfer of objects (both files and folders) from one device to another. The final case covered by this specification is file manipulation [2]. Folders are also included here. File manipulation is defined as the deletion and creation of new files and folders. In client-server relationships, the client is able to manipulate files and folders in addition to copying, renaming, and moving objects found on the server. The client is also able to initiate file transfer.

Serial Port Profile sets requirements for radio frequency communication (RFCOMM). This specification uses services for applications to set the features and procedures for communication between Bluetooth-enabled devices [4]. Communication methods include RS-232 and other standards for the serial transmission.

5.0 Design Constraints

5.1 Economic and Time Restraints

Researching similar products in the market today, very few exist with our complexity and range of drink selections. The prices of most products found in the market today range from $499 all the way up to $3999. These different products vary in the number of different drinks that came be made, the type of connectivity between devices offered, and the amount of each beverage than can be held in each container. The nearest model that comes as close to our Drink Wizard would be the Monsieur which retail starts at $3,999. *Figure 5.1* shows the Monsieur and will be very similar in size to our projected completed D*rink*W*izard*. The only major difference in our design will be that the touch-screen portion will be mobile via the android tablet.

Figure 5.1: Monsieur

Image courtesy of Monsieur.co

Our project, the D*rink*W*izard*, will have the sincere attempt to be funded completely by all four members of our team. We proposed an initial budget of $150 per member totaling $600 for the completed unit. Our major budget constraints will be allocating enough project funding to be able to handle 8 different types of liquid to dispense. The major constraint will be finding a moderately priced fluid handling system without going over budget. Our D*rink*W*izard* will be focused on primarily producing a well-made, repeatable drink versus a drink made very quickly. The correct type of fluid delivery and handling system will help us overcome this budget constraint.

The only types of time restraints that must be considered for our D*rink*W*izard* project is found within the class ending time and correlating schedules between team members. The D*rink*W*izard* is our UCF Senior Design Project, which means that it must be completed within two semesters. With the first semester mostly containing the team meetings for brainstorming and updating statuses, the D*rink*W*izard* must be completed from design to final product by the end of Senior Design 2. The summer semester is slightly shorter than most other semesters and leaves us approximately 12 weeks to complete our finished product.

Our team contains students of varying ages and degree types. This means that some team members have different schedules that involve work and well as families to attend to. The time constraint of completion within the two-semester window will be incorporated into our varying schedules and meeting times by setting milestones that must be met and get constantly updated accordingly.

Another time constraint that comes to be potentially considered is the time to market aspect. In theory, the longer that our team takes to complete the D*rink*W*izard*, the sooner a similar or competitive product can hit the market. By following a strict goal and milestone plan, our D*rink*W*izard* will be available for retail purchase before any similar products are fully developed.

5.2 Ethical, Health and Safety Constraints

A few interesting points can be written about when exploring the possibility of ethical constraints with the D*rink*W*izard*. When looking at society as a whole, it easy to understand that many people will think that creating an automated drink maker is the wrong thing to do. Some people may also state that, electronic research should be directed toward something more beneficial to society. Our team met some opposition from friends and family when deciding to create the D*rink*W*izard*.

The ethical constraints that can be directly correlated to the possible health and safety constraints that the D*rink*W*izard* proposes. A medical reference found on WebMD states many possible conditions and side effects linked alcohol consumption. Heavy drinking causes lower than normal amounts of red blood cells in the body, this condition called anemia and can lead to fatigue, shortness of breath and lightheadedness. Alcohol also increases the risk of cancer because it is believed that the body converts alcohol into acetaldehyde, which is a carcinogen. Heaving drinking also makes blood platelets more likely to clump together to form blood clots. These blood clots can lead to a heart attack or stroke. Another form of cardiovascular disease created by drinking alcohol is known as cardiomyopathy, it is basically a weakening of the heart muscle and sometimes failure.

Another one of the possible conditions listed by WebMD is Cirrhosis. The basis behind this disease is the toxicity of alcohol to liver cells. Alcohol actually creates scar tissue inside the liver that inhibits its ability to function correctly. The damaging effects of alcohol consumption are staggering. Alcohol consumption has been known to speed up the rate at which our brains shrink. This condition can be a direct link to dementia. Dementia inhibits the ability to plan, make decisions and solve problems. Researchers also correlate depression as a condition of alcohol. Depressed people often turn to alcohol to cope with emotion pain.

Some other conditions linked to alcohol consumption are epileptic seizures. These seizures can be triggered in people to don't even really have epilepsy. Gout is another condition that can be linked to alcohol consumption. Gout is largely believed to be hereditary, but researchers have found that alcohol can aggravate existing cases of gout. The effects of alcohol seem endless; it has been found that alcohol can cause alcoholic neuropathy, also known as nerve damage. The condition produces pins-and-needle sensations in the body's extremities. One of the final conditions causes by chronic drinking is pancreatitis; alcohol can inflame the pancreas and interfere with the digestive process, researchers state that around 70% of all cases of pancreatitis stem from alcohol consumption.

Another concern in health when dealing with the food and beverage delivery is the overall cleanliness of our D*rink*W*izard*. Our system will have closed containers that hold the fluids, but there is always the possibility of contamination throughout our system. If the unit is allowed to sit too long for extended periods of time, certain types of liquids can begin to spoil or create bacteria. For this reason, The D*rink*W*izard* will have container systems that are easily removable for cleaning. The D*rink*W*izard* will also have a purge or flush cleaning mode that will pump rinse water throughout the entire system for storage. This allows for a simple an efficient way of dealing with possible bacteria growth our system contamination.

Our D*rink*W*izard* obviously cannot address or state all of the health risks associated with alcohol consumption. It would be impossible to inform every user of the D*rink*W*izard* about its potential health hazards, in fact it may be almost detrimental to our product if we over-emphasize these risks. We believe that the health constraints of alcohol including the use of the D*rink*W*izard* rely primarily with the end user. The government dictates the legal drinking age. The government also allows the production and consumption of alcohol in the United States. Our D*rink*W*izard* is simply a medium that simplifies the delivery process.

The safety constraints of the D*rink*W*izard* are fairly basic. The D*rink*W*izard* has electronic components that operate on electric power. One of our basic safety constraints is to ensure that all components being used in the D*rink*W*izard* are

able to handle all required power ratings to prevent a fire. Our system will operate on dc voltages, which means that the power from the ac wall outlet must be converted and regulated properly. The D*rink*W*izard* will have all of the correct rated transformers within the power supply to achieve this requirement. Another fundamental safety constraint is to prevent physical harm to users of the D*rink*W*izard*. This constraint can be interpreted directly or indirectly. Because the D*rink*W*izard* has very few moving parts it is extremely unlikely that our system could cause direct physical harm. The only possible foreseeable harm would be if the vending unit would tip over and fall onto someone or something. The D*rink*W*izard* will be constructed out of thin plywood. The unit will also be constructed so that it is extremely stable and will have a low center of gravity to prevent tip-over.

The indirect safety constraints of the D*rink*W*izard* are numerous. Statistics from the US Department of Transportation state that in 2012, alcohol impaired driving accounted for 31% of all traffic-related deaths in the United States. In 2010, over 1.4 million drivers were arrested for driving under the influence of alcohol *(Dept of Justice).* The average person doesn't understand how much alcohol it takes to start having your senses impaired. The CDC states that about two beers increases the Blood Alcohol Concentration (BAC) to approximately .02%. The typical driving effects even at this concentration are a decline in visual functions and a decline in the ability to perform two things at the same time. At about 4 beers the BAC of an individual is about .08%, which is legally over the limit. A BAC of .08% can effect concentration, can cause short-term memory loss, inability to regulate vehicle speed, and causes impaired perception.

While these types of safety constraints are not directly related to the D*rink*W*izard*, our team is still trying to make an effort to inform users by implementing a Breathalyzer test into our drink-ordering platform. We hope to that this step will drastically reduce the amount of alcohol-impaired drivers that are using our D*rink*W*izard*. Our team is aware that this type of test can easily be circumvented but just allowing patrons to order a drink with a push of a button on an Android tablet can quickly lead to over consumption of alcohol just by the nature of its simplicity. We decided that be implementing the Breathalyzer for each drink ordered, it is allowing the user to be actively notified of their sobriety level. It also makes the user of the D*rink*W*izard* to make conscious effort to make the correct choice when consuming alcohol. When each drink is ordered, the BAC will be displayed. We believe that implementing this type of control into the D*rink*W*izard* will allow people to become more informed and better educated on how quickly alcohol can begin to take effect on someone. Ultimately, it is the end users decision to drink and drive. We just hope that our efforts could start to slowly change societies views on alcohol-impaired driving. Our D*rink*W*izard* may possibly have an override mode to disable the Breathalyzer control; this option would allow the D*rink*W*izard* to be used for personal in-home use.

5.3 Sustainability Constraints

The D*rink*W*izard* is like nothing seen in today's market. While researching similar products, we found similar products, but none of them offer all of the capabilities of the D*rink*W*izard*. This leads us to believe that there could be a long-term market for the D*rink*W*izard* when it is complete. According to the Nightlife and Club Industry Association of America, there are about 70,000 bars operating in the US *(nciaa.org).* We believe that if our product were to hit the market, we would have a viable attempt to market our product to these bars. If we keep our price-point low enough, our D*rink*W*izard* could catch on and have a long-term sustainability for a minimum of 5 years.

The technology being used throughout our D*rink*W*izard* is robust and mature. The prototype structure will be most likely built out of wood, which is a renewable resource. Our initial choice of microprocessor was from Texas Instruments; we believe that Texas Instruments is a profitable company and should be around for a long period of time. This is an advantage for our group because if we need to go to a large-scale production over the span of a few years, the microprocessor chips that we will be installing should be readily available.

The fluid pump system chosen for the D*rink*W*izard* are medical grade peristaltic pumps. These types of pumps were patented in the 1800's and made popular by the 1930's for medical use. They are extremely widespread throughout the medical field and should be fairly easy to source over a long period of time. We found many distributors and vendors to choose from while researching our fluid pump system.

Our power supply and other electronic peripheral components will all be chosen and designed from standard value parts. The needs for our electronic circuits are straightforward and basic; therefore we want to make sure that we do not try to incorporate any custom-designed values or components in our schematics.

The final component aspect of our D*rink*W*izard* that could have a sustainability constraint is our drink-ordering platform. We decided to choose that Android development platform for its widely growing popularity in the application design market. The Android platform currently powers over one billion devices worldwide and a staggering 1.5 million devices are activated each day. The sustainability of using the Android platform to support our D*rink*W*izard* looks very strong. Our team cannot find a foreseeable reason that our product would not be able to be supported do to the lack of Android platform support.

6.0 Hardware and Software Design Details

6.1 Initial Architectures and Related Diagrams

The system has several different subsystems. These subsystems all work together to make one large system. If integrated properly, the end user will only be able to notice the system as one system. The subsystems consist of the cup sensors, liquid level sensors, pumps and pump controllers and the power supply. The block diagram of the subsystems can be seen in *figure 6.1-1*. Each block will be broken into single components and design to be integrated into a larger system.

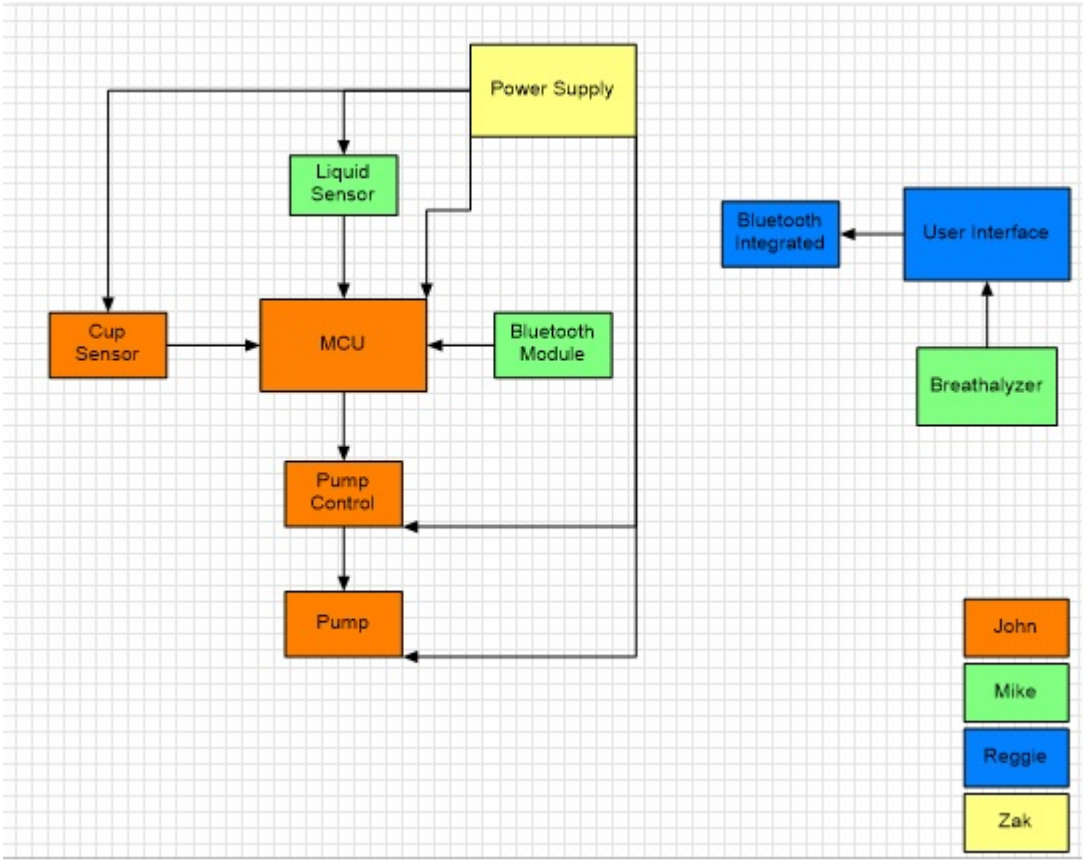

Figure 6.1-1: Block Diagram of the DrinkWizard

As seen in the *figure 6.1-1*, there are two main blocks. The main blocks are the MCU and the User Interface. The MCU will be the Texas Instrument MSP430G2553. The user interface will be an android device, either a tablet or a phone.

The Bluetooth device will be a preassembled module. This module will be connected to the MCU block and establish communication between the MCU and

the android device. This connection will be the way to wirelessly transfer the drink order from the android device to the D*rink*W*izard*. The connection from MCU to Bluetooth will be a direct hardwired connection and can be seen in *figure 6.1-2.* The actual Bluetooth module can be seen in *figure 6.1-3*.

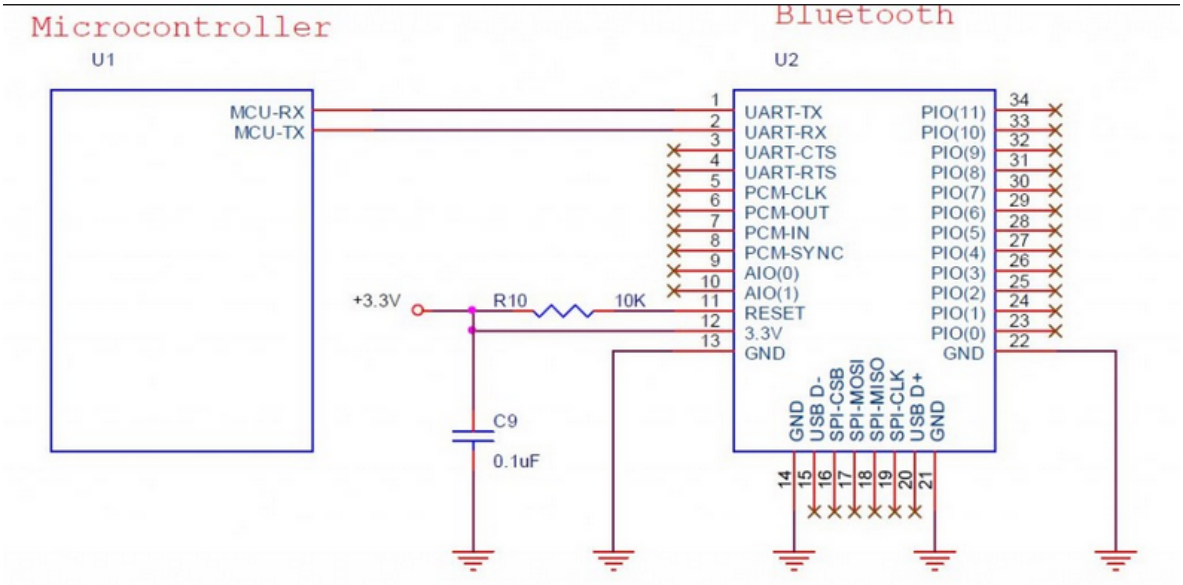

Figure 6.1-2: Bluetooth Module With MCU

Image courtesy of Mcuoneclipse.com

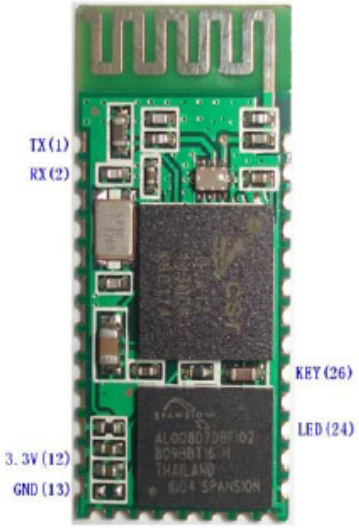

Figure 6.1-3: Bluetooth Module

Image courtesy of Mcuoneclipse.com

The Texas Instrument MSP430 chipsets are very powerful, ultralow power devices. They contain several systems inside the chip, such as memory, ALU, etc. These systems will be used to control the entire D*rink*W*izard* system. The MSP430 can be seen in *figure 6.1-4*. This allows us to see how the internal structures of the MSP430 are connected. These connections will determine how the code will be written and how the internal structures can be used.

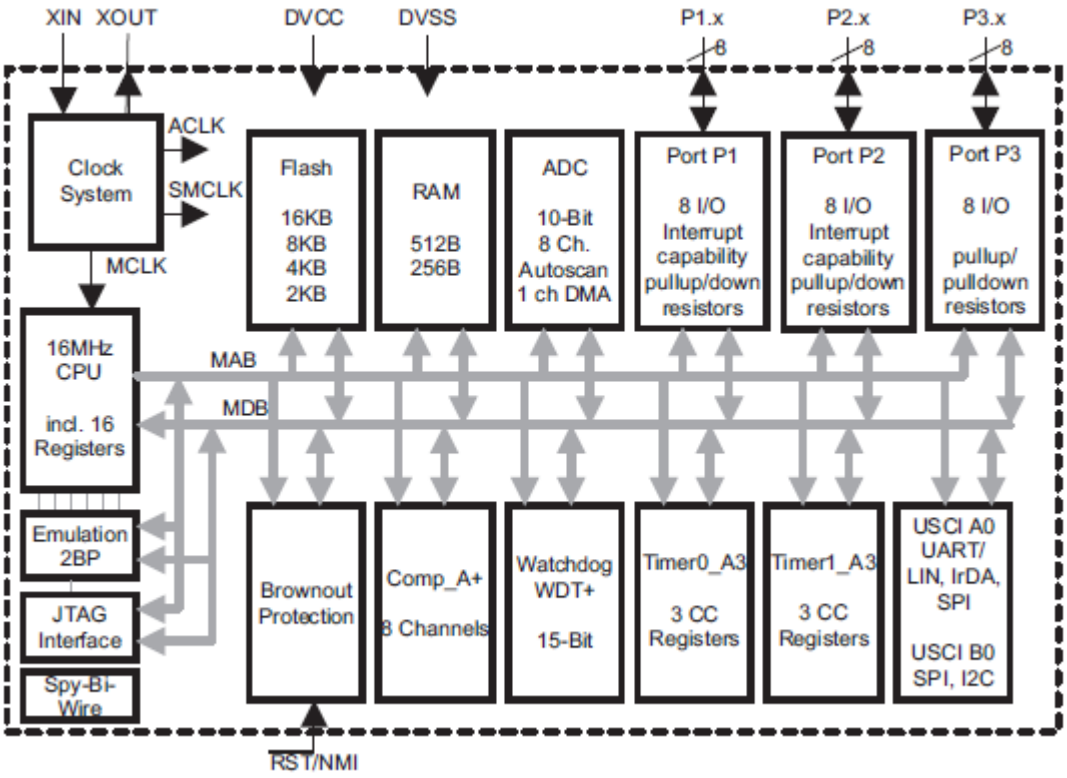

Figure 6.1-4: Functional Block Diagram of MSP430G2553

Image courtesy of Texas Instruments

Inside the MSP430 there are registers. Some of those registers are used by system for specific tasks by the CPU. Those reserved registers are dedicated counters or place holders. Only certain registers can be used by the user and programmed in the code. A table of these registers can be seen in *figure 6.1-5*. Of the sixteen registers, only twelve of the registers are general-purpose registers. Meaning changing these registers will not affect the performance of the microcontroller. Some of these registers will be responsible for counters for our system.

