Portable Pill Dispenser

Final Report

Justin Shien-I Couvignou
Sayan Pico Sur
Phillip Zhou

Final Report for ECE 445, Senior Design Laboratory, Fall 2019
TA: Kristina Miller

11 December, 2019

Abstract

Management of medications is a critical yet often difficult task for patients with lengthy prescriptions, physical ailments, and/or cognitive disabilities. This project proposes a novel portable pill management device which can be autonomously loaded with solid pills or tablets and dispense them to a patient at correct times. The portable device also displays basic dosage information for each pill to the user. The autonomous loading and dispensing of medications creates a seamless medication management system which requires minimal human intervention, among other benefits.

Table of Contents

1. Introduction

1.1. Problem and Solution Overview

Currently, management of medications is typically achieved with two devices: (i) stationary at-home devices, such as the Hero medication dispenser [1]; or (ii) passive container-like devices. The at-home devices comprise the benefit of full medication management with minimal user input, but restrict a user to remain at home as these devices are typically not portable. The passive container-like devices are portable, however require a user to manually load and track their medications which may be difficult for users with lengthy prescriptions, physical disabilities, and/or cognitive disabilities.

To reduce user involvement in the management of their medication and to remove restrictions to users with at-home medication dispensers, a portable pill management device was designed. The portable device is configurable to receive pills from an at-home device in an autonomous loading process, store relevant dosage information of the pills, dispense the pills to a user at correct times, and provide the user with basic dosage information of the dispensed pills. The portable device borrows benefits from both of the aforementioned products while providing a low-cost solution which aims to solve deficiencies found in both.

1.2. High-Level Requirements

- Portable device must be able to be autonomously loaded with pills and receive corresponding pill data wirelessly from an at-home unit.
- Portable device is able to dispense at least 3 different pill sizes at the correct time for each pill.
- Portable device must be able to display human readable text on a 16x2 LCD screen that briefly describes limitations or directions for taking the dispensed pills.

1.3. User Workflow

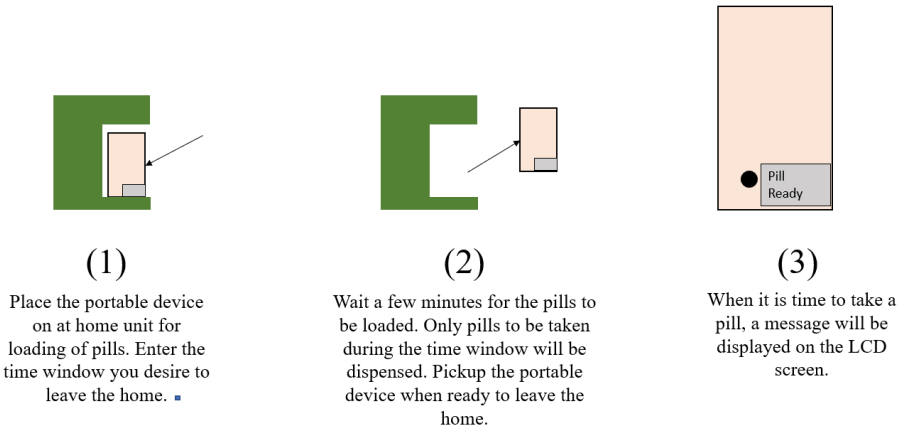

FIG. 1: User workflow when using the portable device and at home unit.

The indented user workflow for autonomous loading of the portable device includes a user first placing the portable device on an at home pill dispensing unit. The user may then specify a time window within which pills are dispensed (e.g., from 3PM to 6PM) to the at home unit and press a “begin loading” button. The user may return for the portable device within a few minutes and find it fully loaded and ready to be taken outside the home.

2. Mechanical Design

1.2. High-Level Mechanical Design

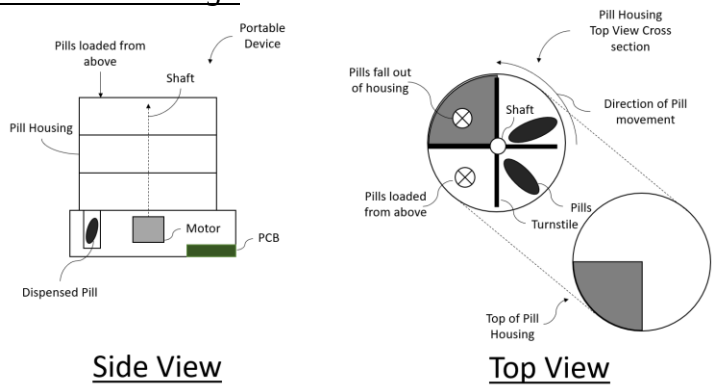

FIG. 2: (Left) A side view of the portable device which illustrates pill housings stacked vertically upon a base which houses electronics and motor. (Right) A cross section of a single pill housing section.

The portable device is intended to provide all of the aforementioned benefits in addition to being inexpensive to an end user (e.g., $100 or less). To achieve this, all electrical components and a motor are disposed in a lower section of the portable device to allow for the pill housings to be electrically passive devices, thereby enabling them to be manufactured inexpensively. This may allow for future iterations of the portable device where a user may expand or reduce its pill capacity by simply attaching or detaching passive pill housing sections depicted above in FIG. 2. Each pill housing comprises a turnstile disposed therein, the turnstile being coupled to a shaft which is further coupled to the motor which provides means for actuating the pills through the device and eventually to a user in a first in first out order. All turnstiles of each pill housing section align and move together when viewed from above, moving pills through the device in a downward spiral pattern.

1.2. Machine Shop Design

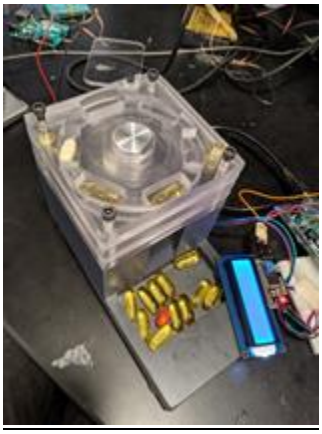

FIG. 3: Machine shop prototype of the Portable Device.

As shown in FIG. 3 above, the ECE machine shop design comprises of four stacked layers of polycarbonate about 2 cm thick disposed above a stationary PVC base. The lowest layer and second to top layers each comprise a gear coupled to the shaft and motor; the gears each comprise 6 teeth, leaving 6

spaces in between with a length of roughly 3 cm to hold pills, for a pill capacity of 10 pills. This size was chosen to accommodate the largest pill sizes available, which are 2.5 cm [2]. Fish oil pills [9] were used to demonstrate the device. The PVC base housed a 9V stepper motor, further described below. All electronics were placed outside of the device for the purposes of demonstration.

