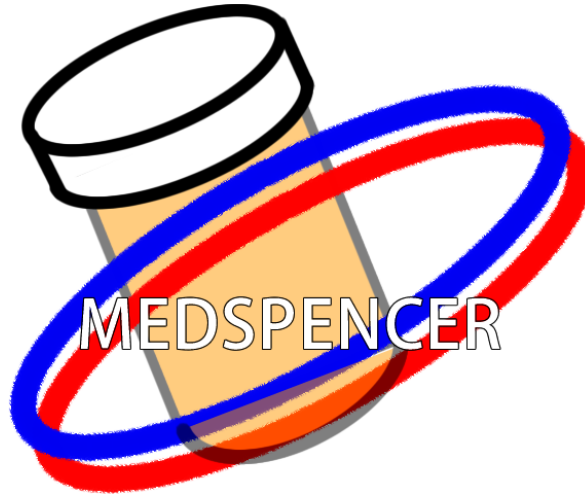


UCF Senior Design II

The Medspencer

A Smart Medicine Dispenser that Dispenses Medicine on Fixed-Schedule or As-Needed Basis, for Individual or Family use



Department of Electrical Engineering and Computer Science

University of Central Florida

Dr. Samuel Richie and Dr. Lei Wei

Summer 2018

Senior Design 2 Document

Group 6

Ivan Alvarez	Computer & Electrical	ialvarez5@knights.ucf.edu
Matthew Hoover	Computer Engineer	matthewhoover@knights.ucf.edu
Sakeenah Khan	Electrical Engineer	sakeenahkhan@knights.ucf.edu
Gustavo Morales	Electrical Engineer	gmoralesburbano@knights.ucf.edu

Table of Contents

List of Figures	vi
List of Tables	ix
1.0 Executive Summary	1
2.0 Project Description.....	2
2.1 Motivations and Goals	2
2.2 Objectives	3
2.3 Requirements Specification	4
2.4 House of Quality.....	5
3.0 Related Standards and Constraints.....	6
3.1 Project Related Stands	6
3.1.1 IEEE Standards	6
3.1.2 FDA Standards.....	7
3.1.3 Soldering Standards and Safety	9
3.2 Design Constraints	11
3.2.1 Economic Constraints	11
3.2.2 Time Constraints	12
3.2.3 Manufacturability and Sustainability Constraints.....	13
3.2.4 Parts, Component, and Testing Constraints.....	14
3.2.5 Environmental Constraints.....	16
3.2.6 Health and Safety Constraints.....	17
3.2.7 Ethical and Legal Constraints	19
3.2.8 Social and Cultural Constraints	21
3.2.9 Political Constraints or FDA Standards.....	22
3.2.10 Security Constraints	25

3.2.11 Performance, Functionality, Usability, Reliability and Availability	27
4.0 Research and Background Information	28
4.1 Market Research	28
4.1.1 Existing Products	28
4.1.2 Input from Medical Professionals	30
4.2 Microcontroller	32
4.2.1 Research.....	33
4.2.2 Microchip PIC32MZ DA.....	34
4.2.3 Microchip PIC32MZ DA Starter Kit	36
4.2.4 MPLAB Harmony.....	37
4.2.5 ATMEGA328P Microcontroller	40
4.2.6 Raspberry Pi Compute Module 3 Lite	40
4.3 Display	40
4.3.1 Technical Background	41
4.3.2 Frame Buffer Space Saving Techniques	41
4.3.3 Innolux AT070TN90	43
4.3.4 Resistive Touch Screen.....	44
4.3.5 Resistive Touch Screen Controller	46
4.4 Wi-Fi connection	47
4.5 Fingerprint Scanner.....	48
4.6 Motors	49
4.7 Speaker.....	50
4.8 Power	51
4.8.1 Power Supply.....	51
4.8.2 Power Regulation and Filtering	52

4.9 Communication Protocols.....	52
5.0 Project Hardware and Software Design Details.....	56
5.1 Hardware Design	56
5.1.1 Hardware Block Diagram	56
5.1.2 Hardware Design Overview.....	58
5.1.3 Microcontroller Design.....	60
5.1.4 Wi-Fi Module.....	64
5.1.5 Display Design.....	65
5.1.6 Fingerprint Scanner.....	68
5.1.7 Dispensing Mechanism.....	69
5.1.8 Speaker.....	71
5.1.9 Power Supply	72
5.1.10 Power Regulation and Filtering	73
5.2 Software Design.....	79
5.2.1 ATmega328P Software	80
5.2.2 User Interface.....	81
5.2.3 SQLite	82
5.2.4 Wi-Fi Communication	82
5.2.5 Fingerprint Scanner.....	83
5.2.6 Patient Identification & Adding/Removing a Patient	88
5.2.7 Prescription Parameters and Schedules	89
5.2.8 As-needed Medication Requests.....	90
5.2.9 Fixed-schedule Medicine Notifications and Requests.....	90
5.2.10 Rescheduling Doses.....	90
5.2.11 Recording events in Schedule & Summary Reports.....	91

5.2.12 Python Structure.....	91
6.0 Prototype Construction	95
6.1 Parts Acquisition and Bill of Materials.....	95
6.2 Printed Circuit Board	96
6.3 Prototype Constraints.....	96
6.4 Issues and Challenges	97
6.4.1 Hardware Issues.....	97
6.4.2 Software Issues	98
7.0 Prototype Testing.....	99
7.1 Hardware Testing.....	99
7.1.1 Breadboard and Development Board.....	99
7.1.2 PCB Testing	99
7.1.3 Testing Procedure	100
7.1.4 Microcontroller Testing.....	101
7.1.5 Touchscreen Testing	104
7.1.6 Display Testing	106
7.1.7 Wi-Fi Module Testing.....	108
7.1.8 Fingerprint Module Testing.....	108
7.1.9 Power Supply Testing.....	108
7.2 Software Testing.....	109
7.2.1 IDE Used.....	109
7.2.2 Python Procedures, Simulation, and Physical Testing.....	110
7.2.3 ATMEGA328P Programming Procedures	112
8.0 Administrative Content.....	113
8.1 Milestone Discussion.....	113

8.2 Division of Labor	116
8.3 Budget and Finance Discussion	118
8.4 Project Operation	120
9.0 Conclusion	122
Appendix A: Citations	123
Appendix B: Copyright Permissions	129

List of Figures

Figure 1. House of Quality.....	5
Figure 2. Cooling fan system with no maintenance.....	14
Figure 3. Wi-Fi module ESP8266.....	15
Figure 4. Precise medication window scheme used by e-pill, LLC.....	29
Figure 5. PIC32MZ DA Family Block Diagram	35
Figure 6. J15 40-pin expansion.....	37
Figure 7. MPLAB Harmony generic block diagram	38
Figure 8. MPLAB Harmony Middleware diagram.....	39
Figure 9. MPLAB Harmony System Service example.....	39
Figure 10. Basic 24-bit color frame buffer; Total size: 1.099MB	42
Figure 11. 16-bit color frame buffer; Total size: 0.732 MB	42
Figure 12. Frame buffer with 256-color lookup table; Total size: 0.366 MB	42
Figure 13. Resistive touchscreen’s layers and electrodes.....	45
Figure 14. Measuring the Y component of a touch	46
Figure 15. AR1021 Resistive Touch Screen Controller Courtesy of Microchip.....	47
Figure 16. Filtered full wave rectifier	51
Figure 17. SPI communication using multiple slaves connected in parallel	53
Figure 18. UART communication connections	54
Figure 19. Transfer of data packets using UART communication.....	54
Figure 20. I2C communication connections using multiple masters and multiple slaves	55
Figure 21. Data message organization for I2C communication	55
Figure 22. Hardware block diagram showing team member responsibilities.....	57
Figure 23. Hardware block diagram	57
Figure 24. Main PCB schematic	59

Figure 25. Main PCB layout	59
Figure 26. Raspberry Pi Compute Module 3 Lite (CM3L)	61
Figure 27. Raspberry Pi CM3L schematic.....	62
Figure 28. ATMEGA328P-PU microcontroller	62
Figure 29. ATMEGA328P-PU schematic	63
Figure 30. Bidirectional logic level shifting circuit	64
Figure 31. ESP-12F Wi-fi module	64
Figure 32. ESP-12F schematic.....	65
Figure 33. LCD display schematic.....	67
Figure 34. AR1021 schematic.....	67
Figure 35. Picture of R307 fingerprint module.....	68
Figure 36. Exterior interface of the R307 fingerprint module	69
Figure 37. CD74HC238 3-to-8 Demultiplexer	70
Figure 38. SG90 servo motor.....	70
Figure 39. Dispensing mechanism schematic.....	71
Figure 40. 2W, 4 Ω speaker	71
Figure 41. Audio amplifier schematic	72
Figure 42. 3-pin DC power barrel connector	73
Figure 43. Voltage regulation schematic	75
Figure 44. Voltage regulation PCB layout.....	75
Figure 45. 1.8V regulation schematic	76
Figure 46. -7V regulation schematic.....	76
Figure 47. 10V regulation schematic	77
Figure 48. 3.8V regulation schematic	77
Figure 49. 5V regulation schematic.....	78

Figure 50. 16V regulation schematic	78
Figure 51. 10.4V regulation schematic	79
Figure 52. 12V regulation schematic	79
Figure 53. ATmega328P Software Diagram	80
Figure 54. Data package format of R307 fingerprint module.....	85
Figure 55. Schedule Selection Screen.....	89
Figure 56. Sample prescription label	89
Figure 57. Fingerprint Frame	92
Figure 58. Patient Frame	93
Figure 59. Administrator Frame	93
Figure 60. Fingerprint Frame	110
Figure 61. Patient Screen	111
Figure 62. Administrator or Caretaker.....	111
Figure 63. Division of labor.....	117

List of Tables

Table 1. Connections for Fingerprint Reader on Arduino	15
Table 2. Connections for Servo Motors on Arduino.....	15
Table 3. Typical Operation Conditions	44
Table 4. Processor Comparison	60
Table 5. Pin Assignment.....	66
Table 6. R307 Fingerprint Module Specifications and Ratings.....	68
Table 7. Power Requirements	73
Table 8. Voltage Regulation Devices	74
Table 9. R307 Package Formats for Fingerprint Processing Instructions	86
Table 10. R307 Package Formats for System-related Instructions.....	88
Table 11. Final Budget.....	95
Table 12. Example integration test cases related to page interaction	102
Table 13. Example system test cases	103
Table 14. Test cases relating to electrical properties of the touchscreen.....	105
Table 15. Test cases relating to the software performance of the touchscreen.....	106
Table 16. Test cases relating to the LCD.....	107
Table 17. Milestone chart	115
Table 18. Work Distribution.....	118
Table 19. Updated Budget List.....	119

1.0 Executive Summary

For people with illness and health complications, it can be difficult to keep track of different medicines and each medicine's dosing schedule. It is important to follow the schedule carefully to avoid an overdose or missing a dose. It can be easy to mix up medications and consume the wrong medication as well. In addition, it is possible for prescription medicines to be consumed by the wrong person, especially when they're stored unprotected in a home medicine cabinet. Our solution to these issues is the Medspencer, a device that dispenses the right medicines to the appropriate person at the correct time. Our device will also record the patient's compliance or noncompliance with taking their medicine and relay it to the health care provider, so that they can better advise and prescribe to the patient.

The main goal of this project is to accurately schedule and track multiple individual users' medicines and dispense a single dose to a user at the appropriate times. The Medspencer would be implemented in the household, for use by the family. The goals include: (1) to help patients stick to their medicine schedules, so they don't miss a dose or mix up medicines; (2) to keep medicines secure so that patients cannot overdose, and the wrong person cannot take the medicine; and (3) to record patients' compliance/noncompliance and inform the health care provider.

The features of the Medspencer are described as follows. Each user is programmed into the Medspencer, and they can access the Medspencer by identifying themselves with a fingerprint scan. Prescription drugs can be added to the dispenser into medicine vials (added by the administrator) and can only be dispensed to the user they were prescribed to. Each medicine vial is identified by its NDC number (medicine type), patient name, prescriber information, and schedule type ('fixed schedule' vs 'as needed'). The Medspencer will notify a user when it is time to take their medicine. Users may also request 'as needed' medicines, in which case the Medspencer will check if enough time has elapsed since the last dose and whether it has negative reactions with other medicines; then it will decide whether to dispense the medicine. The Medspencer records when a user takes or misses a dose. If a dose is missed, the Medspencer will either reschedule or skip the dose, depending on the medicine's schedule type. Every two weeks, the Medspencer sends a summary report on the patient's medication compliance to the administrator and health care provider. If any medicines are running low or empty, the Medspencer will notify the administrator. The administrator can be a caretaker or head-of-household, and is responsible for putting medicines into the Medspencer, inputting prescription information (such as schedules, prescribers, etc.), and adding users and fingerprint identification. Only the administrator can open the device and access (add/remove) medicines, or add/adjust schedules, prescriptions, and user information.

The Medspencer is an excellent method to aid patients and improve their medicine compliance, as it takes care of organizing medications, scheduling them, and reminding the patient. It is a good alternative to medication trays that must be sorted by the patient and allow for human error.

2.0 Project Description

This section details the motivations, goals, and objectives for our senior design project. Once the motivation and specific objectives were identified, we could narrow down how to physically achieve said objectives and provide the described features. The project requirement specifications details the necessary requirements to feasibly complete the set objectives. At the end of this chapter, the trade-offs between engineering technology capabilities and consumer desires are illustrated in the house of quality.

2.1 Motivations and Goals

As this is a senior design project, our first motivation for the project is to demonstrate and apply the knowledge we have learned through attending the College of Engineering and Computer Science at the University of Central Florida. Completing this project will allow us to develop professionally and learn how to properly plan, manage, and execute a project while working in a team. To demonstrate our knowledge, the team decided to create the Medspencer, an automated medicine dispenser. The motivation for the Medspencer is to increase medication adherence in patients with multiple prescription medications and schedules.

Medication nonadherence, or noncompliance, is a serious issue and limits the effectiveness of health care services and prescribed medications. According to the World Health Organization's 2003 report on medication adherence, about 50% of patients with chronic illnesses do not take medications as prescribed [73]. Medication nonadherence has serious health consequences. The implications of medication nonadherence include decreased quality of life, poorly managed symptoms, and even death [22, 63, 87]. In addition, nonadherence costs the health care system over \$300 billion a year due to additional doctor visits, emergency department visits, and hospitalizations. The costs associated with nonadherence comprise up to 10% of total health care costs [63, 87]. This problem has become even more serious over the years, as doctors and pharmacists develop and prescribe increasingly complex and potent medication regimens. Research shows that in 2016, the age-adjusted rate of deaths due to overdose in the United States increased to over three times the rate in 1999 [48].

If patients are to become healthy and obtain the maximum benefit from health care services and prescriptions, then they must comply with their prescribed medication regimens. Increased support for patients and consideration of factors that affect medication compliance may help patients stick to their medication regimen. Medication adherence is associated with many factors, including social and economic, health care system related, patient related, condition related, and therapy related factors [73, 87]. Some of the specific factors that the Medspencer will attempt to target and alleviate are complexity of the medication regimen, duration of therapy, frequent prescription changes, poor health literacy, and the patient-provider relationship [87].

Our team's motivation and main goal for this project is to help patients keep to their strict medication regimens and show higher medication compliance. By creating a household

appliance that supports the patient's compliance, the patient can become more independent and will not have to rely on a caretaker to watch them or remind them all the time.