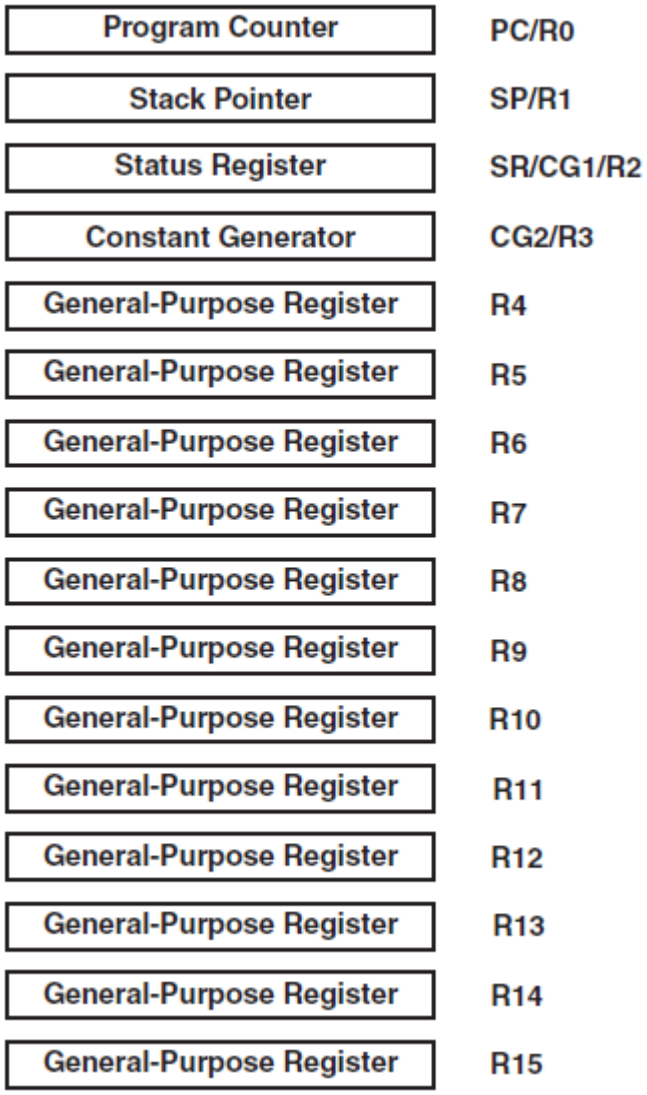

Figure 6.1-5: MSP430 Registers

Image courtesy of Texas Instruments

These registers each have a special purpose in the D*rink*W*izard*. The registers will store the information that is needed to determine the liquid level and the amount of each liquid that should be dispensed. It does so by using a register for each different type of liquor or mixer. The upper byte of the register will be used to store the liquid level and the lower byte will be used to store the amount of fluid that D*rink*W*izard* will need to dispense.

6.2 Cup and Liquid Level Sensors

The system for the D*rink*W*izard* must be able to accurately detect if there is a cup present in order to pour a liquid into the cup. The system also needs to be able to detect if there is an adequate amount of liquid that can be dispensed into that cup from the liquor containers. There will be several on board sensors to determine if a cup is present and also if there is enough liquid to be dispensed for the particular drink that has been ordered. The level of liquid that has been dispensed will also be constantly monitored to ensure that an error doesn't arise from overfilling the cup.

The cup sensors will consist of several ultrasonic sensors. The ultrasonic sensors will be able to determine the presence of a cup and the amount of liquid that is in that cup. The cup for our system will remain a constant throughout the use of the D*rink*W*izard*. This will ensure that the dimensions of the cup will also remain the same in order to decrease the calculations needed to determine the size and volume of the cup that is being used. The size and the volume of the cup will be hard programmed into the microcontroller as a constant value. This will decrease the amount of code needed to operate the cup sensors and will speed the overall system up quite a bit. We have decided to use a short tumbler glass for the D*rink*W*izard*.

There will be a total of three ultrasonic sensors used by this system. All three of these will be the HC - SR04 ultrasonic sensor previously discussed. The characteristics can be seen in the table in the ultrasonic sensor section. These sensors are accurate in the range of 1 inch to 13 feet, well within the design specifications of the D*rink*W*izard*.

The ultrasonic sensors will determine the presence of a cup by using a sensor for each axis. Two of the ultrasonic sensors will determine the sides of the cup by measuring a distance within the range that is determined in the code as a constant value range by being mounted on either side on two separate axes. One will be mounted on the left facing the cup and the other will be mounted on the back facing the cup. This range will again remain constant throughout the operation of the D*rink*W*izard*. These sensors will be mounted onto the D*rink*W*izard* and will only be operational after a drink has been ordered. By having the sensors in standby mode until a drink is ordered will save on power consumption and computing resources.

The last ultrasonic sensor will measure the volume that remains within the cup. Initially the volume that remains to be filled in the cup must be greater than the volume of the drink that has been ordered. This ultrasonic sensor will constantly monitor the level of the liquid that has been dispensed into the cup. There will be a maximum level that the liquid can reach to safeguard against overfilling a cup.

This ultrasonic sensor will be mounted on the top pointing down towards the cup. The configuration of the ultrasonic sensors can be seen in *figure 6.2-1*.

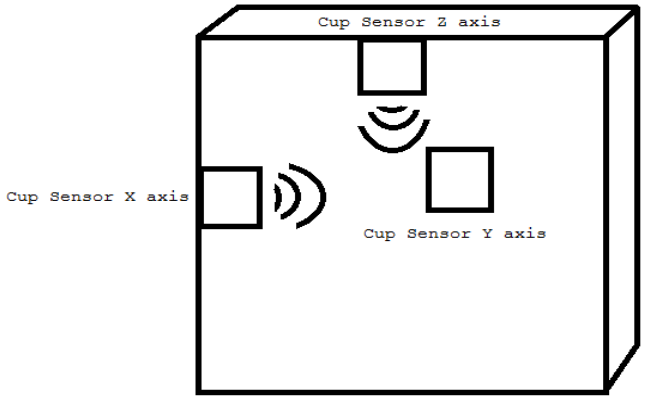

Figure 6.2-1: Cup Sensors

The location of the sensors is a key element in the ability to guarantee a cup will be there to contain the dispensed liquid. The sensors will be mounted in such a manner that they are centered on the correct axis of the cup. The cup will rest in a small recess that is the size of the bottom of the glass. This will make certain that the cup is placed correctly in the dispensing area. This will also ensure the reliability and repeatability of the D*rink*W*izard* and the cup sensors.

The cup sensors can measure distance. Since the distance from the cup to the sensor will remain a constant, the sensor should be able to detect the cup accurately and repeatedly. Two-cup sensors are used to ensure that a hand or other object is not just placed in the dispensing area and giving a falsified response to the sensors. Two sensors will check the cup in both the x and the y-axis simultaneously. This is not entirely accurate but will work for our D*rink*W*izard* for the time being.

We also want to safeguard against the cup becoming overfilled so as to avoid spilling. An ultrasonic sensor makes a great instrument for determining the amount of fluid that has been dispensed into the cup. The distance from the bottom of the cup to the ultrasonic sensor can be measured. This is shown in Figure 6.2.1. A measuring device such as a tape measure will first do this. Then the ultrasonic sensor will be set up and used to measure the same distance. The distance will be calibrated to ensure that the sensors' readings are as accurate as possible. This is essential for this sensor, as an inaccuracy could result in an overflowing cup and possible damage to the rest of the components of the D*rink*W*izard*.

As the fluid is pumped down into the cup, the liquid level will rise towards the top ultrasonic sensor. That means that there will be a change in the distance from the liquid to the ultrasonic cup sensor. This change will be continually calculated by the microprocessor for the duration of the dispensing procedure. If the distance from the ultrasonic cup sensor to the liquid falls below a certain threshold, the pumping will immediately cease and an overfill error message will occur on the screen of the user's android device. If this error arises, then the user should check the level of the liquid in the glass. If the level is below the rim of the glass, a foreign object blocking the sensor could have caused the error. The object must be removed to continue operation of the D*rink*W*izard*. The ultrasonic cup liquid level sensor is shown below in *figure 6.2-2*.

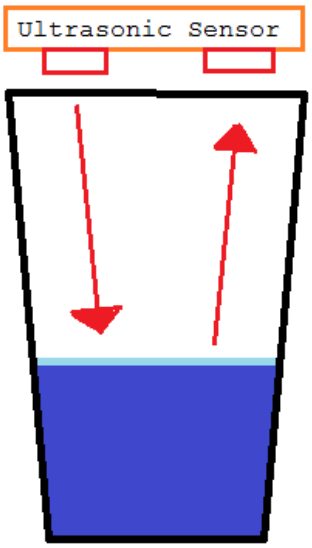

Figure 6.2.2: Ultrasonic Cup Liquid Level Sensor

These three sensors will work in unison to accomplish the task of detecting a cup and the amount of fluid in that cup. However, these sensors will be turned off during the systems' downtime and will only be used by the system when they are needed. This will help to increase the battery life of the system and will reduce the power consumption.

Of course the containers that hold the liquor and the mixers for the drinks must also be monitored. There are two ways to do this. We can implement a hardware device to measure the amount of liquid that remains or we can implement a software counter to count the amount of each liquid that has been dispensed. Each way has its advantages and disadvantages. I believe that the best way is to implement both methods. This way the systems can be used to check each other and if one system fails the other will still be there to alert the user that a bottle will need to be refilled or replaced.

For the hardware implementation there are several different methods that can be used. A light sensor can detect if the fluid gets below a certain level by using the liquid as a means to block the light from transmitter to receiver. When the fluid gets below the transmitter and receiver the signal is no longer being blocked and the system will alert the user about the low level of that liquid. Another hardware device that can be used is a load cell, which can measure the force that the container exerts to determine the amount of liquid in the bottle. When the load cell detects a load below the threshold, it would then cause an error message on the android device and alert the user of the low liquid level.

We have decided to use a load sensor for the D*rink*W*izard*. A load sensor for each individual container will be used to determine the level of liquid that remains to be dispensed. The container is a plastic rectangular bottle that will be the same size and dimensions for the liquors that the system will use. Each bottle has a maximum capacity of 50 fluid ounces. Given the weight of each liquor bottle the liquor weight can easily be calculated, the load of each container can be determined using the measurements calculated for each liquor type. Each liquor type can vary in weight depending on the proof, the amount of alcohol contained by that specific liquid, of the liquor. The formula for the weight of the liquor can be found in *figure 6.2-3*.

$$Weight = \left(\frac{0.79 * proof}{2} + \left(1 - \frac{proof}{2}\right)\right) * amount\ of\ liquor(mL)$$

Figure 6.2-3: Formula for Weight

This formula will be used to determine the weight of each of the five different liquors. These calculated values would then be programmed into the microcontroller. Then the system will be able to calculate the amount of liquor that remains in each of the five liquor bottles. Solving the equation for the amount of liquor that remains will do this. The system will check each liquor type that is to be used for the drink that has been ordered before it starts to dispense liquid. The other bottles that are unused have no relevance at the time and therefore should not be checked, this will reduce the time it takes to calculate the data that will be necessary to produce the drink that had been ordered, shortening wait times. This will also reduce the power consumption and when the system will be running on battery power this will play a larger factor than normal.

All the different mixers will also need a load sensor as well to measure the amount of fluid that is left and ready to be used. The method of determining the amount of fluid is not as easy as the method for calculating the amount of liquor remaining in the reservoir. Each mixer liquid will have a different density and therefore will have independent equations for each different type of mixer. This

method will only work if the mixer does not change brand or type. So if the D*rink*W*izard* is setup to use orange juice with pulp, it must remain that way for this system to work properly and accurately. If a change is made the system will have to be recalculated for that new liquid.

We want the system to alert the user before the liquor runs out. To do this we will alert the user before the liquid runs below the greatest amount that any one drink could possibly use. For most drinks this would be two to three ounces of liquor or five or six ounces of mixer liquid. So this means that the warning threshold will have to be slightly higher than that. All of these numbers will be hard programmed into the msp430. The data will remain the same unless a liquor or mixer is exchanged for a different type.

Another way to determine the amount of liquid left in the reservoir is software based system that measures the amount of liquid that was supposed to be dispensed. Once the liquor bottles and the mixer bottles are a constant uniform size and will never change, a software implementation will be available for a liquid level measurement system. The liquor bottles are smaller than the mixer bottles, due to the higher demand for a mixer. But each drink will have a preset amount of liquid that is to be dispensed and the software can track that amount.

Each liquor type will have up to fifty ounces and each mixer will have up to one hundred ounces available to begin with or after a liquid has been refilled. After a drink has been ordered the software will track the amount of liquid that has been dispensed. The system will do so for each individual liquid. Meaning that each liquor level and each mixer level will be tracked independently. The amount of liquid left in its reservoir can be determined by subtracting the amount of that liquid that has been dispensed from the size of that liquid's reservoir. The calculated amount that is remaining will be compared with the amount of liquid that is being demanded by the person ordering the drink. If the amount left is less than the amount that is needed for that drink then the user will again be alerted.

Both of these systems will be able to accurately check the liquid level. There are some advantages of each system. The hardware implementation takes into account the physical load that is exerted by the liquid and therefore does not need to be reset after a replenishment of a liquor type or a mixer. The hardware implementation also can detect if a bottle is removed from the D*rink*W*izard* system.

The software implementation also has some advantages. The biggest advantage of the software implementation is that no external parts are needed. This can be a big factor if size or cost plays a large factor. Also the software will

take less power than the hardware, which can be essential if battery life is important.

Using both of these systems at the same time also has some advantages. If one of these systems fails, then the other system will still remain to keep track of the remaining liquids. Another benefit of using both systems is to be able to detect a catastrophic failure of the system or it someone has been using a liquid without the D*rink*W*izard* doing the dispensing of that liquid.

To detect a catastrophic failure or a possible theft, the system will check both sensors. If one sensor determines that the liquid is below the value of the other then there is a possible theft problem. But if the load sensor determines that the bottle is empty and the software insists that the bottle still has liquid remaining, then a possible leak has occurred and should be diagnosed before any further dispensing happens. This is to save the integrity of the rest of the system, as liquids and electricity are not usually a good mix for the printed circuit board.

6.3 Bottle Liquid Level Sensors

After the extensive research does on the types of liquid level metering available, we concluded that the most efficient and inexpensive way to accurately measure the amount of liquid in each one of our containers was to weigh them using a load cell. The load cell chosen is found from an overseas vendor. The image shown below in *figure 6.3-1* below provides a representative example of our strain gauge.

Figure 6.3-1: Strain Gauge

Image courtesy of Seedstudio.com

This strain gauge has 3 wires that come out of it. It acts similar to a potentiometer that changes resistance when an external mechanical force is applied. This load cell is one major piece of the liquid level monitoring system. It is going to act as the resistive element of a voltage divider system. Each bottle will have a reference to the 5-volt supply voltage. The 5-volt reference will then have a resistor and the potentiometer in series to ground to form a voltage divider. The portion of the schematic below in *figure 6.3-2* will show how the strain gauge is

integrated into our system. The voltage across the strain gauge potentiometer will then be sent to the non-inverting input of operational amplifier. Each operational amplifier will be tuned using V ref. V ref will be the threshold voltage that will cause the output voltage to change. Assuming that the voltage across the potentiometer will be between 0-5V depending on the applied mechanical strain, our V ref will need to be chosen so that its voltage is greater than the inverting input. This ensures that the output logic will only go “low” when “V ref” is lower than our “V sense”.

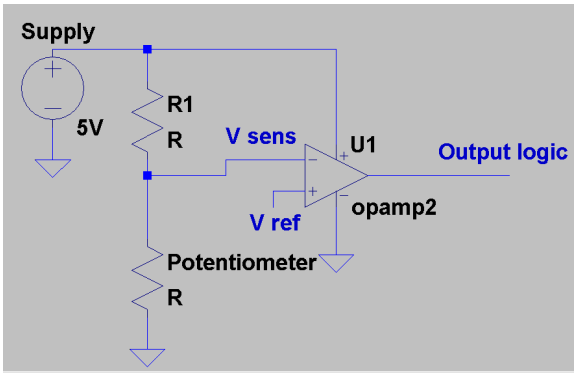

Figure 6.3-2: Strain Gauge Schematic

The operational amplifier package that we will be using is the LM324 integrated circuit package. The LM324 is a 14 pin IC package that is generally produced in thru-hole or surface mount applications. In our design we will be using the ceramic dual in-line package. It measures 0.785 inches long, 0.300 inches wide and 0.330 inches high. We believe that it will be easier to integrate strain gauge onto our circuit board because of its larger size. The size of our overall PCB really isn’t a major factor in our D*rink*W*izard* and can be easily incorporated inside the internal housing. *Figure 6.3-3* shows a representative example of the ceramic dual in-line package type and the pin-out.

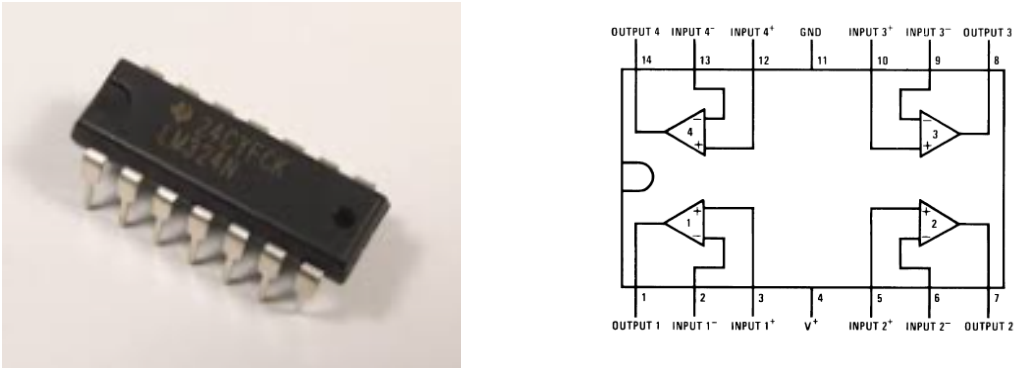

Figure 6.3-3: Ceramic Package

Image courtesy of Texas Instruments

There are some key features of the LM324 that make it very advantageous to use. One feature is its very low supply current drain. This is very important when our Drink Wizard is operating on battery power. Another important feature is the requirement of only needing one supply voltage. This will simplify our power supply subsystem circuitry. All of the operational amplifiers used in our liquid level monitoring system will use as comparators. The LM324's ability to swing from V+ to GND makes it easy to incorporate into the digital logic that our microcontroller will be utilizing. The LM324 only contains 4 operational amplifiers per package; since we are trying to monitor the levels of 9 liquid containers we will be using 3 LM324's.

The importance of having digital logic at this point in the circuit is to allow the use of gates to direct the decision-making in the liquid level monitoring subsystem. All of the outputs of the operational amplifiers will be "ANDED" together using a 74ALS133. A small example of how all of the operational amplifier outputs will be "ANDED" together can be seen in the *Figure 6.3-4*. The multiple inputs of the NAND gate IC package will allow our liquid level circuit to actively monitor the levels of all 9 liquid containers simultaneously. The NAND gate extremely low response times provided immediate feedback from the outputs of the operational amplifiers, this immediate feedback can be quickly read from the microcontroller and pause the entire system until the proper bottles are refilled.

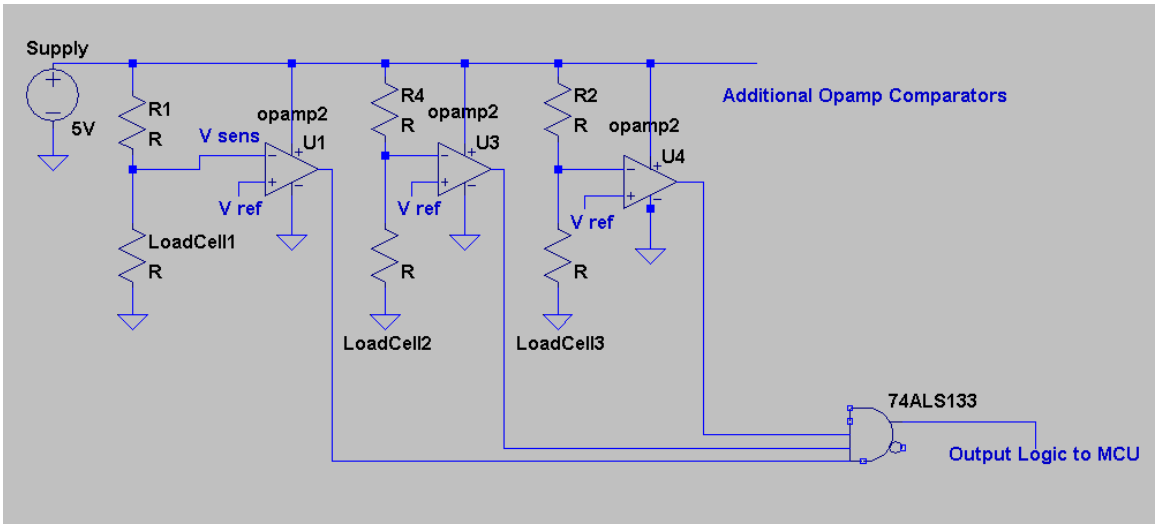

Figure 6.3-4: Operational Amplifier Outputs

The 74ALS133 from Texas Instruments is a 13 input Positive NAND Gate IC. It has an operating temperature of -55C to 125C, which is well within the D*rink*W*izard's* ambient operating temperature. We will ultimately be using the ceramic dual in-line through-hole package in our circuit; its dimensions are extremely close to those of the operational amplifier packages that we are using as well. The package measures 0.785 inches in length, 0.300 inches in width, and 0.330 inches in height. An image and pin out of the 74ALS133 can be seen

in *Figure 6.3-5*. It also uses a nominal 5V VCC Supply Voltage level, which works perfectly in for our power supply subsystem. The 74ALS133 also has a very wide input voltage logic level input. Anything about a minimum of 2V registers as a logic level “1” and anything below a 0.8V registers as a logic level “0”. This digital logic-tuning window provides the type of basic noise filtering that is required in our circuit to prevent false readings of empty bottles. Since the output of our comparator op-amps are the only inputs being applied to the NAND gates, our simple logic circuit should work flawlessly during operation.