3. Submodules

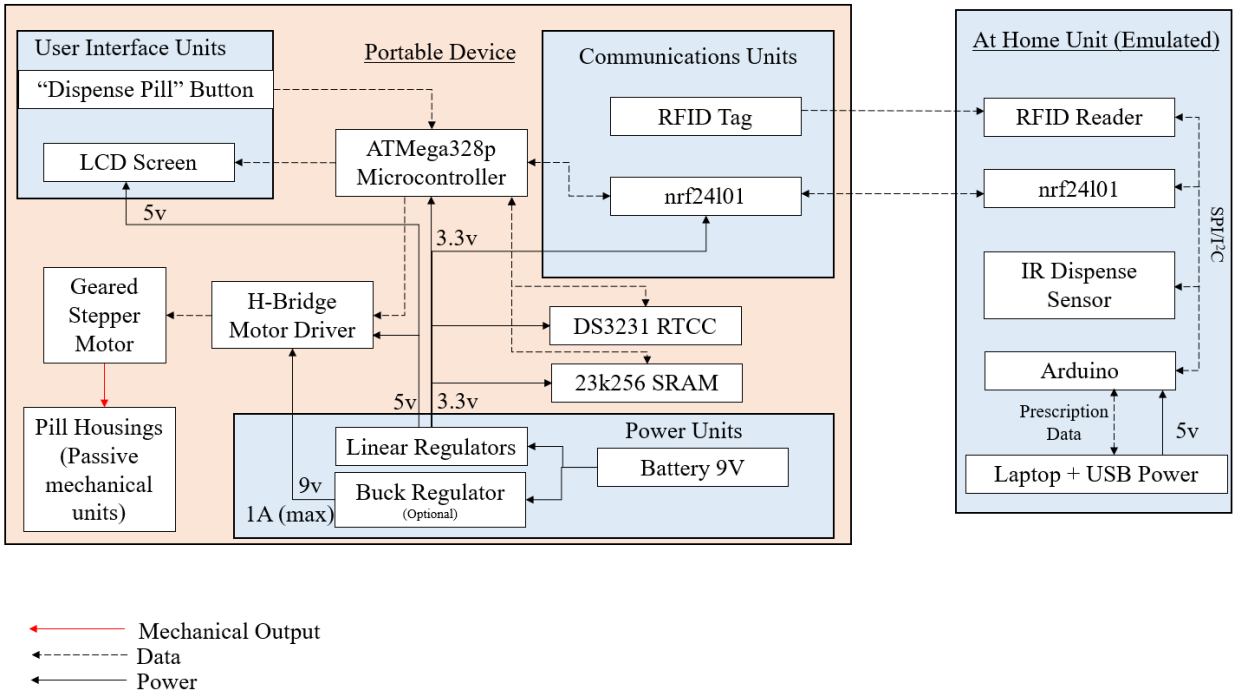

FIG. 4: Block diagram of the portable device. Each region in blue may be considered as a subsystem of the overall pill management system, each subsystem being further described below.

The system comprises of two main parts: (i) the portable device, and (ii) an at home unit. The portable device is the primary focus of the project, however some basic functionalities of an at home unit will be emulated to facilitate automated loading of pills into the portable unit. The portable device handles timekeeping and provides electromechanical means for dispensing pills to a patient. The portable device further displays basic information related to dispensed pills and customizable messages which may be pre-programmed by a user.

3.1. Communication Units

The two primary components of the communication submodule include the RC522 RFID module and the NRF24L01+ (NRF) wireless transceiver module. The RFID is used to ensure that the at-home unit only dispenses pills when a recognized portable device is nearby and the NRF is used to transfer pill data between the at-home unit and the portable device. The primary factors in designing the communication module is the need to secure user data and the design choice to keep the portable device wireless. In order to protect user data, any information being transmitted is encoded, as described in section 4.4, and physical distance for successful wireless communication is limited. The RFID has an extremely short communication range of less than 0.1 meters, while the NRF has a somewhat larger maximum range of approximately 2 meters. Only allowing short range communication enhances data security while also eliminating the impact of external noise and other background signals being picked up by either of the two modules. Use of the portable device involves the use of no wires or connectors as some users may potentially physical disabilities (e.g., tremors) that may make management of small

cables and connectors difficult. By designing a wireless experience, physical difficulties in handling the device are minimized.

The RFID module comprises a passive tag active reader system. The passive tag holds a unique ID which is read by the at home unit when the portable device is proximate to the at home unit. Use of RFID to identify the portable device may enable a single at home unit to dispense pills into a plurality of portable devices (e.g., in a hospital where a single dispensing system provides medication for many patients' portable devices). Additionally, RFID eliminates the possibility of the at home unit dispensing a medication when no portable device is nearby and/or dispensing of medication into an incorrect device, thereby enhancing the safety of the medication management system.

The NRF module operates in the 2.4 GHz range with multiple channels, allowing for the potential of several portable devices being connected to a single at-home unit. The NRF communicates through SPI to the ATMega microcontroller. Each message is limited to 32 Bytes per transmission packet, with internal handshaking between NRF transceivers to ensure message delivery. The NRF module's circuit can be placed directly onto the main PCB in order to reduce the overall size of the PCB, further improving the portability of the portable device. Following Nordic Semiconductor's recommended design [3], along with a Meandering Inverted F Antenna to minimize antenna footprint centered at 2.4GHz [4], an operational custom NRF24L01 module can be implemented directly onto the PCB of the portable device. This design is shown in FIG. 5 below. Testing indicates the antenna design is sufficient to transmit data, however the custom built NRF module could not receive data.

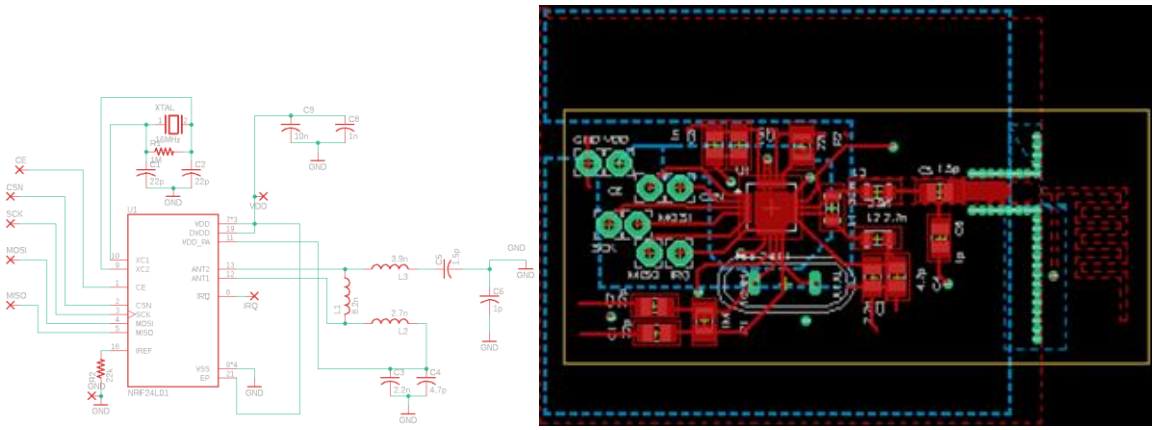

FIG 5: Custom NRF24L01 circuit and board layout.

A primary issue encountered with the NRF24L01+ module is an empty message bug, wherein the module indicates there is a message is stored in a buffer to be read, however the message comprises 0 characters. This is a known issue with the NRF24L01 chip and may be solved with more robust verification of received messages (i.e., checking the received message contains characters).

3.2. User Interface Units

The User Interface is comprised of a 16x2 character LCD connected to an I2C backpack module and a single button. The User Interface module is primarily designed to minimize the amount of interaction between the user and the portable device. Similar to the design reasoning in section 3.1, reducing the amount of interaction minimizes any issues that may occur as a result of impaired movement from the user.

The LCD screen displays any ASCII characters, with the screen showing up to 2 rows of 16 characters, for a total message length of 32 characters. This size has been deliberately chosen in order to match the NRF24L01 packet transmission size. A smaller screen promotes shorter custom messages, although longer messages can be displayed by scrolling messages. Shorter message are encouraged, however, as to not obfuscate medication directions. The I2C backpack module produces outputs to the

eight LCD pins based on serially communicated data from I2C bus of the ATMega. This decreases the amount of pins used to control the LCD from the ATMega, thereby reducing the number of traces and ultimately the overall size of the final PCB.

Upon the portable device dispensing a pill, the LCD displays a custom text message and cycles through preset messages associated with the pill. The preset messages include common medication restrictions/requirements (e.g., MAOI conflicts, food/water requirements, alcohol warnings, etc.). Currently, this data is input by a user into the at home unit via a prescription Excel spreadsheet. During loading, the data for each respective loaded pill is communicated using the NRF submodule to the portable device.

The button is connected to an analog input of the ATMega328p microcontroller, and is used as an interrupt throughout the entire program. Use of system interrupts for simple inputs is advantageous as the ATMega may be executing other subroutines while awaiting this input, wherein interrupts eliminate the risk of the ATMega missing of the button press. Periodically within the code, the program will wait for user input in the form of a button press (e.g., when a pill is ready to be dispensed). Through this method, only a single button is required to use the device, removing any ambiguity from the user about which buttons to press to properly use the portable device.

