AUTOMATED PILL DISPENSER

By

Chris Lee

Eric Groeteke

Jayan Hewaparakrama

Final Report for ECE 445, Senior Design, Spring 2016

TA: Benjamin Eng

04 May 2016

Project No. 17

Abstract

This paper discusses design decisions made by group 17 during the creation of the Automatic Pill Dispensing System. The paper takes a top-down approach in describing comprehensively how the machine is designed. It is a full overview of all components used in the system and how these components work together to accomplish its goal of dispensing pills according to a schedule set by a user. It will go on to discuss the production cost of a machine such as this. The paper concludes by discussing what accomplishments were made, what difficulties and challenges the group had, possibilities for future work on this project, and ethical considerations of the design.

Contents

1. Introduction

1.1 Statement of Purpose

Adherence is how closely patients correctly follow prescription instructions. According to a CVS report on adherence as seen in [1], the estimated cost of non-adherence in the United States is $290 billion. Many people have prescription medications in the form of pills and are non-adherent to these prescriptions. We want to improve the quality of life for all users, by preparing the medication regimen automatically to eliminate non-adherence cases with regard to pill-based medication.

This project is an attempt to create something that is more useful than current products on the market, by minimizing the work that the user must do in order to regularly take their medication(s). Current pill dispensing machines involve the user preloading compartments with the correct combination of pills. These pill compartments then get dispensed at time intervals set by the user. The hope is to help those who have difficulty with fine motor skill by making the loading process far less of an obstacle. Also, it is our hope to make the scheduling process extremely intuitive even for new users. Those with memory disabilities will no longer have to constantly remember when they have to take certain medications. In addition, recent design projects implement pill dispensing machines with a limited number of compartments. The pill dispensing machine that we are proposing will automatically dispense the right combination of pills, and will be capable of accommodating more medications.

2 Design

The intention of this section is to provide a comprehensive understanding of how the automatic pill dispensing system is built. Starting with a basic high level understanding is key to understanding the more detailed components of the machine. From the high-level understanding it is then possible to discuss every component in a very detailed manner. The machine's modular design highly lends itself to this kind of top-down analysis.

Inputs to the machine include, the time and date, how many cartridges are currently loaded into the system, how many pills are in each cartridge, and when pills need to be dispensed. Outputs include the pills themselves and any warning or informative notifications. The notifications are intended to alert the user of the status of the machine. At any time the user should be able to access the user interface and have an understanding of what it is doing. In addition to informing the user of its regular processes, it will also display if the machine needs maintenance or if there is some kind of internal error within the machine. The idea is that if something happens that is out of the machine's control, the user is at least notified so the issue can be corrected. For example if too many or two few pills are dispensed, the user needs to be notified so they can then properly double-check the pills that were actually dispensed and make adjustments to adhere to their medical schedule.

2.1 Block Diagram

One layer of abstraction deeper, we can take a look at each distinct but essential module of the machine. The block diagram in Figure 1 is split into four important modules. The power module (yellow) produces all necessary voltages to power every other component in the machine. The circuitry/hardware module (pink) includes all mechanical components and all circuitry used to operate these components. The software module (red) is what sends signals to the circuitry that instruct when and how the mechanical parts will be moved. The user interface module (green) is used to define all components that the user will interact with directly on a regular basis. It is where all user inputs are stored and dispensed pills are retrieved. Saved schedules in this module tell the hardware module when and how to operate.

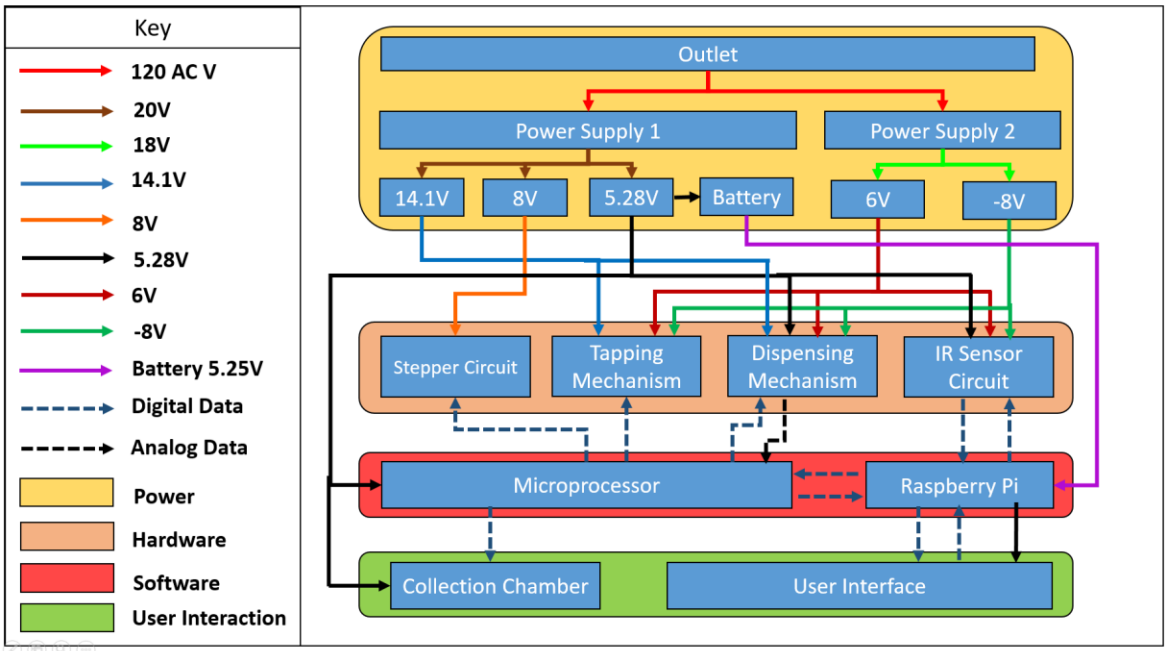


Figure 1: Block Diagram for Automatic Pill Dispensing System

2.2 Block Descriptions, Circuit Schematics, Simulations, Calculations

To go more in depth, this section will describe each block specifically. It will list all inputs and outputs of every block as well as provide all schematics, charts, and diagrams that are necessary for the analysis of each block.

2.2.1 Power Module

The power module consists of two independent DC power supplies providing 28V and 14V while sharing a common ground. The 28V supply is regulated to +14.1V, +8V, and +5.28V using linear variable voltage regulators. The 14V supply is regulated to +6V and -8V. The +5V power line is used to charge a back-up battery which provides power to the Raspberry Pi. Multiple power lines are used because there are multiple hardware components which require multiple voltage supplies. The power module consists of 2 transformers, because the initial transformer used in power supply 1 was not adequate to optimally power the op-amps. In order for the op-amps to function properly, they require both negative and positive voltage. Therefore, a center-tapped transformer was

implemented to power the op-amps. In addition, the power module is designed to withstand 3A, because at any given time the total current being drawn by the circuit loads will be less than 3 A.

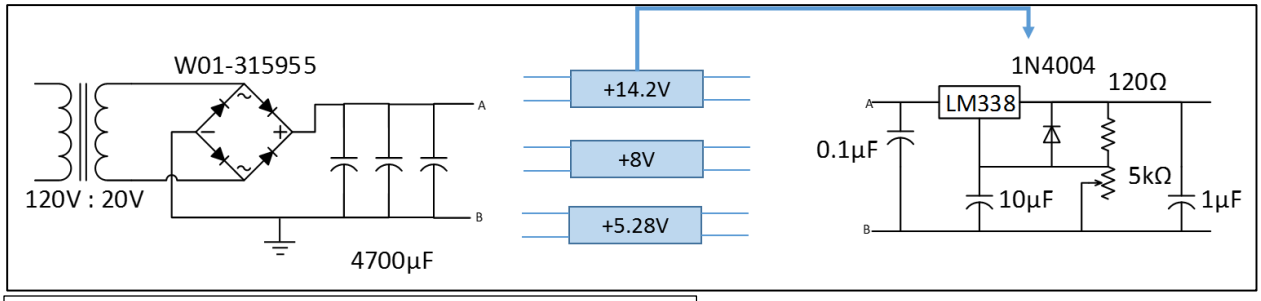


Figure 2 (Above): Schematic of power supply 1 with expanded voltage regulator circuit.