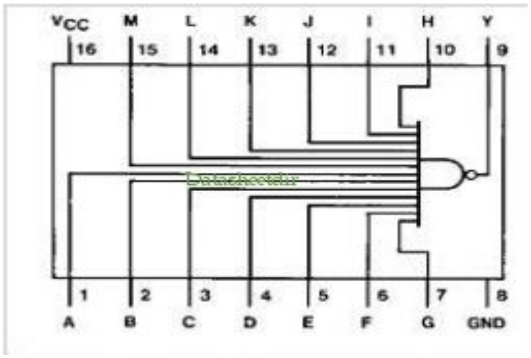

Figure 6.3-5: 74ALS133 Positive NAND Gate IC

Image courtesy of Texas Instruments

The principle theory behind the operation of our logic circuit is the operation of the comparator op-amp. The voltage being applied to the non-inverting input of the op-amp terminal is a constant Voltage labeled “V ref” this voltage will be properly calibrated to be slightly higher than that of the voltage of an empty bottle being measured by our load cell. As the levels of liquid decrease in the containers, the resistance across the potentiometers will lower. This will cause the voltage across them to lower as well. This voltage is feed to the inverting input of each operational amplifier. As long as the “V sense” voltage is higher than the “V ref” voltage, the output of each operational amplifier will saturate to the ground rail. When voltage across the non-inverting input goes under the constant “V ref”, the output of the comparator will saturate to VCC, which is the supply voltage. These two different output voltages are being fed directly to the inputs of the NAND gate. The logic “high” coming from the operational amplifier is

+5V, which is the exact nominal High-level input voltage requirement for the NAND gate. The logic “low” coming from the operational amplifiers is 0V, which is well below the 0.8V max Low-level input voltage. These operating digital voltage levels should be well within the required limits for the NAND gates to function correctly.

During building and testing of the comparator circuit networks, we will need to do some experimentation to determine if any hysteresis will be necessary to stabilize the output of the operational amplifiers during the transitional periods of the output. The hysteresis may be required when both the inputs on the operational amplifiers are almost exactly the same and is causing the outputs of operational amplifiers to switch extremely rapidly. The digital circuit is using combinational logic and will be able to switch outputs with little to no propagation delay. While this is good in most applications, the sampling rates of the microcontroller being used could possibly get confused our malfunction. We could also try to rectify the high switching states with coding. Our microcontroller coding could be constructed so that it ignores any additional inputs after a single logic bit received has changed states.

The output state of the 74ALS133 NAND gate will be fed into one the inputs of the microcontroller. The microcontroller being used is the Texas Instruments MSP430g2553; it is a 20-pin device with 16 general input/output pins. An image of the microcontroller and pin-out can be seen in *Figure 6.3-6*. We decided to use this microcontroller because we had them readily available and are experienced with the writing code from previous classes. This microcontroller only has a couple of tasks to perform in this subsystem. It will monitor the status of the 13 inputs NAND gate through one of the input ports and display an error message on the LCD display through a UART connection. The only other task that it must perform is to send an interrupt signal to the pumping subsystem while any of the containers are empty or are being refilled.

Figure 6.3-6: MSP430g2553 Microcontroller

Image courtesy of Texas Instruments

We will successfully perform these operations by declaring one of the input pins Port1.0 as an input. Port1.0 will have the output of our NAND gate connected to

it. We will also need Port1.1 declared as an output. Port1.1 will be connected to an available input pin on the pumping subsystem microcontroller. The task of Port1.1 is to send the correct logic output to cause the pumping system to hold until all of the correct empty containers are refilled back to correct levels.

The other important task of the MSP430 is this subsystem is to display a string of characters to an LCD Display. The display that we chose was the Newhaven 16 x 2 blue-white backlit display. We will accomplish that by writing code in C Language. The C code will begin by initializing the microcontroller for UART communication for outputting characters. The UART will be initialized for 1 stop bit, no parity, and 9600 baud, polling operation. The initialization will be accomplished by setting the P2SEL register to transmit and receive to port 2 bits 4 and 5. We then need to set UCA0CTL0 to 0; this ensures 8 data, no parity, and 1 stop, UART. The next step is to set UCA0CTL1 to 0x41; this will select an ALK of 32768 and put in software reset for the UART. The initialization continues by setting the upper byte of the divider clock word UCA0BR1, as well as setting the clock divide from a clock to bit clock with UCA0BR0. We then set UCAMCTL to use low frequency mode and UCA0STAT to not loop the transmitter back to the receiver for echoing.

Once we have confirmed that we have proper UART initialization, our C code will have a string of characters that will serve as the error message to be displayed on our LCD display module. The error message will be similar to “Please check the levels of my containers”. The C code will then set the direction of the proper port pins with the P1DIR instruction. The code will then create a function called “send”. The purpose of this function is to print out every character in a string until a null character is reached. The code will run in an infinite for loop, inside this for loop will have an “if” statement that is comparing the value of P1.0 with 0. If the statement evaluates true then that code with then execute the send function and print out each character is our display error message and set the output PORT1.1 to a logic level “high”. This status will remain in this condition until the liquid of each container refilled to an adequate level. Once the proper levels of liquid have been refilled inside the containers, the LCD display will then output a message similar to “Thank you for refilling me.” This thank you message will be displayed by calling the “send” function once more.

6.4 Communication

The device chosen for interfacing with the drink mixer is an Android tablet with a custom application designed for connecting and sending information to the mixer. Android was chosen for its portability and accessibility during the time of production along with its access to various communication methods, specifically wireless local area networks (Wi-Fi Direct), Bluetooth, and near-field communication (NFC) on most devices. The Android operating system provides access to all of these methods through the use of the Android software

development kit. This section will cover all considerations for each communication method and why the choice was made for the D*rink*W*izard*'s technology usage. A chart highlighting the differences between Bluetooth, Wi-Fi, and near-field communication has been included at the end of this section.

6.4-A Range

Current plans for the D*rink*W*izard* show primary usage will occur within an area of about ten square meters. For this reason, the chosen method of wireless communication should have a maximum range of at least ten meters. This will allow for unrestricted (by wireless communication method) access to the mixer from the majority of the area and act as a security measure to prevent unwanted access to the mixer from devices not currently in the vicinity.

6.4-B: Security

While security is not a major concern for the D*rink*W*izard*, due to the fact that no personal information will be stored in the either the machine or the Android application, and a cup is required to begin dispensing a beverage, it is still necessary to have a secure connection. Since there may be personal information stored within the Android device that could be accessed by malicious software, the chosen method of wireless communication should be secure enough to deter or protect against such software.

6.4-C: Power Consumption

The chosen method of wireless communication should use very little power in case plans are made in the future to change the power source. However this is not a major concern at the time as the mixer will be plugged into a standard United States 120-volt power outlet. Portability is not currently a driving factor. But should there be another iteration of the D*rink*W*izard* drink mixer, it is highly likely that there will be a major focus on portability so to keep the design as close to the original as possible, low-power components are being used now.

6.4-D: Data Transfer Rate

It was determined that the rate at which data is transferred between the Android device and the drink mixer would not be driving factor in design. The data that will be sent is a very small amount, generally measuring in mere bytes. All possible methods of wireless communication will make the data transfer process appear instantaneous to users. Therefore, the bit-rate is not a major factor in the decision of which method of wireless communication to use.

6.4-E: Wireless Local Area Network

One of the primary factors in determining which method of communication to use is range. Using a Wi-Fi Direct connection, a connection can be established up to two hundred meters away [8]. This is well within the range of the typical building, which would house the D*rink*W*izard*. However, general usage will most likely see the D*rink*W*izard* being used at no more than ten meters therefore while the added range could only be beneficial as an extra feature for the D*rink*W*izard*, it is not a necessity. Another factor necessary in the determination of which method to use is security. Wi-Fi Direct is also considered to be secure and can be accessed through NFC, a passcode, and also physical means such as pressing a button to activate it.

Finally, power consumption of Wi-Fi is much greater than that of Bluetooth. This extra power is how the maximum range of two hundred meters is achieved so it could be worth the extra power to gain the extra range [7]. Taking these things into consideration, it was concluded that a wireless local area network could be successfully used to connect the Android device to the drink mixer.

6.4-F: Bluetooth

Using Bluetooth to communicate with the device would also prove to be a viable option, as it is standard in most mobile devices on the market today. These devices generally use Bluetooth, Class 2 that has a maximum range of ten meters. While this is less than Wi-Fi Direct, it is still a reasonable range for typical usage of the D*rink*W*izard*. Since security is not a major concern, a standard Bluetooth passcode will suffice. Keeping the range to Bluetooth's approximate maximum of ten meters acts also acts as a security measure. Users will have to be fairly close to access the D*rink*W*izard* and any information on the Android device should they have the means to do so.

Bluetooth modules are also relatively cheap, reliable, and consume less power than Wi-Fi. This makes Bluetooth a good alternative as a low-power module. This will allow for minimal change to communication module should another iteration of the D*rink*W*izard* drink mixer with a focus on portability be created.

6.4-G: Near-field Communication

Near-field communication is the final method considered for communication between the user device and the D*rink*W*izard* mixer. Although NFC was a possibility, this technology requires both devices to be very close together (if not in direct contact) to communicate. This would limit usability of the D*rink*W*izard* and force the user to be next to the mixer. Such contact may also create sanitation issues should the D*rink*W*izard* be in a public setting. Common

modules were also found to be more expensive than those for other methods of communication.

Near-field communication's security comes in the form of range. Any malicious software would have to come into contact with either the Android device with the D*rink*W*izard* application or the drink mixer's near-field communication chip. Since the D*rink*W*izard* is designed primarily for home usage, this is not a major issue, making this characteristic of NFC less important than the range itself. Taking these things into consideration, it was decided that NFC would not be the best choice for the D*rink*W*izard*.

6.4-H: Conclusion: HC-06 Bluetooth Module

The primary factors for deciding which form of communication should be used was technology range and module cost. Careful consideration led to Bluetooth being the method of choice for the D*rink*W*izard*. Near-field communication posed problems and did not have enough benefits to outweigh them. The range was also found to just be too short for initial plans.

While Wi-Fi Direct has considerably more range, it was deemed unnecessary. The excess range would lead to outside, unwanted devices being able to access the D*rink*W*izard* drink mixer and possibly any device paired with it. Thus, Bluetooth was chosen. A connection through Bluetooth will provide appropriate security and not create unnecessary insecurities. In addition Bluetooth's appropriate range, Bluetooth modules are relatively cheap, which will allow the D*rink*W*izard* to be produced at minimal cost. The specific Bluetooth module used is the HC-06.

The HC-06 is a small, cheap, and effective module for communication through Bluetooth. This is a class 2 module with four pins: receiver, transmitter, positive supply voltage, and ground. A working voltage of 3.3 volts is required to use activate the HC-06. Shown below is a wireless communication table, *table 6.4-H*.

	Wi-Fi	Bluetooth	NFC
Max Range	Approximately 200 meters	Approximately 10 meters	Approximately 10 centimeters
Data Transfer Rate	Up to 250 Mbit/s	2.1 Mbit/s	106, 212, 424 kbit/s
Frequency	2.4 GHz	2.4-2.485 GHz	13.56 MHz
Standards	IEEE 802.11a/b/g/n	802.15 (previously) Bluetooth SIG	ISO/IEC 18092 ECMA-340

Table 6.4-H: Wireless Communication Method Comparison

6.5 Main Control

A device must control the system in order to work autonomously after a drink order has been placed. The Texas Instrument MSP430G2553 will control the D*rink*W*izard*. This chip will control the action of turning on and off the pumps for the desired amount of time in order to fill the cup up with the appropriate amount of liquor or mixer.

First an order for a drink must be made on the android device. The android device will then send a signal to the D*rink*W*izard*. The Bluetooth module will then receive that signal and send it to the MSP430G2553. The microcontroller will then decipher the data received. That data will determine what drink to make and the amount of each liquid that should be dispensed. The MSP430 will then begin to check all of the sensors necessary to perform the task at hand.

These sensors include the liquid level sensors and the cup sensors. To ensure that the drink can be properly made the liquid level sensors must guarantee that there is an adequate level of that liquid that is to be dispensed. The liquid level sensor will begin to check the fluid levels and report back the information received to the MSP430. If a liquid is empty or too low to make a certain beverage, then the microcontroller will alert the system and display the location or type of liquid that is too low. The drink manufacturing will also come to a halt. Once the liquid is refilled the system will continue where it left off.

The cup sensors are there in order to prohibit the dispensing of a liquid without a container to hold that liquid. The glass that will be used with this system must be the same cup every time, once the system sensors were calibrated to use that specific cup, with a certain volume and overall dimensions. The cup sensor confirms that a cup has been placed in the dispensing zone and that the cup is not going to be overfilled if dispensing begins or continues.

The system does have one final check. The system must ensure that the person ordering the drink has an approved blood alcohol concentration level. The amount of alcohol that is contained in the blood needs to be below a certain level in order to continue the drink making process. This will also determine whether you can legally drive, setting morals aside for this part. In the state of Florida, the legal level for a person to drive is having a BAC that is less than 0.08%.

The D*rink*W*izard* will be able to determine the BAC of a person and will display it on the screen of the android device. If above the legal limit to drive, a warning will be displayed to verify that the user is aware of their condition of inebriation. If said person continues to drink, their blood alcohol concentration will continue to rise and if it goes above a certain level certain health related problems could arise and that user should be cut off. This is to prevent possible death or other adverse conditions that could arise. The maximum level of a person's BAC should be 0.2%.

A Breathalyzer is used to check the blood alcohol concentration of the user. This will be the first step before a drink can even be ordered. The breathalyzer is going to be attached to the android device and the android device will be able to check the BAC of the user and allow the user to continue and order drinks or to look that user out for an allotted amount of time depending on the BAC that was measured.

If the system sensor are all working properly and no errors arise, then the system will begin to dispense the drink that had been ordered. The system has a total of nine fluids that it can dispense. Each fluid will have its own pump and each container of liquid will have its own liquid level sensor.

The overall system procedure is visually displayed in the flowchart in *figure 6.5-1*. This will flowchart will help in the design and troubleshooting stages to ensure that the proper steps are performed and in the correct order. Each step will have a method of checking the system states to ensure the proper performance of the D*rink*W*izard*.

Each individual pump will have its own circuit to turn the pump on or off. The circuit will consist of a resistor, a MOSFET, and a MOSFET driver. The MOSFET is basically acts as an electrical solid-state relay. The MOSFET will control the current that flows to the peristaltic pump and will determine whether the pump will be on or off due to the voltage at the gate of the MOSFET. The voltage that is sent to the gate of the controlling MOSFET, via the MOSFET driver, gets a signal from the microcontroller.

The microcontroller will receive the input of a drink order from the Bluetooth module. The module will already be paired with an android device prior to use

and will remain connected to this device will the unit is powered on. If the system is shut off for any reason, the next time the system is powered on the unit will automatically connect with the default android device. This signal will contain the data for the amount of each fluid that is to be dispensed. For example if a screwdriver beverage is ordered then the android device will send the correct data to dispense 2 ounces of vodka and 6 ounces of orange juice.

The microcontroller will then send the on signal to the vodka pump for the correct amount of time and the same for the orange juice. The amount of time will be measured and hard programmed in as a constant into the microcontroller. Each register will have a special purpose. Registers will hold the value for individual liquid levels and for the amount of liquid that should be dispensed.

The registers are used to allow for a faster response time from the CPU. Speed is essential for the manufacturing of the delicious drinks. Since the pumps are on the slower side, we try to compensate in other areas. One area we can gain speed is the speed of the code. With minimal code and using registers instead of memory the speed of the code can be increased. Each register will designated for a specific liquid, 5 for the liquor and 4 for the mixers. The link can be seen in *table 6.5-2*. This will show the relationship between the liquids and the registers.

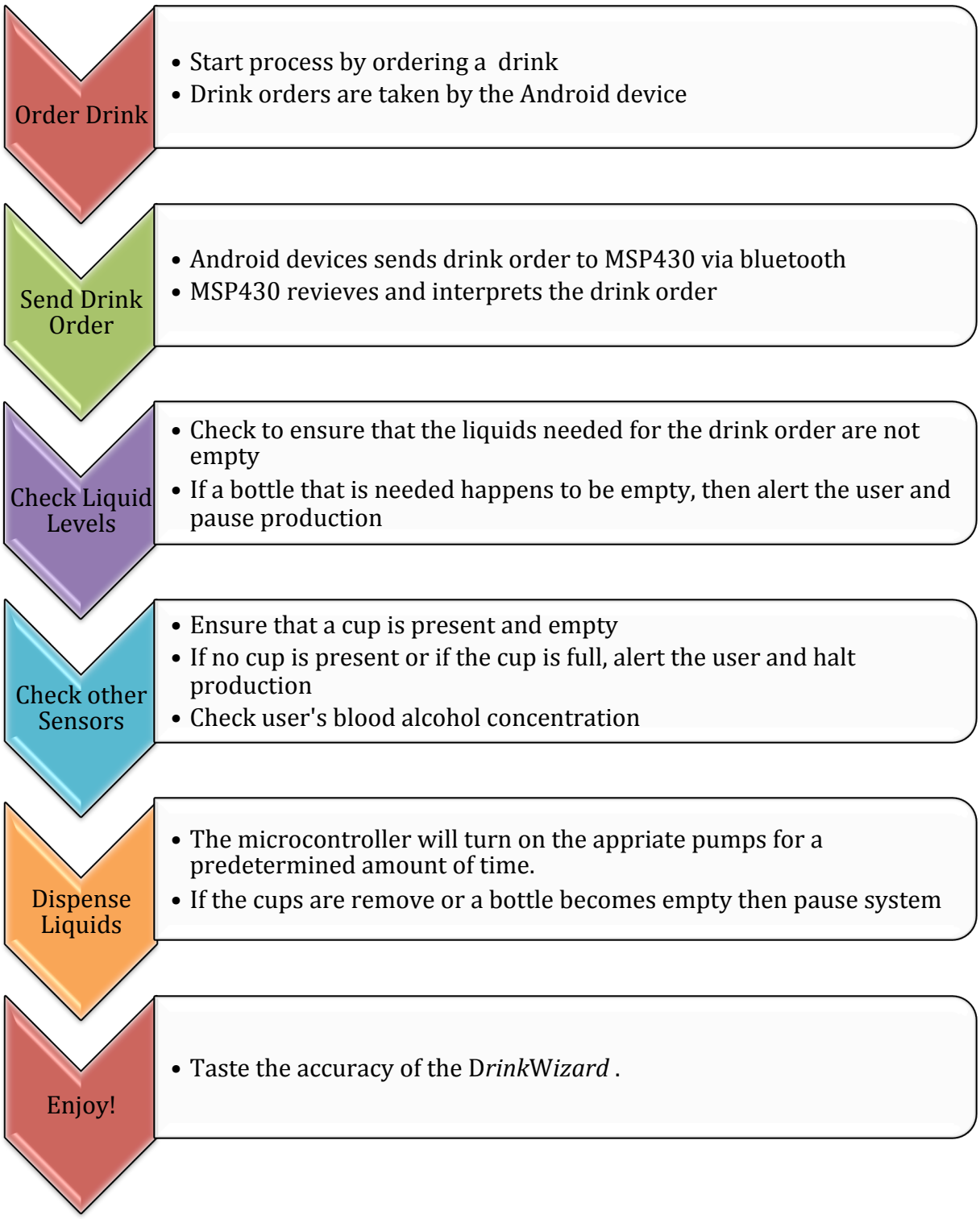

Figure 6.5-1: System Procedure Flowchart

Register	Register Use
R4	Used for vodka level and amount of vodka to be dispensed
R5	Used for peach schnapps level and amount to be dispensed
R6	Used for tequila level and amount to be dispensed
R7	Used for rum level and amount to be dispensed
R8	Used for Triple Sec level and amount to be dispensed
R9	Used for orange juice level and amount to be dispensed
R10	Used for cranberry juice level and amount to be dispensed
R11	Used for pineapple juice level and amount to be dispensed
R12	Used for sour mix level and amount to be dispensed
R13	Used to store data from Bluetooth device
R14	Reserved for later use
R15	Reserved for later use

Table 6.5-2: Register Uses

The microcontroller will then take the registers and send that data to the MOSFET to turn the pumps on or off. The MOSFET require a minimum of five volts and will need a transition step in between the MOSFET and the microcontroller, due to the fact that the microcontroller can only put out a maximum of Vcc – 0.3 volts and having a maximum Vcc of 3.6 volts. This means that the maximum output of the microcontroller is 3.3 volts.

Due to the low output voltage of the general-purpose I/O pins, a MOSFET driver will be used to increase the voltage from the microcontroller. The MOSFET driver works as a voltage controlled switch. In *figure 6.5-3* the schematic of the microchip TC4427 can be seen. The output voltage will be the same level as the input voltage of the MOSFET driver and is the non-inverting variety.