3.3. Motors, Drivers, and Calibration

Three types of motors were experimented with during the design of the portable device: geared bipolar stepper, geared unipolar stepper, and brush DC encoded motors. Common DC motors comprising encoders are typically expensive and/or are lengthy, making them unsuitable for a portable device. Initially, a unipolar 6V (28BYJ-48) stepper motor was chosen for actuation as stepper motors may be turned with high precision. The small size of the 6V motor made it ideal for a portable device. Later, after working with the ECE machine shop, an upscaled 9V bipolar stepper (17HS13-0404S-PG5) motor with active torque of 4Nm was chosen for the final demonstration to leave sufficient margin of error in the mechanical design. 4Nm of torque may turn about 25 turnstiles of the turnstile design depicted in FIG. 2 or about 5 geared layers of the machine shop design depicted in FIG. 3 above. All motor types tested comprise gear boxes to allow for sufficient holding torque (i.e., resistance to rotation when the motor is not powered) to hold the turbines or gears in place when the motor receives no power.

All of these three motor types may be driven with an H-Bridge driver, wherein bipolar steppers and DC motors may also be driven with a Darlington array (ULN2003) [10][11]. Accordingly, an H-Bridge (L293D) driver was chosen in the final hardware revision to enable driving of as many different motor types as possible. This eases later downsizing of the 9V motor as the mechanical design is refined as no hardware revisions are required to the motor driver, aside from a voltage regulator tuned to the voltage of the respective motor.

Lastly, a method for calibrating the motor was configured. Due to the circular symmetry of the device, calibration of the motor may only require an indication that the motor is at a reference 0^o pose. To detect and define this reference 0^o pose, a small permanent magnet was disposed in the distal end of an actuated arm or gear of the portable device and a stationary hall sensor was positioned below the magnet. The hall sensor produces an analog voltage output based on a proximate magnetic field. When the magnet is directly above the hall sensor, the sensor outputs a minimum or maximum value (based on relative orientations of N-S of the magnet and sensor). This minimum or maximum value is measured during initialization of the device by performing a minima/maxima search as the motor rotates at least 540^o, over rotated to ensure at least one full rotation is made and the minima/maxima is found.

Each motor activation step (i.e., dispensing/loading of 1 pill) increments a pose counter in software, wherein the pose counter reaching $N_{slots} - 1$ (N_{slots} = number of pills/layer) causes a calibration process to begin. In this process, the motor is rotated slowly as the ATMega reads the analog hall output and compares it to the minimum/maximum measured during initialization. System interrupts could not be used here due to sensor noise triggering constant interruption. If the measured value of the hall sensor is within 20% of the measured minimum/maximum sensed during initialization, the pose counter is reset to

0 and the motor is stopped at the 0º pose. This 20% margin of error was implemented to account for sensor noise while being small enough to not cause substantial mechanical misalignment.

3.4. Power Units

The portable device is powered using a 9V Li-ion battery. The 9V supply is split by two parallel linear voltage regulators: a 3.3V regulator (LM117-T) and a 5V regulator (APK2210). The initial design further comprised of a 6V buck switching regulator (LM2576T) used to power a 6V motor, however this regulator was removed to accommodate a larger motor as per machine shop request, wherein the voltage of the motor and battery where the same thereby requiring no regulator. Linear regulators where chosen to supply power to the digital circuitry as the low AC noise output of linear regulators is preferred, whereas AC ripple is permissible when supplying power to the motor.

Linear regulators, however, are inefficient as they produce an output voltage using a variable resistance voltage divider. That is, a portion of the input power, proportional to the step-down voltage difference is dissipated as heat. It was empirically observed that both linear regulators where warm after operating for more than 10 minutes, corresponding to a substantial waste of power from the battery supply. Future changes to the power supply units may account for this by (i) reducing the output voltage of the battery and using a buck-boost converter to power the motor, or (ii) using a low voltage battery with buck regulators to power the digital elements and couple bypass capacitors to the output to reduce the AC ripple.

Measurements from the PCB found the current draw from the logical circuitry to be 200-280 mA with the 9V stepper motor drawing an additional 400 mA when active. This is substantially higher than the estimated 25 mA of the design document, which reduces the device lifetime to 2.5 hours using a 500mAh battery. The reduction in the device lifetime may be attributed primarily to losses from the linear regulators, as mentioned above. The lifetime may be further improved by disabling the H-Bridge driver and LCD backlight when not in use, which is a substantial majority of the time the device is operative as the motor and LCD are only activated when a pill is being dispensed.

3.5. Memory

The ATMega328p comprises 2kB of dynamic memory, limiting the amount of pill data which may be stored as local variables to about 8 pills. Accordingly, an external SRAM (23k256) memory was implemented in the portable device to store received pill data. The SRAM comprises 32kB of byte-addressable memory, equating to roughly 1000 pills worth of data. As mentioned above in section 2.2.1, each transmission packet from the NRF radios is of size 32B. Accordingly, data for each respective pill is stored in a page in memory, or a 32B section, as shown below in FIG. 6.

Pill time data is compressed to two bytes of data prior to being communicated to the portable device via the NRF. The third byte comprises dosage information. The dosage information is encoded using a 1-hot encoding method further described below in section 4.4. Each bit denotes: (i) if the pill has food restrictions (some pills don't care), (ii) if the food restrictions include with or without food restrictions, (iii) water requirements, (iv) MAOI conflicts, and (v) alcohol warnings. The remaining 29 bytes of the 32 byte packet are utilized for the custom messages. This time and dosage information data is decoded in the idle subroutine, further described below in section 4.2.

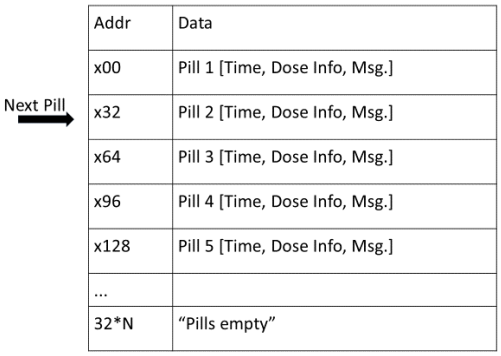

FIG. 6: Storage of pill data in 23k256 SRAM memory. In the figure, pill 1 has been dispensed and pill 2 is the next to be dispensed. The portable device is loaded with N-1 pills.

3.6. Real Time Clock (RTCC)

Real time timekeeping of the portable device is handled by a highly accurate, temperature controlled DS3231 real time clock chip. The DS3231 uses the I2C bus to output a current time as a string of characters as either "hour:minute:second" or "hour:minute" formats, the portable device uses the ladder output. The time of the DS3231 may be set using a library function [7] to allow for time-zone changes (e.g., if the portable device is taken on a longer distance trip) or synching of the device's time keeping.

3.7. At Home Unit (Emulated)

As mentioned above, a plurality of at home solutions for medication management currently exist on the market. As the portable device is designed to interface with these at home solutions, some basic functionality and communication between the two devices is required. Accordingly, only the basic requirements for the at home unit were emulated (e.g., mechanical dispensing of a pill from a reservoir is not considered). These basic functionalities include: (i) verifying that the portable device is ready to be loaded based on RFID; (ii) detecting that a pill is loaded into the portable device during a loading subroutine, using IR distance sensors described next; and (iii) communicating dosage information to the portable device using the NRF submodule.

The physical design of this at home unit may be similar to existing products but further include a base or section for the portable device to be placed during loading. The RFID Reader would be disposed within this base or section thereby requiring the portable device to be placed properly prior to any loading. Although the emulated at home unit is denoted as belonging "at home", the emulated at home unit may also be embodied within a larger pill management system configurable to dispense pills into a series of portable devices, such a setup could be used in a nursing home for the elderly. That is, the electronics platform of the emulated at home unit may be implemented into a plurality of systems to integrate the use of one or more portable devices. It is additionally appreciated that an at-home unit, as described herein, may also be illustrative of automated pill dispensers in a pharmacy or hospital and is not intended to be limited to consumer devices for home use.