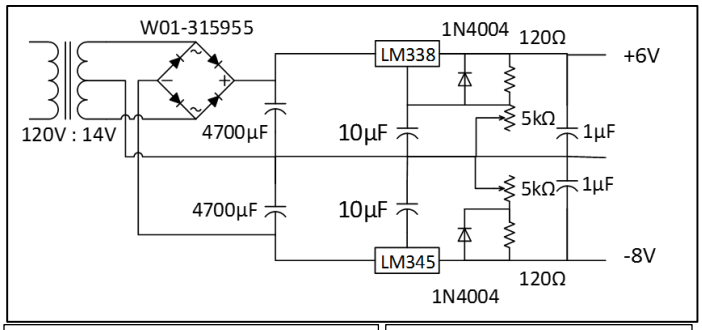


Figure 3 (Above): Schematic of power supply

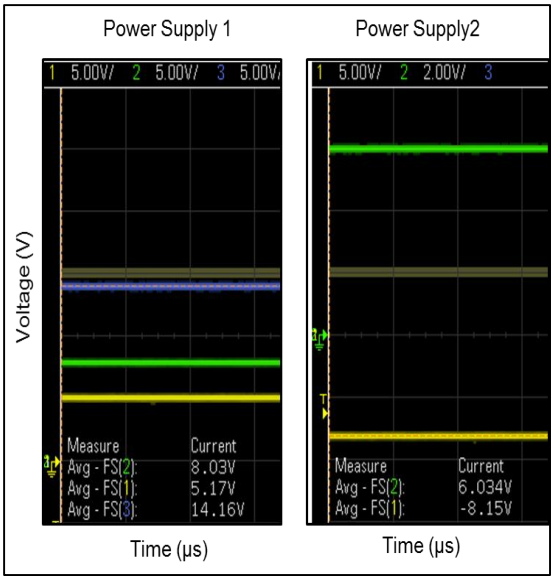


Figure 4 (Right): Oscilloscope readings

The outputs of power supplies 1 and 2 shown in Figures 2 and 3 are verified via the oscilloscope readings shown in Figure 4.

2.2.2 User Interface

The User Interface and Graphical User Interface are implemented using a 3" TFT display for giving and receiving user feedback. Large font/Large images to accommodate those with poor eyesight. The (G)UI is powered off of a Raspberry Pi Zero, due to its small size and ease of use for implementing graphical interfaces. There is a large number pad for entering information such as pill quantity, schedule times, etc. with the idea that any non-numeric input required can be selected from a menu presented on the screen with a number representing a menu selection. This also reduces complexity and risk of user error or typos as the user is not required to type any text into the device. User information will be recorded from user input and stored within its internal memory. Then the Pi will create instruction packages to send to the Atmega168, which controls the motors, at the scheduled times. The Pi does not handle the hardware (motor) directions directly and instead passes this off to an Atmega168 because we wanted to keep the timing-sensitive operation of moving parts as synchronous as possible and remove any chance of latency from dealing with scripts running off of an operating system which is simultaneously handling UI.

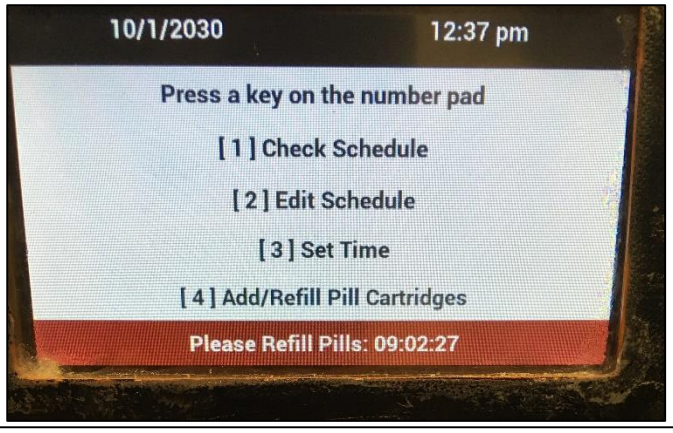


Figure 5: The main screen of the GUI for the pill dispensing machine.

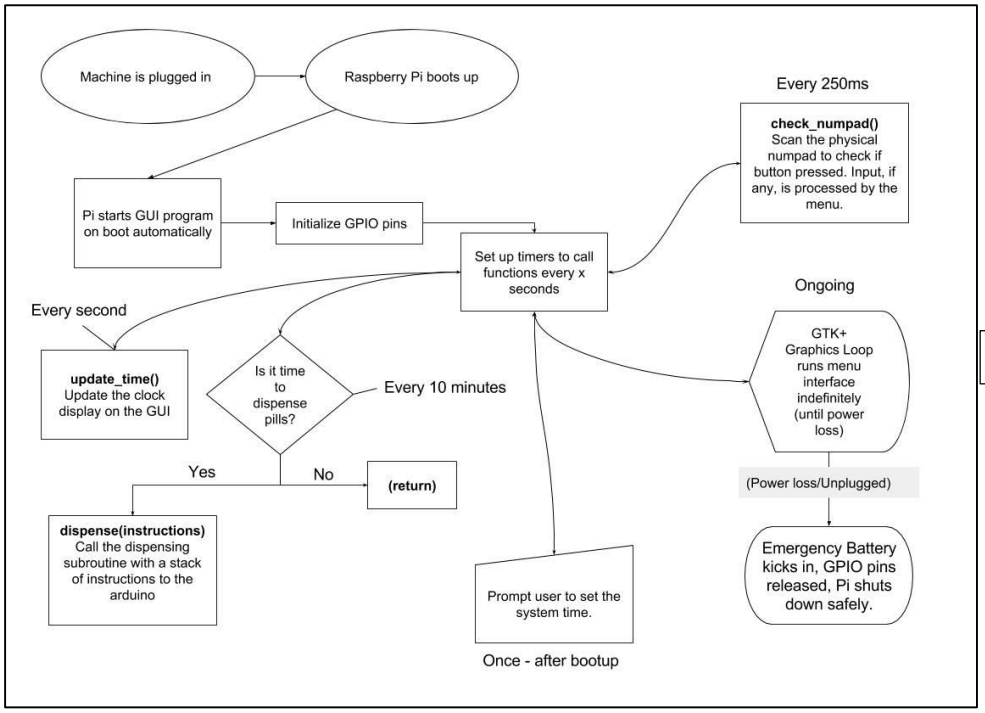


Figure 6

Figure 6 (Above): Shows the flowchart for the underlying program which controls the Graphical User Interface (GUI) and non-graphical interface (UI) which is used to display and receive information from the end user. The GUI program is run immediately upon boot up of the raspberry pi as a C++/GTK+ full-screen program which will prevent the user from access to the underlying Linux system unintentional or otherwise. In order to display the menus and other such information on the screen, the GTK+ graphical library is used which requires calling the GTK infinite main loop in favor of our own. As a consequence, any interaction we wish to handle with our own code is primarily done by scheduling periodic function calls as indicated by the “every x” lines of the flowchart. As the Pi requires an internet connection to sync up time on boot, we instead required the user set the time every time the dispenser is unplugged/plugged back in in lieu of complicating things with a required internet connection, much like many other common appliances.

Figure 7 (Below): Shows the flowchart for the Graphical User Interface (GUI) menus. Upon boot, the user only has access to one menu which is to set the time. Schedule information, stored in FLASH

(SD-card) memory is retained regardless of if power is cut to the dispenser, but the clock does not have a secondary power source and so must be set (just once) each time the power to the dispenser is removed. Menu 2 is the main menu and presents many options to the user such as viewing and editing schedules, time, and pill replacement - the final option keeping the dispenser from dispensing until the pill replacement has been completed. The read-only schedule option is present to prevent accidental schedule changes if all the user wants to do is verify information. Whenever it is time to dispense pills, so long as the user is not changing pill cartridges in the dispenser, the GUI halts all user input and displays warning messages while the dispenser dispenses. Upon finishing pill dispensing, the GUI will return to where the user left off.