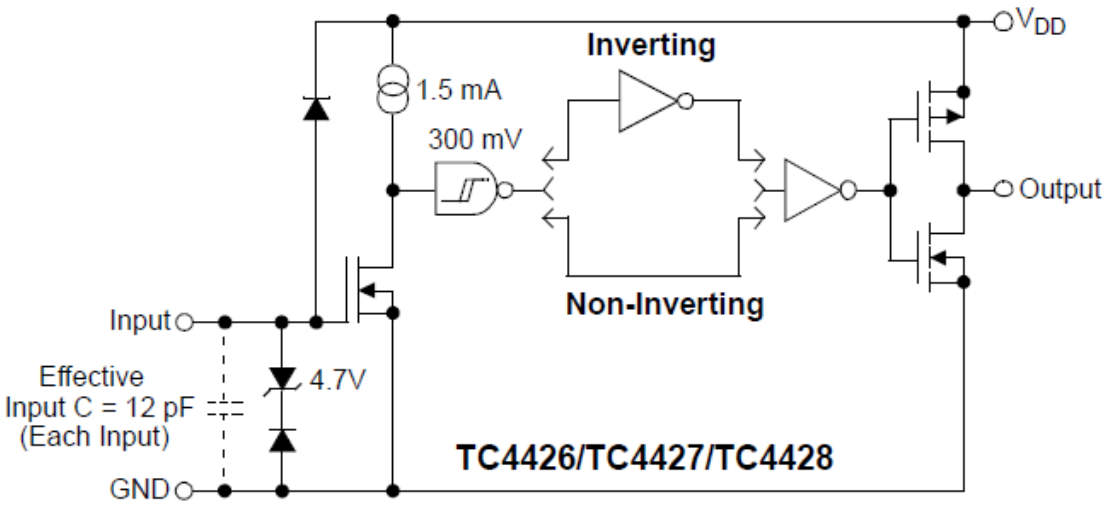

Figure 6.5-3: Microchip TC4426/27/28 MOSFET Driver

Image courtesy of Microchip

The MOSFET driver is used to turn the MOSFET on. With the MOSFET on the current can flow through the pump. Thus turning the pump on. The MOSFET can be used like a relay or a voltage controlled switch. The MOSFET that is going to be used is the Fairchild IRF630B. This MOSFET is an n-channel enhancement mode field effect transistor. It has the capability to handle up to nine amps and is fast switching. In *table 6.5-4* the characteristics of this MOSFET can be seen. This MOSFET was chosen because of the low cost and ease of acquiring these MOSFET's.

	Minimum	***Typical***	***Maximum***
Vdss	--	--	200 V
Id	--	--	9.0 A
Vgss	-30 V		30 V
Vgs	2 V		4 V
Rds		0.34 Ω	0.4 Ω
Td(on)		11 ns	30 ns
Td(off)		60 ns	130 ns
Price		$0.40	

Table 6.5-4: Characteristics of Fairchild IRF630B

When the MOSFET receives the necessary five volts at the gate, the MOSFET will turn on. This will engage the pump motor and thus start the dispensing process. The voltage must remain high for the duration that the pump is on. To determine the length of time that is necessary for the pump to stay on for, simply divide the amount of fluid that should be dispensed by the flow rate of the pump.

Our pump has a flow rate of 100 ml/min. For a one ounce shot of liquor the pump must remain on for exactly 29.57 seconds. This will be done in the coding portion of the controller.

To get the pump to stay on for the allotted amount of time, a delay in the code is used. An interrupt or a software delay can be used. Either of the two will work but the software delay will take more power due to the fact the processor is performing tasks the entire delay. Unfortunately, no one in the group is familiar with using an interrupt so more research is necessary to use this in our code.

The pump is connected to the positive side of the power supply and the negative connection of the pump is connected to the MOSFET. This connection can be viewed in *figure 6.5-5*. This is ultimately how the pump is controlled.

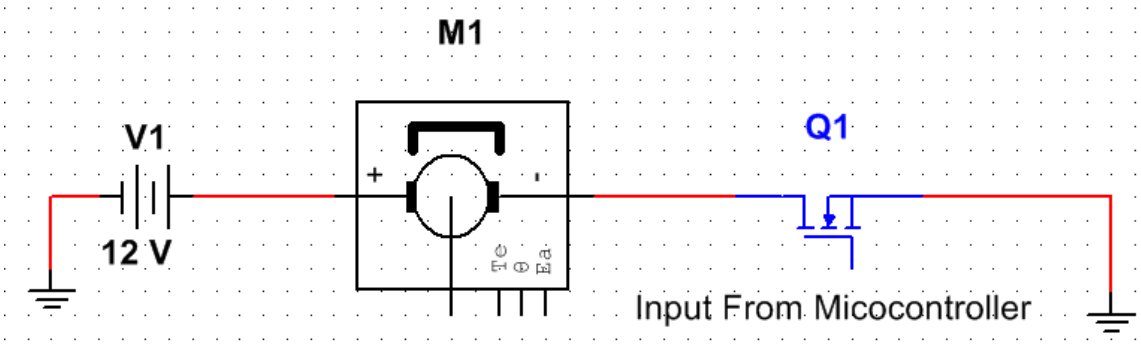

Figure 6.5-5: Single Motor Controller

The single motor controller will be duplicated for the entire pump. Each pump will be connected to its own MOSFET. The only shared parts between the pumps are the battery and the ground source. The large schematic of all the pump controllers can be seen in *figure 6.5-6*. Each pump only pulls about 200 mA and if all pumps were to be on at the same time then the system of pumps would pull about 1.8 A. Our power supply is able to handle up to 5 A, so upgrading this system would not be difficult at all.

All of these components put together make up the D*rink*W*izard*'s controllers. Each sensor will ensure a smooth and trouble-free operation of the D*rink*W*izard*. Through much research and trial and error, we believe this to be a suitable design and hope it will work in the practical implementation of our design.

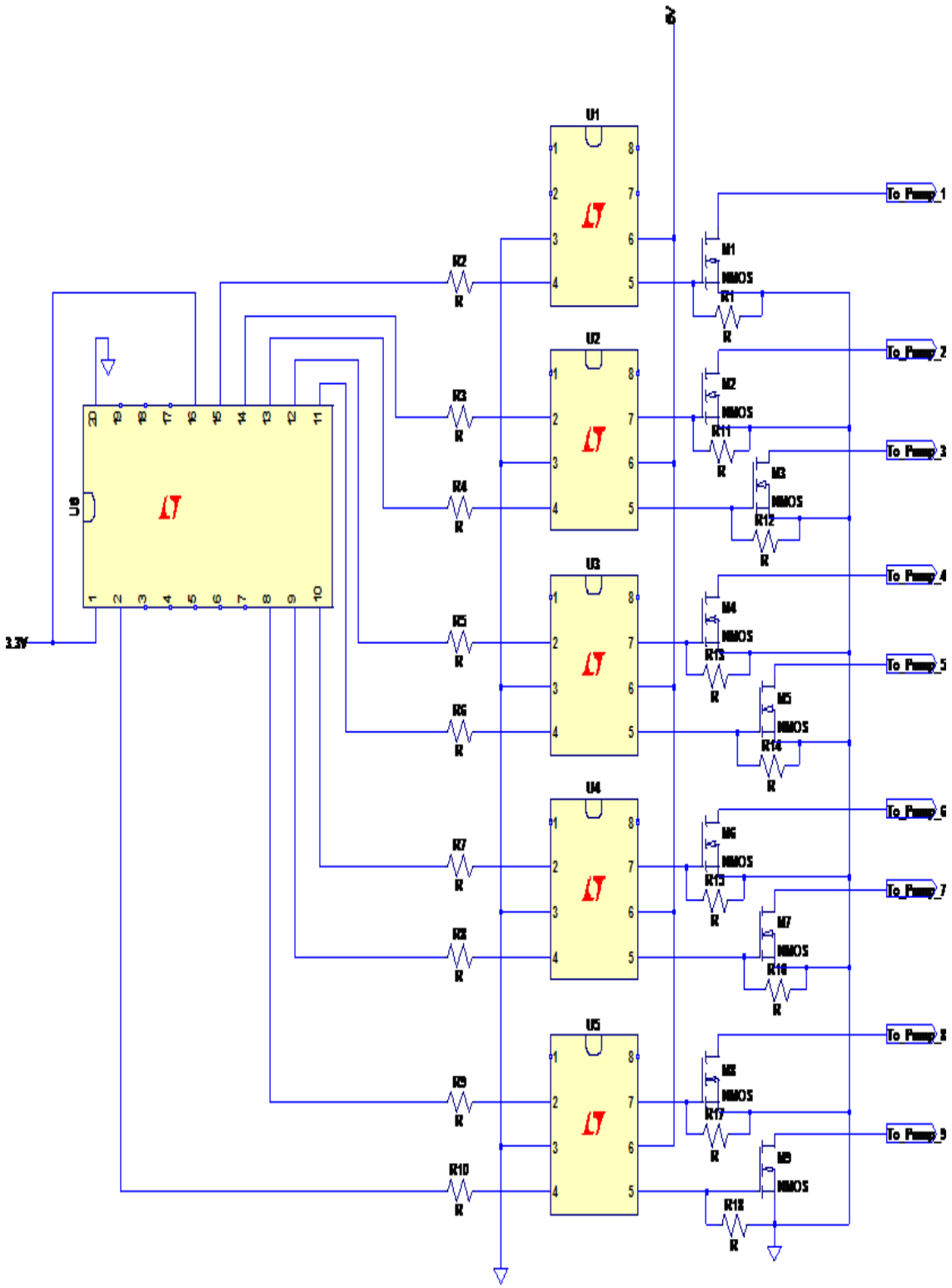

Figure 6.5-6: Pump Controller

6.6 Breathalyzer

The D*rink*W*izard* is going to have an additional selling feature to potential buyers. This feature is an active alcohol-measuring sensor that will actively monitor the alcohol content on the user's breath. The unit is going to be tethered to the Android Tablet through a very small USB cable. While the alcohol sensor is not as reliable or calibrated as a professional Breathalyzer, it will still give a percent of alcohol measurement from a given sample. We believe that this should be a relatively strong and repeatable measurement to search its purpose in the D*rink*W*izard* application.

During the research portion to find relative technologies for alcohol sensors and complete Breathalyzer units we determined that the easiest and best solution to achieve our design was to go with the MQ3 alcohol sensor and board. Keyes produces the unit. The key feature of this unit is that the integrated board provides a simple 4-pin output. An image of the Keyes MQ3 alcohol sensor and board can be seen in the *figure 6.6-1* below.

Figure 6.6-1: Keyes MQ3 Alcohol Sensor

Image courtesy of Gearbest.com

The Keyes alcohol sensor and board is 1.2 inches long x 0.7 inches wide x 1.1 inches high. It uses a 5V input voltage. There are a few main advantages of using this unit. The first main advantage is its relative small size, which will be easily integrated into a small enclosure. The second advantage of this unit is its ability to utilize a digital output on one of the 4 pins that this already built into the board.

The Keyes board that comes with MQ3 alcohol sensor has one key feature that allows us to easily integrate it almost perfectly. The Keyes board comes with an adjustable potentiometer. This potentiometer allows the user to adjust when the

digital output signal will turn to a logic level “high”. This feature allows us to exactly decide how much alcohol the sensor will detect before the D*rink*W*izard* will no longer allow the user to order another drink.

The Breathalyzer unit should be ideally compact and unobtrusive to the user. With this in mind the Breathalyzer is going to be powered directly from the USB port located on the Android Tablet. The USB port will provide power and ground for the sensor to function properly. The digital out coming from the Keyes MQ3 board will be fed to the Rx pin of the USB header. The image of the USB header below is the type of connector going to be used. The Keyes board offers an analog output as well but will not be required. *Figure 6.6-2* shows the USB port with 4 pins.

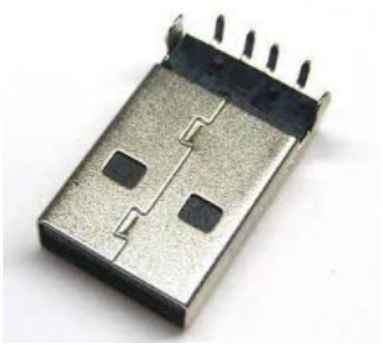

Figure 6.6-2: USB Port

Image courtesy of Ebay.com

The most difficult and intricate part of finalizing the design of the Breathalyzer component is the final fit and finish of the housing. The enclosure that we selected was from OKW Enclosures. The small enclosure that we selected is constructed from Duroplastic. OKW enclosures are specifically made for potting of small electronic assemblies. They are especially resistant to deformation and extreme temperatures. The specific enclosure selected can be seen below in *figure 6.6-3*.

Figure 6.6-3: Duroplastic Enclosure

Image courtesy of Duroplastic

As seen from Figure 3, the enclosure is black, completely RoHS and REACH compliant. The enclosure is 1.776 inches long x 1.181 inches wide x 0.980 inches high. The key to integrating the Breathalyzer is to avoid having a bulky item that sticks out or does not seem to fit properly. The enclosure selected will have very little room to fit all of the components, but should provide an overall better fit and finish to the D*rink*W*izard* because of its small compact design.

The alcohol sensor works optimally when the air being measured is being blown across or over the sensor. This will be achieved with the addition of small tubing inlet that will lead directly to the sensor. A small hole will be drill on the front and backsides of the enclosure to allow the air being measured proper airflow across the sensor. Ideally this tubing will be approximately 3/8” diameter tubing with a small cutout directly over the sensor area.

The final assembly of the Breathalyzer system will have the Keyes MQ3 Sensor and board secured to the Duroplastic enclosure by a form of electronic safe adhesive. Ideally the adhesive being used would be a 3M doubled sided tape that is approximately 1” wide. The USB header would we soldered directly to the pins of the sensor board. The required hole for the USB jack would be cut out using a rotary tool with a cut-off wheel or routing bit. The required holes for the inlet and outlet of the air sample with we drilled through the enclosure with the proper size drill bit.

The finished product of the Breathalyzer will be a unique and fun aspect of the D*rink*W*izard* that is not offered or has been seen in any other products on the market.

6.7 Android Application

The Android application is a critical component to the D*rink*W*izard* and serves as a controller for the user. Most importantly, it allows the user to order drinks; however, it is also capable adding customized drinks from the user to the database and alert the user when their drink is ready to be picked up.

6.7-A: Programming Language

The language the Android application will be written in is Java. While the Java application programming interface (API) and Android API has some differences, they are very similar as are many of the classes. Furthermore, Java is the most common option for developing Android applications and is has the most support at this time which will aid the developers should any complications arise. Documentation for Android is also very well written and relatively easily understood. For these reasons, Java is the optimal language for writing the Android application for the D*rink*W*izard*.

6.7-B: Software Development Kit and Libraries

The Android Software Development Kit (SDK) is used for the creation of the D*rink*W*izard* application. It provides the APIs needed to simplify development. Also included are useful libraries, which will be imported into the project and then used to develop the application.

6.7-C: Integrated Development Environment

The Integrated Development Environment (IDE) used to write code for the Android application is Android Studio. Developed specifically for the production of Android applications, Android Studio is currently the official IDE for Android development. It is built around IntelliJ IDEA and provides multiple features in addition those that come with IntelliJ.

Useful features provided by Android Studio include but are not limited to: an Android application package (APK) generator, a user interface layout editor, and the ability to easily sign APKs. Debug versions of applications should never be used during the final release and Android Studio provides options to build release versions of APKs. Related to this is the ability to sign generated APKs. A signature provides a way to identify the author of the APK. While this is not critically important for the D*rink*W*izard*'s application upon initial release, it should not be overlooked and serves as a measure of completeness. Finally, the user interface layout editor will be useful for designing an effective, familiar user interface. It has the ability to insert buttons, text boxes, and other elements that will make the D*rink*W*izard* application feel familiar to anyone who uses it. These

are just some of the features that make Android Studio an excellent IDE for developing this application.

6.7-D: Android Compatibility

The D*rink*W*izard* Android application is developed using API Level 14, also known as Ice Cream Sandwich. Any device with a platform version lower than Android 4.0 will not be able to run the D*rink*W*izard* application. According to Android Studio, this will allow the D*rink*W*izard* application to run on 87.9% of all devices active on the Google Play Store. Developers decided upon this level because of the Wi-Fi Direct and Bluetooth functionality.

The other major component needed for successful use of the D*rink*W*izard* application is Bluetooth. All communication between the mixer and the Android device will occur through Bluetooth; therefore without Bluetooth capability, the app will not be able to connect to the mixer.

Another requirement for the device with the potential to run the D*rink*W*izard* application is storage space. The application itself will also be no larger than fifty megabytes. This size is a standard set by the Google Play Store and is not anticipated to be an issue for most devices.

6.7-E: Application Structure

The application has three major activities in which the user can interact with. They are labeled Main Activity, Order Activity, and Custom Drink Activity. These three govern the entire application and make simplify the processes of connecting to Bluetooth, ordering a drink, and creating a custom drink. There is another activity called Paired List Activity but it is only used for pairing with devices, breaking bonds with devices, and connecting to bonded devices and shouldn't be accessed as often. The user interface was designed with the layout editor in Android Studio. The *diagram 6.7-E* below outlines how the three major activities interact with each other.

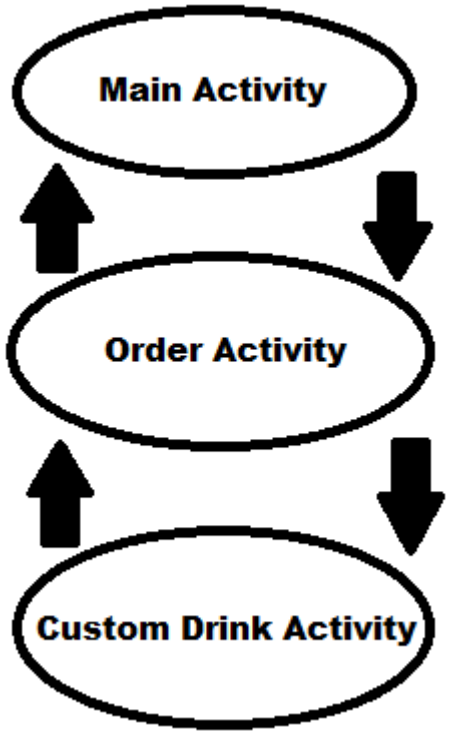

Figure 6.7-E: Activities Diagram

Main Activity handles administrative activities such as connecting to Bluetooth and building the database of drinks. It is able to turn Bluetooth on and off, pair with a visible device, show the currently connected devices, and take the user to Order Activity. If Bluetooth is turned off when the user attempts to show the connected devices, the application will alert the user that Bluetooth is not enabled and present the option to turn it on.

Order Activity provides the major functionality the D*rink*W*izard* application was designed for. It is familiar, intuitive and allows the user to order a drink. It also updates the user when their drink is being created and when it is finished. It will also have the option to go back to Main Activity in case an administrative action needs to be performed. This may happen if the Android device loses the connection to the mixer for any reason. Another point about Order Activity is that it keeps a private instance of the Drink List class, which has an Array List of objects that hold the information about the ingredients, which compose a drink. As noted earlier in this document, this ensures that the D*rink*W*izard* Android application will be able to store over two billion drink combinations. The number of combinations that can be created defines a more practical limit.

Finally, Custom Drink Activity is used to create new drinks. The user is allowed to add ingredients to a beverage of their choice. Sliders are displayed and can be moved back and forth to add or remove a liquid. Should the user craft a drink that

the developer's feel is not safe, a warning will be displayed that will tell the user why it is unsafe and ask if they wish to proceed. The D*rink*W*izard* application will never disallow a drink from being made and the responsibility is left to the user to continue creation or go back and edit the drink. This activity will also allow the user to return to Order Activity without creating a drink, completing the cyclical structure of the application.

6.7-E-1: General Bluetooth Functions

Android requires permissions to keep applications from accessing unnecessary features on the device. Developers must manually include permissions and as a security measure, when the application installed, the user is shown a list of the included permissions. This serves as a security measure and keeps applications from gaining access to personal information. Bluetooth also requires permissions. There are two permissions, which must be written into the Android Manifest called Bluetooth and Bluetooth Admin. The Bluetooth permission is needed for all Bluetooth actions such as data transmission and connection management. Bluetooth Admin handles administrative capabilities such as allowing the device to be discovered and discovering devices in the area. Without these two permissions, the application would not be able to communicate through Bluetooth and would likely crash.

The Bluetooth Adapter class is used to represent the hardware Bluetooth adapter on the device. It is capable of typical Bluetooth actions such as scanning for visible devices, enabling and disabling Bluetooth on the device, retrieving a list of the paired devices, and many other actions. When Main Activity starts, it calls the getDefaultAdapter() method of the Bluetooth Adapter class to give the application access to the Bluetooth module of the Android device. A button to toggle Bluetooth on and off is included in Main Activity. If Bluetooth is currently off, the application launches an Android Intent for Bluetooth.

Intents are Android's way of allowing an application to start another Activity or Service. The Intent launched by the D*rink*W*izard* application requests that Bluetooth be enabled. Because enabling Bluetooth could open the device up to malicious software, user permission is required to enable it so a dialogue box opens and asks for confirmation before turning Bluetooth on. The user will be alerted when Bluetooth has successfully been turned on. If Bluetooth is on when then button is pressed, Bluetooth is simply turned off and the user is notified.

Connecting to the D*rink*W*izard* mixer is also handled partially by Main activity. A button for searching for discoverable devices is included and when pressed, it calls the Bluetooth Adapter object's startDiscovery() method. This function signals the device to begin searching for devices. Other actions will be suspended while the search is being performed. For this reason, the

cancelDiscovery() is called if the search process is cancelled to make sure that the discovery process has stopped. During the discovery process, all other connections experience latency. After the discovery process has finished, to connect to a device the user can simply touch the button next to the device to pair with it. The application calls the createBond() method to pair the two application with the D*rink*W*izard* mixer.

Another button is used to enable and disable Bluetooth, depending on the current state. The Broadcast Receiver determines what happens in the application based upon the incoming Bluetooth states. If the state of Bluetooth on the device is changed, the application detects this and shows a toast with the text “Enabled” to alert the user that Bluetooth has been turned on. If the device discovery process has been started, the Broadcast Receiver overwrites the old list of devices and shows a dialog box which says, “Scanning for local devices…” This box will go away on its own accord once the scanning process is complete or the user can cancel it. If it is cancelled, cancelDiscovery() is called. The next signal the receiver listens for is received when the discovery process is finished. Once it is finished, the dialog box is closed, and a new intent is created to start Paired List Activity. This activity will be covered in depth next. The final signal the Broadcast Receiver listens for is when a device is found. When a device has been found, it is added to the list of devices known to the application and a toast is shown to the user displaying the name of the device.