2.2 Objectives

For the Medspencer project, the main goal and objective is to increase medication compliance in patients. This goal can be broken down into several specific objectives that contribute to the main goal. First, we want to prevent patients from missing a dose or taking a dose at the wrong time. These are easy mistakes to make, especially for complex medication regimens or in the case of frequent prescription changes. To achieve this objective, the Medspencer will schedule doses and notify the patient when it is time to take the dose. In the case of a missed dose, the Medspencer will reschedule the dose. Another objective of the Medspencer is to keep patients from overdosing, whether intentionally or by accident. We also wish to prevent people from consuming other patients' prescriptions. Preventing these two events is especially important when it comes to strong prescription medications, such as opioids. To this end, the Medspencer shall be a secure device that can only be opened by an administrator/caretaker. Medications will only be dispensed to a patient if it is time for the next dose (for fixed schedule prescriptions), or if enough time has elapsed since the last dose (for as needed prescriptions). To access the device, a fingerprint scan is required. This ensures that the correct patient is getting the prescription. An additional objective is to strengthen the patient-provider relationship, by informing the primary care physician (PCP) on the patient's medication compliance. Accomplishing this will allow the PCP to obtain a better understanding of the patient's health conditions. This will ensure that the PCP can accurately diagnose health issues, prescribe medications and therapy, and advise the patient. By accomplishing these specific objectives, we believe that the Medspencer will support the patient and help them stick to their medication regimen.

Another objective of our team is to reduce the monetary and power costs associated with using an automated medication dispenser. The automated medication dispensers for the household that currently exist on the market are quite expensive, making them less feasible options for many patients. Thus, our team aims to reduce the cost of research and development so that the final cost for the consumer will be reasonable and less than the competing products on the market. We will also try to reduce the power costs associated with our device, as that is an additional cost to the consumer. To save power, we will put the device in sleep mode until an event occurs (such as a fixed-schedule dose notification, or an as-needed dose request).

We also aim to make the Medspencer as user-friendly as possible. We will utilize a touchscreen display so the user can interface and interact with the Medspencer, and we will keep the interactions simple. When it is time for a fixed-schedule dose, the Medspencer will notify the patient, and all the patient must do is scan their finger and the medicine will be dispensed. For an as-needed medicine request, the patient can scan their finger, then select from the available medicines using the touchscreen. We will also make the set-up for the Medspencer as simple as possible. When entering new medicines or new patients, we will provide a template with different sections to parameterize the new addition. The template will include preset options for the medicine type, schedule type, and so on.

2.3 Requirements Specification

“The system” refers to the Medspencer in this case.

Overview

- The system will have an expandable amount of cylinders that contain the medicine
- The system will have motors that dispense the appropriate medicine
- The system will determine patients’ identity via a fingerprint scanner
- The system will alert patients to schedule via a speaker
- The system will connect to the Internet over Wi-Fi

Physical

- The system will be no larger than 2 ft. x 1 ft. x 1 ft.
- The system will be no heavier than 20 lbs.
- The product’s cost will not exceed \$300
- The system will have a wall plug adapter
- The system will rectify and step down power to necessary voltages

Data

- The system must store the following data for each medicine:
 - Medicine type
 - Medicine time restrictions
 - Prescribing doctors’ email addresses
- The system must store the following data for each patient:
 - Patients’ name
 - Patients’ dose schedules
- The system will make records of medicine dispenses and will be able to send them via email to the appropriate doctor
- The system will support the following administrator interactions:
 - Add/edit/remove medicine
 - Add/edit/remove patient
 - Make changes to patients’ dose schedules
- The system will support the following user interactions:
 - Dispense medicine at proper time
 - Request “take anytime” medicine

Display and Touchscreen

- The system will have a screen at least 7” across
- The system will support the following human interfaces:
 - Touch screen menu navigation
 - Touch screen keyboard

2.4 House of Quality

		Marketing						Engineering Targets
Engineering		Quality	Ease of Use	Pill Capacity	Security	Installation Time	Cost	
		+	+	+	+	-	-	
Quality	+	↑↑	↑↑	↑	↑	↓	↓	
Power Usage	-	↓						< 500 W
Screen Size	+	↑	↑		↑			7" WVGA
Memory Size	+	↑	↑	↑			↓↓	> 1 MB
Dimensions	-	↓		↓↓			↓	< 2' x 1' x 1'
Cost	-	↓		↓	↓		↓↓	< \$2000

Legend	
↑	Positive Correlation
↓	Negative Correlation
↑↑	Strong Positive Correlation
↓↓	Strong Negative Correlation

Figure 1. House of Quality

3.0 Related Standards and Constraints

This section outlines the official standards and design constraints that are related to our project, the MedSpencer. While some categories do not apply to this project very much, there are a large number of standards that apply to any piece of medical hardware, as well as technological restrictions, and legal requirements.

3.1 Project Related Stands

There are several standards that we must consider as we plan out this project. Standards provide the guidelines and protocols that must be followed when managing projects and creating/implementing products. The standards must be followed during the design process to avoid scrutiny or liabilities. The standards that we have considered include IEEE standards, FDA standards, soldering standards, and the standards of programming languages we may use.

3.1.1 IEEE Standards

The Institute of Electrical and Electronics Engineers (IEEE) has the duty to regulate and standardize many proposal and laws for better performance and usage of technology within the technology industry. From the different sorts of connections that every domestic or industrial project should use, to the proper documentation of tools and language used and interconnection between systems, IEEE gives us the easiest and most effective way to perform any engineering duty. The following IEEE standards were considered relevant for the project.

IEEE 1588 Precision time Protocol (PTP) synchronizes clocks in a Local Area Network (LAN). Its accuracy is adjusted to microseconds making it useful for control systems and measures [36]. This is useful in our project due to the fact that MedSpencer will be connected to a Wi-Fi module to for constantly update the clock to alert the patient about his or her medication. The dispenser compares the current time with the time saved in the patient's profile for the scheduled medication.

IEEE P1619 Security in Storage Working Group (SISWG) [50] refers to the protection of stored data and its corresponding cryptographic key management. This standard talks about how the information within should be encrypted and decrypted for its use and not so easy to decode for wrong purposes. There was a discussion about how the data will be managed, by narrow-block or wide-block, resulting the narrow-block the most effective way to operate information.

IEEE 830 software requirement specification [34] states all the requirement needed for a project or product that uses a customized software. This standard establishes some agreements between the customers and developers in order to fulfill the criteria and avoid redesign. As our project will require its own software, the computer engineer in charge will meet with the sponsor to discuss further details about the limitations and advantages of the software to-be.

IEEE 11073 Medical Health device communication standard enables the communication between a health-care device with an external computer. It provides detailed information of the client's health or device's operational data [55]. This standard is extremely useful for the MedSpencer due to it will generate weekly reports that will be sent to the authorized doctor informing the patient's compliance. Also, let us remember that there will be an administrator that will take care of the device maintenance, when the medications are not enough for the rest of the month or to add or remove patients from the device memory. This administrator will also receive alerts concerning the device condition.

IEEE 829 Software test documentation [51] states the procedure for testing not only the software but the system too through a system of eight stages, each one of these will create its own document that goes from master test plan to master test report. [51] The test plan will be carried away by our computer engineering team. They better than anyone knows how the software will run. They created the flowchart that also gives a rough scope of the system's behavior.

IEEE 1016 Software Design Description [52] is a document that describes the software behavior and features in order to inform the software developer for better understanding. The MedSpencer will require a very detailed description because of its various inputs and cases. The test that will be run on this software will be done according to the description document created beforehand.

IEEE 1149.1 Joint Test Action Group (JTAG) [54] is the standard that is defined for verification of proper connectivity within the printed board circuit (PCB). It plays a complementary role with the digital simulation. It implements a serial communication interface, that connects to an on-board test access port that runs through a series of test registers to present chip logic levels and device capabilities. This standard is not only useful but fundamental, because is the main test of the main board after the simulation has been run and approved. This standard will be implemented to each individual PCB purchased.

3.1.2 FDA Standards

In the Food and Drug Administration, states that all Medical devices should be designed and manufactured in such a way that, when used under the conditions and for the purposes intended and, where applicable, by virtue of the technical knowledge, experience, education or training of intended users, they will not compromise the clinical condition or the safety of patients, or the safety and health of users or, where applicable, other persons, provided that any risks which may be associated with their use constitute acceptable risks when weighed against the benefits to the patient and are compatible with a high level of protection of health and safety [80]. The medical devices has be effective and have high performance standards. Each device must be designed with a purpose to be a clinically effective when it produces the effect intended by the manufacturer relative to the medical condition. For example, if a device is intended for pain relief, one expects the device to actually relieve pain and would also expect the manufacturer to possess objective, scientific evidence, such as clinical test results, that the device does in fact relieve pain.

Clinical effectiveness is a good indicator of device performance. Performance, however, may include technical functions in addition to clinical effectiveness. For example, an alarm feature may not directly contribute to clinical effectiveness but would serve other useful purposes. Furthermore, it is easier to measure objectively and quantify performance than clinical effectiveness.

Performance is closely linked to safety. For example, a blood collection syringe with a blunt needle would perform badly for collecting blood and could inflict injury. A patient monitor that does not perform well could pose serious clinical safety problems to the patient. Thus, the safety and performance of medical devices are normally considered together [27].

In Food and Drug Administration standard, an average phase in the life span of a medical device are conception and development, manufacture, packaging and labeling, advertising, sale, use, and disposal. The section for this project is in conception and development, manufacture, packaging and labeling which is called Manufacturer.

In the manufacturer stage, the creator of the device, must ensure that it is manufactured to meet or exceed the required standards of safety and performance. This includes the three phases (design/development/testing, manufacturing, packaging and labelling) that lead to a product being ready for the market. The term “user error” is defined as an act that has a different result than that intended by the manufacturer or expected by the operator. User error may result from a mismatch between variables, for example the operator, device, task, or environment. By incorporating human factor engineering principles in design, and appropriate training for users, the risk of user errors can be minimized.

The key advantage regarding quality systems is that they represent a preventive approach to assuring medical device quality versus the previous reactive approach by inspection and rejection at the end of the manufacturing line. Prevention has been proven to be more efficient and cost effective in controlling manufacturing processes and maintaining medical device quality. It is important to note that since the majority of medical devices are in the medium- to low-risk classes, their compliance with regulations often depends upon the declarations of manufacturers, thus the question of quality assurance naturally arises. This is why it is critical for manufacturers to conform with quality system standards and for this conformity to be subject to periodic audit by governmental or third party agencies.

The Food and Drug Administration has developed a guidance to provide the Agency’s initial thinking on technical considerations specific to devices using additive manufacturing, the broad category of manufacturing encompassing 3-dimensional (3D) printing. Additive manufacturing (AM) is a process that builds an object by sequentially building 2-dimensional (2D) layers and joining each to the layer below, allowing device manufacturers to rapidly produce alternative designs without the need for retooling and to create complex devices built as a single piece. Rapid technological advancements and increased availability of AM fabrication equipment are encouraging increased investment in the technology and its increased use by the medical device industry. The purpose of this guidance is to outline technical considerations associated with AM processes, and

recommendations for testing and characterization for devices that include at least one additively manufactured component or additively fabricated step.

From the research, this guidance is broadly organized into two topic areas: Design and Manufacturing Considerations (Section V) and Device Testing Considerations (Section VI). The Design and Manufacturing Considerations section provides technical considerations that should be addressed as part of fulfilling Quality System (QS) requirements for your device, as determined by the regulatory classification of your device and/or regulation to which your device is subject, if applicable. While this guidance includes manufacturing considerations, it is not intended to comprehensively address all considerations or regulatory requirements to establish a quality system for the manufacturing of your device.

This guidance focus on five broad themes: (1) materials; (2) design, printing, and post-printing validation; (3) printing characteristics and parameters; (4) physical and mechanical assessment of final devices; and (5) biological considerations of final devices, including cleaning, sterility, and biocompatibility. A variety of different types of materials can be used in additive manufacturing [27].

From the research found to patient a medical device for future expansion, it was found that there are multiple view to be consider for standard-sized devices for example an interacting design models often made by altering the features of a standard sized device for each patient within a pre-determined range of device designs and size limits. This is typically accomplished through the use of anatomical matching or design manipulation software that may be developed specifically for the AM device, or through other third party software. Patient-matching may also be accomplished by manual methods using specific measurements on radiographs or key anatomic landmark measurements. Any software or procedure used to make modifications to the device design based on clinical input should include internal checks that prevent the operator from exceeding the pre-established device specifications documented in the device master record. We recommend that the design manipulation software or procedure used to make modifications to the device design identify the iteration of the design being changed by the operator. In addition, because this is a medical device, cybersecurity and personally identifiable information so the device in to properly manage and care of personally identifiable information (PII) and protected health information (PHI) is essential in any clinical application. More information is found in protecting PII and PHI, please refer to the HHS Guidance on Significant Aspects of the Privacy Rule. And the Food and Drug Administration include interactive steps in their patient matching workflow be familiar with implementing the FDA's Guidance on the “Content of Premarket Submissions for Management of Cybersecurity in Medical Devices” [77].

3.1.3 Soldering Standards and Safety

The Institute for Printed Circuits (IPC), has some standards to follow regarding the proper soldering, wiring and design of electronic circuits. As there are many standards to take in consideration, this section will only name the most important considered for this project.

As stated in standard IEEE 1149.1 Joint Test Action Group (JTAG), the standard defined for verification of proper connectivity within the printed board circuit, there comes the IPC standards that are not only limited to soldering but also for measurements, stripping tools, sorts of wire used for specific tasks and the proper way to run some test considering every aspect in order to ensure the perfect functionality of the printed board.

IPC-2221 Generic Standard on Printed Board Design: [10] This standard considers the generic requirements for the organic boards and its interconnectability with inorganic materials. The requirements established in this standard are for design and how to connect when there are different components and materials in order to achieve the physical and/or electronic function.

This standard is very specific with the terms in which the instruction for either user or supplier SHALL follow and which ones recommend to follow.

IPC-A-600F Acceptability of Printed Boards: [8] this is a documentation that only states the ways in which the board should look in order to be considered as acceptable. In the not so often case that the requirements does not meet the board's performance there are protocols and documents to follow.

According to this standard the characteristics can be divided into two groups: externally observable where all features and internal phenomena can be seen from the exterior; and internally observable where the imperfection within the board need to be seen by microsectioning. During evaluations the illumination should be precise so it lights the area of interest, without being shadowed by any component.

IPC J-STD-001 Requirements for Soldered Electronic Assemblies: [12] the document describes requirement to manufacture soldered equipment. It gives a controlled methodology to ensure the good quality of work during the manufacture process.

This document classifies the finished product according to complexity, producibility, performance and verification: general electronic products, dedicated service electronic product, and high performance electronic products.

IPC-A-630 Acceptability Standard for Manufacture, Inspection, and Testing of Electronic Enclosures: [9] Regarding the enclosure of the project, this standard sets the requirements for the manufacture, inspection and test of it. For purpose of this standard, the enclosure is defined as a box, a drawer, or a cabinet forming a top level system assembly.

It includes the mechanical parts to assemble a finished system, and the finished system must be made of modular components for the an easy replacement. In a similar way as the J-standard, this standard classifies the end products according to its producibility, complexity and performance requirement into the general, dedicated service and high performance electronic products.

IPC-4101C Specification for base materials for rigid and multilayer printer boards: [13] this document specifies the requirement for the materials needed in order to be used in the

rigid or multilayer printed boards. According to this document there are two kinds of base materials: the laminate base material (L) and the Prepreg base material (P).

IPC 9701A Performance Test Methods, and Qualification Requirements for Surface Mount Solder Attachments: [11] this document establishes the test methods to evaluate the performance of the soldered mount of the electronic assemblies. This test also provide an approach of the equipment's behavior under usage environment.

This document has the purpose to create standardized methods, and reports in order to ensure and give the confidence of optimal performance of electronic assemblies.

IPC D 620 Design and Critical Process Requirements for Cable and Wiring Harnesses: [6] This document provides design, critical process requirements and technical scope that have been removed from the acceptance standards for cable and wire-harness assemblies. This document is intent for mechanical and assembly engineers that will need to consider the different cables and wires to accomplish a specific task.