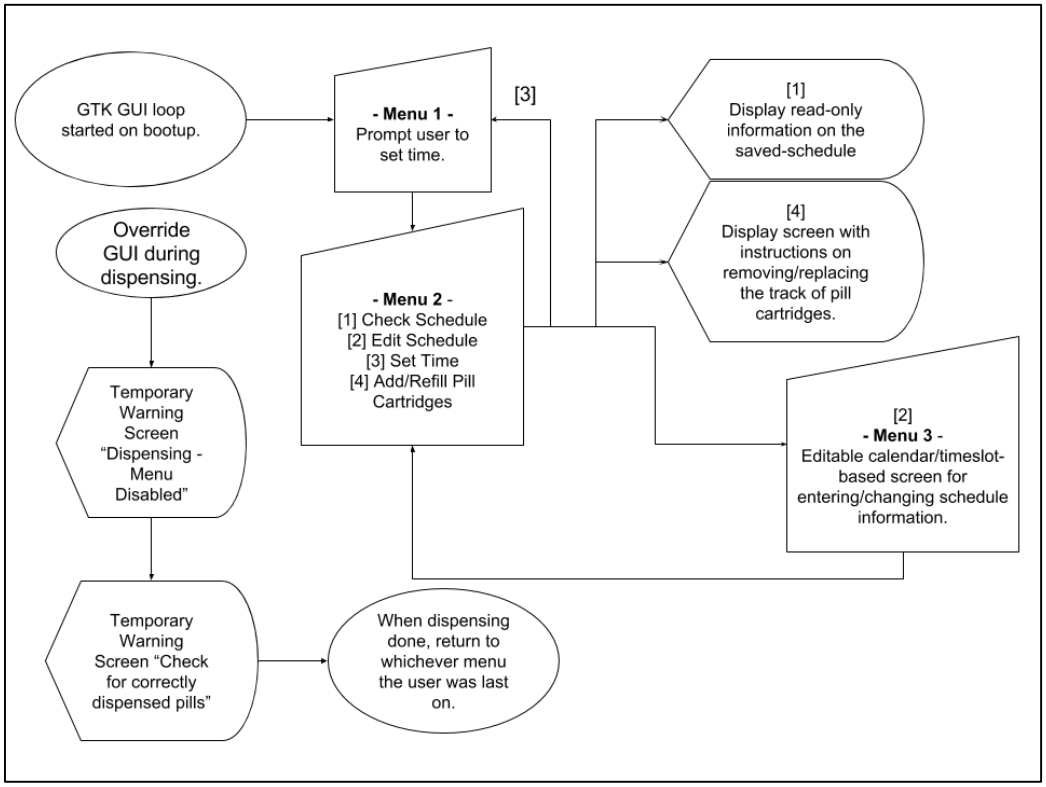


Figure 7

2.2.3 Raspberry Pi/ Microcontroller

The Microcontroller is an Arduino that will act as a driver for all of the hardware used in this project. The Raspberry Pi will be running software programs in C++ to control all of the hardware using specially defined functions used to control the driver in the Arduino. The Raspberry Pi sends the Arduino a 3-bit encoded instruction. Three bits are needed to differentiate between the five functions that the machine can carry out. These instructions include moving the track to the left, the right, picking up a pill, dropping a pill, and opening the trap door inside the collection chamber. The instructions are carried out only once the Raspberry Pi also gives a high signal on the execute bit and this only occurs if the Arduino sets its ready bit to 'high,' indicating it is ready for another instruction. A ready signal will only be sent when the machine has finished carrying out its previous command.

2.2.3.1 Raspberry Pi

The Raspberry Pi receives a high signal from the detection system, specifically via an SR latch, so that it can record that a pill has been dispensed properly. It sends a signal back to the detection system so that it can reset the latch and wait for the next pill to be dispensed. The software contains its own methods and algorithms used to control the hardware via the microcontroller driver based on the schedule determined by the user. The main algorithm will dispense pills based on the schedule using methods "Move Right", "Move Left", "Pick Up Pill", "Drop Pill", and "Empty Tray." The Empty Tray command has its own command bit separate from the other command bits and execute bit. It will operate the trap door immediately upon a high signal without the need for the execute bit to be high. Move Right is activated by sending '00' to the remaining command bits of the Arduino. Move Left is activated by sending '01' to the remaining command bits. Pick Up Pill is activated by sending the '10' to the remaining command bits. Drop Pill is activated by sending the '11' to the remaining command bits. These methods are defined to tell the drivers when to operate.