Paired List Activity shows the devices seen by device with the D*rink*W*izard* application. First, it receives a list of all devices seen by the application then creates a list and displays the devices in it. A class to complement Paired List Activity’s list view named Device List Adapter has been included. It contains helper methods to format and get information from the list. The adapter handles input from the user touch a device in the list. When a device is touched, the device at that position is retrieved and checked for a bond. If a bond exists, the user is given the choice to connect to the device or break the bond. Otherwise the device is paired with D*rink*W*izard* application’s Android device. The methods createBond() and removeBond() are invoked to handle bonding. A separate thread called Connect Thread handles connecting to the device. It will be covered next. Like Main Activity, Paired List Activity also contains a Broadcast Receiver. This one waits for signals about a change in the state of bonds. If this state is received, the user is notified via a toast about the current state of a bond and the adapter is notified of a change within the list. The list then refreshes itself. Paired List Activity also contains the methods showToast() which makes it easier for developers to show toasts and onDestroy() which releases the Broadcast Receiver.

6.7-E-2: Communication from the Android Perspective

Threads are used for connecting and sending and receiving data via Bluetooth. Connect Thread is the link between the application and the serial port through which data travels. When the thread is started, the application attempts to create a radio frequency communication socket with the device the user requests to connect to. To do this, a Universal Unique Identifier (UUID) is used. For safety, a fallback socket is also created. A connection is established with this socket only if an exception is thrown by when the application attempts to connect to a remote device. Without the fallback socket, if a connection is not made, the application cannot communicate with the mixer.

After a successful connection has been made, the application will begin listening for data from the mixer. Constantly listening will cause the central processing unit (CPU) usage to remain at relatively high levels for the duration of the application's run time. To remedy this, the device will only begin listening to the channel once an order for a drink has been submitted. This is possible because the mixer should not be transmitting any data to the application when it is idle (when no drinks are being made). A class, which extends Runnable called Receive Runnable, handles the incoming data from the mixer. Upon running this class, a method called readChannel() is called. This method attempts to get the length (the number of the bytes) of the incoming data then iterate through the bytes one by one and add each one to an array, which will later be turned into a human-readable string. If a delimiter is found, the bytes are copied from the buffer they were being appended to and decoded to a readable string. The data is now ready to shown to the user. This process is done multiple times to inform the user of the state of their drink.

A class handles sending data, which is also a child of Runnable. When the thread is started the data needed to make a beverage will be prepared. Once it is ready to be interpreted by the MCU, it will be sent using the open socket's output stream. This completes the Bluetooth-related functionality of the D*rink*W*izard* Android Application. The application is now connected to the mixer and data to make a drink can be sent to the mixer. Likewise, information of the stage of the drink can also be received by the D*rink*W*izard* application. Two class diagrams have been included below to outline the process of creating the previously described Bluetooth functionality. They were created using ObjectAid's class diagram plug-in for Eclipse. The Bluetooth Class *figure 6.7-E-1* is shown below.

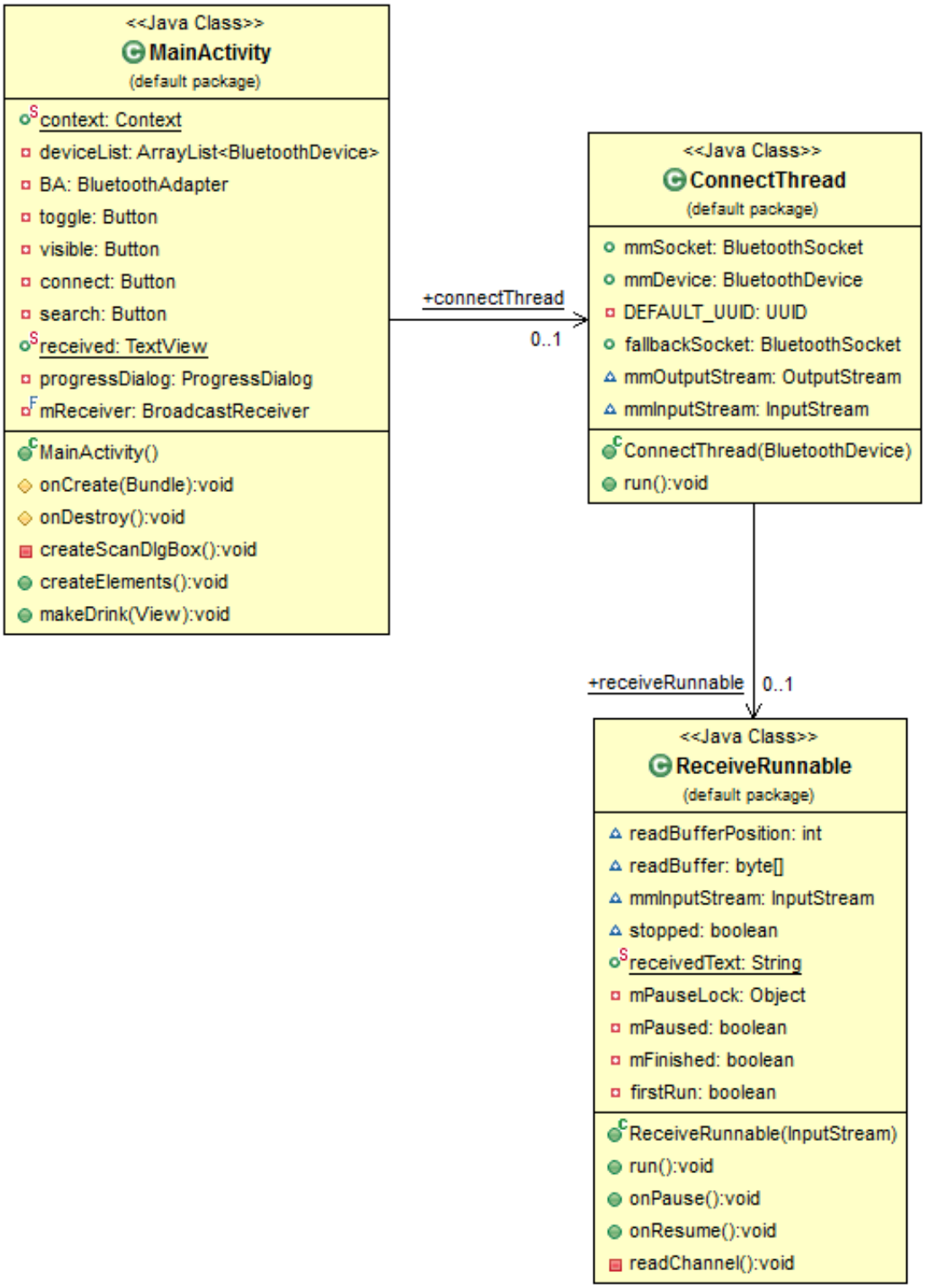

Figure 6.7-E-1: Bluetooth Class Diagram 1

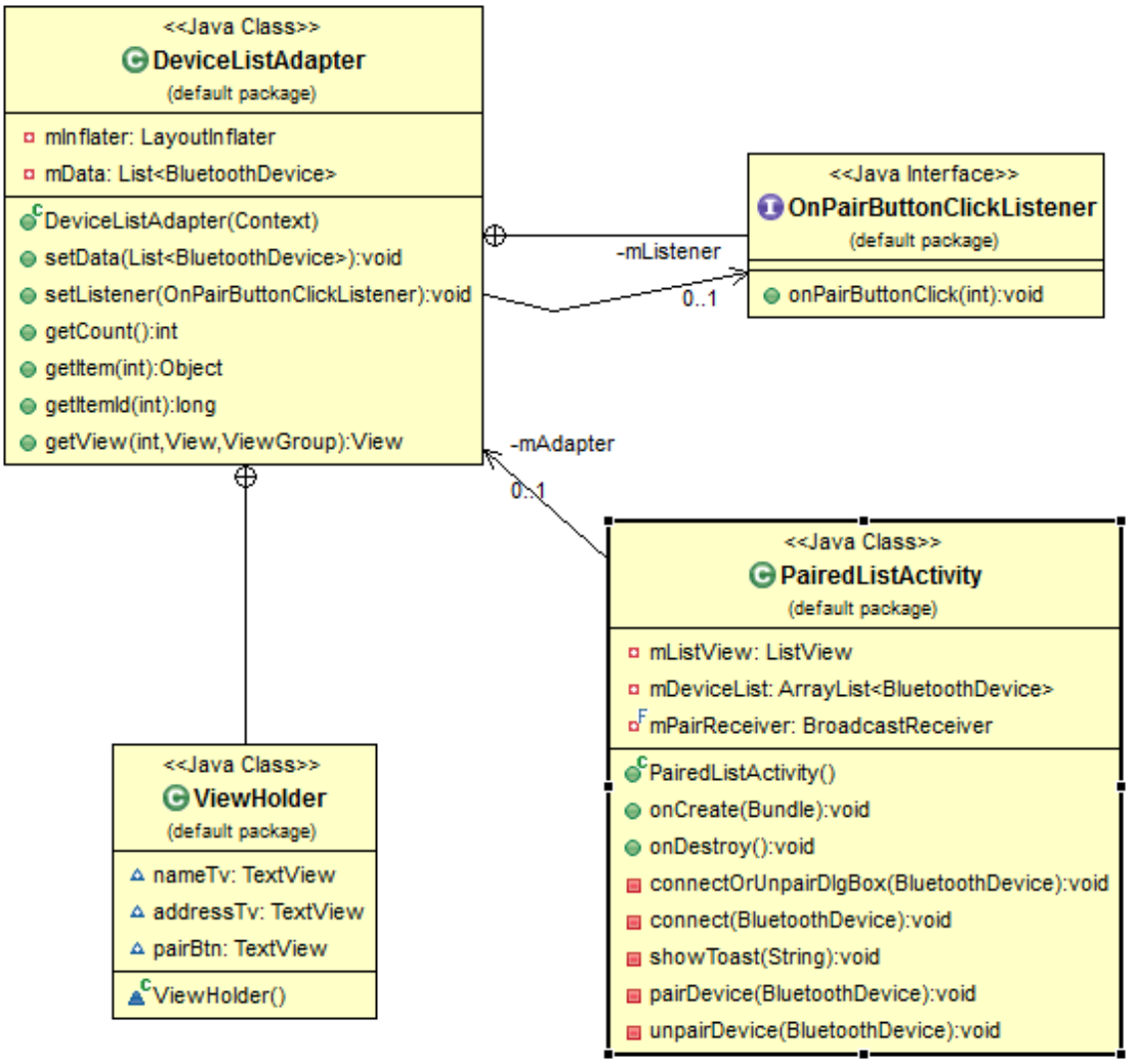

Figure 6.7-E-2: Bluetooth Class Diagram 2

Universal Serial Bus control is integrated into the default Android libraries. A Broadcast Receiver controls the communication with the USB device. The onReceive() method is overridden and the incoming intent is used to find the device. Included in the application will be a button to signal a thread to begin listening for data from the USB port so that the thread is not continually running in the background, thus causing the device's central processing unit (CPU) to run harder. As with Bluetooth and all other hardware devices, Android requires an Intent to communicate with and access the device USB port (if one exists on the device). The permission required for USB access is "USB Accessory Attached" and without it the application would not be able to recognize the device. When the application is closed, communication with the USB device should be closed by calling the USB Accessory's close() method.

6.7-E-3: Ordering from the Android Perspective

Order Activity is a subclass of Android's List Activity and is the primary interaction point between the user and the mixer. This class begins by creating a database of drinks, which has been created by hand by the developers. It then pulls the names of the drinks from the database and uses them to fill in the list. An Array Adapter, which contains the names of the drinks, is created and passed into the List Activity. The previous two actions are done by a method called buildAdapter(). The default listener of List Activity is used for user input. When an item is touched, the application checks to see if the user has touched the option to create a custom drink. If this was the option selected, Custom Drink Activity is started. If one of the items in the list was a drink, the program calls the requestConfirmation() method. A dialog box is constructed which asks the user to confirm their selection. A screen shot has been included later showing this process. Other notable methods in this class include buildMessage() which constructs a string to show in the confirmation dialog box, showToast() which allows for Android toasts to be shown more easily, and showOptions() which asks the user if they would like to delete an drink. It is shown after a long press on a drink. This class also contains a Drink List object.

The Drink List class contains an Array List object, which holds all the drinks in the application. A getter method for the list of drinks has been included for convenience and so has a method to get an individual drink by name. The most used method in this class is the addDrink() method which takes in all the necessary information for a drink, creates it in the application, and adds it to the master list. The addDrink() method creates Drink objects. Another class Drink holds all information about a given drink. This information includes how much of each ingredient goes into the drink. Possible ingredients are vodka, tequila, rum, whiskey, peach schnapps, orange juice, cranberry juice, sour mix, and pineapple juice. For convenience a method has also been included to get the index of a drink. Getter methods for how much of each ingredient goes into a drink have also been included. Setters are also in the class and are used by the SQL database, which is covered later. A class *figure 6.7-E-3* has been included below. It illustrates the order process. This diagram was also created using the ObjectAid plug-in in Eclipse. Following the class diagram is a screenshot of Order Activity. The black box is not a flaw in the software. It is used for censorship in this document and is not present in the application itself. Also shown is an activity drink list in *figure 6.7-E-4*.

Figure 6.7-E-3: Order Class Diagram

Figure 6.7-E-4: Order Activity Drink List

Included in the list of drinks is an option to create customized beverages. These drinks can be created from nine Android Seek Bars. Each of the Seek Bars corresponds to one of the nine beverages in stored in the mixer and the user can slide the thumb of the Seek Bar up and down to add or remove the amount of the corresponding liquid. There are also buttons included to save the drink and go back to the drink list without saving it. If a drink is believed to be unhealthy or dangerous, a dialog box will pop up warning the user that they should not continue with the creation. They will have the option to dismiss the dialog box and continue if they wish to make the drink anyway.

The Custom Drink Activity class has many fields for the seek bars, buttons, and text views. All of these are initialized in the onCreate() method of this activity. The setupViews() method connects all views from the layout to the activity. The setupSeekBars() method finds all Seek Bars and creates a separate listener for each of them which update the views that show the amount of each liquid that will go into the custom beverage. The buttons are created in the setupButtons() method. If back is pressed, the activity will simply be finished and the user will be taken back to Order Activity. When the save button is pressed, the intent that started the activity is retrieved and the information is sent back to Order Activity where it is extracted in the onActivityResult() method. From here, the addDrink() method is called and it is added to the list of drinks. The user can then select it as

if it was a standard option. The following screenshot shows an example of the current drink creation process. Below is *figure 6.7-E-5*.

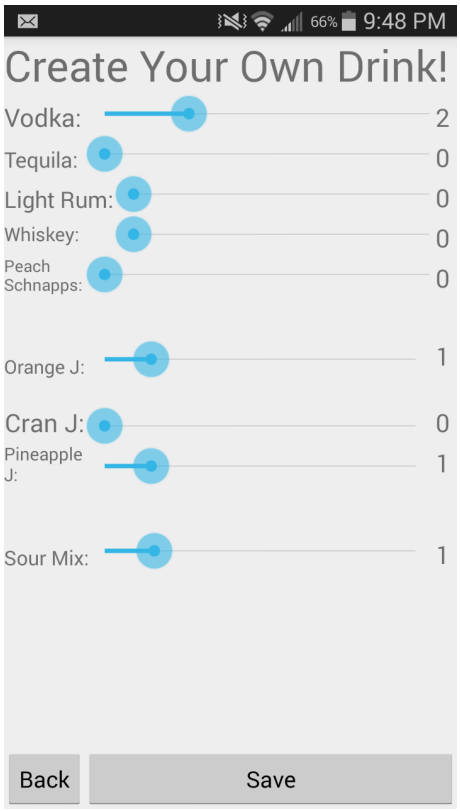

Figure 6.7-E-5: Custom Drink Activity

One final note about Order Activity is that it checks with the user first to make sure the selection they have made is what they want. To do this a dialog box is shown to the user, which contains the name and ingredients of a drink. All the information from the dialog box is pulled from the database. In the future, the dialog box may include a short description of each drink and a picture may be included. The following screenshot shows what the confirmation screen looks like currently. All methods check to determine whether a chosen option is one of the default drinks generated by the application or whether it is a custom drink created by a user. This ensures that the application will behave properly whenever a drink is added or deleted. Below is *figure 6.7-E-6*.

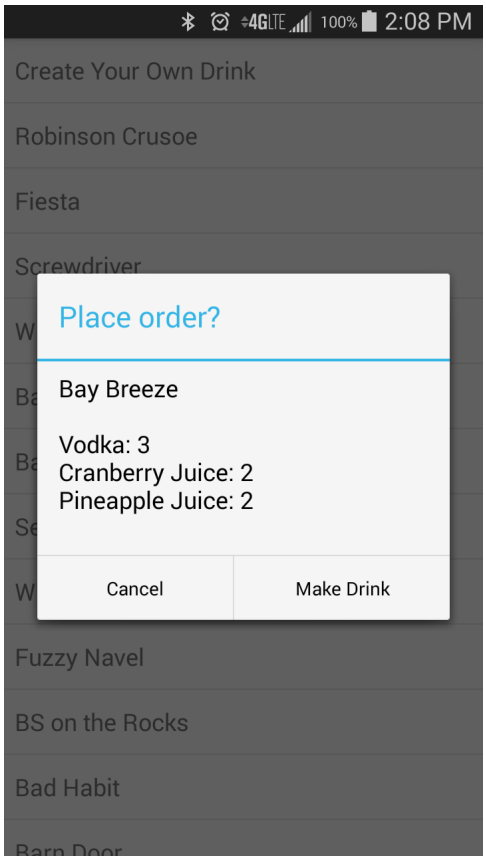

Figure 6.7-E-6: Drink Confirmation Dialog Box

6.7-E-4: Storing Custom Drinks

Without a database, custom drinks would immediately be deleted whenever the device was shut off or even when the application was closed. Because new data can be actively generated at almost any point in Order Activity's lifetime, developers decided to either write new drink data to a file or keep a dedicated database for the data. Instead of keeping track of a file, which could easily be deleted by going through the file system of the device, it was decided that a true database should be used. For this reason, an Android SQLite database is used to keep track of all custom drinks.

The class Drink SQLite Helper extends Android's SQLite Open Helper class. Contained within this subclass are fields of all possible ingredients for a drink and the name of the drink. The onCreate() method has been overridden and it creates a new database using the database create statement. The onUpgrade() has also been overridden to delete the old table and create a new one if a later version of the table has been created. This class is only for the initial, administrative functions of the SQLite database. The main functions come from another class called Drink Data Source.

Drink Data Source defines all the standard activities for database. It has fields for the database itself, the helper class, and all the columns of each entry the database. The constructor instantiates the helper class; the database is created when the source is opened. Open and close methods have are included to open the database, which should be done before any functions are performed on it, and close the activity when no more actions need to be performed on it. One of the most used functions is createDrink(). It is supplied with all the parameters to create a drink and creates, then adds a drink to the database. The opposite deleteDrink() takes a drink as a parameter and finds the drink's id in the database before removing it. The final two methods are cursorToDrink() and getAllDrinks(). The former returns a drink with the all the parameters at the current position of the cursor and the latter returns a list of all drinks within the database. Order Activity and Custom Drink Activity have been included in the following class diagram, which covers the SQLite database section of the application. Order Activity is the same class from the above class diagram. *Figure 6.7-E-7* is shown below.

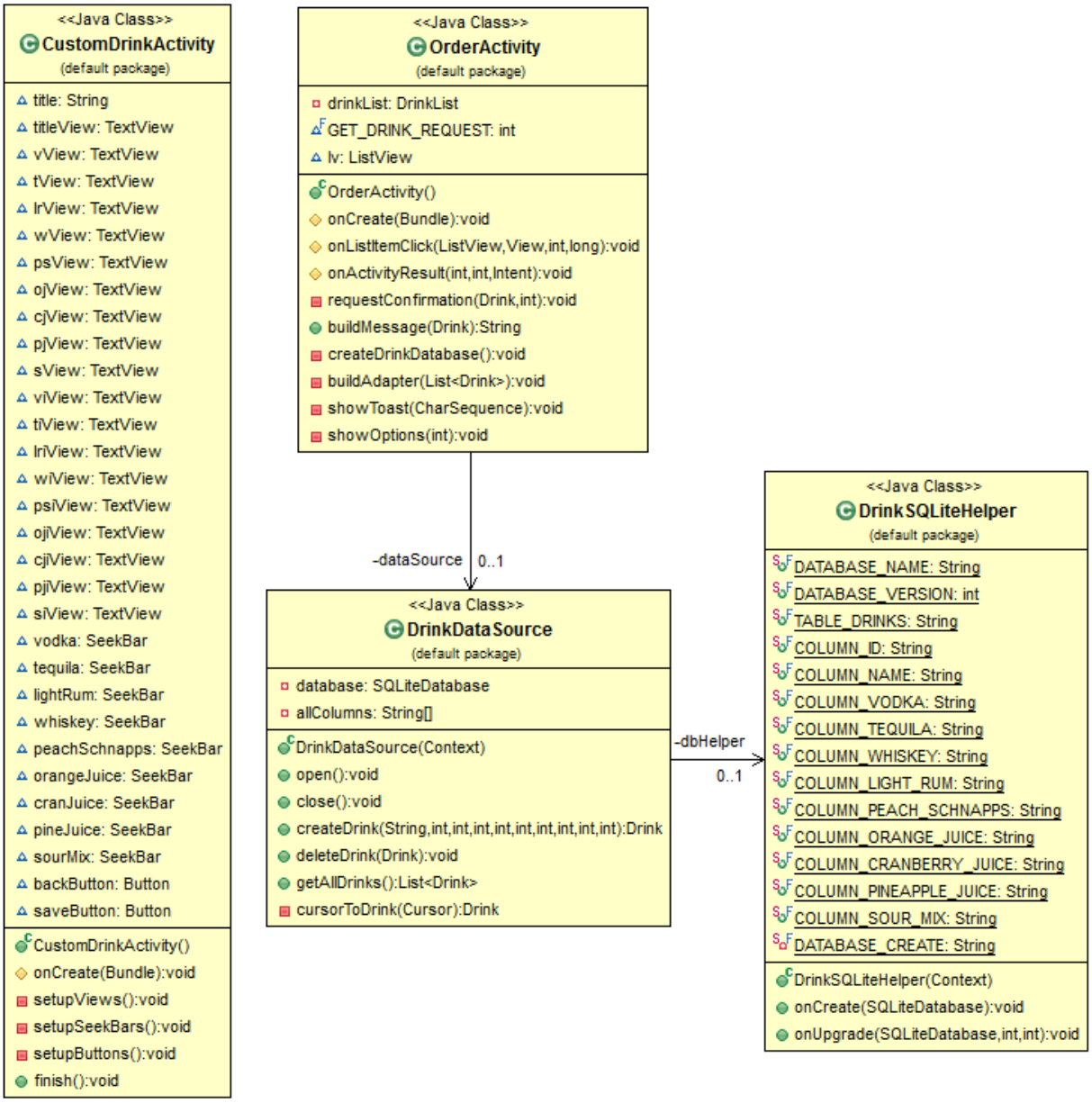

Figure 6.7-E-7: SQLite Database Class Diagram

7.0 Prototype Construction

7.1 Parts Acquisition

The D*rink*W*izard* project was not sponsored or funded by any outside sources. Our team agreed to attempt to split the projected $600 budget 4 ways totaling $150 per project team member. The general understanding between the team is that each member has a budget to purchase as many components as possible for their respectively portions of the project.