Manual loading of the portable device was not implemented, primarily as autonomous loading using two synchronous devices is the more challenging of the two loading methods to design and implement. Further, building the emulated platform to facilitate autonomous loading maximizes the utility of the portable device by allowing for simple integration into many contemporary medication management systems and environments. Lastly, the portable device was designed in part to take advantage of existing technologies to solve engineering challenges not quite met by these existing technologies, as mentioned in the introduction above.

3.8. IR Sensors

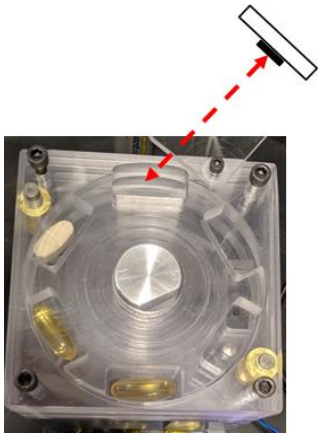

FIG. 7: The red line illustrates the line of sight of the at home unit distance sensor, which is between the sensor and a base of a loading slot of the portable device.

IR time of flight distance sensors (VL53L0CXV0DH) where implemented in the at home unit and tried in the portable device. The distance sensor of the at home unit is configured to sense a distance between the sensor and a loading slot where pills are loaded into the portable device, as shown above in FIG. 7. During loading of the portable device, the distance sensor measures a reference distance, or a distance between the sensor and a bottom of the loading slot. When a pill is loaded into the loading slot, the distance sensor measures a distance different from the reference distance, thereby indicating a pill has been loaded. Due to the timescale of the project, pills were placed by hand into the loading slot, wherein special care was made to ensure the sensor does not measure distance to the hand during demonstration of the system.

An attempt was made to implement the same distance sensor in the portable device to detect and verify a dispensed pill was truly dispensed. This would comprise of the sensor detecting a change in measured distance due to a pill falling through its line of sight. It was found that the sample rate of the sensor was too slow to detect a pill falling past it, leading to a 90% detection rate, the remaining 10% however were all false positives (i.e., the pill was dispensed but not detected). This could lead to a frustrating user experience so the sensor was not implemented in the final demonstration version of the portable device.

4. Subroutines

This section will describe the various subroutines executed by the ATMega328p (ATMega) microprocessor of the portable device. The ATMega has three primary functions to meet the high level requirements: (i) manage data including which pills are loaded into the system, their order, time to take them, and any additional dosage information; (ii) activate the motor during loading and dispensing of pills; and (iii) communicate information to the user via the LCD. All of these functions are handled by the following subroutines.

4.1 Loading Subroutine

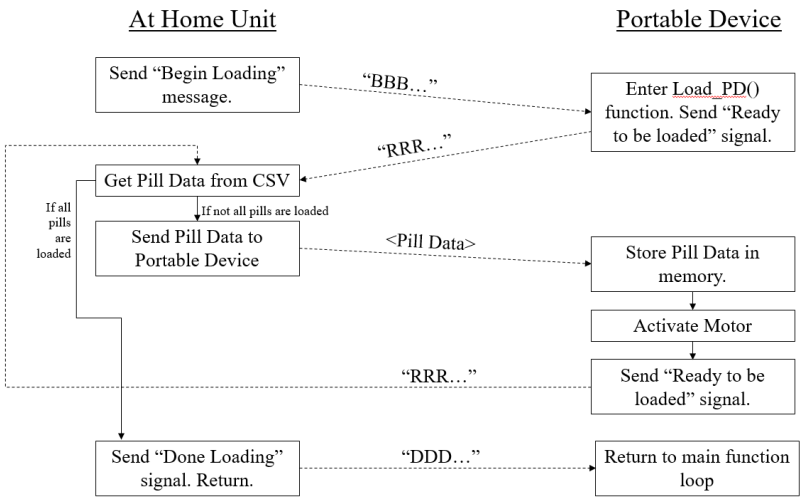

FIG. 8 – Loading subroutine flowchart illustrating the various processes involved with autonomous loading of the portable device.

The loading subroutine requires the use of all of the submodules described above, except the LCD display. The loading subroutine is invoked when the at home unit detects (i) a recognized ID of an RFID tag of a known portable device, and (ii) a user presses a "begin loading" button on the at home unit. Upon these conditions being met, the at home unit outputs a "begin loading" radio message to the portable device, comprising of a string of 31 capital B's. Upon receipt of this message, the portable device activates its motor to empty all pill contents, erases all memory contents, and emits a ready signal of 31 capital R's back to the at-home unit.

The ready signal causes the at home unit to parse a CSV file containing pill information. The Arduino of the at home unit, illustrated in the block diagram of FIG. 4 above, interfaces with a host PC via the serial I2C bus and Processing3, an open source data processing platform with a plurality of Arduino libraries and functions. The host PC simply acts as a terminal where the user may enter their prescription data into an Excel spreadsheet, which is later converted to a CSV (Comma-Separated Variable) file. If there are pills of the CSV file to be loaded into the portable device, Processing3 sends the respective pill data to the Arduino. Upon the Arduino receiving the pill data and sensing a pill is dispensed into the loading slot of the portable device, the pill data is communicated via NRF to the portable device. Receipt of this pill data causes the portable device to store the data in the SRAM memory and activate the motor, opening the loading slot for another pill. When the motor has completed its rotation, a ready signal is then communicated back to the at-home unit and the cycle repeats until all pills are loaded into the portable device.

To complete the loading process, the at-home unit sends a done signal, 31 capital D's, which causes the portable device to store a "Pills empty" message in the next memory page, shown in FIG. 8 above. Based on the number of pills loaded and number of pill slots available, the portable device activates the motor until the first pill loaded is in a location wherein the next motor activation will dispense the first pill. After this activation, both the at-home unit and portable device leave their respective loading subroutines.

As mentioned above, the NRF modules comprise an empty message bug, wherein a message of size 0 is stored in the NRF message buffer which causes the NRF module to indicate a message is available. This occasionally causes desynchronization of the at home unit and portable device as performing a string comparison with a first string, of length 0, and a second string, comprising an expected message (e.g., 31 capital R's), using the strcmp() function (a standard string comparison function in C) causes the strcmp() to output 0, indicating that character 0 of the first string is different.

Unfortunately, this is the same output when zero characters are different. Additionally, the automatic handshaking and verification of sent/received messages performed by the NRF modules were empirically found to fail regularly, i.e., the NRF module would indicate a message was received successfully, even when it was not.

4.2 Idle Subroutine

The idle subroutine comprises the default state where the device spends a substantial majority of its operative lifetime. This subroutine is handled within the loop() function which is the base subroutine of the device and a default subroutine of ATMega processors. This subroutine comprises of the ATMega looping through a series of checks in the following order:

1. Check for radio messages from the at home unit in the NRF message buffer. If no begin loading signal is found, move to step 2.
2. Check the memory and read the next pill's data.
3. Decode the pill information, including the time and dosage information data.
4. Compare the time data with an output from the real time clock. If the times match or if current time is later than the pill time, enter the dispense subroutine. Else, repeat steps 1-4.

4.3 Dispense Subroutine

The dispense subroutine is entered when a time stored in SRAM memory, comprising a time to take a pill, matches or is exceeded by a time output from the real time clock. This subroutine, when called, saves the current time and adds 30 minutes to the current time, the current time being measured using the millis() function which provides a number of milliseconds since the device was turned on.

During this subroutine, the ATMega displays a "Pill Ready!" message to the LCD and awaits the user to press the dispense button. Provided the user presses the button within the 30 minute window, the motor will activate to dispense the pill and the dosage information will be displayed to the LCD. This button press is handled by an interrupt service routine (ISR) as the ATMega is preoccupied with measuring the current time for enforcing the 30 minute window.

If the button is not pressed within the 30 minute window, a "Pill Missed!" message will be displayed on the LCD, prompting the user to press the dispense button twice. In doing so, the user inputs a form of acknowledgement that the medication is taken late and should not be taken. Future work would comprise of making this window optional and/or of variable length for specific medications.