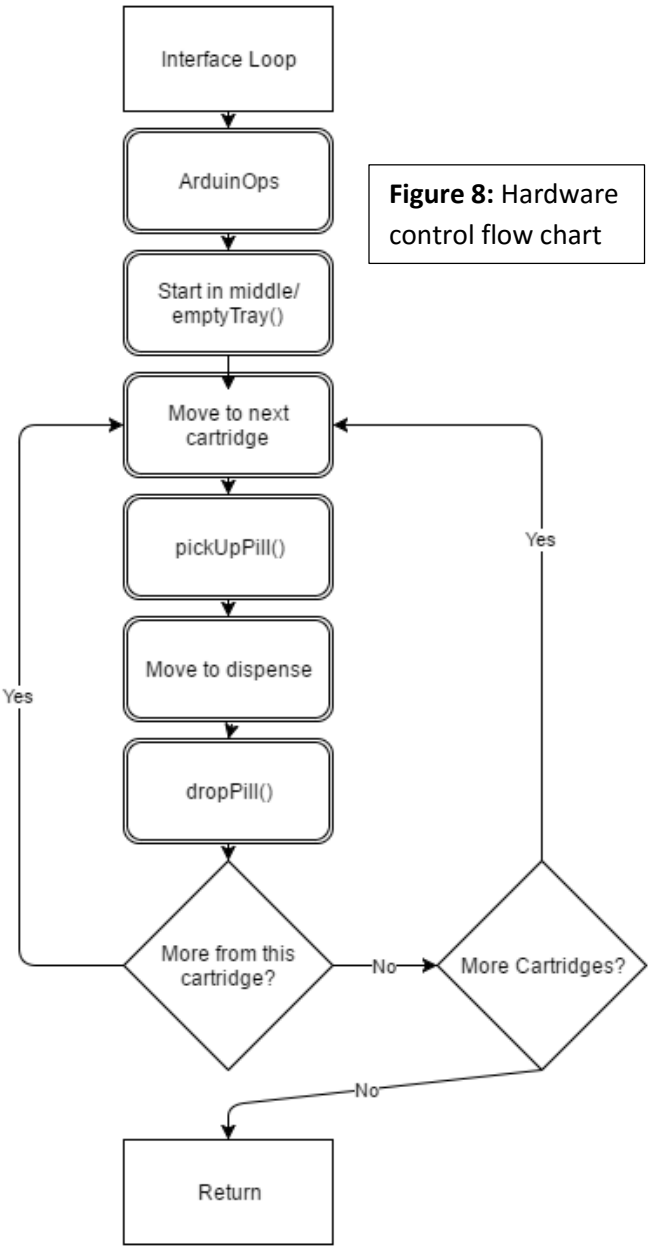


Figure 8: Hardware control flow chart

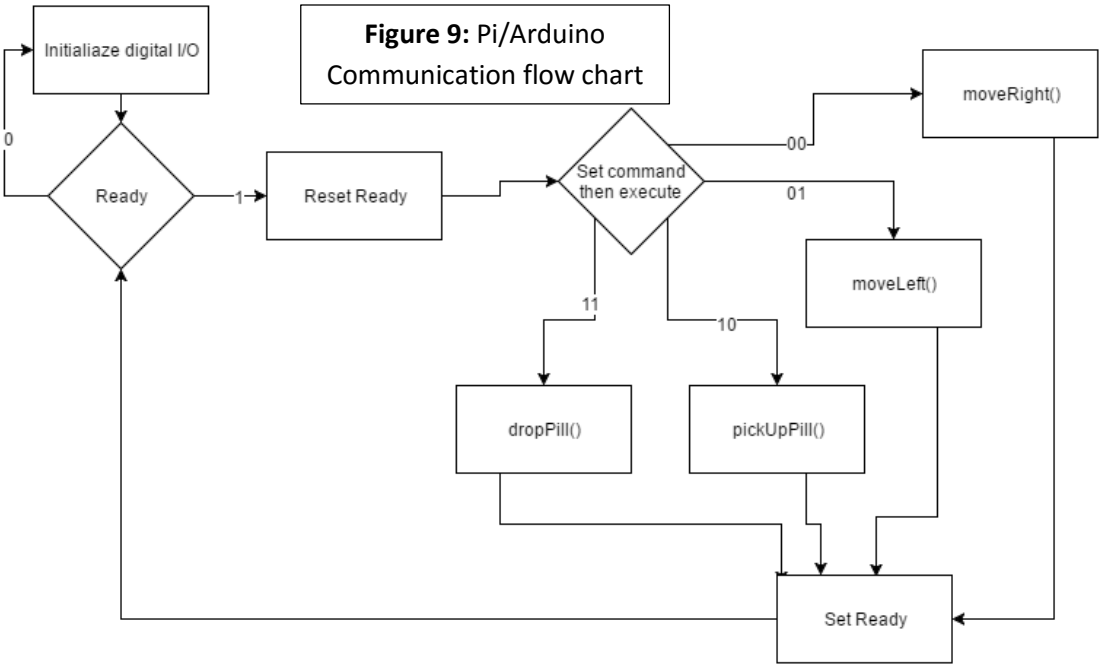


Figure 9: Pi/Arduino Communication flow chart

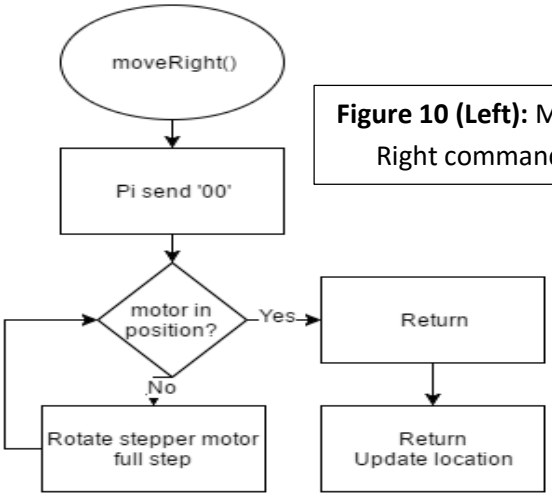


Figure 10 (Left): Move Right command

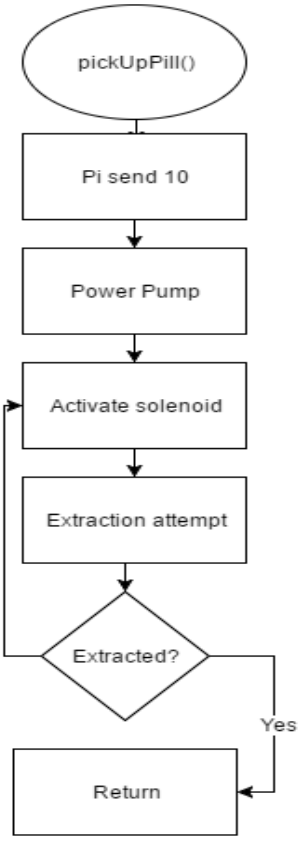


Figure 11 (Left): Pick Up Pill Command

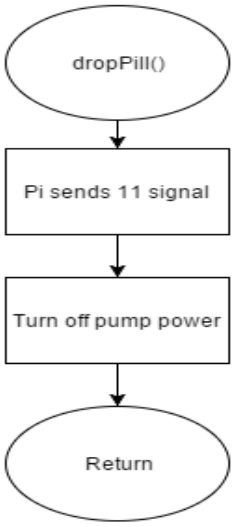


Figure 13 (Above): Drop Pill command

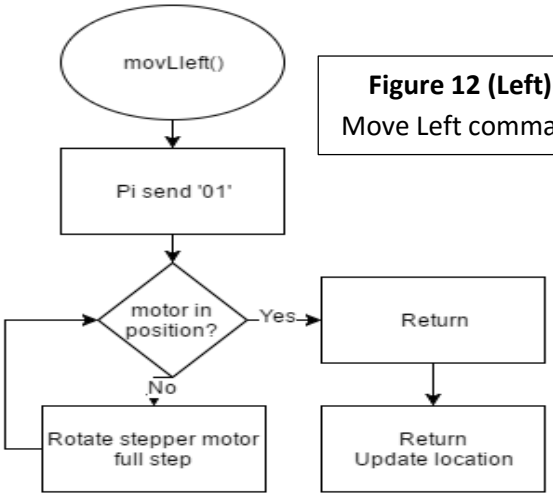


Figure 12 (Left): Move Left command

2.2.3.2 Microcontroller

The microcontroller uses the commands it receives from the raspberry pi to change signals that directly operate all of the mechanisms in the machine. A 4-bit signal is used to operate the transistors of the stepper motor. Giving a high to these transistors in the correct order will cause the stepper to rotate in one direction. Signaling in the opposite order will cause the stepper to rotate in the opposite direction. Two PWM signals are used to operate both servos. One servo rotates the vacuum tube into position to pick up a pill. The other servo is attached to the trap door in the collection chamber, and can open the trap door by converting the circular motion of the servo into linear motion via. This motion is converted via a rod that is attached to the door and the servo that is allowed to rotate around a bolt at each end. A 1-bit signal is sent to the pump to control the power of the pump. A 1-bit signal is sent back from the pump indicating whether or not a pill is clogging the intake. In order to get an accurate reading of this, the microcontroller computes the average of the last thirty readings. If it is below a threshold of 2.8 V, it is an indication that a pill is not blocking the intake (i.e. we are picking up a pill). If the average is above 2.8 V, it indicates a pill is blocking the intake.

2.2.4 Stepper Circuit

The cycling mechanism consists of a set number of cartridges attached to a geared rack. A stepper motor with a gear connects to the gear rack, and allow the cartridges to move. The stepper motor's rotational motion is converted to linear motion by the use of a gear and a gear rack. Each of the cartridges are evenly spaced such that the distance between the cartridges will equal a set number of stepper motor steps. There are 42 whole steps of the stepper motor between each cartridge. The position of the track is monitored by the Arduino by tracking how many full steps the stepper has made.