This standard is important for our project when building the bread and in case of hard-wire two ports of the final PCB. As many components comes with a built-in wire connector, we need to consider if that wire is specific for voltage purposes. Let us remember than most wires comes with different resistance that can affect the project functionality.

IPC 7711B/7721B Rework, Modification and Repair of Electronic Assemblies: [7] this documents talks about the repairing procedure for integrated circuits. This version of standard includes a coverage for process that lead is not present and guidelines for repair operations. This document goes from guidelines, tools, procedures and methods for the repair and restoration of electronic products.

Although we as a group are not going to repair the device, we must build it repairable so the administrator can give proper maintenance and it must be easy to operate.

3.2 Design Constraints

Several realistic constraints limit and guide our project design. These design constraints account for not only the identified requirement specifications, but also several outside factors. These outside factors include economic constraints; time constraints; manufacturability constraints; sustainability constraints; environmental constraints; health constraints; safety constraints; ethical constraints; social constraints; political constraints; security constraints; and parts, components, and testing constraints.

3.2.1 Economic Constraints

The budget cost of constructing the Medspencer was offset by the sponsor, Dr. Fredesvinda Jacobs-Alvarez, MD, head physician of Esperanza Behavioral Health and Services. The total cost amount to design and produce the Medspencer was around \$450. This amount includes all the components that were incorporated into the final Medspencer product, such as the ATMEGA328P microcontroller, Raspberry Pi Compute Module 3 Lite, the PCB,

Wi-Fi module, fingerprint scanner, touchscreen display, servomotors, control unit for the servomotors, wiring and electrical components, and so on. The predicted cost also includes the research and development costs, which include the development boards, breadboards, wires, resistors, capacitors, and other electrical components. The team kept receipts for all purchased material. The team strived to be as economically frugal as possible.

Regarding the customer costs, our team aimed to provide an effective product for less than current products on the market. The Livi Automated Medication Dispenser is the most advanced home medicine dispenser that we found on the market and is the only one that automatically sorts and dispenses medicines, so it does not require pre-sorting of medicines into trays or cups. The Livi costs \$1999 to buy [60]. If we manage to build the Medspencer using \$300, then potentially the Medspencer could be marketed at \$600. This would provide a good profit margin of 100%. \$600 is a comparable price to some other medicine dispensers on the market and significantly less than the Livi, and still allows us to develop a device with enhanced performance and added features (as compared to the current dispensers on the market).

The team attempted to keep costs as low as possible, so no waste is incurred. To this end, the team carefully designed the circuit connections and tested the hardware components and connections appropriately. This way, parts were not damaged. If parts were damaged or broken, we would need to repurchase them, which would incur additional costs. By paying attention to detail and testing our designs by simulation and physical testing, we kept the total costs for the Medspencer low.

At the end of the Senior Design 2 semester, the total costs incurred to create the Medspencer project was \$440.92. However, \$221.95 of that was pure research and development costs, as we needed to purchase the CM3L development board and Arduino development board in order to do initial testing and prototyping. With that in mind, the manufacturing cost of the Medspencer is \$218.97. This is within our target goal of <\$300 for manufacturing.

3.2.2 Time Constraints

Because this design project is a Senior Design Project, time is limited to the two semesters given to complete the project. The first semester involves designing the Senior Design project and writing the Senior Design 1 Document. The second semester involves implementing the project design and creating a working device.

Two months (March and April) were allotted to conduct the necessary research, design the project, and plan the execution, which includes prototyping the Medspencer and building the finished product. Researching the Medspencer involved conducting market research and investigating the available technologies we may use. Designing the Medspencer involved choosing the specific hardware components we will use for our device, designing the hardware connections and circuits, and planning the software design for the Medspencer.

Then, three months were allotted to buy the chosen parts, prototype the Medspencer device, design the PCB, assemble the device hardware and write the software, and test the finished Medspencer device. Because this was such a limited time frame, it is imperative that the team planned the milestones wisely to complete the project on time and allow time for any necessary modifications. We ordered all the necessary parts during early May. By the early June, we began prototyping the device using the development board and breadboard, ensuring that the circuitry and basic coding is correct. Then the PCB was designed received by late June. After soldering and testing the PCB, a second revision for the PCB was required, which was designed and received by mid-July. During the second half of July, we worked on on soldering the board, assembling the final device, finishing up the software, and testing to ensure everything worked correctly.

3.2.3 Manufacturability and Sustainability Constraints

Manufacturing the Medspencer may require more research considering that the device will store medication for longer periods of time. Most of the medication need a cool environment where to be store at. Since the medication will share room and will be enclosed with many servos, PCBs and a voltage regulator, a ventilation system must be added in order to keep an ideal environment for the medication.

Since the device will require a voltage regulator to handle all the charge burden, it must be powerful enough to run all the module pieces, but also small enough to fit in the box. So far the elements such as the fingerprint reader, the servo motors, the cooling fan, the touchscreen display and the speaker do not use more than 7V, so a 12V-DC voltage supply will suffice our criteria. We can separate the project into different modules so it would be easier to make everything work. After every module is properly functioning, they must fit within the box for proper showing. We must remember that the dispenser will be for domestic use, that means it should not exceed certain size and certain weight.

For the sustainability of the device, there must be a certified administrator that will change, remove and provide the medication whenever is needed. As the MedSpencer have Wi-Fi connection and is capable to report via email to the doctor in charge about the patient compliance, by similar means it will alert the administrator when the device is running out of medication and also when to add or remove a patient from the device's system.

The administrator will also be in charge of the maintenance of the hardware and software of the dispenser. A simple and common issue that will face the dispenser is the dust. As the machine has a cooling fan, it will drag dust from the air to the inside part. Neither the patients, nor the caretakers can do the inside cleaning because they do not know the proper procedure for maintenance. As the medications will be sealed in the plastic containers, dust or other contaminants cannot get to them.

As we can see in Figure 2, a common issue regarding any device that uses a cooling fan system. It is no so important when this is within the Central Processing Unit (CPU) because it only involves electronic components that can be cleaned every once in a while, but when this happens to a unit where medicine is involved, it is unacceptable. The administrator

should keep a regular schedule for the device maintenance that includes cleaning the cooling fan system and also to clean the plastic medicine containers.



*Figure 2. Cooling fan system with no maintenance [39]
Courtesy of dirtycomputer.com*

3.2.4 Parts, Component, and Testing Constraints

In order to achieve the complete functionality of the project we can divide it into modules. There will be the servo motor modules, the fingerprint reader module, the display module and the cooling-fan module. Each one of them are only hardware components. Everything regarding software will be explained in another sub-section. Summarizing the project, the Medspencer takes into account the clock from the internet and compare it with the pre-assigned schedule of each one of the patients on file. Once a match is found the device will sound to alert the patient and it will not stop sounding until a fingerprint is placed on the fingerprint reader and this matches to the patients’.

The Medspencer has a touchscreen display that will help the administrator to input the information about patients, change the medication slots and allows more options for caretakers. These options make it possible to take medication out for their patients in case of travel. The administrator options is only enabled when his/her fingerprint is read by the dispenser.

In order for the internet connection to happen we are attaching a ESP8266 Wi-Fi module so the clock will always be updated. This component will have email purposes that will be described later on in chapter 5.



*Figure 3. Wi-Fi module ESP8266 [44]
Courtesy of generationrobots.com*

What is unique to this project is that the dispenser will identify each patient through his or her fingerprint. The fingerprint reader used is a R307 optical fingerprint reader that it can be easily connected to an Arduino microprocessor in order to do some test. This reader needs an input voltage of 5V to work. While doing some test on an Arduino board, we had to make the connections described on Table 1 so the reader could work properly:

Table 1. Connections for Fingerprint Reader on Arduino [45]

Reader	Arduino
Power input (red)	5V
Ground (black)	GND
Data in(white)	Port 3
Data out (green)	Port 2

Our first goal with the fingerprint reader is to store different samples of fingerprint. Once the fingerprint is recognized and linked to an existing patient, the proper servo is activated to dispense the correct amount of pills. These servos also work with an small input voltage of 5V. Running some tests on the Arduino, we noticed that we can manage the degrees that the servo can make its arm to turn and also the speed of it. In Table 2 there is described the proper connections for the servo to work with the Arduino.

Table 2. Connections for Servo Motors on Arduino [5]

Servo	Arduino
Power (red)	5V
Ground (brown)	GND
Signal (yellow)	Port 9

The medication was stored in plastic containers that meet the FDA standards for storing. For the purpose of this project, these containers were modified at the bottom to add special filters that were attached to the servo motors. Once the servo motors rotate, they let the medications leave the container and fall through the tubes that will lead to the base of the device so the patient can take it. The tubes, according to the FDA Guidance on Container Closures, can be made of glass, high density polyethylene or metal, so they cannot alter the composition of the medication.

The cooling fan system does not consume too much energy and is only there to refrigerate the inner environment of the dispenser so the medication does not suffer from the heat and lose its effectiveness. All those components, the main board, the servo motors and the closure can create a warmer environment where the medication containers will become poisonous for the them. This cooling fan is be triggered by a temperature sensor that monitors the inside temperature in order to avoid to get warm enough to be considered dangerous.

The enclosure for this project was made of acrylic and has a door through which the administrator should replace empty containers or replace medications and give maintenance to the parts.

3.2.5 Environmental Constraints

The Environmental constraints on having this device will use power however, the device will not use huge power; it only requires 16-20V and 2A. On the other hand, this device can save the environmental by not allowing the patients and caretakers to throw the medication away down the sewage line or into the ocean because some medication can have a 1 to 2 week half-life in the human body but longer in the ocean and fishes where other humans can absorb these deadly efforts. In the EMBO reports, “Medicines have an important role in the treatment and prevention of disease in both humans and animals. But it is because of the very nature of medicines that they may also have unintended effects on animals and microorganisms in the environment. Although the side effects on human and animal health are usually investigated in thorough safety and toxicology studies, the potential environmental impacts of the manufacture and use of medicines are less well understood and have only recently become a topic of research interest. Some of the effects of various compounds—most notably anthelmintics from veterinary medicine and antibacterial therapeutics, but there are many other substances that can affect organisms in the environment. Pharmaceutical substances may also be degraded by biological organisms in treatment systems, water bodies and soils as well as abiotic reactions. Generally, these processes reduce the potency of medicines; however, some breakdown products have similar toxicity to their parent compounds. Furthermore, degradation varies significantly depending on chemistry, biology and climatic conditions. For example, the half-life of the antiparasitic ivermectin under winter conditions is six times greater than in the summer and the compound degrades faster in sandy soils than in sandy loam soils. The natural estrogens 17β -estradiol and estrone degrade in the aerobic and anoxic tanks of activated sludge systems, whereas 17α -ethinylestradiol only degrades under aerobic conditions. All this

adds to the complexity of the problem and calls for individual solutions for individual pharmaceuticals and applications” [21].

“The US Food and Drug Administration (FDA) requires environmental risk assessments of human and veterinary medicines on the effects on aquatic and terrestrial organisms before they allow a product to the market, and the EU introduced similar requirements in 1997. These environmental impact studies investigate the potential negative effects on fish, daphnids, algae, bacteria, earthworms, plants and dung invertebrates. Much of the data are publicly accessible—many of the environmental assessments are published on the FDA's web site—and provide a reasonable body of data for further study. However, there are valid questions about the real-world value of these studies. Risk assessments usually use standard ecotoxicity tests, which are often short-lived and focus predominantly on mortality as the endpoint. Moreover, aquatic tests tend to focus on the water compartment and do not take into account pharmaceuticals residing in sediments. In general, the effects observed in these studies occur at much higher concentrations than those that are measured in the environment. What is less known are the more subtle effects that therapeutically active substances can have on organisms in the environment, such as growth, fertility or behaviour” [21].

3.2.6 Health and Safety Constraints

The medical profession has long subscribed to a body of ethical statements developed primarily for the benefit of the patient. As a member of this profession, a physician must recognize responsibility to patients first and foremost, as well as to society, to other health professionals, and to self. The following Principles adopted by the American Medical Association are not laws, but standards of conduct that define the essentials of honorable behavior for the physician.

The nine principles of medical ethics are a physician shall be dedicated to providing competent medical care, with compassion and respect for human dignity and rights, a physician shall uphold the standards of professionalism, be honest in all professional interactions, and strive to report physicians deficient in character or competence, or engaging in fraud or deception, to appropriate entities, a physician shall respect the law and also recognize a responsibility to seek changes in those requirements which are contrary to the best interests of the patient, a physician shall respect the rights of patients, colleagues, and other health professionals, and shall safeguard patient confidences and privacy within the constraints of the law, a physician shall continue to study, apply, and advance scientific knowledge, maintain a commitment to medical education, make relevant information available to patients, colleagues, and the public, obtain consultation, and use the talents of other health professionals when indicated, a physician shall, in the provision of appropriate patient care, except in emergencies, be free to choose whom to serve, with whom to associate, and the environment in which to provide medical care, a physician shall recognize a responsibility to participate in activities contributing to the improvement of the community and the betterment of public health, a physician shall, while caring for a patient, regard responsibility to the patient as paramount, and a physician shall support access to medical care for all people.

They also must Food and Drug Administration when prescribing medications. In addition, medical profession must follow the Medicare regulations and law on treating and servicing patients where all health insurance companies follow Medicare laws but add their own requirements and recommends.

One example of where this device will benefit for the patient is using Celexa (generic: Citalopram hydrobromide). It is a drug used for antidepressant medication. However, the FDA issued a Drug Safety Communication (DSC) stating that citalopram should no longer be used at doses greater than 40 mg per day because it could cause potentially dangerous abnormalities in the electrical activity of the heart. Citalopram use at any dose is discouraged in patients with certain conditions because of the risk of QT prolongation, but because it may be important for some of those patients to use citalopram, the drug label has been changed to describe the particular caution that needs to be taken when citalopram is used in such patients. The revised drug label also describes lower doses that should be used in patients over 60 years of age. Changes in the electrical activity of the heart (specifically, prolongation of the QT interval of the electrocardiogram [ECG]) can lead to a risk of an abnormal heart rhythm called Torsade de Pointes, which can be fatal. Patients at particular risk for developing prolongation of the QT interval include those with underlying heart conditions and those who are predisposed to having low levels of potassium and magnesium in the blood.

The facts about Celexa is that it works by increasing the amount of serotonin in the brain. Available as 10 mg, 20 mg, and 40 mg tablets. Also available as an oral solution (10 mg/5 mL). In 2011, a total of approximately 31.5 million prescriptions were dispensed for citalopram from U.S. outpatient retail pharmacies. In 2011, approximately 7.2 million patients received a dispensed prescription for citalopram from U.S. outpatient retail pharmacies. In 2011, according to U.S. office-based physician practices, approximately 89.5% of citalopram drug use was at doses of 40 mg and below and 6% of drug use was at doses above 40 mg per day.

The maximum recommended dose of citalopram is 20 mg per day for patients with hepatic impairment, patients who are older than 60 years of age, patients who are CYP 2C19 poor metabolizers, or patients who are taking concomitant cimetidine (Tagamet®) or another CYP2C19 inhibitor, because these factors lead to increased blood levels of citalopram, increasing the risk of QT interval prolongation and Torsade de Pointes. Electrolyte and/or ECG monitoring is recommended in certain circumstances Consider more frequent ECG monitoring in patients for whom citalopram use is not recommended, but is, nevertheless, considered essential.

Patients at risk for significant electrolyte disturbances should have baseline serum potassium and magnesium measurement, with periodic monitoring. Hypokalemia and/or hypomagnesemia may increase the risk of QTc prolongation and arrhythmia and should be corrected prior to initiation of treatment with periodic monitoring. Citalopram should be discontinued in patients found to have persistent QTc measurements greater than 500 ms. Advise patients on citalopram to contact a healthcare professional immediately if they experience signs and symptoms of an abnormal heart rate or rhythm (e.g., dizziness,

palpitations, or syncope). If patients experience symptoms, the prescriber should initiate further evaluation, including cardiac monitoring.