4.4 Encoding and Decoding of Prescription Data

Prior to transmission, all pill data is encoded, which is subsequently decoded when later read from the 23k256 memory by the ATMega of the portable device. Due to the fact that each transmission packet from the NRF is 32 Bytes, and that each pill must be constrained to a single transmission, the encoding must be as short as possible. For each encoding byte, the maximum custom message character length is also reduced.

The minimum number of encoding bytes is 3, including the hour and minute of the pill, as well as 5 preset bits that are used to display common pill restrictions to be displayed by the LCD during a pill dispensing. Due to how the real time outputs time, the time of the pill is separated into 10 hour, 1 hour, 10 minute, and 1 minute bins, split between each of the three encoding bytes. The five preset bits include a 1 bit MAOI restriction, a 1 bit alcohol restriction, a 1 bit water restriction, and a 2 bit food restriction. The food restriction requires two bits due to including the possibility of a "don't care" state, in which there is no requirement specifically for food. The maximum length of each useful byte is only 7 bits, rather than 8 bits - this is because all of the characters that are being transmitted are in ASCII format. ASCII code is a representation of 7 bits, with each bit representing a basic character. In order to encode the 3 bytes as

readable ASCII characters, the last bit must be padded to a 1. The custom text message is reduced to 29 characters which are also ASCII characters. Once this completed packet is concatenated, the message is sent.

The message is received by the NRF within the portable device, at which point the entire message is decoded. The custom message is taken and stored, leaving only the encoding bytes. The encoding bytes are then run through a bitmask, in which each bit or set of bits is filtered and analyzed. The time bits are converted to a string of time that can be used to compare directly to the output of the output of the real time clock IC on the PCB. Once the real time clock output matches a pill dispense time, the preset message bits are fed to the LCD to determine whether or not the LCD will display the common preset message. The custom text message is displayed alongside the preset messages.

5. Verification of Submodules and Subroutines

This section details results and measurements found when verifying the submodules and subroutines of the portable device an at home unit. Verification procedures are listed in appendix A below.

5.1. Verification of Communication Submodule

Verification of this submodule included both the RFID module and the NRF module. The RFID module was connected to an Arduino, which would serial print out the correct ID of the passive tag once the tag was brought close enough to the reader device at a distance of at most a few inches (i.e., 0-2 inches). The NRF device was tested through sending a test message to another NRF, and then receiving and subsequently printing another message to the LCD screen. Furthermore, the strength and range of the NRF were tested, showing at least a 20dB SNR at a range of 1 meter.

5.2. Verification of User Interface Submodule

The user interface module was testing through sending a message hard-coded from the ATMega328p directly to the LCD, and comparing the input versus the output. Additionally, other verification procedures repurposed the LCD screen as a means of confirming correct output, further testing the LCD's ability to display proper information. The multipurpose button was tested through cycling through stages of the LCD display message, wherein a new message would be displayed once the button was pressed. A successful screen change indicated that the button signal was correctly received by the microcontroller.

5.3. Verification of Motor, Driver, and Calibration

Requirements and verification of the motor and its power supply have changed as per machine shop request to upscale our motor. The smaller 28BYJ-48 stepper only drew about 200mA when activated which meets its requirements, whereas the larger 9V motor drew 450mA, a value which is within reason given the current draw from the small motor. Both the larger and smaller motors meet their respective requirements, as specified in table 5 in appendix A below. It is noted that the increased maximum current draw in the power requirements for the 9V motor have been assigned based on prior requirements for the smaller motor stated in the design document and reasonable estimation of an increase in current draw.

Both motors had their rotations successfully interrupted by the ATMega when the hall sensor read a minimum value, corresponding to the magnet being directly overhead. Initially, a digital hall sensor was used to draft the requirements, however it was found that an analog hall sensor provided a more accurate

calibration of the device and the requirement has since been reworded slightly from the design document to accommodate this change.

5.4. Verification of Power Submodule

Verification of this submodule was completed through directly measuring the output voltages of the two linear regulators with respect to the ground plane of the PCB. The voltage of the 3.3 Volt regulator read a voltage of 3.29 Volts when given a supply of 9 Volts, with a maximum deviation of ±0.2 Volts. The voltage of the 5 Volt regulator read a voltage of 4.98 Volts, with a maximum deviation of ±0.5 Volts. Both of these deviations meet the requirements specified within the R&V tables 4 located in Appendix A.

5.5. Verification of Memory Submodule

Verification of this submodule included the use of the LCD display to display contents written to the 23k256 memory. This submodule did not pass verification on the final PCB due to suspected trace delays between slave input/output pins and the ATMega SPI clock or improper wiring as the final PCB did not comprise a required 10kOhm resistor to 3.3V for a !HOLD pin on the 23k256. The memory module was verified to work on a perf-board coupled to an arduino by programming the arduino to write a message to the memory, read the message, and print the message to the serial monitor.

5.6. Verification of Real Time Clock Submodule

Verification of this submodule was completed on the final PCB by displaying a real time output to the LCD in “hour:minute:second” format. It was empirically observed by the LCD output that the clock was operational and accurately sensed time. Additionally, the time on the DS3231 may be set to any time in software, as shown during the final demonstration wherein the time was set two minutes prior to a first dispensed pill (at 1:27 AM) which caused the portable device to enter the dispense subroutine shortly after being loaded.

5.7. Verification of the At Home Unit (Emulated)

Verification of the at home unit was essential for the final demonstration, wherein all three requirements needed to be met to facilitate autonomous loading of the portable device. The emulated at home unit was able to communicate with the portable device using their respective NRF transceivers. The at home unit, during loading, was successfully able to detect a dispensed pill using a distance sensor, verified in the following section. Lastly, the ID of the passive tag was successfully read and stored in dynamic memory of the arduino of at home unit as a local variable. This variable is later referenced to ensure the ID detected from the RFID reader is familiar.

5.8. Verification of the IR Sensors

Verification of the IR sensors included the use of the serial plot from the arduino of the at home unit, wherein distance measured by the IR sensor as a function of time was graphed. The distance plotted was found to be proportional to the distance of a target. Setting of the threshold for detection of a pill first comprises of the IR sensor measuring a distance from the sensor to a base of a loading slot, as shown in FIG. 7 above. This distance is the reference distance. Upon a pill being placed in the loading slot, the IR sensor measures a distance which deviates from the reference by a threshold value. The threshold value was set to either 80% of the reference distance or if the difference was equal to at least 2 cm. These two thresholds together enable the IR sensor to detect a dispensed pill from a short range (e.g., 5 cm or less) and longer (e.g., 10 cm) distances, thereby allowing for small variations in how the user places a portable device on the at home unit to be loaded.

5.9. Verification of the ATMega328p Subroutines

The ATMega328p microcontroller (ATMega) on the PCB drew about 16mA alone, which is nearly exactly its specified value of 15 mA [8]. As will be discussed below in section 6.1, system integration onto the PCB posed challenges. Individual submodules were unit tested on the PCB using the verification procedures discussed above, wherein the 23k256 did not meet requirements.

The perfboard prototype circuit was functional and met the requirements specified in table 9 below. Upon receiving the "begin loading" signal, depicted in FIG. 8 above, the portable device successfully entered the loading subroutine, thereby indicating proper functioning of the ATMega. Similarly, when a time in the SRAM memory matches an output of the real time clock, the ATMega successfully displayed a "Pill Ready!" message to the LCD. Initially, the ATMega was required to sample an RF signal from the radio transceiver, however the nRF24L01 handles the sampling and processing of radio messages and provides the messages instead using the serial-parallel interface (SPI). Lastly, as mentioned above, the ATMega was successfully able to print outputs to the LCD screen, thereby meeting its final requirement.

6. Conclusions

6.1. System Integration Issues

While the portable device has been completed on a perfboard with use of breakout modules and an Arduino Uno, the PCB layout of the device was not as successful.

One of the first major issues on the PCB is the fact that the SRAM is unable to properly communicate with the ATMega328p. When the microcontroller writes to the SRAM, the data that is read back is not the same. In this case, the problem is either through incorrect trace positioning or differences between the ATMega328p and the Arduino Uno. Either a redesign of the PCB or changing out the SRAM could resolve these issues.