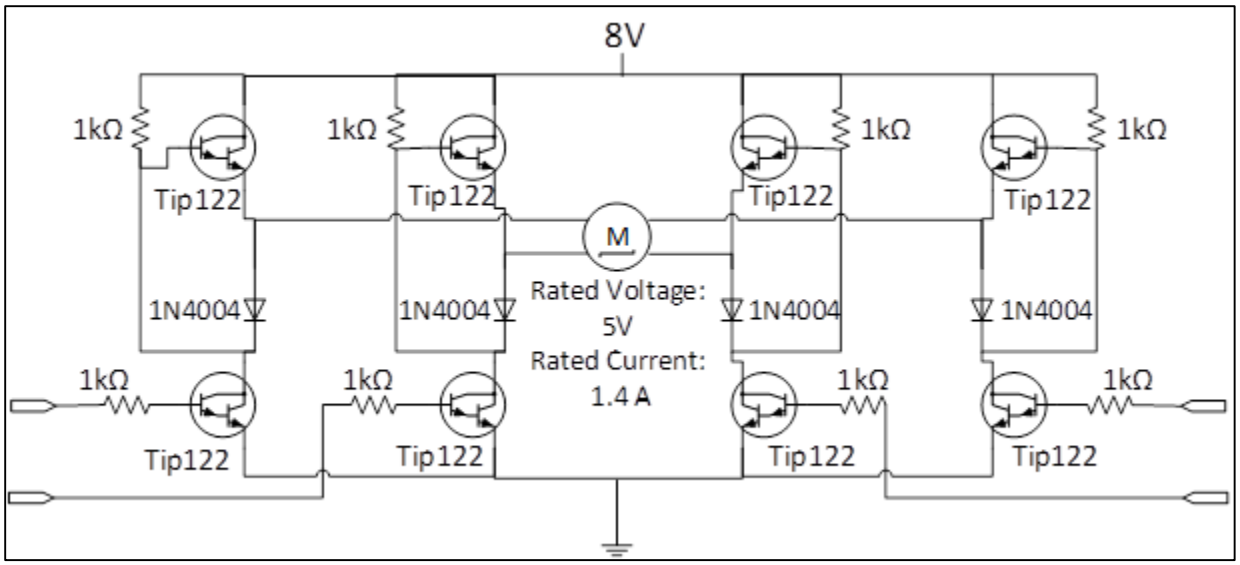


Figure 14: Stepper Circuit Diagram

The stepper motor contains two coils which are each driven by two independent H-bridges shown in figure 14. Each of these H-bridges comprise of four NPN transistors. The h-bridges are used to control the flow of current in both directions through the motor when commanded. This allows the

control of the stepper motor in both the clockwise and counterclockwise directions. The stepper motor has a rated current of 1.4 A and a rated voltage of 5 V. The measured resistance across each coil is ~3.8 Ω, the voltage drop across each Darlington transistor is 1.4V, and the voltage drop across each diode is 0.7V. Equations (1) and (2) demonstrates the current through the motor. Since two transistors and a diode are used in series, the voltage drop across the motor is 4.5 V.

$$I_c = \frac{V_{cc} - V_{ce}}{R_L} \tag{1}$$

$$I_c = \frac{8-(1.4+0.7+1.4)}{3.8\ \Omega} = \frac{4.5\ V}{3.8\ \Omega} = 1.18\ A \tag{2}$$

The calculated current value agrees with the measured current through the stepper motor.

2.2.5 Tapping Mechanism

The tapping mechanism is in place to ensure that the pills are in the location where the vacuum tip will descend. This mechanism is in place to prevent pills from being stuck somewhere in the cartridge where it cannot be retrieved by the dispensing mechanism. This system is implemented via a solenoid that raises and lowers to tap the cartridge that is currently above it. It is operated via a 1-bit signal from the Arduino. The Arduino rapidly turns the solenoid on and off to produce the tapping effect. The solenoid is driven by a very simple circuit shown in Figure 15. The solenoid functionality was verified via mechanical observations.

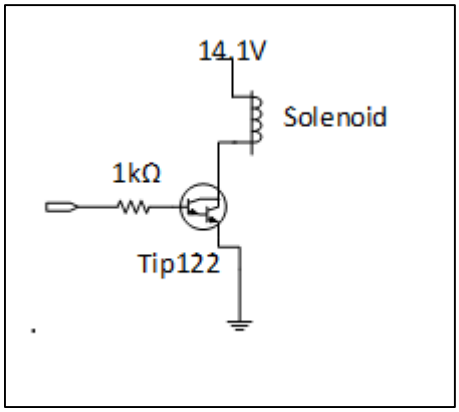


Figure 15: Solenoid Circuit Diagram

2.2.6 Dispensing Mechanism

The dispensing mechanism consists of a vacuum pump with tube that leads to a specially designed tip, and a servo. The main purpose of the dispensing mechanism is to isolate a single pill from the cartridge, and dispense that pill into another location. The tip of the vacuum is designed for isolating one small item such as a pill. It was originally intended to be used for designers to pick up singular sequins. Once the cartridge is in a specified location, the servo will move the vacuum tube from its elevated position to pick up a singular pill from the specially designed cartridges. The vacuum will then raise back up and send signals to the Arduino for it to check if the vacuum intake is blocked. If it is not blocked, it will re-attempt to pick up a pill until it is. Once the intake is blocked with a pill, the pill will be dropped off in the slide to the dispensing tray through a designated drop location in the cartridges. The solenoid and servo are operated by signals they receive from the Arduino. The pump is powered via the Arduino.

Figure 16: Vacuum Tip

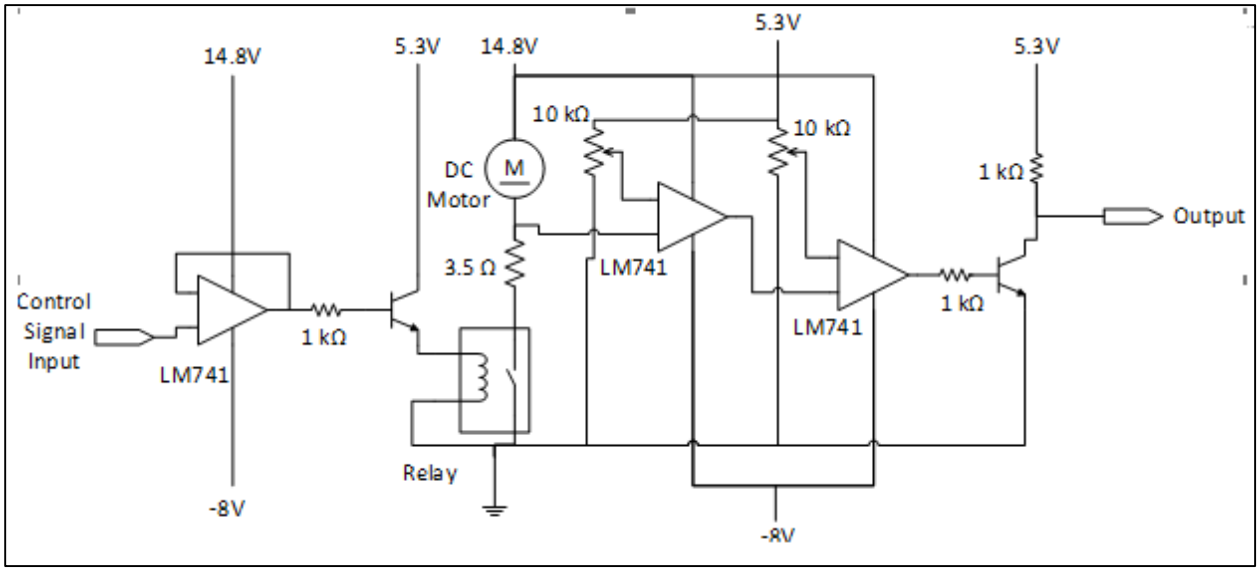


Figure 17: Vacuum Circuit Diagram

The control input signal shown in Figure 17 is used to activate a transistor which engages the relay. This relay activates the vacuum pump, and a set of comparators is used to monitor the voltage drop across the 3.5 Ω power resistor. When a pill clogs the air-intake valve, the output voltage results in an average voltage of 3.6 V. When a pill does not clog the air-intake valve, the output voltage results in an average voltage of 1.7 V. These output voltages are verified in the oscilloscope readings shown in Figures 18 and 19.