In conclusion, this medication example can be the reference to set the project standard and if this device is cleared but the FDA to hold and dispense this medication then any medication is possible.

3.2.7 Ethical and Legal Constraints

Pharmacists and physicians both follow the code of ethics. They must know what they are giving out and why they are giving it out. From the 1970, the “Controlled Substances Act (CSA) has been used by law enforcement to decrease drug abuse and dependence among Americans by regulating the production, sale, purchase and use of many drugs. This act gives authority to the Drug Enforcement Administration (DEA) to monitor and control the use of substances, both legal and illegal. Because of the many differences between the various types of substances, and between each individual substance, the CSA puts each substance into one of five categories, called Schedules. These categories give each substance a simple classification that helps both law enforcement and the medical community to easily understand its nature” [82].

The CSA states the following factors affect a drug’s schedule which are potential for abuse, scientific information available regarding the drug’s pharmacological effect, scientific understanding of the drug, historical and current patterns of abuse, magnitude of abuse, possible risks to public health, risk of developing psychological or physical dependence.

“The DEA explains, Schedule V drugs have the least potential for abuse, while Schedule I drugs are considered to have the highest abuse and dependence potential. The DEA also lists the following examples for each drug schedule meaning schedule I: heroin, LSD, marijuana and ecstasy, schedule II: Ritalin, Adderall, oxycodone, methadone, cocaine and methamphetamine, schedule III: anabolic steroids, testosterone, ketamine, products with less than 90 milligrams of codeine per dosage and products with less than 15 milligrams of hydrocodone per dosage, schedule IV: Xanax, Valium, Ambien, Soma and Ativan, and schedule V: cough medications containing less than 200 milligrams of codeine or per 100 milliliters, Lyrica, Motofen and Lomotil” [82].

Furthermore, the DEA has some Federal Regulations for example in Part 1304 of the Records and Reports of Registrants under the Inventory Requirements which has 5 parts which states in the general requirements, “Each inventory shall contain a complete and accurate record of all controlled substances on hand on the date the inventory is taken, and shall be maintained in written, typewritten, or printed form at the registered location. An inventory taken by use of an oral recording device must be promptly transcribed. Controlled substances shall be deemed to be “on hand” if they are in the possession of or under the control of the registrant, including substances returned by a customer, ordered by a customer but not yet invoiced, stored in a warehouse on behalf of the registrant, and substances in the possession of employees of the registrant and intended for distribution as complimentary samples. A separate inventory shall be made for each registered location and each independent activity registered, except as provided in paragraph (e)(4) of this

section. In the event controlled substances in the possession or under the control of the registrant are stored at a location for which he/she is not registered, the substances shall be included in the inventory of the registered location to which they are subject to control or to which the person possessing the substance is responsible. The inventory may be taken either as of opening of business or as of the close of business on the inventory date and it shall be indicated on the inventory [83].

Initial inventory date. Every person required to keep records shall take an inventory of all stocks of controlled substances on hand on the date he/she first engages in the manufacture, distribution, or dispensing of controlled substances, in accordance with paragraph (e) of this section as applicable. In the event a person commences business with no controlled substances on hand, he/she shall record this fact as the initial inventory [83].

Biennial inventory date. After the initial inventory is taken, the registrant shall take a new inventory of all stocks of controlled substances on hand at least every two years. The biennial inventory may be taken on any date which is within two years of the previous biennial inventory date [83].

Inventory date for newly controlled substances. On the effective date of a rule by the Administrator pursuant to §1308.45, 1308.46, or 1308.47 of this chapter adding a substance to any schedule of controlled substances, which substance was, immediately prior to that date, not listed on any such schedule, every registrant required to keep records who possesses that substance shall take an inventory of all stocks of the substance on hand. Thereafter, such substance shall be included in each inventory made by the registrant pursuant to paragraph (c) of this section [83].

Inventories of manufacturers, distributors, registrants that reverse distribute, importers, exporters, chemical analysts, dispensers, researchers, and collectors. Each person registered or authorized (by §1301.13, 1307.11, 1307.13, or part 1317 of this chapter) to manufacture, distribute, reverse distribute, dispense, import, export, conduct research or chemical analysis with controlled substances, or collect controlled substances from ultimate users, and required to keep records pursuant to §1304.03 shall include in the inventory the information listed below [83].

Inventories of manufacturers. Each person registered or authorized to manufacture controlled substances shall include each controlled substance in bulk form to be used in (or capable of use in) the manufacture of the same or other controlled or non-controlled substances in finished form, the inventory shall include the name of the substance, and the total quantity of the substance to the nearest metric unit weight consistent with unit size. For each controlled substance in the process of manufacture on the inventory date, the inventory shall include the name of the substance; the quantity of the substance in each batch and/or stage of manufacture, identified by the batch number or other appropriate identifying number; and the physical form which the substance is to take upon completion of the manufacturing process (e.g., granulations, tablets, capsules, or solutions), identified by the batch number or other appropriate identifying number, and if possible the finished form of the substance (e.g., 10-milligram tablet or 10-milligram concentration per fluid ounce or milliliter) and the number or volume thereof [83].

For each controlled substance in finished form the inventory shall include the name of the substance; each finished form of the substance (e.g., 10-milligram tablet or 10-milligram concentration per fluid ounce or milliliter); the number of units or volume of each finished form in each commercial container (e.g., 100-tablet bottle or 3-milliliter vial); and the number of commercial containers of each such finished form (e.g. four 100-tablet bottles or six 3-milliliter vials) [83].

In this section (damaged, defective or impure substances awaiting disposal, substances held for quality control purposes, or substances maintained for extemporaneous compoundings) the inventories shall include the name of the substance; the total quantity of the substance to the nearest metric unit weight or the total number of units of finished form; and the reason for the substance being maintained by the registrant and whether such substance is capable of use in the manufacture of any controlled substance in finished form [83].

3.2.8 Social and Cultural Constraints

When a patient walks out a faculty, clinic, or outpatient office, physicians and other providers always worry when patients are out in sociality. Especially from the hospitals, “patients are being discharged sooner, often in the process of convalescence rather than at baseline health status. This requires physicians to more effectively communicate instructions for post-discharge care to patients, family members, and outpatient providers. A comprehensive review on improving all aspects of this process is beyond the scope of this article; we will focus specifically on improving medication use. Patients are very susceptible to medical errors in the days immediately following hospital discharge. Forty-nine percent of hospitalized patients experience at least one medical error following discharge, most commonly involving medication use. An estimated 19–23% of patients suffer an adverse drug event (ADE) after discharge. Most of these errors and ADEs could be prevented through better communication. This commentary will discuss several common barriers to proper medication use after hospital discharge and review potential solutions based on the experience in the United States” [31].

Another social constraint is when patient states that the medication is working for the first month and they stop taking that medication automatically without the physician’s approved. Later, they compliant to the staff or physician that the medication is not working and need other medication. While the medication that has not been taking is at home and if the patient takes the new medication and the medication leftover they can get a big complication or death.

A cultural constraints example is a study found in “Rituals of Silence’ Haafkens examines the use of benzodiazepines – medicines prescribed for mental distress – by women in the Netherlands. She describes how the medicines not only make it possible for women to live on with their mental health problems, but also provide society with a means to control anxiety and stress. The ambiguous relation between self-control of female distress and medicalized social control of life problems is the main theme of another chapter in our book.

On the one hand, women use benzodiazepines like other medicines to enhance the quality of their individual lives. Medicines liberate them from bodily discomfort, and give them means to control natural bodily processes such as conception, menstruation and menopause. Medicines are part of day-to-day body regimes, in which women strive to fulfill societal expectations of work capability, appropriate fertility, attractive appearance, and mental stability. On the other hand, benzodiazepines and other medicines also function as a medical means of social control. In modern societies, where medicine has replaced religion as a dominant moral ideology and social control institution, more and more of everyday life has come under medical dominion, influence and supervision, a process known as medicalization. Around 1990, at the time of Haafkens' study, about one in every seven women had a prescription for benzodiazepines filled annually in the Netherlands; men received such prescriptions much less often. Dutch medical guidelines acknowledged that the drugs were not as safe as initially thought, and recommended that the duration of benzodiazepine use be limited to one to two weeks, a few months at the most. Despite this advice, long-term use of benzodiazepines remained relatively common in the Netherlands, affecting an estimated three percent of the Dutch population.

The efficacy of the medicines, therefore, is ambiguous. For the women who use them, the medicines are indispensable to stay in control of their lives. At the same time, many of those using them feel that the medicines have taken control of *them*. They would like to stop but cannot. The medicines are empowering as well as disempowering” [31].

Another social and cultural constraint is the language barrier, if the patient is Hispanic or Russian and they recently came to the USA.

3.2.9 Political Constraints or FDA Standards

In the Food and Drug Administration, it states that all Medical devices should be designed and manufactured in such a way that, when used under the conditions and for the purposes intended and, where applicable, by virtue of the technical knowledge, experience, education or training of intended users, they will not compromise the clinical condition or the safety of patients, or the safety and health of users or, where applicable, other persons, provided that any risks which may be associated with their use constitute acceptable risks when weighed against the benefits to the patient and are compatible with a high level of protection of health and safety [80]. The medical devices have to be effective and have high performance standards. Each device must be designed with a purpose to be a clinically effective when it produces the effect intended by the manufacturer relative to the medical condition. For example, if a device is intended for pain relief, one expects the device to actually relieve pain and would also expect the manufacturer to possess objective, scientific evidence, such as clinical test results, that the device does in fact relieve pain.

Clinical effectiveness is a good indicator of device performance. Performance, however, may include technical functions in addition to clinical effectiveness. For example, an alarm feature may not directly contribute to clinical effectiveness but would serve other useful purposes. Furthermore, it is easier to measure objectively and quantify performance than clinical effectiveness.

Performance is closely linked to safety. For example, a blood collection syringe with a blunt needle would perform badly for collecting blood and could inflict injury. A patient monitor that does not perform well could pose serious clinical safety problems to the patient. Thus, the safety and performance of medical devices are normally considered together [27].

In Food and Drug Administration standard, an average phase in the life span of a medical device are conception and development, manufacture, packaging and labeling, advertising, sale, use, and disposal. The section for this project is in conception and development, manufacture, packaging and labeling which is called Manufacturer.

In the manufacturer stage, the creator of the device, must ensure that it is manufactured to meet or exceed the required standards of safety and performance. This includes the three phases (design/development/testing, manufacturing, packaging and labelling) that lead to a product being ready for the market. The term “user error” is defined as an act that has a different result than that intended by the manufacturer or expected by the operator. User error may result from a mismatch between variables, for example the operator, device, task, or environment. By incorporating human factor engineering principles in design, and appropriate training for users, the risk of user errors can be minimized.

The key advantage regarding quality systems is that they represent a preventive approach to assuring medical device quality versus the previous reactive approach by inspection and rejection at the end of the manufacturing line. Prevention has been proven to be more efficient and cost effective in controlling manufacturing processes and maintaining medical device quality. It is important to note that since the majority of medical devices are in the medium- to low-risk classes, their compliance with regulations often depends upon the declarations of manufacturers, thus the question of quality assurance naturally arises. This is why it is critical for manufacturers to conform with quality system standards and for this conformity to be subject to periodic audit by governmental or third party agencies.

The Food and Drug Administration has developed a guidance to provide the Agency’s initial thinking on technical considerations specific to devices using additive manufacturing, the broad category of manufacturing encompassing 3-dimensional (3D) printing. Additive manufacturing (AM) is a process that builds an object by sequentially building 2-dimensional (2D) layers and joining each to the layer below, allowing device manufacturers to rapidly produce alternative designs without the need for retooling and to create complex devices built as a single piece. Rapid technological advancements and increased availability of AM fabrication equipment are encouraging increased investment in the technology and its increased use by the medical device industry. The purpose of this guidance is to outline technical considerations associated with AM processes, and recommendations for testing and characterization for devices that include at least one additively manufactured component or additively fabricated step.

From the research, this guidance is broadly organized into two topic areas: Design and Manufacturing Considerations (Section V) and Device Testing Considerations (Section VI). The Design and Manufacturing Considerations section provides technical considerations that should be addressed as part of fulfilling Quality System (QS)

requirements for your device, as determined by the regulatory classification of your device and/or regulation to which your device is subject, if applicable. While this guidance includes manufacturing considerations, it is not intended to comprehensively address all considerations or regulatory requirements to establish a quality system for the manufacturing of your device.

This guidance focus on five broad themes: (1) materials; (2) design, printing, and post-printing validation; (3) printing characteristics and parameters; (4) physical and mechanical assessment of final devices; and (5) biological considerations of final devices, including cleaning, sterility, and biocompatibility. A variety of different types of materials can be used in additive manufacturing [27].

From the research found to patient a medical device for future expansion, it was found that there are multiple views to be considered for standard-sized devices: for example, an interacting design models often made by altering the features of a standard sized device for each patient within a pre-determined range of device designs and size limits. This is typically accomplished through the use of anatomical matching or design manipulation software that may be developed specifically for the AM device, or through other third party software. Patient-matching may also be accomplished by manual methods using specific measurements on radiographs or key anatomic landmark measurements. Any software or procedure used to make modifications to the device design based on clinical input should include internal checks that prevent the operator from exceeding the pre-established device specifications documented in the device master record. We recommend that the design manipulation software or procedure used to make modifications to the device design identify the iteration of the design being changed by the operator. In addition, because this is a medical device, cybersecurity and personally identifiable information so the device in to properly manage and care of personally identifiable information (PII) and protected health information (PHI) is essential in any clinical application. More information is found in protecting PII and PHI, please refer to the HHS Guidance on Significant Aspects of the Privacy Rule. And the Food and Drug Administration include interactive steps in their patient matching workflow be familiar with implementing the FDA's Guidance on the "Content of Premarket Submissions for Management of Cybersecurity in Medical Devices" [77]

Another Political constraint is the cost of health care which has been escalating slowly but steadily. "Lately, the speed of this development has become harder to accept as it has greatly outpaced the growth of the national economy in many countries. The multiple reasons for this trend include the growth in expensive technologies and increasing demand for services fueled by a complex mix of social changes—for example, improved levels of education, the commodification of medicine, and the general medicalization of life.

There is also rising demand for quality improvement and quality control in health care. Some of the equally complex social developments underlying this are the increasing autonomy, rights, and knowledge of patients; commodification of the doctor patient relationship towards a provider client relationship; declining trust in professionalism and professional judgment; greater trust in statistical research; and less toleration of medical

error. These trends affect the different parties involved in health care—patients, doctors, and administrators or payers—differently, and can lead to conflicts of interest.

It has also become commonplace to argue that increasing resources will not necessarily produce any good if not spent effectively. Thus, when more money is promised for healthcare, it is done on the condition that it can be proved that the money is spent on effective interventions. This trend has created an unprecedented need for the medical profession to explicitly justify its actions in both medical and economic terms. Evidence based medicine (EBM), or the whole outcomes movement, is a central tool in this process of increasing the accountability of medicine.

Cost control and quality improvements are often combined in practical health care development—to get better for less is an obvious ideal for healthcare administrators. Even if, however, this might be logical and practical on the administrative level, for individual patients and doctors it is often the cause of much of the confusion that surrounds both EBM and rationing. We argue that this confusion of costs and quality has made it difficult for doctors to foresee and react to the changes EBM implies for the whole profession” [72].