Another issue encountered during testing is that the LCD is not isolated from the motor. Due to the large inrush current of the motor, the LCD backlight may flicker. This problem results from the motor not being isolated properly from the linear regulators. Simply inserting a buck regulator or an isolation circuit between the main 9 Volt power rail and the motor should eliminate the issue.

A critical issue during the final stages of testing of the PCB includes the inability to program the ATMega328p, seemingly at random. While using the Arduino as an ISP to program onto the ATMega328p, the ATMega328p can at times become unresponsive due to a corrupted device address. Once this occurs, the microcontroller must either have its fuse bits reset, or a new microcontroller must be used.

6.2. Accomplishments

The device was able to be successfully loaded, meeting the first high level requirement, and display pill information to a user when pills are dispensed, meeting the second high level requirement. The mechanical design, however, caused small pills with a minor axis of 1 cm or less to occasionally fall through to the lower geared layer one slot ahead of where it is intended to be. Additionally, larger pills occasionally got jammed when falling from the first geared layer to the second, lower geared layer. This was greatly improved by simply slowing down the motor, however improvements may be made to further reduce pill jamming and improve actuation of smaller pills.

6.3. Final List of Parts and Cost

Table 1 below illustrates a final list of electrical components and their cost for the portable device only, which totals to about $52.17. It is noted that the 17HS13 9V stepper motor comprises a substantial portion of the cost but may be downgraded to cheaper, smaller motors (e.g., 28BYJ-48 for $5) for a final product, which would bring the total cost to $21.27.

The total cost for the RFID reader and IR distance sensor of the at home unit where $13 retail, but would cost under $5 in bulk and if breakout board modules are not used.

Part	Manufacturing	Description	Quantity	Retail Cost	Bulk Cost
AP2210K	Texas Instruments	5V linear regulator	1	0.41	0.10
LM1117	Texas Instruments	3.3V linear regulator	1	1.10	0.44
ATMega328P	Atmel	Atmel 8-bit AVR Microcontroller IC chip	1	1.26	1.22
DS3231	Texas Instruments	24 Hour real time clock IC chip	1	7.26	3.66
23K256	Texas Instruments	32kB SRAM IC chip	1	1.06	1.02
L293D	Texas Instruments	4 Half-Bridge Motor Driver IC chip	1	3.91	1.83
LCD-16X2Y	HiLetGo	16x2 character LCD screen	1	11.95	3.32
I2C TWI 1602	SunFounder	Parallel Interface to I2C converter for 16x2 LCD Screen	1	1.60	0.65
17HS13-0404S-PG5	OSM Technology Co. Lt.	Bipolar Stepper Motor	1	35.90	35.90
NRF24L01+	HiLetGo	2.4GHz RF transceiver module	2	3.64	1.73
Passives	N/A	Capacitors, resistors, inductors	9	0.10	0.02
Passives	N/A	16MHz Quartz Crystal	1	0.24	0.12
Passives	N/A	Tactile Button Switch	1	0.04	0.01
PCB	PCBWay	Printed Circuit Board	1	1.00	0.26
Total Cost					52.17

Table 1: Finalized list of parts and cost.

6.4 - Ethical Considerations

In addition to the IEEE Code of Ethics [15] codes cited in the design document, patient privacy was a primary concern in developing the communications subsystem. Accordingly, all wireless transmissions were sent using near field radio with a maximum range of about 2 meters and device specific addresses. Further, drug names are never communicated verbatim, further enhancing patient confidentiality. Additionally, the mechanical design fully encloses each pill and ensures no contact is made, preventing any external or cross contamination of medication. Lastly, cost of the device was considered. As discussed above in section 6.3, a portable device may be manufactured for under $22, making it an economic option for users experiencing restrictions from at home dispensers or for larger scale applications to integrate into existing systems.

6.5 - Future Work

The most important feature to implement next would be manual loading of the device. This would eliminate the need for an at-home unit which may be expensive. Further, not all potential users of this device may desire a large at-home unit but may instead use the portable device for short term medication management (e.g., once per week, month, etc.). Manual loading may further include the use of a phone app or other means for providing pill data for pills being loaded into the portable device.

Power management leaves much to be desired as the battery lifetime of the device requires substantial optimization in future revisions. Improvements to the battery lifetime may include the use of a lower voltage (e.g., 1.5V), higher capacity, rechargeable battery and buck boost converters, as described in section 3.4. Additionally, disabling the L239D motor driver and LCD backlight when not in use would substantially increase the battery lifetime. Recharging of the battery may be performed by, for example, an inductive charger disposed within an at-home unit.

The mechanical design may be downsized substantially using the turnstile design depicted in FIG. 3 above. This would in turn reduce the weight of the device which further reduces the size of the motor, both would greatly enhance the portability and improve battery lifetime of the device.

As mentioned above in section 3.8, the IR sensors proved insufficient in detecting dispensed pills. Other means for sensing a dispensed pill may include the use of touch sensors in the base of the device or may become trivial with revisions to the mechanical design allowing for other sensors to be disposed within the portable device.

The simplicity of the at home unit demonstrated above illustrates that a portable pill management device may be easily integrated into many existing systems. This allows for potential large scale applications of the portable device, in addition to being a solution to consumer level medication dispensers. For example, a single worker in a hospital or elder care facility may provide many people with a portable, autonomous pill management device loaded from a single dispenser.

Citations

[1] TheSeniorList.com. (2017). 'Automated Medication Dispensers: Pill Dispensers for Seniors'. [online]. Available at: https://www.theseniorlist.com/medication/dispensers/ [Accessed 29 Aug. 2019].

[2] Capsuleconnection.com. (2013). 'Capsule Sizes'. [online] Available at: https://capsuleconnection.com/capsule-sizing-info/ [Accessed 4 Sep. 2019].

[3] Nordic Semiconductors. 'nRF24L01+ Single Chip 2.4GHz Transceiver Product Specification v1.0'. [online]. Available at: https://www.nordicsemi.com/-/media/DocLib/Other/Product_Spec/nRF24L01PPSv10.pdf [Accessed 9 Sep. 2019].

[4] T. Pattnayak. (2014). "Antenna Design and RF Layout Guidelines". [online]. Available at: https://www.cypress.com/file/136236/download [Accessed 21 Oct. 2019].

[5] DroneBot Workshop. (2018). "Stepper Motors with Arduino - Getting Started With Steppers". [online] Available at: https://dronebotworkshop.com/stepper-motors-with-arduino/ [Accessed 9, Sep. 2019].

[6] DroneBot Workshop. (2018). "Controlling DC Motors with the L298N Dual H-Bridge and an Arduino". [online]. Available at: https://dronebotworkshop.com/dc-motors-l298n-h-bridge/ [Accessed 9, Sep. 2019]

[7] Rinky Dink Electronics. (2019). "Library: DS3231". [online]. Available at: http://www.rinkydinkelectronics.com/library.php?id=73[Accessed 9 Oct. 2019].

[8] Atmel. (2015). "8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash". [online]. Available at: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf [Accessed 12 Sep. 2019]

[9] Nature Made, "Fish Oil 1200 mg," *Fish Oil 1200mg Supplement | Nature Made®*. [Online]. Available at:http://www.naturemade.com/fish-oil-and-omegas/fish-oil-and-omega-3/fish-oil-1200-mg#44o75sHPEUOTITLl.97 [Accessed: 11-Dec-2019].

[10] DroneBot Workshop. (2019). 'Stepper Motor with Hall Effect Limit and Homing Switches'. [online]. Available at: https://dronebotworkshop.com/stepper-motor-hall-effect/ [Accessed 28 Sep. 2019].

[11] DroneBot Workshop. (2018). 'Controlling DC Motors with the L298N Dual H-Bridge and an Arduino'. [online]. Available at: https://dronebotworkshop.com/dc-motors-l298n-h-bridge/ [Accessed 9, Sep. 2019].

[12] Eagle Library Project. Diyproject. (2017). 'nRF24L01+ Module". [online]. Available at: https://www.diymodules.org/eagle-show-object?type=dm&file=diy-modules.lbr&device=WIRELESS-NRF24L01 [Accessed 2 Oct. 2019].