The PCB hardware has an initial cost estimate of $100.00. One of the team members works locally for an engineering company name MtronPTI with access to a LPKF quick turn prototype-milling machine. The material for our PCB will be constructed out of a two sheet of copper that is sandwiched between a dielectric material. The projected material is Rogers's duroid 5880, as it is ready found in scrap sizes at the team member's place of employment. The cost of routing out the board will also be completed saved by completing the milling and routing at MtronPTI while not on company time. We believe that this will be a valuable asset to our team as the turn-around time to complete a finished PCB can be less than one day.

The all of the components on the PCB will be primarily chosen to be surface mount and through-hole devices. The team member currently employed at MtronPTI also has access to numerous kinds of Hako soldering irons, heating plates, pre-heating ovens and a complete reflow oven if the application was a necessity. The parts to construct the PCB and routing materials will be acquired through MtronPTI.

There will be two microcontroller units located on the D*rink*W*izard*. The main microcontroller will be used to control the fluid pumps and the ultrasonic sensors. The smaller subsystem microcontroller will be used in handling the inputs of the liquid level load cells and outputting characters to an LCD through a UART connection. Both of these microcontrollers are going to be the ones initially projected as the ones to be used from our Launchpad kits from a previous class, the MSP430g2553. These have been already acquired from the UCF bookstore.

The Bluetooth module is going to an HC-06 Bluetooth to UART converter. It is an extremely popular Bluetooth communication module and can be readily found on most Internet retailer's website. Our team decided to acquire this part from Amazon.com prime to utilize the free 2-day shipping.

The components of the power supply will be sourced and acquired through Internet vendors and retailers. Mouser and Digikey will be the two main sources

that we will use to acquire any various components being used in our power supply subsystem.

The main vending unit is going to going to be constructed of either some form of plywood or Lexan Plexiglas. The construction of either proposed type is going to require various forms of nails, screws and adhesives. Since most of these materials are general construction materials, we will acquire the various components from a local construction material retailer such as Home Depot or Lowes.

The LCD screen display will be acquired through Newhaven Display International. We will be acquiring the display through their website. Newhaven is located in Elgin, Illinois and we believe we should not have trouble allocating one of their blue-white LED backlit serial LCD displays.

7.2 Construction Techniques and Materials

Vending machines have been around since about 215 B.C. where the first vending machine accepted bronze coins and in return dispensed holy water for the temples in Egypt. It was not until 1888 that vending machines became a viable market in the United States. Vending machines today are primarily constructed using four leading materials. These materials are galvanized steel, Lexan, acrylic powder coatings, and polyurethane insulation. The D*rink*W*izard* Is envisioned as being an alcoholic drink vending machine. Since the *DrinkWizard* will be a vending machine style dispenser the research for construction materials started with what most vending machines are made of today. Since the group will fund the DrinkWizard some alternative, lower cost, materials were considered as well. The materials that were reviewed are galvanized steel, Lexan, acrylic, and wood.

7.2–A: Galvanized Steel

Galvanized steel has many benefits and becomes quite apparent why it was chosen as the material for drink machines today. That first benefit of galvanized steel has lower initial cost, galvanizing is now lower that painting. Galvanized steels second benefit is less maintenance, which equals lowest long-term cost. Galvanizing has a long life expectancy; most structural members are in the area of 50 years and 20 to 25 years while even in severe urban and coastal contact. The fourth benefit of galvanizing is reliability. Galvanizing has a minimum coating thickness which is dictated by the Australian / New Zealand Standard 4680. The 4680 standard produce a coating life that is predictable and has extremely reliable performance. The fifth benefit of galvanized steel is it has the toughest coating. The galvanized coating has an exclusive metallurgical configuration which gives superior resistance to mechanical damage in transport as well as when it is assembled and serviced. An added benefit of the tough coating is

automatic protection for damaged areas. The galvanized coating corrodes favorably to steel; the coating provides protection to small areas exposed due to damage. The galvanized coating is unlike other typical coatings because the small damaged areas do not need to be touched up. An added benefit of the galvanized coating is the galvanized coating provides complete protection. The galvanized coating protects all parts of the material; it gets in the corners and sharp recesses that other coatings could miss. Galvanized steel sheets are reasonably priced; a typical 4-foot by 4-foot sheet is $30.00. The price becomes a factor since the individuals in the group will fund this project. Galvanized steel, with all its benefits, is a very favorable choice.

7.2-B: Lexan

Lexan is another durable material and consequently it is ideal for vending machines. The first benefit that Lexan provides to vending machines is its strength. Lexan's most distinctive attribute is the strength and high impact resistance it provides. The strength that Lexan resin provides is shatter-resistant and virtually unbreakable. This durability is consistent in very hot and very cold temperatures. The added benefit of Lexan's strength is walls can be made thinner which leads to a lighter weight product. This would help the D*rink*W*izard*, reach the goal of being a more portable vending machine. The durability of Lexan is impressive. The heat resistance is up to 212 degrees Fahrenheit and can withstand temperatures down to -40 degrees Fahrenheit. Lexan is water, weather, flame and ultraviolet light resistant. This allows Lexan to stand up to any typical, as well as unpredicted, conditions we could throw at it. The appearance of Lexan is another benefit; it comes in many different forms.

Lexan can take on an optical purity where it is crystal clear, can have dramatic surface textures, unlimited colors, consist of metal flakes, and even translucent. The options for Lexan are virtually endless. One of the biggest benefits of Lexan is the electrical compatibility it offers. Since Lexan can have a broad range of flame retardant grades it makes it ideal for fuse boxes, switches, plugs and sockets. This means we do not have to worry about Lexan conducting current or arching to our power supply. The downfall to Lexan is the cost. The same 4-foot by 4-foot sheet is double the cost, about $70.00. This is something to consider since this is coming out of the group members pockets. Lexan, besides the cost, is a very appealing choice for the durability as well as the weight reduction it can provide.

7.2-C: Acrylic Sheet

Lexan was an extremely appealing choice, the cost is what was concerning. Acrylic sheets caught our attention because it is very comparable to Lexan. Acrylic is benefit is that it provides glass like characteristics such as clarity but at 10 times the impact resistance and at half the weight of class. Acrylic, like Lexan,

is impact resistant. Acrylic is also weather resistant. Acrylic sheets can be treated with a specialized UV treatment to help safeguard that the acrylic wont turn yellow after a sustained exposure to sunlight. Acrylic sheets are easy to cut. This is an added benefit to the group members because this will be built at our house and not in a manufacturing plant. Acrylic has the benefit of being very scratch resistant, acrylic is typically found in display cases. This scratch resistant quality is very appealing for a vending machine that is going to be made to be transportable. Acrylic can also stand up to the cold conditions. This durability also allows acrylic to be shatter resistant. This shatter resistance would be a good quality for individuals trying to break into our vending machine. Acrylic, as with Lexan, can be infused with colors and can be translucent.

The one drawback with acrylic is that it is not as flame resistant as Lexan. Acrylic given the right amount of heat will become moldable and can change shapes. If we want the product to withstand a fire this would be a major consideration. The D*rink*W*izard's* typical use will be inside or outside. The heat exposure from the sun is not enough to mold the acrylic sheet or change the shape of the sheet. Acrylic is also chemical resistant, spilt alcohol on the inside or the outside will not affect the acrylic. Acrylic is also a good insulator; it has a higher resistivity than most plastics. Acrylic is extremely appealing for the D*rink*W*izard* due to the fact that it has the properties of Lexan but at half the cost. Acrylic sheets, that measure 4-foot by 4-foot, cost about $37.00 dollars each. The cost is very attractive and this makes acrylic a solid contender for the material of choice.

7.2-D: Plywood

Another option to save the group money is to use plywood. When plywood is first mentioned the face usually makes a scowling look and the thought is that plywood would not work. Contrary to what you might think plywood is a very feasible option. Plywood exhibits an extremely high uniform strength. Wood tends to be 25-45 times stronger along the grain versus across the grain, and then plywood manufacturing crosses the adjacent sheets to equalize the strength in all directions. The next advantage of plywood is the lack of shrinking, swelling and warping. Thanks again to the manufacturing process the adjacent panels balance the stress from shrinking, swelling and warping across all the panels. The most important quality of plywood is the non-splitting characteristic. Typical wood sheets split fairly readily along the grain; thanks to the crisscross adjacent sheets plywood can be nailed or screwed near the edges without damage from splitting. Another advantage of plywood is that it comes in large sheets, these sheets can then be cut down to the size we need. The advantage of larger sheets is that the larger sheets lower the cost per sheet that saves the group money on building cost. The disadvantage of plywood is that it is not resistive to fire or potential break in. Thief's can easily use saws to cut into the D*rink*W*izard* and take the alcohol; this would defeat the purpose of having age verifications. The price of plywood is what makes it really attractive for this group project; a 4-foot by 8-foot

sheet of plywood goes for $16.00. Plywood has some downfalls, however the price point is very appealing since the group is paying for everything out of pocket.

7.3 PCB Design and Assembly

After all of the extensive research and development is completed, a working prototype must be made. The prototype will consist of all the part that would be necessary to build a fully functional unit. One of the items that will be on this working prototype will be a printed circuit board that was designed by our group for the D*rink*W*izard*. The printed circuit board will be designed using the schematic that was also designed exclusively for our D*rink*W*izard*. A program called Eagle will be used to design the schematic and the printed circuit board. This is a new program for all of us and took a significant amount of research to learn how to use this.

Eagle is a powerful program that contains several different features. One of these features is the ability to design a schematic and create a board from that schematic. However, as a student at the University of Central Florida, we had never learned anything about printed circuit board design and at best the board created for the D*rink*W*izard* can be considered a hobbyist level design.

The entirety of the D*rink*W*izard*'s components is through hole soldering parts. This will make it easier to assemble and to troubleshoot. All of the components will fit easily inside of the D*rink*W*izard*. The electrical boards will be in their own enclosure fitted within the D*rink*W*izard*. This will give a nice completed look to the product.

The printed circuit board will be a double-sided board, meaning that both the top and bottom layer of the printed circuit board will contain a copper sheet. This copper sheet will be cut to allow parts to be connected via trace on the board. These traces will need to be able to handle the correct voltages and currents that are required for each individual part to work properly as needed unlike so many other senior design students have trouble in the past semesters.

Research has shown that signal and power traces should be separated to ensure that unnecessary noise is not induced by these traces being too close to each other and creating interference. Another essential component to the board layout is something called planes. For example a ground plane is the remainder of the board not being used by other traces and signals that contains all of the ground connections using as much copper as possible, creating a large heat sink like sheet. This will increase the power that the circuit can handle.

Eagle has an auto router feature that will be used as a starting point for our board design. The auto router will try to route all of the traces for us. But the auto router is not always able to do this. Sometimes it is because of the amount of parts or just the way the parts are configured on the board. Routing the board may take a lot of trial and error to get a decent board layout that will work the way it was intended to.

For the D*rink*W*izard* each subsystem will have its own circuit board. This will make each design a little bit simpler and will help in the ease of testing each circuit individually. Each board will contain only the parts necessary to complete the task of that subsystem excluding the power. The power supply will have its own circuit board and the power will be distributed from that board to each board with the correct voltages that are needed.

When designing the actual printed circuit board a lot of different factors need to be considered. Size of the components usually plays one of the biggest factors. It is impossible to create a working board that doesn't even fit the parts necessary for the board to work on it. Another factor is the power and heat created from the parts dissipating any power. Resistors can create a lot of heat and burn up traces that may run under the resistor, so rerouting traces should be considered when they are close to heat producing parts. The last factor is the amount of layers that the board will have. Like mentioned before, our boards will have a maximum of two layers and the fewest amount of via connections as possible. A via is a plated through a hole connection between different layers of a printed circuit board. A circuit board for the D*rink*W*izard*'s pump control that was designed using Eagle can be seen in *figure 7.3-1*.

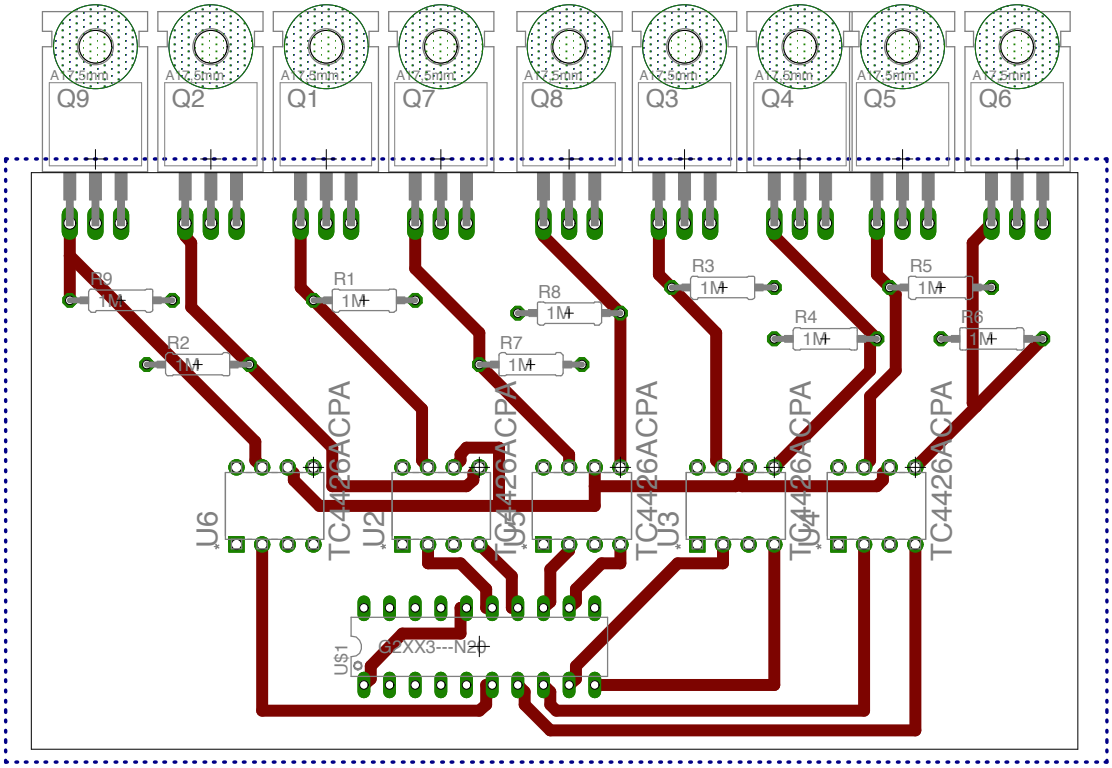

Figure 7.3-1: PCB Board for Pump Controller

All parts will be soldered to the board using a rosin core solder, containing 60% tin and 40% lead. The soldering will be done at the house of the group member John Brushwood. All soldering will be done to industry standards and will be double checked by another group member to ensure proper soldering techniques and guarantee that all parts were assembled properly. Also all ICs will be held to the board using a dip socket and not directly soldered to the board. This is done to safeguard against possible damage to the chip from the heat of the soldering iron.

7.4 Liquid Containers

The containers that house the liquid are also an important part of the D*rink*W*izard.* The liquid containers will hold all of the contents to make the end user a beverage of their desire. The containers need to be large enough to hold an ample amount of liquid. They also need to be small enough to minimalize the overall size of the D*rink*W*izard.* After all, we do not want the D*rink*W*izard* to be an overwhelming size.

The containers that will be housing the mixer solutions need to be larger than the containers for the liquor. They need to be larger because in any give drink there contains more mixers in the beverage than there is liquor. The container for the

mixer liquid will hold a minimum of one gallon. This will be enough to produce several different drink concoctions.

The liquor contains can be smaller than the mixer containers because the demand for them is less than that of the mixer fluid. The liquor will be held in five 50-ounce plastic containers. The containers are BPA free and safe for storing the liquors in. Since liquor bottles vary vastly in size, a uniform bottle will be used.

7.5 Final Plan

The D*rink*W*izard* had a sizeable amount of options for microcontrollers, as well as materials to house the vending unit, and even PCB board construction techniques. After reviewing all the possibilities for the D*rink*W*izard,* the group members narrowed the selections down to what we feel is an absolute winner.

After a lot of debate and discussions the group narrowed the material, for the construction of the vending unit, down to Plexiglas. The galvanized steel was a very viable solution, however the group was concerned about the look and weight of the galvanized steel. The group's goal is to build a product that the customer will want to use and proudly show off to their friends and family. The group also wanted to design the D*rink*W*izard* with a commercial application in mind, having a more ascetically pleasing design would meet this goal. The galvanized steel could have easily been painted or vinyl wrapped with any design we could think of, however the group felt this was a basic design and did not inspire bar owners to invest in our product. The same basic look and feel applied to the plywood option as well. The group also felt that over time the plywood would deteriorate and not be a product bar owners would want in their establishment. Lexan is exactly the ascetically pleasing material we are looking for, however the cost of Lexan is currently out of the scope of the group member's budget. The group member's decided that the Plexiglas was a great, cost-effective alternative to Lexan.

The projected design for the D*rink*W*izard's* vending unit is to have a box that contains six sides made of Plexiglas. The front Plexiglas sheet will be transparent, so as to allow customers to see the internal operation of the vending unit. This will also allow of visual inspection of fluid levels. The other 5 sides will be translucent and will be sandblasted to leave only the D*rink*W*izard's* logo to be translucent. This is important because the Plexiglas sides will be fitted with LED's that will illuminate the D*rink*W*izard's* logo. The LED's will bring an eye-catching factor to the D*rink*W*izard* and draw in more customers to come check it out. The back translucent panel, the one that is opposite the transparent panel, will have a hinge system installed. This hinge system will allow the panel to be opened and access to the fluid containers as well as the electronics of the vending unit. Since

access to the fluid containers will be through this access door, a locking system will also be in place as to prevent unwanted access to the fluids; we do not want minors getting ahold of the alcohol.

For the PCB board assembly the choice was really of no contest. One of the group members has access to a LPKF, Laser and Electronics AG, milling machine. The company manufactures PCB boards on a regular basis and after this manufacturing is done, a decent amount of blank boards is left over. Permission has already been obtained from the company that these blank waste boards can be used for our project. Permission has also been obtained for the group members to accompany and learn how to use the LPKF machine to cut the PCB boards that will be used for this project. Since PCB board manufacturing is a big cost for most projects this was a huge advantage to the group. This cost savings is a great asset and it made sense to take advantage of the help we could receive.

8.0 Testing

Every aspect of the D*rink*W*izard* will be testing fully under multiple circumstances and by different designers. The code for Android application will be tested throughout development to attain the best possible results. Testing the code as it is written will ensure not only that it works properly but will also help to maintain modularity within it and make maintenance and modification easier in case it ever needs to be updated.

The logic in the microcontroller units will be tested for accuracy. It is critical that the correct drink is made and that each one is made correctly. Ideally, the D*rink*W*izard* should be able to make drinks extremely similar in taste to how they would be crafted by a professional bartender. Too much or too little of one element could make a drink taste bad. Adding too much alcohol to a beverage may even be dangerous in certain situations.

Continuing with safety, hardware must be tested completely to make sure that not only the appropriate amount of liquid is being dispensed but also that liquid is kept separate from all electrical components. Should liquid be allowed to come into contact with the electrical components, the user could experience electrical shock, the mixer may stop working properly, and/or a fire could also be started. For these reasons, it is critical that all hardware is tested under not only standard use but also more extreme cases.

Integration testing must be done once all parts are deemed to be working precisely and accurately. At this point, the build will be nearing completion. These tests will be used to make sure the D*rink*W*izard* works correctly as a whole and under standard use conditions. Final tests will be conducted with a fully assembled drink mixer and begin with a clean install of the Android application to simulate a running a brand new product.

To test the final product, the D*rink*W*izard* will be set up as it would be in its final environment. Multiple people who would be expected to be average users will then be allowed to order drinks and perform other standard operations for D*rink*W*izard*. Should the Android component and the physical mixer perform properly, testing will be concluded and the D*rink*W*izard* will be considered finished in its current build state with only minimal cosmetic changes. Finally, it will be prepared for delivery to its final location.