Moreover, “costs inevitably play a role when CPGs are created and individual patients treated, even if in many cases only implicitly, for at least two reasons. First, because for CPGs to be relevant their recommendations must be realistic—that is, possible to apply in practice. A guideline recommending a treatment that nobody can or wants to provide is futile, and more likely to be damaging if there is no realistic alternative recommendation—for example, Western HIV treatment guidelines in most African countries. Second, because opportunity costs—that is, the alternative goods (not just monetary) that will be lost if a treatment is given—must be considered. Considering costs as part of trying to find the best possible solution to an individual patient’s problem is a traditional and important part of medical praxis. It is common that patients do not buy medication they consider too expensive, or use a lower dosage than prescribed, without reporting this to the doctor. Patients always consider opportunity costs on their individual level, and these naturally depend on the way the health care is financed. On the individual level, cost considerations can in theory be dealt with relatively simply by letting individual patients decide what they are willing to pay for certain services. In insurance based systems people can choose the insurance policy they like and can afford, although, at least in the US, this appears true in most cases only in theory. And even in theory, it is quite optimistic to think we could correctly predict our reactions to some possible future diseases and ailments that we have no experience of at the moment” [72].

3.2.10 Security Constraints

The device will be a home appliance placed in an open area like the kitchen where the patient(s) and caretaker/administrator can access the device. If the device can be opened without a key or any other permission then the device is vulnerable. They would have access to the medication and the data storage. Thus, it is important to secure the device so only the administrator can access all of the medicines and data.

Another security constraint is the wireless communication capabilities present in many modern Medical Devices (MDs). This is a major source of security risks, particularly while the patient is in open (i.e., non-medical) environments. To begin with, the implant becomes no longer “invisible”, as its presence could be remotely detected. Furthermore, it facilitates the access to transmitted data by eavesdroppers who simply listen to the (insecure) channel. This could result in a major privacy breach, as Medical Devices (MDs) store sensitive information such as vital signals, diagnosed conditions, therapies, and a variety of personal data (e.g., birth date, name, and other medically relevant identifiers). A vulnerable communication channel also makes it easier to attack the implant in ways similar to those used against more common computing devices, i.e., by forging, altering, or replying previously captured messages. This could potentially allow an adversary to monitor and modify the implant without necessarily being close to the victim [24].

Another Security constraint is software or web application to defined the roles of the patient and caretaker to have specific access to the each of them. “The content to be secured is declared using one or more web-resource-collection elements. Each web-resource-collection element contains an optional series of url-pattern elements followed by an optional series of http-method elements. The url-pattern element value specifies a URL pattern against which a request URL must match for the request to correspond to an attempt to access secured content. The http-method element value specifies a type of HTTP request to allow.

The optional user-data-constraint element specifies the requirements for the transport layer of the client to server connection. The requirement may be for content integrity (preventing data tampering in the communication process) or for confidentiality (preventing reading while in transit). The transport-guarantee element value specifies the degree to which communication between the client and server should be protected. Its values are NONE, INTEGRAL, and CONFIDENTIAL. A value of NONE means that the application does not require any transport guarantees. A value of INTEGRAL means that the application requires the data sent between the client and server to be sent in such a way that it can't be changed in transit. A value of CONFIDENTIAL means that the application requires the data to be transmitted in a fashion that prevents other entities from observing the contents of the transmission. In most cases, the presence of the INTEGRAL or CONFIDENTIAL flag indicates that the use of SSL is required.

The optional login-config element is used to configure the authentication method that should be used, the realm name that should be used for the application, and the attributes that are needed by the form login mechanism” [57] .

The auth-method child element specifies the authentication mechanism for the web application. As a prerequisite to gaining access to any web resources that are protected by an authorization constraint, a user must have authenticated using the configured mechanism. Legal auth-method values are BASIC, DIGEST, FORM, and CLIENT-CERT. The realm-name child element specifies the realm name to use in HTTP basic and digest authorization. The form-login-config child element specifies the log in as well as error pages that should be used in form-based login. If the auth-method value is not FORM, then form-login-config and its child elements are ignored [24].

With this web command, they will need to be update to the newer standards which contain higher security parameters and a bit complex algorithm to sort out unwanted or unnecessary character and number from the input of the user.

3.2.11 Performance, Functionality, Usability, Reliability and Availability

The performance is the more of a quality to do survey for a given population.

From the House of Quality and comparing to other device out in the market, functionality of the timing when the patient scan their fingerprint to the time the device dispense the medication. The device will have a 1-2 min to respond.

Usability is for the patient to access the medication which is predicted to be less than 2 mins. The patient will scan their fingerprint and push on the touch screen to confirm that the patient is the patient shown. Later, the patient will push a button on the screen if the medication or “as needed” meaning when the patient just request the medication at random times. Then the next screen will show in simple terms instead of the medication name for example Zolpidem (Ambien) it will say and show the reason why they are taking the medication. For example, Ambien is for sleep problems. In addition, this device will be used for a family of 4 users or patients and a caretaker.

For now, this device will be 99% reliable for 1 yearly check-up due to the importance of patient’s medication for live and death scenarios. On the other hand, the device will be plug in to the outlet however in future improvement that device will be powered by battery for 12 hours due to weather and disaster in the area.

The availability of the device is 365 days a year and operational 24 hours a day and 7 days a week.

4.0 Research and Background Information

The research conducted in preparation for designing the Medspencer product includes research on similar products in the market; technologies that we plan to use to develop the Medspencer; and power considerations. When researching technologies, we considered our project requirements, then studied the available technologies and how they work, in order to ascertain how we can implement the technologies to accomplish our goals for the Medspencer. Researching available technologies and figuring out the Medspencer's specific needs helped us later when we chose what specific hardware components to purchase to develop the Medspencer.

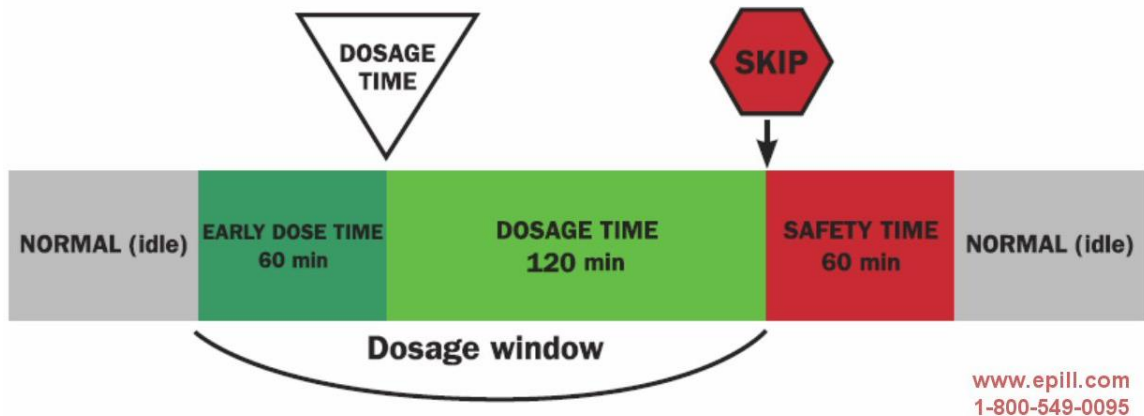
4.1 Market Research

Market research is an important part of researching when it comes to product development. By researching what products are already available, we can ascertain what features are already offered or not, and whether they are effective. This helps us design a unique product with new and improved features and/or at a more agreeable price to consumers.

4.1.1 Existing Products

While there are a few household automated medicine dispensers on the market, they are quite expensive and do not have the same features that have been described for the Medspencer. All the medicine dispensers on the market that we studied are only for one person use. Also, the devices do not verify that the correct person is taking the medicines. None of the dispensers can reschedule in the case of a missed dose. All the dispensers mentioned will sound an alarm and/or light up when it is time for a scheduled medicine dose. They are locked devices, and do not allow patients to access medicines outside of the schedule dose time. Researching other automated medicine dispensers that are already on the market is advantageous, because the team can consider possible new features that could be added or existing features that may be improved upon.

e-pill, LLC sells a series of automatic medicine dispensers [17]. The cost of most dispensers ranges from \$450 to \$595. The e-pill medicine dispensers require pre-sorting of the medicines into the medication tray. The medication tray has a finite number of compartments, so the tray must be refilled more frequently if the patient must take multiple doses in a day; the frequency of refilling ranges from once per month (for one dose a day) to every few days. Doses can be scheduled up to 3 or 6 times per day at different specified times (depending on the product), and there is an optional early dose feature. The Medication Window used for e-pill medicine dispensers is illustrated below in Figure 4. Our team is considering a similar scheme for the Medspencer's dosage window. The dispenser is locked; however, reviews say it is easy to tamper with the device to open it. The tamper-resistant medicine dispensers range from \$650 to \$995. The e-pill MedSmart Plus will send a call, text, or e-mail to the caregiver if the medication has not been taken within 60 minutes of the scheduled time, and costs \$895. If a dose is missed, the device simply skips that dose.



*Figure 4. Precise medication window scheme used by e-pill, LLC [38]
 Courtesy of e-pill © Medication Reminders*

Another product on the market is the Philips Automated Dispensing Service, which is available at a monthly subscription price of \$59.95 a month [16]. The Philips dispenser requires pre-sorting of medicines into 60 cups, and can accommodate up to 40 days of medicine. Up to 6 doses per day can be scheduled, and there is an optional early-dose feature. The Philips can give reminder alerts for non-pill medications that are not housed in the dispenser as well. There is a program for as-needed (PRN) medications. If the dose is missed, that box's medicine will be dumped in a secure container, so the dispenser does not keep the medicines organized and separated. Alerts can be sent to the caregiver for the number of missed doses, reminders for refilling, dispenser errors, and loss of electricity. The dispenser is connected to the patient's telephone landline, so the caregiver can be contacted if a dose is missed. The patient's activity can be viewed through the online Monitoring Report. Schedules and reports can be accessed using an online portal or app.

A cheaper option is the MedReady 1700 Automatic Dispenser, which costs \$149 [15]. The medications must be pre-sorted, and the tray can hold 28 doses. Up to four doses can be scheduled per day. The MedReady 1700PRN dispenses only as-needed (PRN) medications by allowing the user to set a minimum time between doses. It costs \$159. The Medready 1750 (Landline monitored) and MR-357FL (Cellular monitored) offer cellular and landline monitoring to track medication compliance. The call center will be contacted when a dose is missed or when there is a malfunction or power outage. Then the call center notifies the caregiver via email, text messages, or phone calls. A complete compliance history is available on the website. The cellular model allows caregivers to change settings via the online portal. The 1750 costs \$197 with a \$16 per month monitoring fee, while the MR-357 costs \$297 [15, 30]. Devices with a flashing light, in addition to the audio alarm, is available for \$10 more than the regular prices.

A more advanced medicine dispenser on the market is the Livi Automated Medication Dispenser [60]. The Livi can be purchased for \$1999 or rented for between \$79-\$99 a month. The Livi can handle a variety of pill shapes and sizes, however to do so, someone must measure the pill size and fit the matching "fit kit parts" to the medication container. The Livi can hold up to 15 medicines in separated vials, and medicines do not need to be hand-sorted. This saves time from sorting medicines or loading the machine, as up to 90

days' worth of each medication can be stored. In addition, it makes adding or changing out prescriptions easier in the case of a new prescription. Doses can be scheduled up to 24 times a day, and the machine can dispense as-needed (PRN) medication. There is an optional early-dose feature. The Livi ensures only the quantity allowed for a given time-frame is dispensed, including daily do not exceed needs. The Livi can also give up to 15 additional reminders for items not stored in the Livi. The Livi has its own telephone service and can send text messages or emails to the caretaker when a dose is taken or missed, when the lockable device cover is opened, and when inventory levels are low. The Livi can generate reports and statistics about medicine adherence. The Livi web portal allows easy scheduling and real-time monitoring of medications. The Livi cloud stores medication history and adherence data for easy sharing with the physicians and other caregiver(s).

Four medicine dispensers; e-pill, MedReady, Philips, and Livi; were studied in depth to analyze the current market, and ascertain what features are currently available. Most of the automated medicine dispensers require pre-sorting of medicines. This takes a significant amount of time, and requires more frequent refilling. When new medicines are prescribed or a prescription must be stopped, it can be difficult to re-sort the medicines. In contrast, the Livi has 15 separate medicine vials. This allows for easy organization, and medicines can easily be refilled or swapped out. However, the Livi requires involved physical set-up by the caregiver to facilitate this. We planned to fill the Medspencer with separate medicine vials, to avoid the need of pre-sorting. Many of the studied products allow early doses, and have a dosage window between 30 min to 180 min (depending on the device). We planned to utilize this optional early-dose scheme for the Medspencer. Some of the medication dispensers allow for online access, where the schedules could be modified remotely. This can be helpful when a patient needs the assistance of a caregiver to modify the scheduling. The Philips and Livi dispensers provide as-needed (PRN) medications, in addition to the scheduled medicines. We also planned to provide PRN medications for the Medspencer. All the dispensers have options for phone or email alerts to the caregiver. Some of them also have online portals, where schedules and compliance reports can be accessed. Our team planned to send phone or app alerts to the caregiver, and email compliance summary reports to the physician and caregiver every few weeks. We also planned to incorporate some unique features into the Medspencer. One such feature is a rescheduling algorithm, which will adjust the dosing schedule if a dose is missed. We also planned to add options for medicines that must be taken alone. Currently the home medicine dispensers on the market are one-person use. The Medspencer handles multiple users and uses fingerprint identification to ensure the correct person gets their medicine.

4.1.2 Input from Medical Professionals

The team met and spoke with Dr. Fredesvinda Jacobs-Alvarez, a psychiatrist that heads a private practice in Orlando. We discussed the project with her to better understand the needs of physicians and patients who would use the Medspencer. Dr. Jacobs-Alvarez explained about how physicians interact with the EMR (Electrical Medical Records) system, prescribe medications, and how an alert system to identify patients with poor compliance may improve the safety and compliance for patients.

In medicine, compliance describes the degree to which a patient correctly follows medical advice. Most commonly, it refers to medication compliance, as we refer to it here. There are various cases of patients being noncompliant with their prescribed medications; patients may take more or less than prescribed or may take the wrong medication. Providing support to the patient can help them stay compliant with their medication schedule, and often results in increased compliance even after support is stopped. Thus, the Medspencer, which provides support for patients to take their medicine on time, should help improve compliance even if use of the device is discontinued. Dr. Jacobs-Alvarez also impressed upon us the importance of the physician and pharmacist's awareness of the patient's compliance, so that they can accurately gauge the patient's health and give better medical advice and treatment. After discussing this with Dr. Jacobs-Alvarez, the team decided to incorporate the summary report feature into the Medspencer. The Medspencer will automatically email a summary report to the PCP (Primary Care Physician) every two weeks to report on the patient's medication compliance.

Dr. Jacobs-Alvarez explained that medical practices have EMR systems that record information on all the patients, visits, prescriptions, diagnoses, and notes. An alert system that automatically alerts the Pharmacy and the PCP if the patient is not being compliant by sending an alert via internet to the provider's EMR would be ideal. If the patient's compliance was automatically sent to the EMR, they physician would have an easier time, as they could see all of the information about the patient summarized in one place. It would avoid any records getting lost or mixed up. Also, sending an alert on noncompliance to the pharmacy would allow the pharmacist to withhold medicines if they see the patient is not taking them, to prevent the patient from hoarding medicines or paying extra money for medicines they are not using. Unfortunately, for this Senior Design project, we do not have the permissions to access the EMR. However, if this product were to be developed after the end of Senior Design, then we could investigate gaining access to the EMR and creating and maintaining secure access. If we gained access to the EMR, then the Medspencer could send compliance/noncompliance reports directly to the EMR to be reviewed by the PCP. In addition, the Medspencer could automatically download the prescriptions and prescription schedules for the given patient from the EMR data. This would allow for easier set-up for the device, and automatic data retrieval when new medicines are prescribed.

Dr. Jacobs-Alvarez also advised us to keep all prescriptions separated. For example, if two patients are prescribed Medicine A, each patient's prescription should still be kept in a separate vial. That way, each person gets the right amount of medicine they were prescribed, and no one takes another person's medicine.