[13] DroneBot Workshop. (2019). 'Powering your Projects'. [online]. Available at: https://dronebotworkshop.com/powering-your-projects/ [Accessed 28 Sep. 2019].

[14] DroneBot Workshop. (2019). 'Using LCD Displays with Arduino'. [online]. Available at: https://dronebotworkshop.com/lcd-displays-arduino/ [Accessed 28 Sep. 2019].

[15] IEEE.org. (2016). 'IEEE IEEE Code of Ethics'. [online]. Available at: https://www.ieee.org/about/corporate/governance/p7-8.html [Accessed 17 Sep. 2019].

Appendix A - Requirement and Verification Tables

Requirements	Verification
1. Portable device may send a wireless signal to at-home unit at a distance between 0 and 0.1 m with Signal-to-Noise ratio greater than at least 20 dB.	a. Ensure both RF Tx/Rx circuits are configured on the same carrier frequencies and the respective circuits are in the correct "Rx" or "Tx" modes. b. Send a "Dummy" signal to the arduino (e.g., a sine wave). Verify the signal is being emitted using Spectrum Analyzer. SNR of 20 dB minimum at 0.1 meter. c. Either configure the arduino to output the Dummy signal to an oscilloscope or store samples of signal in memory.
2. Transmission of data from emulated "at-home" unit to portable device less than 10 dB attenuation across 0.1 m transmission.	a. Connect transmitter, receiver to spectrum analyzer on separate channels b. Send "Dummy" signal from the arduino to the portable device. The Dummy signal will comprise of some string of data or sine wave. c. Ensure the ATMega328 samples the received signal at at least the nyquist rate and configures the RF Tx/Rx block of the portable device into Rx mode, and less than 10 dB attenuation between sent and received signal. d. Ensure ATMega328 stores sampled data in its flash memory through random sampling via PC.

Table 2: Requirements and Verification for the nRF24L01 RF Transceiver.

Requirements	Verification
1. Transponder outputs an AM signal. The AM signal induces a return signal from the passive RFID tag (RF37S114)	a. Measure input impedance over antenna terminals (should be 50 ohm). b. Design impedance matching circuit between antenna terminals and antenna. c. Connect antenna to impedance matching circuit. Connect TRF7960 to arduino following FIG. 4 above. d. Place the circuit and arduino near a vector network analyzer (VNA). Observe RF output. The RF output should be an AM Sine wave if configured correctly.
2. Transponder receives data from RFID tag at a rage of 1 foot *at most*.	a. The RF37S114 is a purely passive device. Read output from RX_IN pin or serial output from SPI of TRF7960. RX_IN trace can be probed using oscilloscope to verify that a return signal is being detected. b. Move the tag increasingly far from the TRF7960. Determine maximum range (3 feet max). Reduce power to antenna if range exceeds 3 feet.

Table 3: Requirements and verification for RFID module.

Requirements	Verification
1. Linear regulator must be within ± 0.5 volts of 5.0 Volts or ± 0.3 volts of 3.3 Volts.	a. Add resistor divider to reference pin of linear regulator. Vout = Vin(1+ R1/R2); R2 being connected between Ref and GND. R1 between Vout and Ref. b. Measure output voltage. Output voltage must be within ±0.5V of 5V or within ±0.3V of 3.3V. If not within tolerance, adjust resistor values and remeasure.
2. Linear voltage regulator output remains within ± 0.5 volts during motor operation.	a. Connect buck and linear regulators in parallel. Connect motor controller to buck regulator output. Connect both regulators to a function generator. Ensure outputs are at correct voltage levels. b. Probe output voltage of the linear regulator (Open circuit) c. Activate the motor, ensure linear regulator voltage does not deviate from 5v by more than 0.5v. d. Add a resistive load to linear regulator. e. Repeat step C. Probe over the resistive load. f. In part (e), ensure that (i) voltage level does not deviate from 5v by more than 0.5v, and (ii) current drawn by the resistive load does not substantially change. g. Repeat above steps (a-f) with battery as power source. h. If (c) and (f) fail to meet requirements, replace linear regulator with buck regulator
3. Ensure maximum current cannot exceed 300 mA over 50 ohm load.	a. Probe and measure voltage and current over resistance of a 50 ohm test load.

Table 4: Requirements and verification for power management module.

Requirements	Verification
1. Turn the 28BYJ-48 stepper motor using a ULN2003 driver with current draw less than 300mA per coil when powered at 6V. NEW: Turn Bipolar Stepper motor (17HS13-0404S-PG5) using L293D H-Bridge Driver. Maximum current of 500mA per coil when powered at 9V.	a. Connect arduino pins 8-11 to pins IN1-IN4. b. Power the arduino using 5V from a DC source and motor using the buck regulator (IMPORTANT: Dont couple arduino 5V to motor VCC!). Connect 9V supply to buck regulator. c. Download sample code [13]. Tune parameters such as step speed to a suitable rate (5-10 RPM). d. Execute the sample code, motor should turn. e. Measure current drawn by the buck regulator using a voltage over known resistance in series with the Buck regulator. f. Modify sample code to stop the motor after a 90 degree rotation (i.e., after N steps, N = (step size)/90).
2. Interrupt when Hall sensor activates, stopping motor upon the hall sensor reading a threshold value.	a. Connect DRV5053 Hall sensor to arduino. Use arduino 5V output as VCC and analog pin as input. b. Use an alligator clip (or similar) to attach a magnet to the motor shaft. Place hall sensor in a correct orientation (see data sheet) level with the magnet. c. Activate the motor, read analog value from serial monitor. Should observe a spike as the magnet is moved close to the hall sensor. d. Stop the motor. Position the magnet as close to the hall sensor as possible. This will establish a reference 0 degrees. e. Measure the peak analog output from the hall sensor. This will correspond to the magnet being at position 0 degrees. f. Software interrupt to motor step function upon the analog input from the hall sensor reaching a threshold value. The value being based on the value measured at step (e). Use a stopwatch to measure between when interrupt activates, and when motor ceased movement. Note: Set threshold value to be 90% of maximum value if Hall becomes saturated.

Table 5: Requirements and Verification for the Motor Control module.

Requirements	Verification
1. Print to monitor "Test Message"	a. Download sample code [14] onto arduino. b. Connect SDA (A4) and SCL (A5) to SDA and SCL inputs of the I2C adapter. The I2C adapter being connected to the LCD. c. Execute code.
2. Modify message based on dosage information.	a. Receive an 8 bit string (test vectors): [B7-B5] - Pill Number [B4] - Take with water [B3] - Take with food [B2] - Pill Missed (Warning) [B1-B0] - Extra dosage info (TBD). b. Determine locations for sprites on LCD. Sprites include, e.g., "Food" symbol or text which indicates pill should be taken with food. c. Based on the 8 bit string provide input to the print function (see step 1) to produce correct user information to the LCD.

Table 6: Requirements and verification for user interface module.

Requirements	Verification
1. Detect output from sensor and determine reference.	a. Couple sensor to an arduino, display serial monitor, measure output. b. Orient the IR sensor towards a reference surface. The distance measured will be the reference distance. c. Drop a pill in between the reference surface and the sensor. Sensor sampling rate should be sufficient to detect a decrease in distance equal to a value between 0 and the reference distance.. Note: The exact value will be dependant on the physical design of the portable device. To be determined as machine shop progresses.
2. Set detection threshold. Detect pills being dispensed upon this threshold being breached.	a. Based on the minimum reading from part 1(c) and the reference measurement, determine a detection threshold. Threshold should be sufficiently large that noise will not trigger a false positive. b. Drop at least 5 pills in front of the sensor from various heights. Determine limits of detection for the speed of an object. c. Drop at least 5 pills of differing sizes. Determine limits of detection for size of pill. d. Adjust threshold and/or position of the sensor until steps (b-c) are satisfied.

Table 7: Requirements and verification for pill dispenser sensors for both the at-home unit and portable device.