8.1 Software Testing

The software will be thoroughly tested for all possible cases. This will ensure that each drink can be ordered and made properly. The Android application will be

responsible for sending drink orders to the drink mixer. Testing will include, at a minimum, ordering each drink and tasting it to ensure that it has been properly crafted. The order information for a drink comes from the Android application; therefore it will send information to a file where it will be analyzed for accuracy by the developers. The order must be made properly as none of the developers would expect users to consume something they did not order or something they cannot consume for medical reasons. After testing with files, the data is sent via a radio frequency communication channel to the PuTTY terminal. Once these early tests are concluded, testing will continue with different machines.

The microcontroller unit must also be tested. It will be responsible for receiving data from the Android application and parsing it to determine which type of drink should be made and what should go in it. Data will be sent to the device during testing and output will be monitored to make sure the device is not only programmed properly by the developers but also that the device is working properly. Specific and random cases will be chosen and tested for correctness. Output must be correct in all cases to maintain the safety of the users.

8.1-A: Drink Wizard Android Application Testing

The first component of the Android application tested is Bluetooth connectivity. The application should be able to scan for other discoverable Bluetooth devices. A list should be shown to the user and allow he or she to successfully connect to the device. This will be tested with multiple laptop computers, other Android devices, and the Bluetooth module from the D*rink*W*izard* drink mixer to ensure that it has been coded properly for more than one case. Initial testing was performed with PuTTY and simple data was sent to and from the D*rink*W*izard* Android application's device.
Secondly, a list of drinks will be entered into the Android application. It will keep a record of how of much of each liquid is required to make a proper beverage. This information will be sent via Bluetooth to a text file and analyzed by the developers. All beverages will be tested and checked for accuracy before the conclusion of proper functionality has been reached. The ability to make custom drinks will also be included in the Android application. A minimum of twenty custom drinks will be created, named, and stored correctly before this functionality is deemed to be fully operational. Initial testing was done with PuTTY and involved the creation of a drink, saving the drink, then ordering it and sending the specified ingredients to the PuTTY terminal. Final testing will consist of the creation of drinks as well. These will be mixed with the D*rink*W*izard* mixer and served. If the taste is reasonable, the feature is successful and will not be redone.

The Android application will be responsible for alerting the user when a drink is complete and ready for pickup. It will include signals to show the current stage of

a drink. This feature will be tested by sending signals from the Bluetooth module to the application that will mimic the ones sent when production is complete. The application will also be able to accept a signal, which tells when no cup is present. This will stop a drink from being created until the mixer detects a cup. Upon completion of the D*rink*W*izard*, this feature will be inherently tested while making drinks, as this process will happen whenever a drink is ordered. These tests should sufficiently prove that the progression system is working correctly and will be verified visually by developers.

A major aspect of the Android application is usability. It should have a clean user interface and be easy to use to the average user, which would be someone twenty-one years of age or older. It will later include graphics and summary of each drink, which will tell what it consists of. Easy to use also means making it clear to the user how to order. It should take a new user no longer than thirty seconds to figure out how to order a drink and input a selection. After the developers ensure proper functionality, field-testing will begin. The Android application will be shown to multiple typical users and they will be asked to go through the process of ordering a drink.

The following screenshot shows the first prototype used for testing Main Activity. It contains buttons for turning Bluetooth on and off, searching for devices, allowing the device to be discovered by other Bluetooth-enable devices, and pairing with devices. Bonded devices can also be connected to. There is also the option to create a drink. This activity was used to debug the application before finalizing Main Activity's user interface. *Figure 8.1-A-1* shown below.

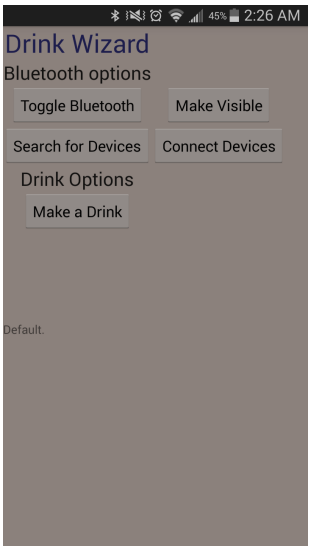

Figure 8.1-A-1: Main Activity First Prototype

Software testing will occur multiple times throughout the development process. All code will also be tested at the time it is written to make the development and

testing as efficient as possible. These tests will look for not only the correct output and functionality but also usability. While correctness is the primary target, should developers find a process to be frustrating or unnecessary during testing, it will be altered to make it user-friendlier. Examples include but are not limited to long wait times for processing, difficulty accessing a feature, and unappealing design. The following *diagram 8.1-A-2* outlines two test cases that are used to test the functionality of the Android application. These are standard processes users may follow when using the D*rink*W*izard* and were chosen for this reason.

In the *figure 8.1-A-2* below, the case on the left side starts with a user starting the application and enabling Bluetooth. They then scan for a device to find the mixer and select it from the device list once it is found. A drink is now ready to be created so they make a drink selection. If they decide to select a different drink, they will be taken back to the drink selection screen; however, if they confirm the selection, the data for the drink is sent to the mixer and after it is finished, it will be ready to be picked up by the user within a few minutes. For the case on the right side, the application is already running and the user decides to create a new drink. After they create the drink, they can either confirm or deny it. They are then returned to Order Activity where they will be able to make a drink selection. This decision requires a confirmation also.

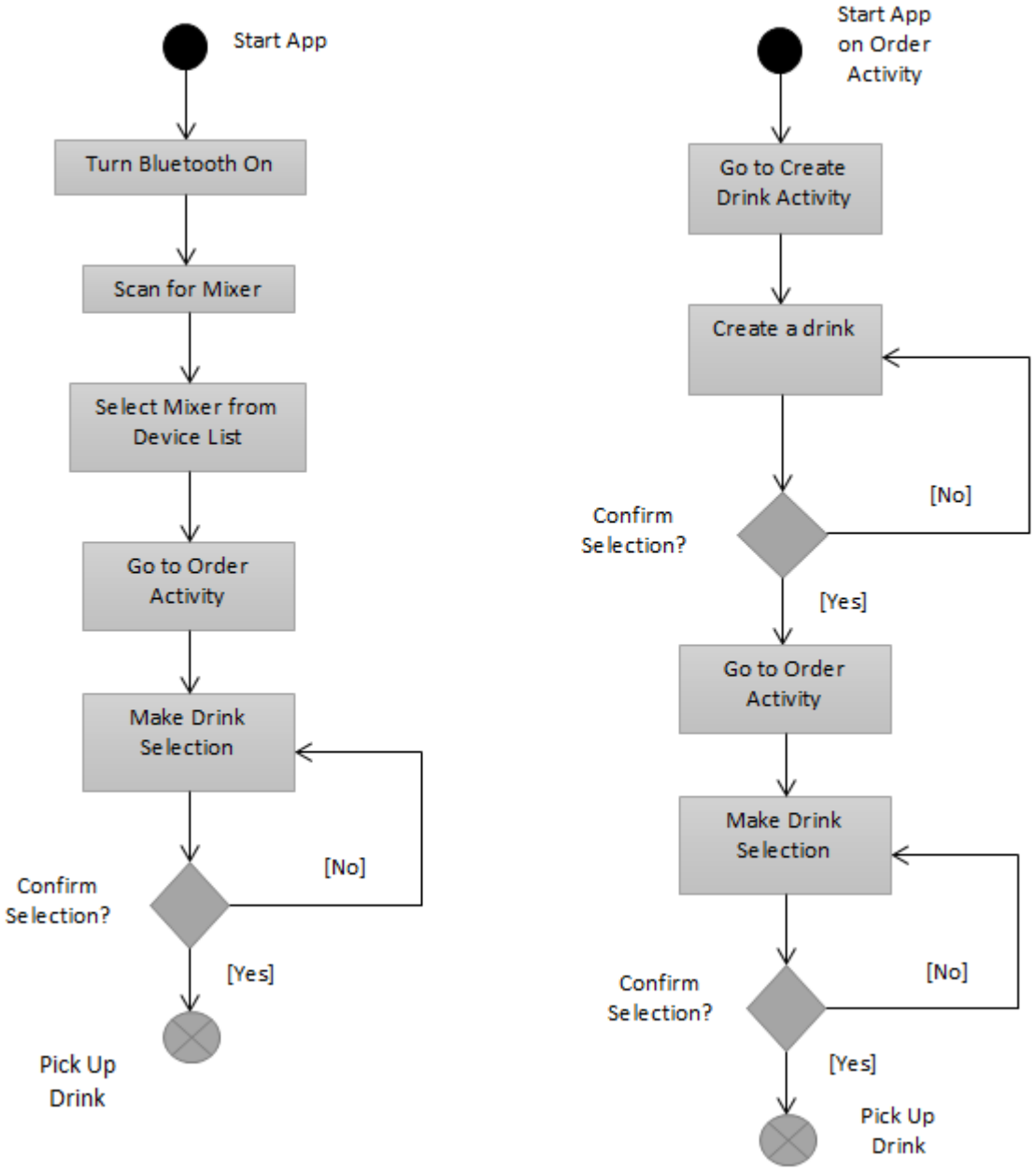

Figure 8.1-A-2: User Test Cases

8.1-B: Microcontroller Unit Code Testing

The microcontroller will be responsible for receiving input from the Android and then interpreting it to create the beverage ordered by the user. The responsibility of transmitting a signal back to the Android application will also fall upon the microcontroller unit. These signals will be used to inform the user when their

drink is being prepared and when it is ready to be picked up. It should also be able to tell the user when there is no cup present. Testing will occur at every stage and constantly throughout the coding process. The output of the received data will be checked and verified for correctness by connecting the output lines of the microcontroller unit to test devices like the ones they will be connected to when the D*rink*W*izard* is finished.

8.2 Hardware Testing

For any system to work properly, it must be tested. Testing is essential to find any problems that could arise. These problems may cause the system to malfunction or at worst case fail. To guarantee that this does not happen, the system must be tested. There are two systems that need to be tested. The hardware and the software are the two systems that need to be tested.

The hardware is not easy to test. Once this system relies on both the hardware and the software to function, the hardware alone will be hard to test. The hardware consists of all the components that are not part of the software and coding side of the D*rink*W*izard.* This will consist of the pumps and the pump controller. It will also consist of the cup and liquid level sensors.

Each pump and its controller will be tested prior to the systems final setup. To test the pumps, a twelve-volt source will be connected to the pump. The pump will have the proper tubing attached to it. Once the twelve-volt source is turned on, the pump should start to dispense a liquid. For testing that liquid will be water. If the pump works it will continue to the next stage where the controller will be tested.

The controller consists of a MOSFET, a MOSFET driver and a high impedance resistor. The resistor is there to dissipate any remaining voltage at the gate after a positive on signal has been changed to a zero off signal. This resistor aids in the fast switching between on and off. To test the controller all of the pins of the MSP430 will be set to high for ten seconds. After the ten seconds of on time the MSP430 will then turn off for ten seconds. This will repeat until the system is turned off. During the cycles of on and off, the system will be checked to ensure that all the pumps, which are already known to be working, are engaging and then disengaging. If any of the pumps do not work then there is a problem with that controller. That controller will then need to be debugged and rechecked for proper working condition.

If the pumps and the pump controller both work then the cup sensors need to be checked. The cup sensor should be able to detect an object inside the dispensing parameters. A simple code will be loaded into the MSP430 Launchpad. This code will display the data collected from each individual cup

sensor on the screen of the computer that the Launchpad is connected to. We will check each sensor by placing a piece of paper that is a known distance from the sensor and is within the range of that sensor. The data will be checked to ensure that the sensor is working correctly. After all three sensors are check and working properly then the liquid level sensor can be checked.

The liquid level sensors also need to be tested to confirm that all of the systems work. The liquid sensors detect that amount of fluid that is left. To check these sensors a simple code will be loaded into the MSP430 Launchpad again. The data collected from the liquid level sensors will be shown onto the computer screen of the computer that is connected to the Launchpad. Each sensor will be tested using the same bottles. One of these bottles will be empty and the other will be filled with water to the full amount of 50 ounces. If all the values match up to a certain degree of accuracy, then the liquid level sensors check out and the individual systems all check out.

If all of the systems are in good working condition they must be tested together. To test the system as a whole, another test program will be loaded onto the MSP430. The program will cycle each pump for one-ounce worth of time. Then the next pump will be cycled for the amount of one ounce of time. This will continue for the entire nine pumps. The system will display the data of the cup sensors and the liquid level sensors on a computer screen. The entire process can be seen in *figure 8.2-1*.

Figure 8.2-1: Hardware Testing Procedure

8.3 Hardware and Software Integration Testing

The testing for proper hardware and software integration in all of our subsystems will include a systematic test plan in an attempt to test proper functionality. The test plan will include a complete procedural step-by-step method in order to find any nuances or communication faults between the drink ordering platform and the main unit housing.

Step 1: Make sure all hardware devices are properly powered

In order to check for proper power to all devices, a digital multi-meter should be used. Starting from the AC outlet from the wall, we need to verify that the power supply first and foremost have all of the correct VCC voltages on the corresponding pins on every integrated circuit package that exists on all of our PCB's. The first thing to be tested will be the power coming into the control box from the wall outlet. At out circuit breaker we should be seeing 120 VAC. After the circuit breaker, the power will run into an on/off switch. We will verify that power is flowing into and out of this switch with it turned on. From the switch the power will run into a 12 volt, 15 amp, step down transformer. The power out of the transformer will be verified to ensure it is 12 volts AC. From the transformer the power will run through a full-wave bridge rectifier. After the rectifier the power will be measured to ensure we are receiving 12 volts DC. After the rectifier is a filtering stage using capacitors. An oscilloscope will be used to ensure all the ripples have been filtered out before voltage is sent to the various devices.

Step 2: Make sure all peripheral devices are properly powered

These peripheral devices include the Ultrasonic sensors, Bluetooth module, and peristaltic pumping units, load cells. Checking that all of the peripheral devices being properly powered are critical to ensuring that the software can properly communicate with the hardware. The first peripheral to be checked will be the liquid pumps; these pumps will be fed 12-volts DC straight from the capacitor filtering stage. The multimeter will measure each motor to ensure we are getting power to each individual motor. The 12-volts DC is also fed to the 5-volt board. This will be metered to ensure we are getting a constant 5-volts out. This board then powers the ultrasonic sensors, microswitches, LCD screen, and MOSFET drivers. These components will also be metered to ensure they are getting the required 5-volts. The 5-volt board will then power the 3.3-volt board. We will meter to ensure we are getting 3.3-volts out and this will then send power to the Bluetooth module and the three MCU's. Each one of these components will be metered as well to ensure they are receiving the correct 3.3-volts of power. The HC-06 Bluetooth module is probably most critical in this element as it will provide the wireless like between the Android ordering platform and the MSP430

microcontroller pumping subsystem. The Breathalyzer subsystem will be plugged in and powered from the USB port on the Android tablet.

Step 3: Make sure that the Android tablet and the HC-06 Bluetooth module can be paired

The first and most important step in testing the software and hardware integration is attempting to search for the Bluetooth module ID being sent from the D*rink*W*izard* main housing. In order to test this step properly, a connected status should be seen on the tablets Bluetooth settings screen. To ensure the Bluetooth module is working properly a phone can be connected to the Bluetooth module to establish a connection. Once the connection is established the Bluetooth module, located in the black control box, will go from a blinking LED indicator to a solid red LED indicator. This allows us to visually inspect that we have established a Bluetooth connection.

Step 4: Check the proper functionality of the Breathalyzer subsystem

The task of the Breathalyzer subsystem is to plug into the tablets USB port and only allow users to access the drink-ordering menu when the alcohol sensor has read a value that is lower than a specified level. In order to test this we need to plug the Breathalyzer into the USB port and blow an air sample over the Breathalyzer that should cause it to read a positive failure test for alcohol. Once the positive test has been recorded, we need to verify that the failed test has successfully locked the user from the drink menu system.

Step 5: Send a drink via the Android tablet

Once we have determined that the user has successfully passed the alcohol test, the next step is to order a drink from the menu and verify if the main unit has successfully received the proper data through the Bluetooth link. In order to test this procedure all other peripherals on the main unit must be functioning and satisfied. These parameters include, having an empty cup in the pouring area and proper levels of liquids are in each container.

Step 6: Order another consecutive drink

This step is extremely important to verify that the D*rink*W*izard* can successfully complete more than one cycle of drink ordering. This step will test if there is any possible infinite loops in all portions of our coding that could potentially cause the D*rink*W*izard* to hang up and stop responding to commands.

A step-by-step flowchart to help demonstrate the logic behind the hardware and software testing can be seen in *Figure 8.3*.

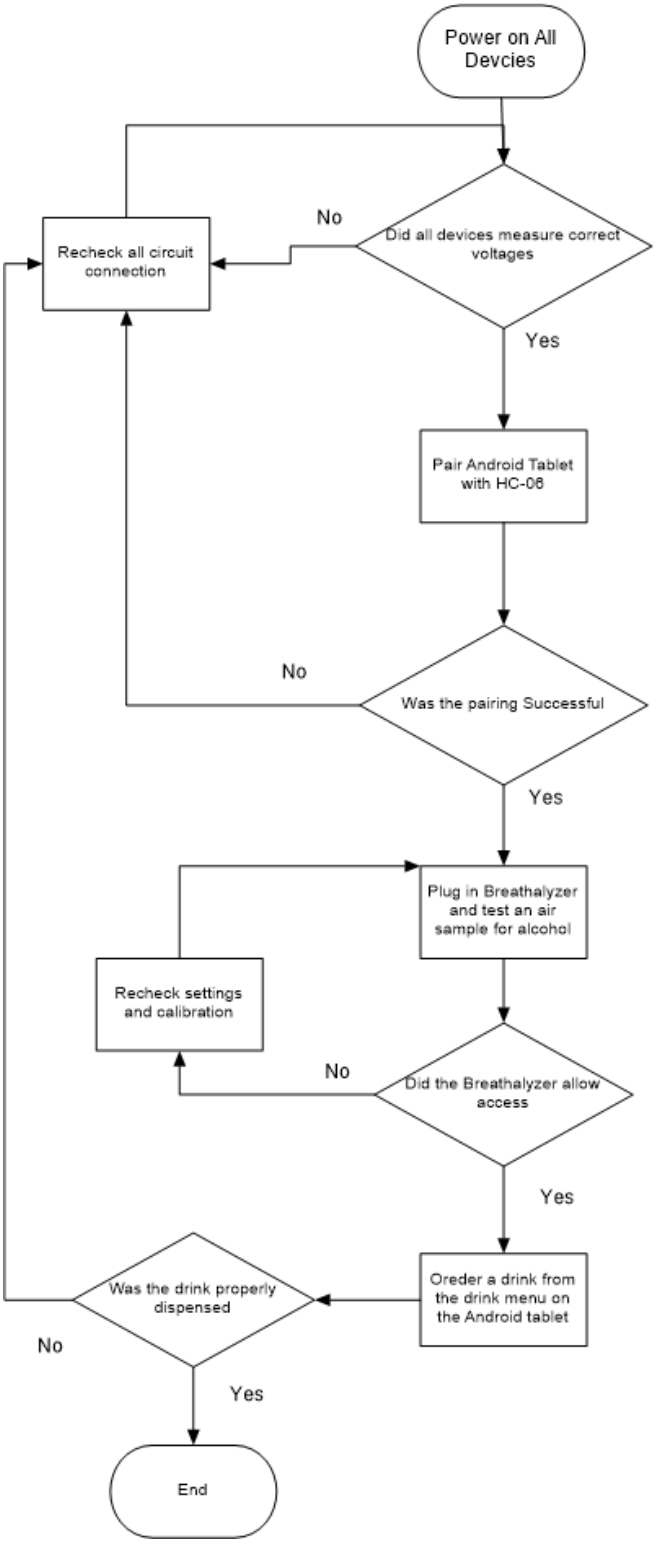

Figure 8.3: Software Testing Flowchart

9.0 Administrative Content

9.1 Task and Responsibilities

This project has the requirement of being completed in two semesters, one of the semesters being a summer semester. The summer semester is shorter and this creates stringent deadlines. Our team decided to place a more stringent deadline of having our prototype, version 1.0, built by June 30, 2015. Having the date of June 30, 2015 allows our group to evaluate version 1.0 and possibly allow time to implement some features the group wanted in version 2.0. The project needs to stay on track for our group to accomplish our group goal of graduating, so we created a table to divide the task among the team members. Dividing the task allows every team member to share in the responsibilities and ensures our group can meet our stringent deadline. Below is *figure 9.1*, a block diagram, showing how the tasks are divided among the group members.