Another point that Dr. Jacobs-Alvarez brought up is that the Medspencer could be advantageous by reducing medicine production costs. A common way to help patients take their medicine on schedule is by putting the medicine in a bubble pack. However, the cost of producing the bubble pack makes the overall medicine cost higher. Also, if the schedule is changed or a new medication is prescribed, then the bubble pack must be discarded and a new one must be purchased. Using the Medspencer can reduce the medication cost because the Medspencer automatically sorts out medicines and dispenses them at the correct time. Also, rescheduling or adding new prescriptions is easy with the Medspencer. Another cost component for medicine production is the informational print-out. If the

Medspencer accessed the NDC Database, then the medicine information could be read from the Medspencer. Then a print-out would not have to be provided, which would save money.

4.2 Microcontroller

The microcontroller is the central hub of an embedded system. A complete computing ecosystem, the microcontroller contains a processor, memory, and numerous peripherals that allow it to be flexible in how it is used. These peripherals, such as digital inputs and outputs, ADC modules, comparators, clocks, timers, and communication interfaces that support UART, SPI, I2C, CAN, USB, or Ethernet, along with more specialized offerings, are implemented into a microcontroller to allow one component to perform many different functions. For the Medspencer, the microcontroller oversees recording patient and medicine data, displaying things on the LCD, handling data from the touch screen and fingerprint scanner, making alerts via speaker, and sending commands to the Wi-Fi module and servo-motor controller.

From a data analysis standpoint, the computing requirements are not heavy. The amount of data that we must record, process, and transmit is very small, and even a budget microcontroller could handle the processing. What really dictates the requirements of the Medspencer's microcontroller is the display. The display we have chosen is a 7" 800x480 touchscreen, which presents a processing and memory threshold that the microcontroller must beat.

One microcontroller we considered is the Microchip PIC32MZ DA. A member of Microchip's 32-bit PIC line and MZ family, this chip's main selling point is its 3-layer hardware-supported graphics processing unit, supported by 32 MB of RAM. Additional products from Microchip and other manufacturers were considered, but none beat the cost and features of the MZ DA.

Additionally, the idea of using a chip not designed for graphics was considered. After all, if the chip had enough RAM and a fast processor, images were just numbers, we could write our own graphics library, right? As it turns out, most chips with high enough statistics to support our screen already had some sort of graphics support, and that without certain hardware structures in the chip, we would spend most of our processing power on buffering the screen, not running our project. Not to mention that writing a custom graphics library when others have already done so and have software to support development would just be a difficult waste of time and energy.

While developing the project in Senior Design 2, we initially attempted to work with the PIC32MZ DA. However, there were many issues with this microcontroller. The graphics library and many other libraries for the microcontroller were supposed to be completed and included in MPLAB Harmony, the development platform for PIC32 microcontrollers. However, many of these libraries had issues or were unfinished or unwritten. We also had many issues when trying to utilize I2C and UART communication with the PIC32MZ DA. With all of these issues, we reconsidered our microcontroller choice and did additional research, and decided to use the Raspberry Pi Compute Module 3 Lite (CM3L) and the

ATMEGA328P-PU microcontroller. The CM3L is the main master and directly controls the LCD display, resistive touch panel, and wi-fi module. The ATMEGA328P controls the dispensing mechanism, speaker, and fingerprint scanner.

4.2.1 Research

Initial research led us to the Microchip PIC32MZ DA. A relatively new product (released in 2017), this microcontroller, while boasting some neat capabilities, seemed like overkill for the Medspencer. The display requirements for this project are very simple: some menus, some text, a keyboard, with a possible addition of simple images for icons if time and space permit. A hardware-accelerated 2D graphics processing unit (GPU, like the thing in your PC that plays Crysis) just didn't seem necessary. Additionally, the cost of the development board (\$130) was considered to be excessive [46].

For cheaper, less powerful alternatives, other Microchip offerings were first examined. 16-bit processors are usually much too small and slow to handle our display. Most are used in ultra-low-power operations, where there is no need to handle the size of data that this application requires. Even Microchip's PIC24F DA, a 16-bit microcontroller with hardware support for graphics processing, can only support a resolution up to WQVGA (480 x 272) with 16-bit color, a far cry from the scale required [1].

When looking at Microchip's 32-bit offerings, the majority still do not have the memory required to support our screen size, though they do have the throughput. The thing is that most of these chips have pretty obvious intended uses that make them specialized for particular tasks. The MX series have a variety of size and speeds to fit a range of applications. The MM is a lower-powered MCU with FPGA-like logic programming capabilities. The MK has more timers, comparators, DACs, and includes motor encoders and decoders, which makes it suited for analog peripherals control. The MZ EF has a faster processor than the MZ DA and has a specialized floating point unit, but it lacks the extra memory and graphics processing unit, making it suitable for intensive signals applications. The MZ DA is simply the member of the family specialized for graphics processing and big screens, and that is what this project is looking for [3].

Many of these microcontrollers, both 16-bit and 32-bit, have some sort of parallel communication bus: this allows them to utilize external memory to hold the frame buffer. While this is a valid option, it would result in much more complexity and more production work, and potentially higher cost as well. The PIC32MZ DA is unique in that it has a 32MB bank of SDRAM incorporated into the chip. This allows it to support such large screen sizes without needing external memory (in effect, it has external memory attached to that parallel master port, just integrated into the chip) [46].

Microchip also has a family of MPUs called SAM. Some series of SAMs are also graphics capable, and could fulfil our requirements. However, microprocessing units usually do not contain as many peripheral interfaces as MCUs do, and require external support, like various memories. This would be an increase in complexity for our project with no discernable gain, so the PIC32 MZ DA wins at Microchip.

Other manufacturers were also considered. TI doesn't make graphical LCD screen controllers; the only "graphics" products offered were for segmented displays. Many of NXP Semiconductors' 32-bit processors that were large enough to support our screen had the support to run an operating system, which is definitely overkill for this project. Renesas Electronics presented a similar situation, with their RZ/A1L being comparable to the MZ DA, but also having support for Linux, and seemed to be more difficult to acquire. Renesas does offer a number of "driver" chips, which take commands from a microcontroller and control the screen by themselves, but that also adds hardware complexity to the project and limits our ability to try new things.

Another major factor is price. The PIC32MZ DA ranges from \$10 to \$15, depending on the exact model, with the development kit clocking in at \$130. At first, this seemed quite expensive compared to a hobbyist product like a Raspberry Pi, but more research shows this is actually rather cheap as development kits go.

At the end of the day, the requirements for this project push the microcontroller into an interesting place: more powerful than the industrial and automotive applications for sensors and controllers, but not quite into the field of OS-capable MPUs powering the Internet of Things. We wanted to keep this project out of Linux to force ourselves to use lower-level tools and technologies; not everything has to be solved by buying the most impressive chip and only using a fraction of it. It seems that the PIC32MZ DA is a bit of a rare breed in the middle, and it's interesting that it was the first thing we found. Additionally, the \$130 price tag on the development board is really very cheap compared to some other dev kits out there.

Additional research and testing in Senior Design 2 pushed us to drop the PIC32MZ DA as our central microcontroller unit. Instead, we adopted the Raspberry Pi Compute Module 3 Lite and the ATMEGA328P microcontroller to take over processing for the project.

4.2.2 Microchip PIC32MZ DA

The first microcontroller that we considered using for our project is the PIC32MZ DA, which would be the main master for the entire device and would directly control each of our peripherals and touch screen display. The PIC32MZ DA has a large number of components, many of which we would not use.

This microcontroller has a three-layer graphics controller that supports 24-bit color, and is powered by a dedicated graphics processing unit (GPU), which is rare for a microcontroller. Once initiated, the GPU runs separately from the CPU to allow for graphics processing to not impede the normal processing of the microcontroller. The GPU can draw lines, fill or clear rectangles, perform bit blitting, and handle transparency. Unlike most other microcontroller functionalities, a direct hardware interface (i.e. register access) is unavailable, and development must use the provided library in the MPLAB Harmony Software [68].

Separate from the GPU is the Graphics LCD (GLCD) Controller, which is responsible for managing layers, color palettes, dithering, cursors, and directly interfacing with the screen

hardware. All of the signal timing and refresh rates related to the screen are handled by the GLCD controller. While the GPU creates and changes the content of the memory buffer, the GLCD controller will be constantly transferring that display data to the assigned data bus; in fact, it even has its own direct memory access to offload this work from the CPU entirely. This controller supports varying screen sizes, varying color depths, layers, alpha blending, programmable cursors, and gamma, brightness, and contrast support [75].

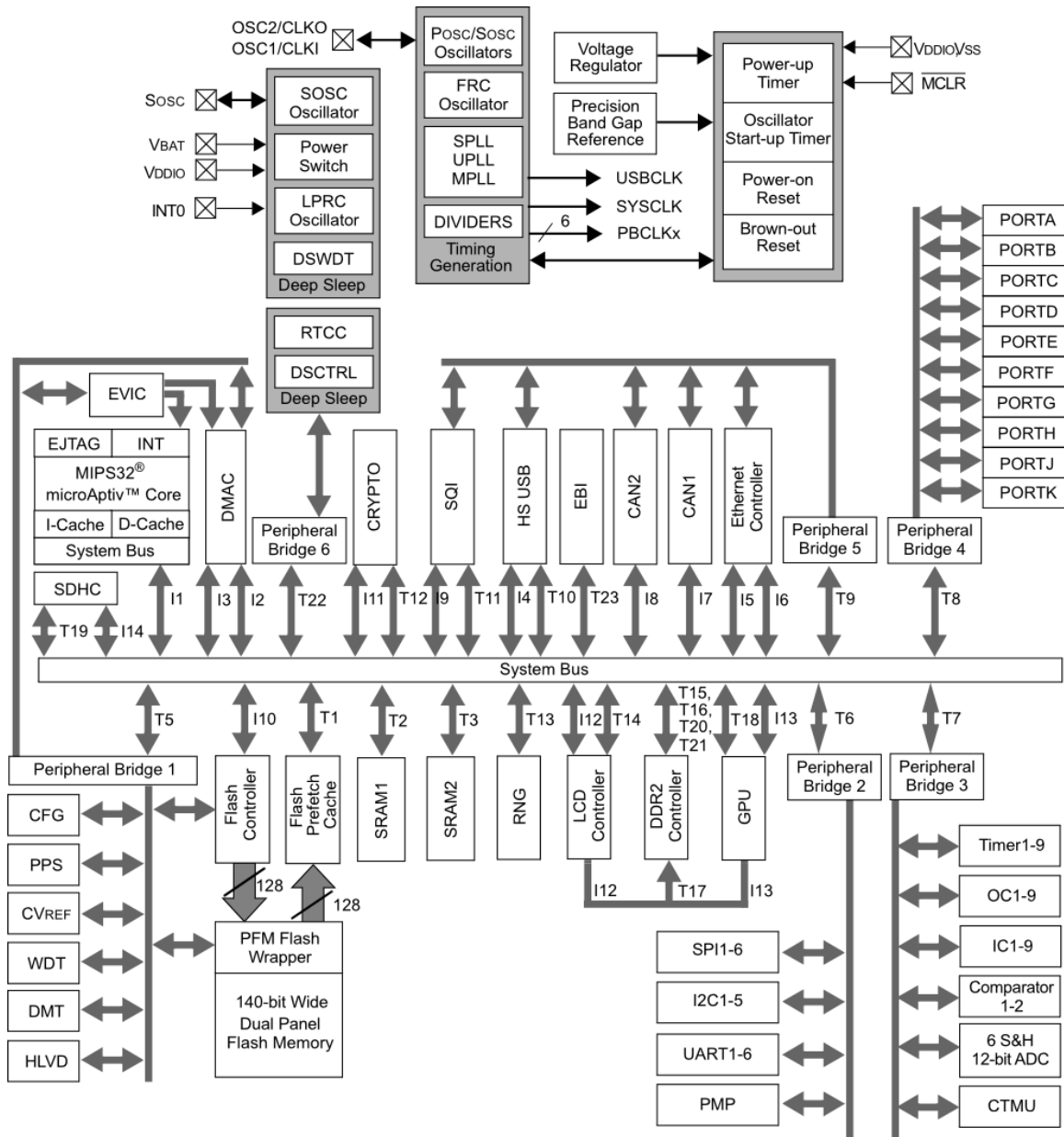


Figure 5. PIC32MZ DA Family Block Diagram [68]
 Courtesy of Microchip Technology

The memory that the GPU and GLCD controller use for the frame buffer has its own controller separate from the main on-chip memory. Depending on which model PIC32MZ DA is purchased, this controller either controls 32MB of SDRAM integrated into the chip itself, or it has pins to control banks of external SDRAM. It is this dedicated memory directly connected to the graphics controller in the MZ DA that make it so well adapted to graphics-heavy projects [68, 75].

The PIC32MZ DA has six ADC modules: these analog-to-digital converters sample a voltage and return a numeric value based on the reading. Once enabled, an ADC will be constantly sampling its input, waiting for a trigger to tell it to hold the signal steady. Once the trigger is received, the ADC module disconnects from its input, and begins its core functionality: successive approximation. Mathematically, the successive approximation register (SAR) is performing a binary search on the held voltage. Using a selection of comparators, the SAR first determines if the voltage is above or below $\frac{1}{2} V_{CC}$, then depending on that result, if the voltage is above or below $\frac{1}{4}$ or $\frac{3}{4} V_{CC}$, on and on until it reaches its smallest comparisons. The ADCs in the PIC32MZ DA have 12-bit resolution, so there are 12 layers of comparison that break down V_{CC} into $2^{12}=4096$ distinct voltage levels [68, 74].

The MZ DA also contains multiple communication interfaces: CAN, UART, SPI, I²C, USB, and Ethernet. Even though the Medspencer will require an Internet connection, it will be achieved by a separate Wi-Fi controller, not this Ethernet interface. However, that chip will have to be interfaced with, as well as the fingerprint scanner and motor controller. These communication interfaces will come in handy when interfacing with such a wide selection of peripherals. Additionally, we have the ability to add a USB port to the Medspencer, should it need computer communication, USB peripherals, or both [68].

4.2.3 Microchip PIC32MZ DA Starter Kit

The PIC32MZ DA Starter Kit is a PCB development board for the PIC32MZ DA. The board features a number of useful hardware expansions to the microcontroller, allowing for functionality testing and rapid prototyping. The Starter Kit can be purchased with a choice of one of four microcontrollers, depending on if the customer wants a model with integrated RAM and/or a crypto engine. The microcontroller comes on a separate PCB called the Daughter Card which attaches to the Development Board. Once attached to the dev board, the microcontroller has a number of peripherals that it can use [67].

The Starter Kit features USB ports for power and UART communication, a Micro-SD card slot, an Ethernet port, three push-buttons, three LEDs, an onboard programmer-debugger, a 40-pin expansion connector, and a 168-pin connector for application boards. In total, there are 5 USB ports on this board, each with a different purpose: power, host-based applications, UART serial communication, on-board debugging, and OTG/device-based applications. Finally, a fun fact we noticed: the 40 expansion pins are mapped to the exact same functions as the 40-pin expansion (J15) on the Raspberry Pi (assumption being that Microchip has recognized the number of products that have been designed for that specification) [67].