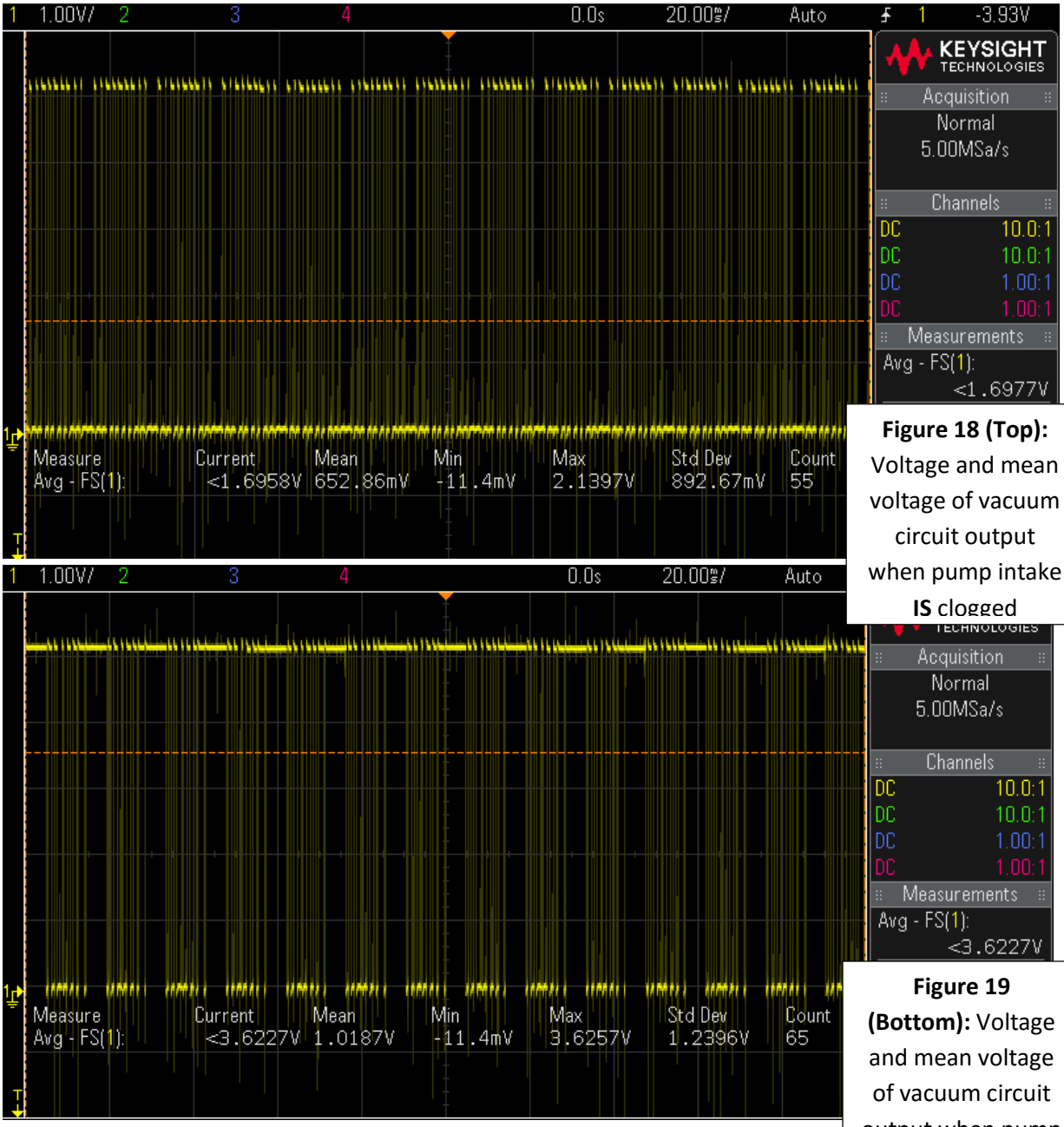


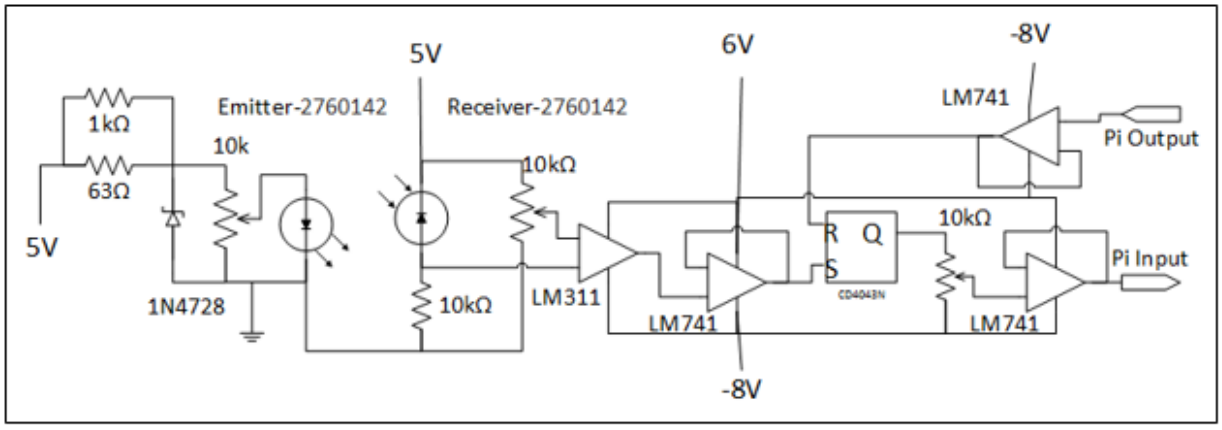
Figure 18 (Top): Voltage and mean voltage of vacuum circuit output when pump intake **IS** clogged

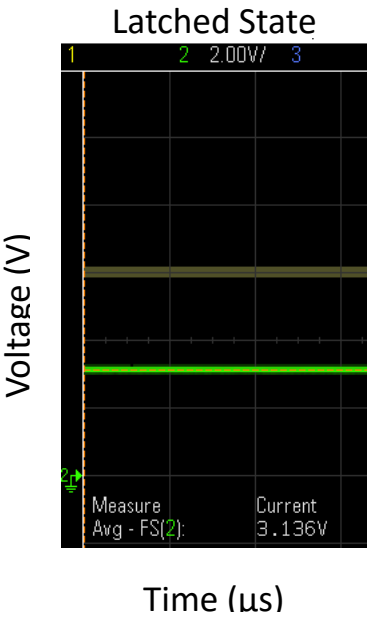
Figure 19 (Bottom): Voltage and mean voltage of vacuum circuit output when pump intake **IS NOT**

2.2.7 IR Sensor Circuit

The IR Sensor Circuit will consist of an infrared emitter and receiver located in the slide to the collection chamber. The purpose of the circuit is to detect whether a pill has been dispensed. Once the vacuum pump dispenses a pill, the pill will fall and this motion will be detected by the infrared sensor. The information gathered from the infrared sensor will be stored in a latch until the Raspberry pi is able to read it. The Raspberry pi uses this information to count how many pills have been dispensed. The Raspberry Pi then sends a signal back to the detection system letting it know that it has received its information and may reset the latch. The time between setting and re-setting the latch is negligible (within milliseconds) compared to the time between dispensing one pill to another so there is no risk of missing the detection of another

dispensed pill.





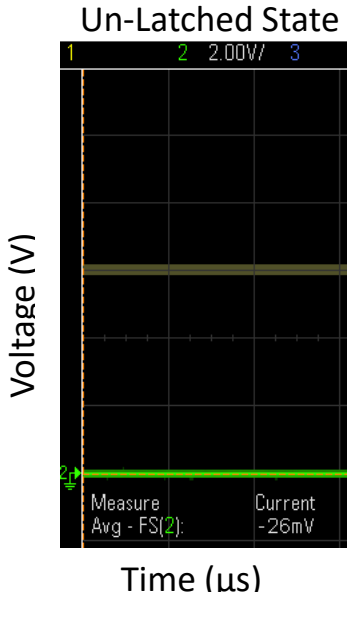


Figure 20 (Above): IR Sensor Circuit Diagram

Figure 21 (Left): Oscilloscope reading of latched and unlatched states from IR circuit

Figure 20 displays the circuit schematic of the IR sensor circuit. The signal from the IR emitter is constantly being monitored by a comparator using the LM311 op-amp. When a pill blocks the IR emitter/receiver beam, the SR latch gets latched and outputs 3.1 V as shown in the left plot of Figure 21. When the Raspberry Pi resets the latch, the output of the latch reads 0 V as verified in the right plot of Figure 21.