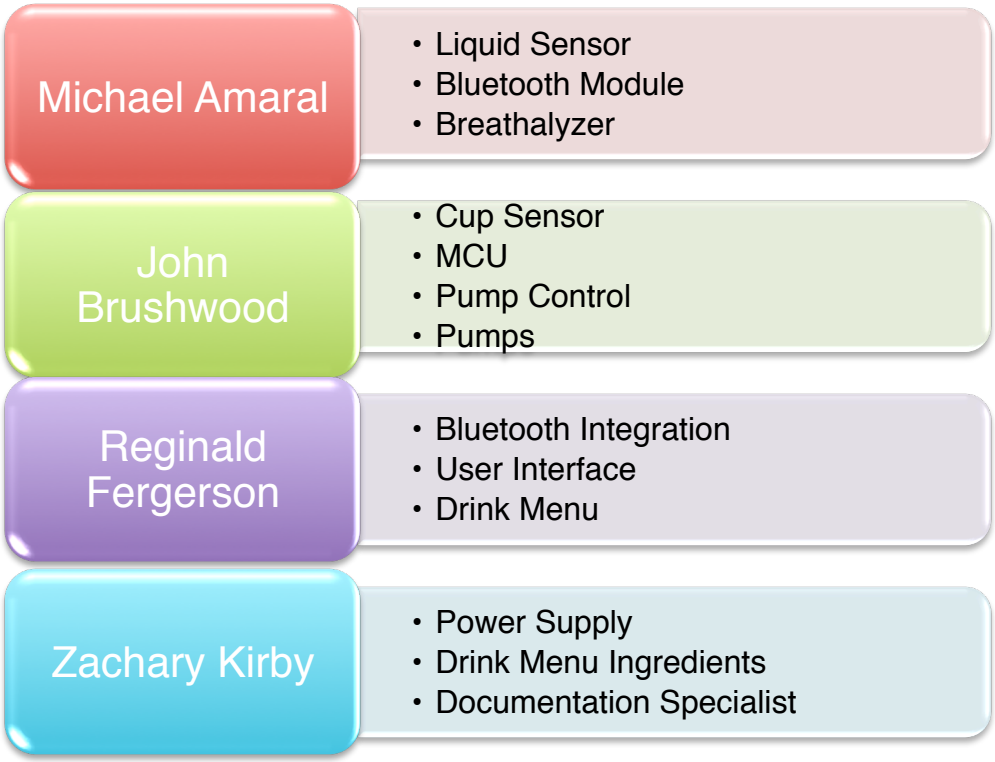

Figure 9.1: Block Diagram

9.2 Milestone Collaboration

Monitoring the project's progress is of the upmost importance, due to the importance of the project's progress, miscellaneous task and objectives will be assigned along with their respective due dates. Having a milestones table will allow this project to be completed in a felicitous fashion over the spring and summer semesters. Each group member will reference the milestones table to institute an individual plan or reference point to gauge their individual progress on completing the major group achievements for the development of the D*rink*W*izard*. The D*rink*W*izard*'s milestones will be composed of related research, writing Senior Design 1 project documentation, procurement of copyright permissions, group acquisition of the imperative project parts, the construction phase of the project, version 1.0 testing, the final presentation and the final demonstration of the finalized product. The amount of research required for the project will be left up to the discretion and preference of the individual group members, however the assumed about of required research will be distributed in a way that will guarantee that each group member is accountable for equivalent amounts of work.

The group members will complete their individual research, at the completion of the research phase the group will then move to the documentation phase. The individual research phase will be completed by April 9, 2015; at that point the documentation phase as well as the procurement of copyright phase will begin. The documentation for Senior Design 1 is due April 30, 2015; this will push the deadline for the documentation and copyright phase to April 25 2015. Having an earlier deadline will allow the group members to combine their individual research and individual documentation to create one main group project document. Once all the documents are submitted the group will move into the acquisition of parts phase, this is where all the parts that were reviewed, discussed and finally chosen, in the Senior Design 1 documentation, will be ordered. While parts are being acquired construction phase can begin at the same time. The building of the main vending unit box as well as the platform can begin right away once we reach the construction phase. The construction phase will be completed by June 30, 2015; this will allow the group to move into the testing phase. Once in the testing phase the final documentation as well as final presentation can begin. The testing phase will be completed once the product meets the group member's minimum requirements. Time permitting some version 2.0 additions may be added at this time before the final presentation. The final presentation will need to be completed by August 1, 2015. These will complete the major milestones for the group project. Displayed below is *Table 9.2: Milestone Table* for the D*rink*W*izard* project.

Date	Milestones
Feb - 01	**Project Overview** • Define project • Objectives for the project • Goals for the project • Individual areas of interest • Divide and conquer documentation
April - 09	**Research** • Previous Projects Research • Microcontrollers • Fluid Pumps • Power Requirements • Power Converter • Communication Protocols • PCB Software • LCD Screens **Research** • Fluid Storage • Pressure Sensors • Bluetooth Modules **Embedded Software** • Requirements • All classes, functions, and methods • Class Diagrams • Bluetooth Connections **Hardware** • PCB construction facilities • Bluetooth interface requirements • Breathalyzer requirements **Power Supply** • Rechargeable Batteries • Voltage Regulators • Buck Converters • Transformers • Rectifiers • Filters

Date	Milestones
April – 25	**Writing Project Documentation** • Table of Contents • Executive Summary • Technical Content • Administrative Content • Project Summary • Project Conclusions • Appendices **Procurement of Copyright Permissions** • Data–sheets Permissions • Diagram Permissions
June – 30	**Construction** • Design of Main Vending Unit • Final Hardware Schematics • Final Software Code • Chosen Controller Board
June - 30	• Chosen Controller Chip • Chosen Rechargeable Battery • Chosen Pressure Sensor • Chosen Bluetooth Module • Chosen LCD Screen • Chosen Liquid Pumps • Chosen Liquid Containers
July - 28	**Testing** • Test Phone Application • Test Charging System • Test Fluid Pumps • Test Quality of Dispensed Beverage • Implement Any Upgrades • Finalize Code • Update All Code • Final Test and Lock Out Mode Till Demo
Aug - 01	**Final Presentation** • Practice Final Presentation • Final Documentation • Final Demonstration • Update Senior Design I Document • Exit Interview

Table 9.2: Milestones Table

9.3 Budget and Finance Collaboration

The D*rink*W*izard* project makes an exceptional Senior Design 1 project due to the fact that it is comprised of both Electrical as well as Computer Engineering aspects. The great thing about the Computer Engineering side is that the programming of the software cost nothing to make. Since the software side of this project is free, the real deliberation of funding comes down to the hardware components and the cost of materials to build the D*rink*W*izard's* main vending unit box. The hardware cost will be comprised of the microcontrollers cost, fluid pumps, fluid containers, PCB construction, and power supply components. The group members themselves will fund the D*rink*W*izard*. The group did not want to give up the rights to the final constructed vending unit so we chose not to seek funding. Since the funding will come out of the individual group members pockets it is imperative to keep a detailed itemized list so the cost can be split equally among the members. The itemized list of parts and cost can be seen in *Table 9.3*.

Item	Price	Quantity	Projected Cost	Actual Cost
Hardware				
PCB Board	$60.00	5	$300.00	Free
Processor	$35.00	3	$105.00	Free
Bluetooth Module	$7.08	1	$7.08	$7.08
Bluetooth				
Power Supply				
Lead Acid Battery (12 V)	$45.00	3	$135.00	Free
Transformer	$28.67	1	$28.67	$28.67
Breaker (6 amp)	$6.95	1	$6.95	$6.95
Step-down Regulators	$2.73	2	$5.46	Free
C,R,L	$35	1	$35.00	$35.00
Vending Unit				
LCD Screen	$22.70	1	$22.70	$22.70
Liquor Pumps	$12.49	5	$62.45	$62.45
Mixer Pumps	$35.05	4	$140.20	$140.20
MOSFET Drivers	$23.01	1	$23.01	$23.01
Tubing 3/32	$29.88	1	$29.88	$29.88
Tubing	$11.75	1	$11.75	$11.75
Miscellanious Costs				
Skycraft	$75.61	1	$75.61	$75.61
Lowe's	$80.44	1	$80.44	$80.44
		Totals:	$1,069.20	$523.74

Table 9.3: Itemized Budget Table

10.0 D*rink*W*izard's* Operational Manual

10.1 Initial Setup

The D*rink*W*izard* is an automated bartender that can serve several different drink concoctions. The D*rink*W*izard* needs to have the liquids that it will be pumping prefilled. There are five different liquors and four different mixer fluids. These liquids all need to be filled in the appropriate containers located on the top drawer of the D*rink*W*izard.* After these liquids have been filled the pumps need to be primed, in order to ensure that there is no air within the dispensing tubes. The primer buttons are located in the black control box at the back of the D*rink*W*izard.* Each button should be held down until the liquid reaches the dispensing area.

To initialize the DrinkWizard the 120 VAC plug must be located and plugged into a 120 VAC wall outlet. Located on the back of the DrinkWizard's vending unit is a black control box. Located at the bottom of the black control box is a three prong 120 VAC plug, this plug needs to be plugged into a wall outlet. Once the plug is plugged in, on the bottom of the black control box is a 6 amp 120-volt circuit breaker. Press the button on the circuit breaker to verify that it did not trip during the last use of the DrinkWizard or on the initial connection to the wall outlet. Once these steps have been performed, the red power button located on the top of the black control box can be pressed. If the previous steps were followed in order, the red power button will illuminate indicating that the D*rink*W*izard's* vending unit has been powered up correctly.

After the D*rink*W*izard's* vending unit has been powered up, the D*rink*W*izard's* mobile application will need to be downloaded. The application is available to android users that have the minimum android software version 4.2. The application is free and the application would be located in the Google Play Store. Once the application is downloaded from the Google Play Store, open the application so initial connection to the D*rink*W*izard's* vending unit may be performed. To initiate the connection please have the D*rink*W*izard's* mobile application open.

With the mobile application open, a connection can be made between the android device and the D*rink*W*izard.* The connection uses bluetooth communication to talk between the two devices. The first step is to turn on the bluetooth capabilities on the Android device. This can be done on the main screen of the D*rink*W*izard* application by using the toggle bluetooth button. The application will toggle the bluetooth of the Android device on and off by using the button shown in Figure 10.1.

Figure 10.1 toggle Bluetooth

After the Bluetooth has been turned on, a secure connection can be made with the D*rink*Wi*zard.* If this is the first time a connection is to be established, then additional steps must be made to initialize the communication. Using the D*rink*Wi*zard* application we can connect to the D*rink*Wi*zard* itself. If this is the first time pairing with the D*rink*Wi*zard*, the device must be found. To do so simply click search for device and all available Bluetooth devices in the range of your Android device will be found. The D*rink*Wi*zard* will automatically be in a mode in which it can be discoverable. Then press the connect/pair button and type in the key code: 1234. If a connection had already been established, simply press on the connect devices button and the D*rink*Wi*zard's* Bluetooth identification should be seen on the screen. Simply press the connect/pair button and follow the onscreen instruction to properly pair with the device. The Bluetooth connection has now been established and drinks can now be ordered.

Drinks can now be ordered from the D*rink*Wi*zard* app. To order a drink, one must go to the make a drink menu. Once at this menu, a predetermined drink list can now be seen. To get a brief description of the drink, just tap the drink name and a short description of that particular drink will show up on a pop up menu. To order the drink, select the make drink button. If that drink is not up to your liking

or if you simply just wish to choose a different drink, click the back button and you will return to the drink order menu.

Once a drink has been ordered, the D*rink*Wi*zard* will start to make the drink. A cup must be present in the dispensing area in order for the proper dispensing of the beverage. Also all bottles must contain the appropriate level of fluid for dispensing to commence. After the appropriate level of fluid has been dispensed all pumps will shut off and the drink can be removed from the dispensing area. Enjoy!

After all drinks have been made and the D*rink*Wi*zard* is to be shut down, remove all of the bottles and drain the tubing of all the remaining liquids. This is to ensure that the system does not become contaminated. Simply power off the system, using the red power switch, if the D*rink*Wi*zard* is not going to be used for a long time the D*rink*Wi*zard* should be stored and the plug unplugged from the wall outlet.

10.2 Troubleshooting

Problem	Solution
Unit not powering on	• Make sure the unit is plugged into a working outlet that supplies the unit with 110 – 120 volts AC. • Ensure that the onboard circuit breaker has not been switched, if it has been, then depress the circuit breaker button. • Make sure the power button has been pressed and the power button is lit up.
Bluetooth not connecting	• Ensure a different Bluetooth device has not already been connected to it. • Only use the D*rink*Wi*zard* app to connect to the D*rink*Wi*zard* to ensure proper functionality.
Drink not being dispensed after a drink has been ordered using the official D*rink*Wi*zard* app.	• Check the liquid level of each bottle and confirm that all bottles are filled and all the bottle indicator lights are off. • Guarantee that a cup has properly been placed into the predestined dispensing area. • Ensure that the unit is powered up and that a satisfactory Bluetooth connection has been achieved.
Drink not dispensing, but pumps are working	• Check the integrity of the tubing inside pumps and certify that there is not a rupture in the tubing. If a rupture occurs replace the tubing and prime the pumps. • Confirm that the tubing is placed correctly at the bottom of the liquid containers and that they are drawing liquid into the tubes.

Table 10.2: Troubleshooting Guide

Appendix A: Works Cited

Citations

[1] *Generic Object Exchange Profile*, 21st ed. Bluetooth SIG, 2012.

[2] *File Transfer Profile*, 13th ed. Bluetooth SIG, 2012.

[3] *Link Loss Service*, 10th ed. Bluetooth SIG, 2011.

[4] *Serial Port Profile*, 12th ed. Bluetooth SIG, 2012.

[5] I. Poole, 'IEEE 802.11b :: Radio-Electronics.Com', *Radio-electronics.com*. [Online]. Available: http://www.radio-electronics.com/info/wireless/wi-fi/ieee-802-11b.php. [Accessed: 03- Apr- 2015].

[6] I. Poole, 'IEEE 802.11g | Wi-Fi WLAN | Tutorial - Radio-Electronics.Com', *Radio-electronics.com*. [Online]. Available: http://www.radio-electronics.com/info/wireless/wi-fi/ieee-802-11g.php. [Accessed: 03- Apr- 2015].

[7] E. Vogler, 'Bluetooth vs. Wi-Fi Power Consumption', *Science - Opposing Views*. [Online]. Available: http://science.opposingviews.com/bluetooth-vs-wifi-power-consumption-17630.html. [Accessed: 29- Mar- 2015].

[8] Wi-fi.org, 'Wi-Fi Direct | Wi-Fi Alliance', 2015. [Online]. Available: http://www.wi-fi.org/discover-wi-fi/wi-fi-direct. [Accessed: 22- Mar- 2015].

[9] L. Cano, M. Dominguez, M. Tyrlik and S. Zimmerman, *Under the Sun Drink Mixer*, 1st ed. Orlando: University of Central Florida, 2013.

[10] J. Price, 'Build A Mobile Bar - BaR2D2', *Instructables.com*. [Online]. Available: http://www.instructables.com/id/Build-A-Mobile-Bar-BaR2D2/. [Accessed: 25- Mar- 2015].

[11] Smart Bar USA, 'smartender', 2015. [Online]. Available: http://smartbarusa.com/. [Accessed: 25- Mar- 2015].

[12] Somabarkickstarter.com, 'Somabar | Robotic Bartender for your Home', 2014. [Online]. Available: http://www.somabarkickstarter.com/. [Accessed: 25- Mar- 2015].

[13] J. Osborne and I. Cooper, 'The Inebriator | Arduino powered cocktail machine', *Theinebriator.com*, 2012. [Online]. Available: http://www.theinebriator.com/. [Accessed: 25- Mar- 2015].

[14] Osenon-group.com,. 'Ultrasonic Liquid Level Measurement-Ultrasonic Sensor,Ultrasonic Transducer Manufacturer - OSENON'. N.p., 2015. Web. 28 Apr. 2015.

[15] Baumer.com,. 'The Four Sensor Types Of Ultrasonic Sensors | Baumer Group'. N.p., 2015. Web. 28 Apr. 2015.

[16] Welco.net,. 'Peristaltic Pump & Dispenser : WELCO'. N.p., 2015. Web. 28 Apr. 2015.

[17] Matthews, Charli. 'Centrifugal Pumps - Empowering Pumps'. *Empowering Pumps*. N.p., 2015. Web. 28 Apr. 2015.

[18] Yourd, Shawn. 'Peristaltic Pump Wear Factors - Blue-White Industries'. *Blue-White Industries*. N.p., 2014. Web. 28 Apr. 2015.

[19] Ti.com,. 'MSP430G2553 | Msp430g2x/l2x | Ultra-Low Power | Description & Parametrics'. N.p., 2015. Web. 28 Apr. 2015.

[20] Microchip.com,. 'TC4426 - Power Management- Power MOSFET Drivers'. N.p., 2015. Web. 28 Apr. 2015.

[21] Randomnerdtutorials.com,. 'Complete Guide For Ultrasonic Sensor HC - SR04 | Random Nerd Tutorials'. N.p., 2013. Web. 28 Apr. 2015.

[22] Measurementsensors.honeywell.com,. 'LOAD CELL- BEAM STYLE | Honeywell'. N.p., 2015. Web. 28 Apr. 2015.

[23] Seeed Studio Bazaar,. 'Load Sensor'. N.p., 2015. Web. 28 Apr. 2015.

Appendix B: Permissions

Honeywell

Honeywell Copyright Notice
Honeywell International Inc. authorizes you to copy documents published by Honeywell International Inc. on the World Wide Web for personal or non-commercial use only, provided any copy of these documents that you make shall retain all copyright and other proprietary notices contained herein. Except, as expressly provided, nothing contained in this paragraph shall be construed as conferring any license or right under any Honeywell International Inc. copyright. No materials available on the Honeywell International Inc. Web site may be stored, transmitted by any means (including but not limited to electronic, mechanical, scanning, photocopying or recording) without prior written permission of Honeywell International Inc.

Microchip

Use of Microchip Copyrighted Material

Microchip documentation (including but not limited to data sheets, manuals, etc.), images, website, and other original creations are valuable assets protected by copyright law. Microchip aims to protect these assets while encouraging the broad dissemination of product literature and related information in order to reach the widest market for Microchip products. To that end, we often consider requests by customers, distributors, and other parties to reproduce, translate, and/or reprint our copyrighted material in a book, CD, magazine, or other reference material to be sold or distributed on the open market. When we approve such requests, Microchip still owns all rights to the copyrighted material including any translations of such material.

If you would like to reproduce, translate, and/or reprint Microchip copyrighted material for commercial purposes you must request Microchip's written permission by following the 3-step process described below. Microchip's written permission is not required for personal use or educational (non-profit) use of copyrighted material.

Personal Use of Copyrighted Material: If you use Microchip copyrighted material solely for your personal use you do not need Microchip's written permission to use such material. However, distribution or reproduction of such materials and images to others (including posting on a website) does require Microchip's written permission.

Educational and Non-Profit Use of Copyrighted Material: If you use Microchip copyrighted material solely for educational (non-profit) purposes falling under the "fair use" exception of the U.S. Copyright Act of 1976 then you do not need Microchip's written permission. For example, Microchip's permission is not required when using copyrighted material in: (1) an academic report, thesis, or dissertation; (2) classroom handouts or textbook; or (3) a presentation or article that is solely educational in nature (e.g., technical article published in a magazine). Please note that offering Microchip copyrighted material at a trade show or industry conference for the purpose of promoting product sales does require Microchip's permission.

Newhaven

You may use data and pictures from our website as long as you cite us as the source of information.

Sincerely,
Atif Khan I Engineering

Newhaven Display International, Inc.
www.newhavendisplay.com
2661 Galvin
Ct.
Elgin, IL 60124
Phone: 847-844-8795
Fax: 847-844-8796

Parallax

We allow use of our pictures, diagrams, schematics, text, etc. as long as it is unmodified. You can use excerpts from our texts, as long as you don't change the original wording.

Respectfully,
Chris Savage
Engineering Tech, Parallax Inc.

Saint-Gobain

COPYRIGHT NOTICE; USE RESTRICTIONS
The information presented on this Site, along with any documents, data files or other materials available for viewing or downloading (including, for example, press releases, product descriptions, customer guides or tips, and FAQs) is the copyrighted work of the Company and/or its suppliers and is protected under US

and worldwide copyright laws and treaty provisions. The Company grants you permission to copy any such information or material so long as each copy (i) is solely for informational use or for permitted commercial uses in support of the Company's products or business interests and is not modified or revised in any manner, (ii) plainly displays all copyright and other proprietary notices, in the same form and manner as on the original and (iii) displays a statement that the materials are used solely with permission of the Company. You also may not, without the Company's permission, "mirror" or "frame in" any material contained on this Site on any other server. Except as expressly granted in this section (or to you specifically in writing), the Company and its suppliers do not grant any express or implied right to you under any patents, copyrights, trademarks, or trade secret information.

Texas Instruments

Texas Instruments is pleased to provide the information on these pages of the World Wide Web. We encourage you to read and use this information in developing new products.

TI grants permission to download, print copies, store downloaded files on a computer and reference this information in your documents only for your personal and non-commercial use. But remember, TI retains its copyright in all of this information. This means that you may not further display, reproduce, or distribute this information without permission from Texas Instruments. This also means you may not, without our permission, "mirror" this information on your own server, or modify or re-use this information on another system.

TI further grants permission to non-profit, educational institutions (specifically K-12, universities and community colleges) to download, reproduce, display and distribute the information on these pages solely for use in the classroom. This permission is conditioned on not modifying the information, retaining all copyright notices and including on all reproduced information the following credit line: "Courtesy of Texas Instruments". Please send us a note describing your use of this information under the permission granted in this paragraph. Send the note and describe the use according to the request for permission explained below.

Appendix C: Datasheets

C.1 Atmel

Atmega16

http://www.atmel.com/Images/doc2466.pdf

C.2 Microchip

Pic24FJ16MC101

http://ww1.microchip.com/downloads/en/DeviceDoc/39997B.pdf

C.3 Newhaven

NHD-0216K3Z-NSW-V3

http://www.newhavendisplay.com/specs/NHD-0216K3Z-NSW-BBW-V3.pdf

C.4 Texas Instruments

MSP430

http://www.ti.com/lit/ds/symlink/msp430g2553.pdf

LMZ21700

http://www.ti.com/lit/ds/symlink/lmz21700.pdf

TPS62150A

http://www.ti.com/lit/ds/symlink/tps62150a-q1.pdf

TPS62177

http://www.ti.com/lit/ds/symlink/tps62175.pdf?keyMatch=TPS62177&tisearch=Search-EN-TechDocs