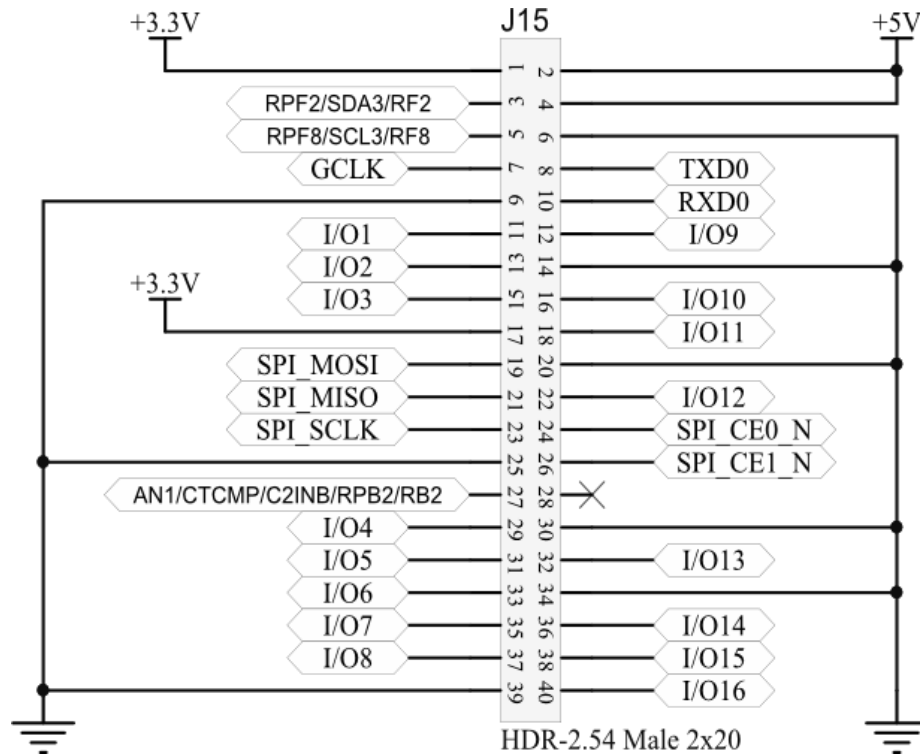


Figure 6. J15 40-pin expansion [67]
Courtesy of Microchip Technology

The 168-pin application board connector (J3) on the back of the dev board is meant to connect to some other prototyping boards that Microchip sells. However, the datasheet provides both a part number for the mated connector (Hirose FX10A-168S-SV) and a pinout, which would allow us to attach our own LCD. The LCD we have chosen also provides a pinout and a recommended connector (Hirose FH12A-50S-0.5SH), and it should be possible to wire these two connectors together to prototype our own screen connection [67].

4.2.4 MPLAB Harmony

MPLAB Harmony is a software framework of modules that provide building blocks for constructing firmware for the PIC32. It is closely associated with other tools from Microchip, such as the MPLAB Harmony Configurator (MHC) and the MPLAB X integrated development environment. The MHC is a plugin for MPLAB X IDE to allow for a graphical representation of the chosen PIC32 MCU configurations. This gives simplicity to setting up the functions of an MCU, including clocks, pins, and software modules. These tools will be used to construct the firmware for the Medspencer’s MCU [64].

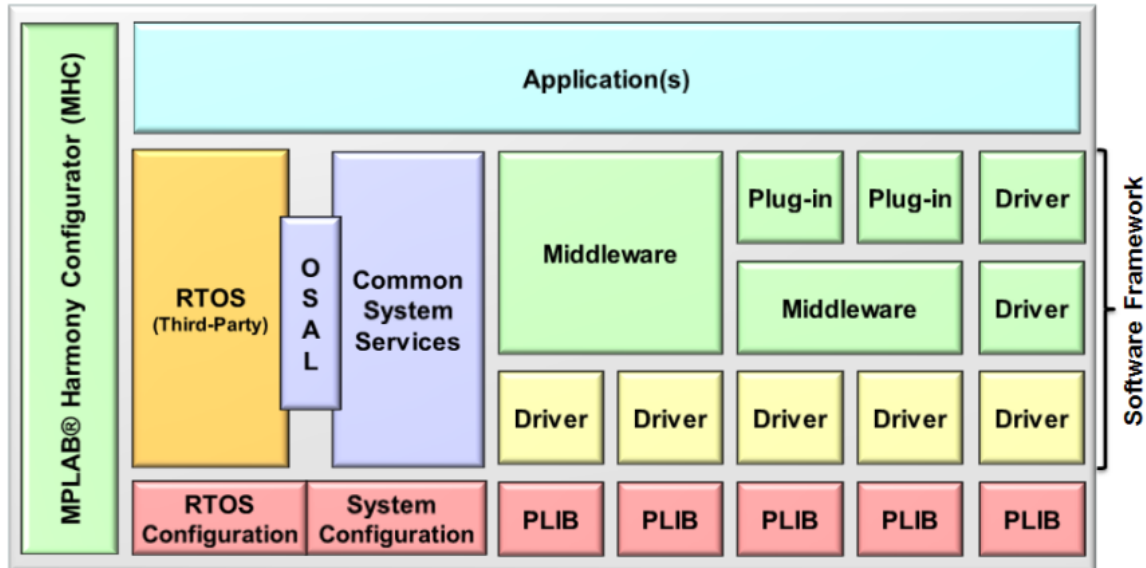


Figure 7. MPLAB Harmony generic block diagram [64]
 Courtesy of Microchip Technology

MPLAB Harmony is a framework of libraries, supported by device drivers, that allow for the development of applications for the PIC32 family of microcontrollers. Closest to the hardware are Peripheral Libraries (PLIB), simple libraries that hide register details to the drivers above to allow more portability between hardware. The device drivers provide an interface between the peripheral and the software, allowing access to functionality through a consistent set of functions. These manage all access to a peripheral, and so monitor concurrent accesses and managing the state of the peripheral.

This multi-layered approach is designed to allow for portability between hardware, which is usually impossible when working closer to the hardware, as is normal with MCUs. The modular approach also encourages code reuse and usage of pre-made libraries, which saves developers time and difficulty.

A different category of module is the middleware library. Some peripherals are too complex to control directly and require extra processing power or interpreting complex protocols. Examples of these complex protocols include USB interfaces, network connections, or LCD controllers. These middleware stacks are built on device drivers and system services, and allow for a more abstracted control of these services. For the Medspencer, the LCD controller middleware will be of special importance.

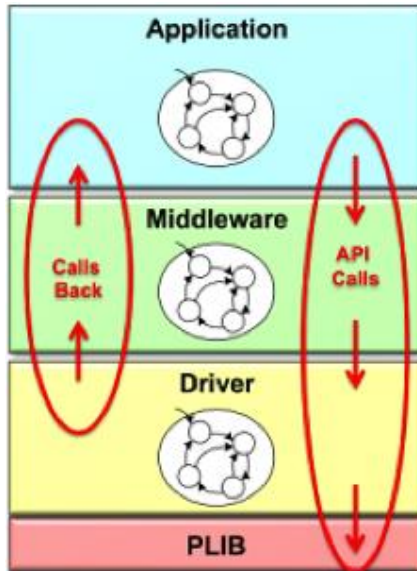


Figure 8. MPLAB Harmony Middleware diagram [64]
 Courtesy of Microchip Technology

Often, a firmware will have more than one module that needs to access a system peripheral, such as a timer. If a module attempted to set the functionality of a timer while it was in use for another module, then they would likely catastrophically interfere. However, a layer of abstraction allows for a System Service to control the peripheral, and handles the needs of multiple modules. The example system timer now has the responsibility to fulfil the modules' requests and keep them from interfering with each other.

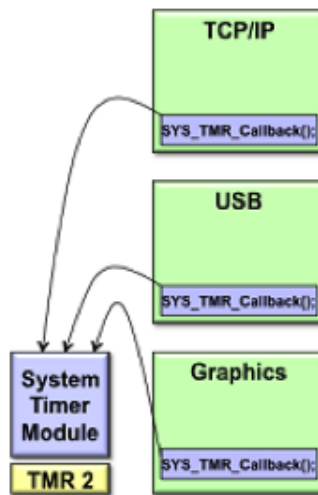


Figure 9. MPLAB Harmony System Service example [64]
 Courtesy of Microchip Technology

Overall, the MPLAB Harmony firmware solution provides support for complex functionality and granular hardware abstraction, allowing unparalleled flexibility in hardware choice and multi-module support.

4.2.5 ATMEGA328P Microcontroller

The ATMEGA328P is a popular microcontroller for embedded designs. It is a low-power 8-bit microcontroller based on the AVR enhanced RISC architecture, and with 32kB of flash memory, 1kB EEPROM, and 2kB SRAM, it is suited to applications where it connects and controls low-power components. There are 23 programmable I/O lines, with USART, PWM, SPI, I2C, and ADC interfaces available. Rugged and reliable, this microcontroller can retain information for 100 years at a temperature of 25 degree Celsius with a projected data retention failure of less than 1 PPM. When the ATMEGA328P is in power-save mode the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping [88].

4.2.6 Raspberry Pi Compute Module 3 Lite

The Raspberry Pi Compute Module 3 Lite (CM3L) is a System on Module (SoM), which is a PCB containing a processor, memory, and supporting power circuitry. It conforms to the mechanical specification for a 200 pin SO-DIMM module, and therefore fits with many DDR2 SO-DIMM sockets available on the market, though it is obviously not electrically compatible with them. The CM3L contains a BCM2837 processor, and the software includes the ARMv8 instruction set and Linux software stack. 1GB RAM is available, and an SD card up to 64 GB can be fitted using the SD/eMMC interface. The CM3L has 48 GPIO pins and supports I2C, SPI, UART, SD/SDIO, HDMI, USB, and DPI communication interfaces [89].

The Compute Module is effectively a stripped-down Raspberry Pi Model 3, with no connectors or 5v power regulation, and all I/O pins exposed onto the socket. This makes it much less usable out-of-the-box, but much more flexible for designers who can design their own PCB.

4.3 Display

The display is the main communication channel between the system and the user. Whether it is general overviews of functionality, live updates of a specific task, or just pictures of cats, the display is how humans can see what the system is doing. Not only does the system communicate to the user through the screen, but with the addition of a touch-sensitive panel over the display, the user can now communicate back. Infinitely more flexible than a keypad or a bank of switches, a touch screen lets the system choose exactly what information it requires from the user, then presents them with a tailor-made interface to communicate through. Buttons, switches, knobs, keypads, even pen and paper can be emulated in the touchscreen. This allows for unparalleled flexibility in intercommunication between machine and man.

Consequently, the screen is a very important piece of hardware for the Medspencer. An administrator will need to enter strings of text into the system for patient names, medicine ids, and email addresses. However, the admin will also have to set up schedules, and that will take a very different kind of interaction than plain text. Additionally, the end users of this system will need to interact to ask for medication, or to answer alerts, or any number of other things. This broad scope of interaction is what makes the touchscreen such an important piece of the Medspencer: it acts as many different kinds of human interfaces.

4.3.1 Technical Background

A screen is simply a large array of pixels, each with the capability to show a color. The amount of unique colors that can be shown on a screen is called its color depth, and this is directly related to the number of bits that indicate the color of a single pixel. The more bits that are used to define a color, the more variety of colors can be reproduced. To communicate this color information to a display, there is a set of data lines that are mapped to the color data bits.

To display an image on the screen, you first set the screen's color data lines, to the value of the color you want the first pixel to be. Then, the clock signal pulses, which causes the screen to move on to the next pixel, and it reads in the value on the color data lines again. As the pixel being updated moves across the screen and then down to the next row, the data lines are set to the color for that pixel, and eventually fills the screen with colors. Now, obviously, this must happen very fast, as screens have a large number of pixels that must be refreshed multiple times per second.

For the microcontroller producing this pixel data, it can be hard to know which pixel needs to be what color. Because the pixels are transmitted line-by-line, complex shapes, color shading, and multiple layers of objects could be very computationally difficult to process for each individual pixel, especially if that pixel's value depends on the values of pixels around it (eg. a blurring effect). The solution to this problem is called a frame buffer. A section of memory is set aside to contain a copy of the screen. This is a bitmap with the value of every pixel recorded. Now the microcontroller can draw anything it wants onto this buffer before it feeds it pixel-by-pixel to the screen.

4.3.2 Frame Buffer Space Saving Techniques

The screen chosen for this project is of resolution WVGA, which means it is 800 pixels wide by 480 pixels tall. It also supports 24-bit color. If a frame buffer was used for a full-color screen this size, the microcontroller supporting it would need over 1MB of RAM ($800 \text{ px} * 480 \text{ px} * 3 \text{ B}$). This is much more data than any budget microcontroller was intended to have to deal with. However, techniques exist to limit the amount of memory needed for the frame buffer.

	1	2	...	800
1	FF0000	C39B6F		D4CE4C
2	3399FF	008000		2F11E5
...				
480	649A67	D76FF7		9604C8

Figure 10. Basic 24-bit color frame buffer; Total size: 1.099MB

The simplest option is to limit the size of the stored colors. Instead of setting aside 24 bits for each pixel, only use 16 bits. A common way to partition 16 bits into red, green, and blue components is the ratio of 5:6:5. This means a microcontroller only needs 16 output data lines: the extra wires for the screen's color data lines would be tied to ground to signify 0. Only the most significant bits of color data would be saved, squeezed into two bytes. This uses 2/3 the amount of memory as a full frame buffer ($800 * 480 * 2B$) and gives 65,536 possible colors, but some color depth is lost.

	1	2	...	800
1	F800	C4CD		D669
2	34DF	0400		289C
...				
480	64CC	D37E		9039

Figure 11. 16-bit color frame buffer; Total size: 0.732 MB

A more complex possibility is to use a color lookup table. This involves a frame buffer that doesn't store full 24-bit colors, but only an index to a table that stores the full colors. A microcontroller using this strategy sends data to all 24 color data lines on the screen, so the screen can show full-precision colors, but the frame buffer can only store a subset of those colors. A lookup table with 8-bit color entries would allow the use of a frame buffer that takes half the memory as above and can only produce 256 different colors, but can utilize all of the color depth of the display.

	1	2	...	800
1	0	1		5
2	3	4		2
...				
480	5	0		6

0	FF0000
1	C39B6F
2	2F11E5
3	3399FF
4	008000
5	D4CE4C
6	9604C8

Figure 12. Frame buffer with 256-color lookup table; Total size: 0.366 MB

There is more cleverness that can be applied to the color lookup tables. If all that is going to be displayed on the screen are flat color menus, text, shapes, etc., then a full gamut of colors is not needed, and the limited palette is acceptable for the given purpose. Even some anti-aliasing and smoothing between colors could be displayed, if properly accounted for in the color table. If a full-color image was going to be displayed on the screen, then it should be pre-processed to determine the most-used colors, and come compressed with a color table. After all, if only a fraction of real colors can be displayed at once, then saving colors that aren't used in the image is a waste of color gamut. Indeed, depending on the size of the color table, the amount of memory available, and the application requirements, it wouldn't be unfeasible to save multiple color tables and treat them as different palettes for different uses (e.g. This object uses palette 1, while that object uses palette 2, etc.). The developers for cartridge-based video game consoles were masters at these sorts of space-saving tricks.

However, with all that research done, we decided to use a microcontroller large enough to handle the screen at full size. In the past, clever methods like these were required to store visual data in small memories, but given that microcontrollers exist today that not only can run a screen of this size, but also provide powerful graphics processing that is easily usable, these techniques are more trouble than they're worth.

4.3.3 Innolux AT070TN90

The screen requirements for the Medspencer are fairly straightforward: it needs to be big enough to use as a keyboard, and the graphics don't need to be too complicated. The screen will be the main human interface for the Medspencer, used for menu systems, a keyboard, and visual alerts. Therefore, the most basic use of the screen will be to show words and boxes: potentially a lot of them, possibly overlapping, and preferably in fun colors. These basic requirements are pretty simple.

However, the MCU chosen to power this project is quite powerful, and will allow for much more than simple boxes and words. With transparencies, color blending, layers, and lots of storage, there are a number of possible ways for the display to be jazzed up (transition effects, full-color images, etc.). In reality, however, style and beauty come second to getting the thing to work, so plans of an aesthetic nature will be made once the team has excess time on their hands.