2.2.8 Collection Chamber

The collection chamber is a chamber with a front, main door and a bottom trapdoor. The latter is attached to a servo motor. Once all the pills have accumulated into the collection area, the patient can retrieve their pills from the chamber through the main door. If the patient does not take the pills, then the trapdoor will open before the next dosage is delivered. This function is performed via the "Empty Tray" command. This will allow the forgotten pills to fall into a separate compartment. The Arduino will provide a signal to the servo attached to the trap door to inform it when it can dump the pills currently inside of it through the trap door.

2.2.9 Alternative Designs

There are few alternative designs that could improve the machine in terms of reliability and appearance. The method of picking-up a pill can be improved such that a linear actuator is implemented instead of the servo motor. This will change the angle of contact between the pill and vacuum tip and prevent the pill from slipping. To reduce the size of the overall machine, a circular track could be implemented instead of the linear track. In addition, surface mount circuit components can be utilized instead of through-hole components to reduce the size of the circuits.

3 Costs

3.1 Labor

Table 1: Labor Costs

	Hourly Rate	**Total Hours**	**Total = Hourly Rate × 2.5 × Total Hours**
Eric Groeteke	$25.00	200	$12,500
Christopher Lee	$25.00	200	$12,500
Jayan Hewaparakrama	$25.00	200	$12,500
Total	$75.00	600	$37,500

3.2 Parts

Table 2: Part Costs

Item	**Manufacture**	**Part Number**	**Quantity**	**Unit Cost**	**Total Cost**	**Notes**
High Torque Stepper Motor	Microkinetics	23HT175D	1	$54	$54	
Vacuum Pump	Airpo	D2028B	2	$18.00	$36	
Servo Motors	Hitec	HS-485HB	2	$17.00	$34	
Solenoids	Amico	a13072900ux1656	2	$13.00	$26	
Gear(s)	ServoCity	615250	1	$8.00	$8.00	
Gear Rack	ServoCity	48P-125-12	1	$10	$10	
Raspberry Pi	Raspberry Pi Foundation	Model-Zero	1	$35	$35	

Keypad	Vetco	VUPN6955	1	$6.95	$6.95	
Wood	Lowes	-	8	$3.20(avg)	$30	
Plastic (spool)	Maker Filament	-	1	$21.55	$21.55	
Tubing	Lowes	-	3	$0.15	$0.45	
Microcontroller	Arduino	Uno	1	$30	$30	
NPN Transistor	Fairchild	TIP122	10	Free	Free	
Voltage Followers	Fairchild	LM7808	4	Free	Free	
Positive Variable Voltage Regulator	Texas Instruments	LM388	4	Free	Free	ECE Service Shop
Negative Variable Voltage Regulator	Texas Instruments	LM345	4	Free	Free	ECE Service Shop
1 kΩ, ¼ W Resistor	-------	1 k Resistor	15	Free	Free	ECE Service Shop
10 kΩ Potentiometer	-------	10 kΩ Pot	10	Free	Free	ECE Service Shop
5 kΩ Potentiometer	-------	5 kΩ Pot	5	Free	Free	ECE Service Shop
120:20 V Transformer	Triad Magnetics	VPT18-13800	1	$60.16	$60.16	
120:14 V Transformer	Triad Magnetics	166P18	1	$33.95	$33.95	
Bridge Rectifier	Multicomp	W02	2	Free	Free	ECE Service Shop
220uF Capacitor	-----	220uF Capacitor	1	Free	Free	ECE Service Shop

4700 uF Capacitor	-----	4700 uF Capacitor	5	Free	Free	ECE Service Shop
10 uF Capacitor	-----	10 uF Capacitor	5	Free	Free	ECE Service Shop
1 uF Capacitor	-----	1 uF Capacitor	5	Free	Free	ECE Service Shop
0.1 uF Capacitor	-----	0.1 uF Capacitor	3	Free	Free	ECE Service Shop
SR Latch	Futerlec	CD4043	1	Free	Free	ECE Service Shop
IR Emitter/Receiver	Radio Shack	2760142	1	$3.99	$3.99	
5 V Relay	Radio Shack	2750240	1	$4.99	$4.99	
5.0 " 40-pin TFT Display – 800x480 with Touchscreen	Adafruit	Product ID: 1596	1	$39.95	$39.95	
TFP401 HDMI/DVI Decoder to 40-Pin TTL Breakout – With Touch	Adafruit	Product ID: 2219	1	$29.95	$29.95	
UPS Pico (Battery Backup power supply – just enough to safe shutdown)	ModMyPi	Product Code: MMP-0045 + Fan Kit	1	$45.84	$45.84	
				Total	**$510.78**	

3.3 Total

Table 3: Total Costs

	Total
Labor	$37,500.00
Parts	$510.78
Grand Total	**$38,010.78**

4 Conclusion

4.1 Accomplishments

By the end of the project, a successful working prototype was made. Although clunky and inefficient compared to what a marketable product would be, the end result was a fully-functional proof of concept that meets all objectives laid out in the "Statement of Purpose" section. The machine is also self-contained in the sense that it can be unplugged from the wall, picked up, moved anywhere (with traditional wall outlets), be plugged in, and function as expected. The modularity of the design was a huge triumph in itself. All modules laid out in the diagram can be disconnected and reconnected with ease via the use of RS232 ports, rainbow wire, as well as various other means to group wires carrying signals together. Its modularity is what made the organization of this report possible. The machine is also modular in a different way, in the sense that tracks with different number of cartridges can be easily installed to the system.

4.2 Uncertainties

Our biggest uncertainty was the calibration and reliability of the machine to dispense pills effectively when it is programmed to. Many factors go into calibrating the system such as track placement, vacuum pump servo placement, tube placement, the angle of approach of the tube, and the list goes on. Though we made strides toward making the machine reliable, the dispensing mechanisms reliability can be greatly improved. On average the system can retrieve one pill per minute. By reducing the amount of times the vacuum has to descend back down again, we can reduce the time it takes to dispense one pill to less than ten seconds.

4.3 Future Work

Although fixing the reliability of the dispensing mechanism would be the primary concern, there are several ideas for improving upon the prototype machine generated by this project. The machine itself is very bulky and can be reduced in size in many ways. The linear motion track can be replaced with a circularly rotating track. All circuit board sizes can be optimized to reduce surface area greatly. The power supply was very large because of our use of linear regulators. These regulators caused us to use several other large components such as heat shields that can be removed entirely by using switching regulators. To increase the user friendliness, helpful sounds can be incorporated into the design. This will further the interaction between the machine and its user by extending the range at which it can communicate. To reduce the monetary cost of the machine, the Arduino could be replaced by a lone ATMEGA328 chip. The product needs to be sealed away from moisture and light in order to store pills for a long period of time. The current prototype is made of wood and glass in order to display the mechanical movement of the system. Replacing these materials with something more lightweight would also lend itself to the overall weight of the machine. Beyond that, the only steps are to make the system more aesthetically pleasing in preparation for commercialization.

4.4 Ethical Considerations

The IEEE Code of Ethics [2] was used to guide the ethical considerations of this project.
1.) The pill dispenser will always dispense the amount of pills specified by the user. This will be ensured by rigorous experimentation and testing. If the user fails to take pills from the collection

tray before the next dispensing cycle, the pills from the old cycle will be removed from the collection tray in order to prevent the user from accidentally taking the wrong number of pills.
2.) Shortcuts in implementation will not be used at the expense of the eventual users of the product.
3.) Performance of device will be based off the results of rigorous experimentation and be presented in raw form.
4.) All motivation behind this project is from sheer interest of producing a novel useable product. Any other forms of motivation, including bribery, will not be used to alter our desired end goal.

5 References

[1] "Adherence Why It's So Hard And What We Can Do About It", CVS Health, 2014. Available at: http://www.cvshealth.com/sites/default/files/INSIGHTS%20ADHERENCE%202014.pdf

[2] Ieee.org, "IEEE Code of Ethics", 2016. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html.