Requirements	Verification
1. Able to send and receive data packet of length 32 bytes to and from the portable device.	a. Meet requirements and verification of nRF24L01. See table 1.
2. Detect that one pill is dispensed at a time.	a. See Table 6. b. Communicate this to portable device ATMega328. If motor control, power units, and nRF24L01 requirements are met, this should activate the motor.
3. Receive a unique ID string of 8 bytes from RFID tag and store in memory of Arduino.	a. Verify RFID reader provides output when tag is within proximity b. Ensure the ID value is correctly stored in registers and/or memory of the Arduino. Short code can be written to output the ID on a terminal.

Table 8: Requirements and verification for the emulated at home unit.

Requirements	Verification
1. Impedance matched ATMega328P interface shows loss of less than 10% of power (>90% power transmission).	a. Supply power from voltage regulator. Ensure input impedance is 50 ohm. If it is not 50 ohm, design an impedance matching circuit. b. Correlate output signals to pins on the ATMega328 to be used for later testing.
2. Loading states/logic send high (2.5-5V) and low (0-1.5V) outputs during specified states - states transition correctly.	This step requires RF Tx/Rx R&V to be met. a. Program ATMega328 to embody a state machine, as described in FIG. 7 above. b. Enter the Preload state by communicating, via RF Tx/Rx blocks, aPill Load = 1 signal in response to the arduino detecting the RFID of the portable device. c. Enter Load1 state. Trigger the photosensor to detect a dropped pill, e.g., using hand to block light, causing the state machine to move to Rotate state. d. Ensure motor rotates 90 degrees and stops. Probe power output to ensure R&V of the voltage regulator is still met. Ensure proper stepping of the motor. e. Ensure proper state transitions from Load2 back to Load1 if more pills are to be loaded. Else, return to idle.
3. Dispense states/logic send high (2.5-5V) and low (0-1.5V) outputs during specified states - states transition correctly.	a. Program flash memory of ATMega328 with pills to be taken. Space dosages out by 1 minute starting at 12:01 AM (0:00 UTC). Do not add timeouts for missed pills. b. Reset 24:00 RTClock to 0:00 by providing a 5V input to a RST pin. c. Verify signal is being sent to LCD display using oscilloscope. The signal may be later verified as correct following the user interface R&V table below. d. Press “Dispense Pill” Button. ATMega328 should move to the Dispense state, rotate the motor 90 degrees, and return to Idle. Repeat this process multiple times. e. Reset UTC clock and reload Flash memory with 1 pill per two minutes starting at 12:01 AM. Additionally, program flash memory to also include timeouts of 30 seconds per pill. f. After one minute and thirty seconds, verify the state machine has moved to Missed Pill state by verifying LCD display output signal using oscilloscope. g. Press “Dispense Pill” button once. This should not do anything. Press the button again or twice rapidly to cause the motor to activate in the Dispense state.
4. Sample signals less than 4MHz of ATMega328P processing speed, with less than less than 20 dB attenuation across 0.1 m transmission.	Ensure Communications R&V are met before this stage. a. Ensure the CLK speed of the ATMega328 is equal to or greater than a Nyquist sampling rate of our signal. b. ATMega328 properly reads values from a hardware register or logical high (5v) input to one or more pins of the ATMega328. c. Emit “Dummy” signal from the arduino, e.g., sine wave. The output of the RF Tx/Rx block of the portable device should register as a “1” or sinusoidal output. d. Sample the dummy signal received by the RF Tx/Rx block and store the data in flash memory of the ATMega328. e. Verify data is properly communicated with no errors. Output the received data in software and verify it matches the Dummy signal.
5. LCD UI displays human-readable text	a. Ensure ATMega328 detects button presses from the “Dispense Pill” button. b. Program default messages such as “Pill Ready” or “Pill Missed” into flash

(by visual inspection).	memory of ATMega328. c. Encode the messages based on formatting requirements of LCD display. d. Output the messages to the LCD display, verify the messages are displayed properly.

Table 9: Requirements and verification for the ATMega328 microprocessor block of the block diagram illustrated in FIG. 1 above.

Appendix B Portable Device Circuit

The PCB below comprises a size of 5.95 by 5.16 cm, making it sizeable for a portable device.

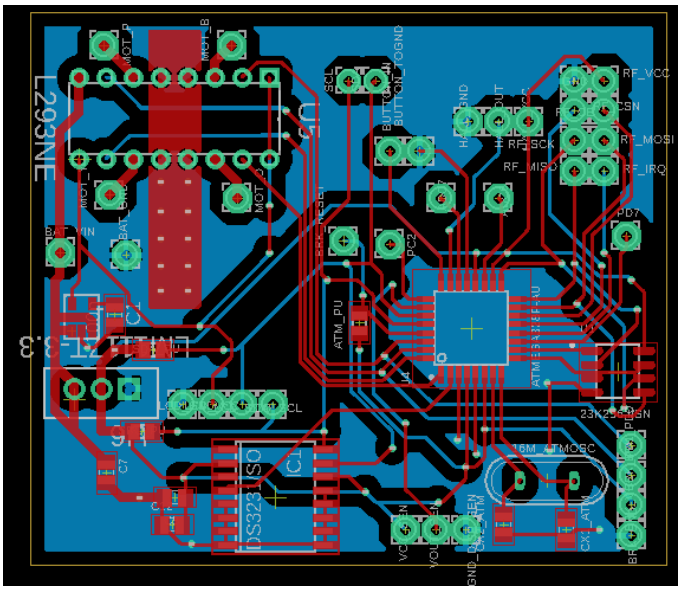

FIG 9: Portable Device board layout

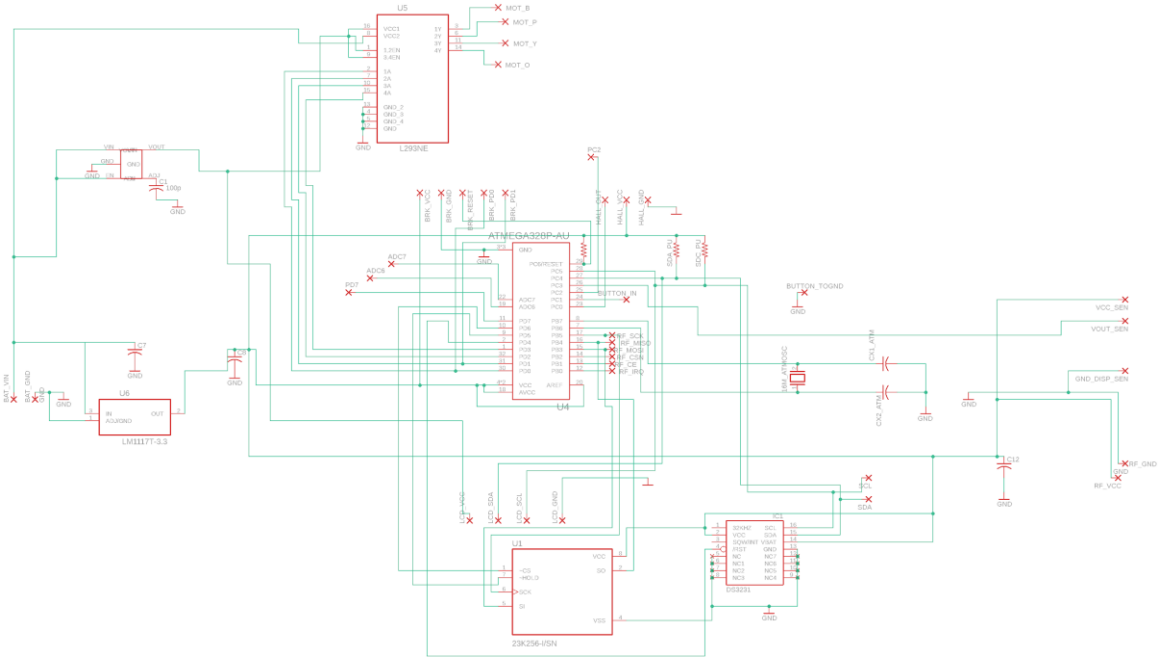

FIG 10: Portable Device circuit