[3] Guidance for Industry Container Closure Systems for Packaging Human Drugs and Biologics, 1st ed. Rockville, MD: FDA, 2016, pp. 2-37.

[4] Fda.gov, "Safety Considerations for Product Design to Minimize Medication Errors", 2016. [Online].
Available: http://www.fda.gov/Drugs/GuidanceComplianceRegulatoryInformation/Guidances/ucm331808.htm. [Accessed: 23- Feb- 2016].

Appendix A: Requirements and Verifications

Block	Requirement	Verification	Points
Raspberry Pi	1) Decrement pill count on infrared sensor trigger and then reset sensor.	1) Write a test program with a dummy test variable at an arbitrary positive integer. Trigger the IR sensor and the test program should receive this signal (passed through a latch), decrement the count, and then output a reset signal to the latch. The test program is a success if it only decrements when the sensor is triggered and decrements only once per latching event; an object continually stuck to the sensor should only trigger the decrement once.	5
User Interface	1) Take in and process numbers 0-9, *, and # from the user keypad.	1) Write a test program such that you: Create an array of 12 integers all initialized to 0. Turn on the GPIO input to row 1 of the number pad, check GPIO output from the number pad for all three columns. For any columns that have an "on" signal, set a corresponding cell of the array to 1. For instance, row 1 corresponds to cells 0-2 or the array, row 2 corresponds to cells 3-5 of the array, etc. Turn off the GPIO input to row 1. Repeat this process for all 4 rows. Afterward, the 12-integer array will hold all confirmed button presses as "1" and all unpressed buttons as "0". Output the contents of this array to the screen, and confirm that the correct cells of the array switch to "1" if and only if the corresponding button(s) is/are pressed. Note: As the interface involves physical buttons that need to be pressed – it is impractical to try to test all combinations of button presses, and digitally running through each combination on the GPIO pins would be testing the GPIO rather than the keypad/interface itself, rather it is sufficient to test that each button works individually and that combinations of keypresses work in general, as this is how the interface is intended to be used.	5
Power	1) 13 V DC Supply: V_{out} = 13 V +/- 0.25 V at 3 A maximum 2) 6.5 V DC Supply: V_{out} = 6.5 V +/- 0.25 V at 3 A maximum	1) 13 V DC Supply: a. Connect digital multimeter(DMM) in parallel with 13 V DC Supply. The voltage should read 13 V +/- 0.25 V b. Connect DMM in series with 13 V DC Supply and the dispensing block. The current should read less than 3A.	9

	3) 5 V DC Supply: V_{out} = 5 V +/- 0.25 V at 3 A maximum 4) Backup Supply: V_{out} = 5V +/-0.25V at 2 A maximum	2) 6.5 V DC Supply: a. Connect digital multimeter(DMM) in parallel with 6.5 DC Supply. The voltage should read 6.5 V +/- 0.25 V b. Connect DMM in series with 6.5 DC Supply and the dispensing block. The current should read less than 3A. 3) 5 V DC Supply: a. Connect digital multimeter(DMM) in parallel with 5 DC Supply. The voltage should read 5 V +/- 0.25 V b. Connect DMM in series with 5 DC Supply and the dispensing block. The current should read less than 3A. 4) Plug in main power supply for 5 minutes of charging. Unplug main power supply. Raspberry Pi (and peripherals attached) should remain powered in an uninterrupted state for (up to) 30 seconds. After which time the Pi should shut down safely. A safe shutdown is determined as one in which the Pi may be powered again and all GUI files can be re-loaded with no data loss. Data verification is done between copies of files saved and the files after reboot by bit-differencing these files.	
Cycling Mechanism	1)Bi-Polar Stepper Motor a. $V_{A\text{-}A}$- = 5 V +/- 0.1V at I_{in} < 1.41A $V_{B\text{-}B}$- = 5 V +/- 0.1V at I_{in} < 1.41A Note: The rated current through each coil is 1.41A. b. For 21 steps, the translated linear motion should be less than 8.5mm Note: According to the Tolerance Analysis, if more than 21 steps are taken, the	Note: Two test programs are used to verify the stepper motor requirements. These test programs are Atmega168 sketches that uses 4 pins for each of the stepper motor inputs. The programs are written to control the sequence in activating the stepper motor coils. One program causes the shaft to rotate clockwise, and the other program causes the shaft to rotate counterclockwise. 1) Bi-Polar Stepper Motor a. Run test program for clockwise Bi-Polar stepper motor. This program will apply the proper stepper motor sequencing for clockwise direction.	9

	vacuum tube will not be properly inserted	b. Connect digital multimeter in parallel with terminals of the Bi-Polar stepper motor on H-bridge 1 side. The voltage should alternate between positive and negative 5 V +/- 0.1 V. c. Connect digital multimeter in parallel with terminals of the Bi-Polar stepper motor on H-bridge 2 side. The voltage should alternate between positive and negative 5 V +/- 0.1 V. d. Repeat the above procedures for the counterclockwise test program. Visually inspect that the stepper motor rotates in the expected directions. e.) Run the clockwise test program, and measure how far the gear track moves. This distance should be less than 8.5mm.	
Vibrating Mechanism	1. Vibrating Motor V_{in} = 3 V +/- 0.25 V 2)Servo Motor a. V_{in} = 5 V +/- 0.25 V	1. Vibrating Motor a. Connect digital multimeter in parallel vibrating motor. Apply 3 DC volts to the terminals of the motor, the voltage should read 3 V +/- 0.25 V. 2)Servo Motor a. Connect digital multimeter in parallel with servo motor. Apply5 DC volts to the terminals of the motoer. The voltage should read 5 V +/- 0.25 V.	4
Dispensing Mechanism	1) Vacuum Pump a. V_{in} = 12 V +/- 0.25 V at I_{in} < 2 A b. When the air-intake line is blocked, I_{in} should increase by 0.1 A +/-0.02 A. 2) Solenoid Valve a. I_{max} <= 300mA at 12V b. Valve should open for V_{in} = 12V +/-0.2V	1. Vacumm Pump a. Connect digital multimeter in parallel with the vacuum pump. Apply 12 DC volts to the terminals of the vacuum pump, and the DMM should read 12V +/- 0.25 V. b. Connect digital multimeter in series with the vacuum pump and apply 12 DC volts to the terminals of the vacuum pump. The DMM should read I_{in} < 2 A. Close the air-intake valve. The DMM should read 0.1 A +/- 0.02A higher than the current when the air-intake valve was open. 2) Solenoid Valve a. Connect DMM in series with solenoid valve, and apply 12V. When the valve is open, the DMM should read less than 300mA.	8

		b. Connect the DMM in parallel with solenoid. Apply 12 DC volts to the solenoid terminals, and observe whether the valve opens when the DMM reads 12V +/- 0.2 V.	
Detection System	1) Infrared(IR) emitter and receiver: a. When nothing is in the line of sight between the emitter and the receiver, V_{out} = 0.5 V +/- 0.1 V. b. When a pill blocks the line of sight between the emitter and receiver, V_{out} should range from 2.3V to 1.3V.	1) IR emitter and receiver a. Connect DMM in parallel with the receiver terminals. Apply 5 DC volts to the transmitter terminals. The DMM should read between 2.3V and 1.3V. Connect DMM in series with the infrared emitter. The DMM should read less than 30mA. b. Drop a pill such that the line of sight between the emitter and receiver is blocked. As the pill blocks the line of sight, the DMM should read between 2.3V to 1.3V across the receiver terminals of the IR emitter.	5
Collection Chamber	1) Solenoid Valve a. I_{max} <= 300mA at 12V b. Valve should open for V_{in} = 12V +/-0.2V	1) Solenoid Valve a. Connect DMM in series with solenoid valve, and apply 12 DC volts to the terminals of the solenoid. When the valve is open, the DMM should read less than 300mA. b. Apply 12 DC volts to the terminals of the solenoid, and connect the DMM in parallel with solenoid. Observe if the valve is opening when the DMM reads 12V +/- 0.2 V.	5