The screen chosen for this project is an Innolux AT070TN90, a Chinese display commonly found in consumer electronics. There are a number floating around Amazon, eBay, and Alibaba, so they are pretty cheap (~\$20). They have a WVGA resolution (800 x 480), support 24-bit color, and can be found with a resistive touch panel incorporated into the screen. The datasheet contains a pinout for the 50-channel flexible flat cable (FFC), which contains all of the color channels, power connections, and timing signals for the display. The datasheet also contains specifications for the exact timing to send the screen color data, though thankfully the PIC32MZ DA's LCD controller will handle the screen's timing requirements if you set it up properly.

The datasheet gives a recommended connector for the FFC, a Hirose FH12A-50S-0.5SH. According to the pinout, while many of the lines are simply data lines, with voltage levels that should switch between 0 and ~3.3, a number of wires are power lines, and require much higher voltages than digital hardware usually calls for. This will certainly have to be taken into consideration when designing our power distribution for the product, as the Gate voltages go outside normal digital operating ranges.

Table 3. Typical Operation Conditions [14]

Item	Symbol	Values			Unit
		Min.	Typ.	Max.	
Power voltage	DV_{DD}	3.0	3.3	3.6	V
	AV_{DD}	10.2	10.4	10.6	V
	V_{GH}	15.3	16.0	16.7	V
	V_{GL}	-7.7	-7.0	-6.3	V
Input signal voltage	V_{COM}	3.6	3.8	4.0	V
Input logic high voltage	V_{IH}	$0.7 DV_{DD}$	-	DV_{DD}	V
Input logic low voltage	V_{IL}	0	-	$0.3 DV_{DD}$	V

4.3.4 Resistive Touch Screen

We decided to use a resistive touch panel. The alternative technology for touchscreens is a capacitive touchscreen. The resistive touch panel relies on mechanical pressure, while the capacitive touchscreen measures changes in electric field due to the conductivity of your fingers. We chose to use the resistive touchscreen because of its lower price, and the fact that it reacts to multiple kinds of touch, whether it's a finger or a stylus.

This display senses touch through a device called a resistive touchscreen. The touchscreen has three layers: one conductive layer with electrodes at across the top and bottom (Y1, Y2) of the screen, a spacer layer in the middle, and another layer with electrodes on the left and right (X1, X2) of the screen. Each of these electrodes has a wire connected to it that will connect to the microcontroller.

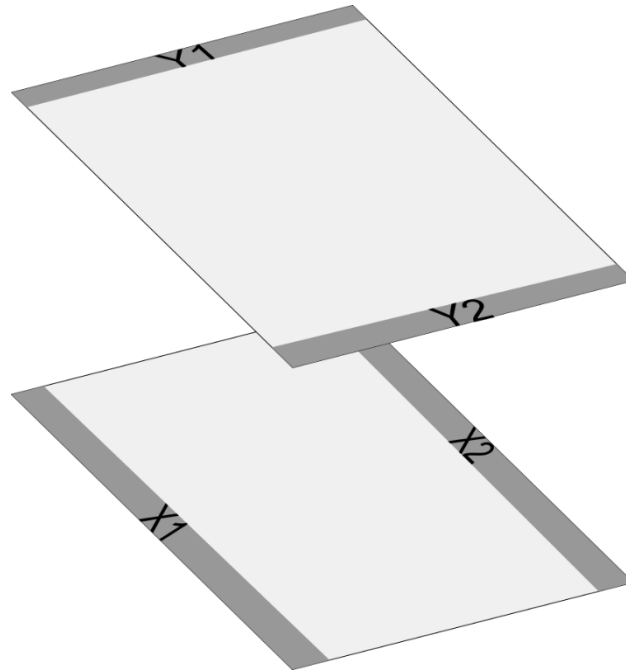


Figure 13. Resistive touchscreen's layers and electrodes

When the screen is touched, the top layer is pressed against the bottom layer and forms an electrical connection at that point. To measure the Y component, two different voltages are placed at Y1 and Y2. This causes a voltage gradient across the Y layer, and the Y component of the touch location is a voltage in that gradient, relative to its location. At the point, the electrical connection between the X layer and the Y layer make the X layer's voltage equal to the point touched on the Y layer. The voltage of X1 or X2 can be measured, and the voltage of Y1 is obtained. This is in some sense similar to a voltage divider.

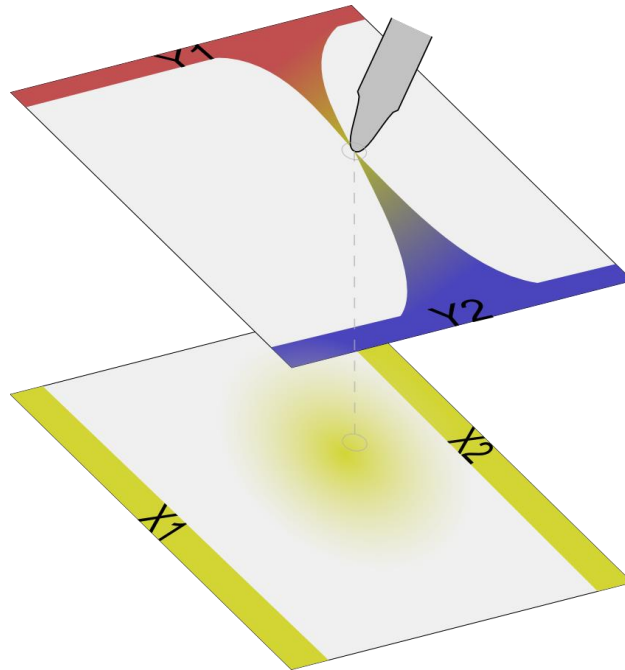


Figure 14. Measuring the Y component of a touch

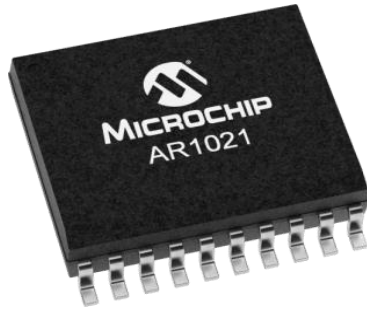
This process can be repeated with the layers swapped to obtain the X component of the touch, and with these voltage measurements, the coordinate of the touch can be calculated. This type of screen is simple and robust, and can detect pressure of any sort, not just a finger.

4.3.5 Resistive Touch Screen Controller

The touch panel requires an ADC module, and touch coordinates are calculated by measuring the voltage gradient across the touch panel. We considered three different possibilities to control the touch panel.

The first method we considered is using the ADC modules on the PIC32MZ DA microcontroller to directly supply power to, measure, and calibrate the touch panel. This was our initial plan, but since the PIC32 didn't work for our LCD display specs, we decided to forego this option.

The method that we decided to utilize is to control the resistive touch panel using the AR1021 Resistive Touch Screen Controller [90]. This controller processes the touch data and delivers calibrated touch coordinates to the host MCU. It communicates with the host over I2C communication.



*Figure 15. AR1021 Resistive Touch Screen Controller
Courtesy of Microchip [90]*

For the Raspberry Pi CM3L, the touch panel may also be controlled using USB. Utilizing the USB bus requires more hardware circuitry and power overhead, however it's more flexible. Since we're planning to have a self-contained manufactured product, we decided it's more economically efficient not to use the USB method.

4.4 Wi-Fi connection

The following sections show the research that was done to give the microcontroller Wi-Fi connection capabilities, and the different approaches that were considered to solve the problem. Having a fast Internet connection is important for any piece of technology these days, so a quality solution was needed.

In searching for a wireless frequency interface module, the module must be secure in the regulations and laws under the Section 3.1.2 and 3.2.2, and send and receive communication signals using protocols and regulations standards for internet connectivity.

The product that was initially considered is the CC3220 SimpleLink Wi-Fi Wireless and Internet-of-Things Solution, a Single-Chip Wireless MCU. This contained the requirement for the device that is being created with added features. The Microcontroller unit has up to 27 GPIO Pins, an Application Microcontroller Subsystem and Wi-Fi Network Processor (NWP) Subsystem, high-level secure system Software Tamper Detection, cloning protection and secure boot. In addition, for the future project the added feature of Internet-of-Things. In the Application microcontroller subsystem, a Cortex-M4 processor is used which has a 64-bit ARM Thumb instruction, an upgrade from previous education courses. Also in the application subsystem is advanced high-performance bus (AHB-Lite) interfaces: system bus interfaces and the low-cost debug solution featuring like Debug access to all memory and registers in the system, including access to memory-mapped devices, access to internal core registers when the core is halted, and access to debug control registers even while SYSRESET is asserted [77].

The other option we explored, which is the option we ended up going with, is the ESP8266EX Wi-fi module by Espressif. This chip is a popular Wi-fi microcontroller with many cheap versions on the market. This module can perform either as a standalone application or as the slave to a host MCU. The ESP8266EX can interface to a microcontroller using SPI/SDIO or I2C/UART interfaces. It compactly integrates antenna

switches, RF balun, power amplifier, low noise receive amplifier, filters and power management modules, and requires minimal external circuitry. The chip integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor and on-chip SRAM [93, 94].

4.5 Fingerprint Scanner

This project requires patient authentication to ensure that only approved patients will use the Medspencer, and that patients receive only the medications that were prescribed to them. Passwords or pins can be easily stolen or forgotten. People who are elderly or have mental issues may also have a hard time remembering them. To these ends, we will utilize a fingerprint scanner. As every human has their own unique fingerprint, fingerprint scanning works as a secure method of identification. Thus, the fingerprint scanner provides an easy and secure way for patients to identify themselves to the Medspencer.

The two major technologies used for fingerprint scanning include optical scanning and capacitive scanning. Capacitive fingerprint scanners generate fingerprint images using electrical current. The sensor is made of one or more semiconductor chips containing an array of cells; each cell contains a capacitor. The sensor is connected to an integrator, a circuit built around an inverting operational amplifier. When placed on the sensor, the surface of the finger acts as a third capacitor plate. The varying distance between the sensor and the finger, caused by fingerprint valleys and ridges, causes a change in the total capacitance. Thus, the capacitor under a ridge will have greater capacitance than if it were under a valley. To scan the finger, a fixed charge is applied to the integrator. The capacitors affect the amplifier's voltage output, which can be measured to determine whether the voltage is characteristic of a ridge or a valley. By reading every cell in the sensor array, an overall fingerprint image can be produced [47].

Meanwhile, the touch surface of an optical fingerprint scanner is a glass surface, and underneath the glass is a light source (such as a light-emitting diode or a layer of phosphor) which illuminates the touch surface for digital capture. The light illuminates and reflects off the finger, and the reflected light passes onto the charge coupled device (CCD), which is an array of light-sensitive diodes called photosites. Each photosite generates an electrical signal in response to photons. The conversion of photons into electrons by the individual photosites allows the information to be stored and transformed into pixels, which collectively make up digital images [47,81]. An advantage of capacitive scanners is that they require a real fingerprint-type shape, which makes them harder to trick. They are also more compact than optical sensors, since the semiconductor chip is smaller than a CCD unit [47]. An advantage of optical sensors is lower maintenance; capacitive scanners are more sensitive, as they require constant surface protection from electrostatic discharge (which could permanently damage the sensor). Other advantages of optical sensors include lower price, larger imaging surface areas, and higher resolution [81].

While biometric fingerprint analysis is an effective way to secure systems, it is not infallible. There are different methods by which a user could trick a fingerprint scanner. Optical scanners sometimes cannot tell the difference between a picture of a finger and the finger itself. Meanwhile, capacitive scanners can be fooled with a mold of a finger. To

make systems more reliable, additional pulse and heat sensors could be used to verify that the scanned finger is real, live finger. Another method to make systems more reliable is to combine biometric analysis with another security measure, such as a password. However, for the Medspencer, we want our system to be easy to use for the elderly and other people with possible mental or psychological issues. These people may have trouble remembering passwords. After considering our target consumers, we decided fingerprint scanning will be our primary method of identification. To make the device more secure however, we could add pulse or heat sensors [47].

4.6 Motors

On choosing a motor between brushless DC motors, servo motors or inverters? When it comes to speed control, the common choice is a three-phase induction motor that controls speed by use of a general-purpose inverter. For many, this may be the natural choice because it allows you to freely set a temporary driving speed that you can change in the future. If speed, torque, or improved controls are needed, upgrading to a servo motor is certainly an option. However, considering the relatively low cost of the inverter driven three-phase motor, by changing to a servo motor you will face the problem of increasing expenditure. The servo motor stabilizes speed and solves the problem of synchronization of multiple conveyors, but taking cost into consideration, compromises in your setup would almost certainly be necessary.

The position of the brushless DC motor is, in simple terms, between the inverter and the servo motor. It is a motor dedicated to speed control that controls speed as effectively as a servo motor for a lower price, closer to that of an inverter.

Most AC induction motor inverters do not communicate with the motor, however recently with additional encoders or analog signal devices added separately to the motor or moving parts, this is becoming an option. The drawback is the additional costs and tuning to a typically low cost solution. The majority of today's inverters still run open loop. Under the open-loop system, when the load changes, the actual speed does not follow the command. This is why the speed changes (slower when more load is added) and depending on the load, why speed synchronization over multiple axes is difficult. Also, because torque is lower in an AC motor at high or low speeds from rated speed, which is an inherent torque characteristic of the three-phase motor, it is difficult to obtain both the speed and torque you want at the same time. The inverter is effective when the operation continues at a fixed speed, but it is not ideal for multi-speed operations. Heat is also a common component to AC induction motors. To combat this a cooling fan is attached to the back of the motor. Due to the AC induction motors design, heat rises when the motor runs slow and the cooling fan is runs slower. Inversely, when the motor is running at high speed, heat from the windings is also increasing.

Both the servo motor and the brushless DC motor adopt a PM motor (permanent magnet is used for the rotor) and come standard with closed-loop speed control, where the motors operation status is fed back to the driver. This ensures the motor speed remains constant at the commanded level and enables the speed of two motor axes to be synchronized. Additionally, flat torque is produced whether operating at high or low speed. Even if the

load changes, at whatever speed is commanded, stable driving speed is ensured. This means these two motors are highly effective in situations in which the inverter struggles.

Needless to say, the servo motor is different from the brushless DC motor. Generally speaking, the difference is that one is capable of high performance and all around speed control, while the other is dedicated solely to speed control [23].

4.7 Speaker

One of the functions that we plan to include in the Medspencer is the function to remind the patient when it is time to take their dose of medicine. In order to alert the patient, we plan to utilize a buzzer or speaker. When it is time for a patient to take a medication dose, the speaker will play a simple sound or beep to alert the patient.

A cheap and commonly used audio device is the piezo buzzer, and it can be used to make simple beeps, tones, and alerts. The working principle of the piezo buzzer is based on the inverse piezoelectric effect, which was discovered in 1880 by Jacques and Pierre Curie. The piezoelectric effect states that when certain solid materials (such as crystals and certain ceramics) experience applied mechanical stress, electric charge will accumulate. The inverse is also true; an applied electric field can result in the generation of mechanical strain [69].

A piezo buzzer makes use of a piezoelectric element such as a crystal or ceramic. When an AC voltage with a frequency in the kHz range is applied across the piezoelectric element, the element undergoes mechanical stress and deformation, resulting in some oscillation. This oscillation will occur at the same frequency as the AC signal and will result in an audible sound. The usable frequency range depends on the materials that are used in the piezo buzzer. Piezo buzzers generate sharp sounds, so they are typically used as alarms or alerts [49].

Another possible audio device we may use is a small passive speaker. This speaker employs a passive radiator and an active loudspeaker (main driver). The loudspeaker is an electroacoustic transducer, and converts an electrical audio signal into a corresponding sound. The most common type of speaker is a dynamic speaker, and it uses a voice coil, which is a coil of wire suspended in a circular gap between the poles of a permanent magnet. When an alternating current is applied to the voice coil, the coil is forced to move back and forth, according to Faraday's law of induction. This causes a conically shaped diaphragm that is attached to the coil to move back and forth, which pushes on the air to create sound waves [62]. The passive radiator is also known as a drone cone. The passive speaker uses the sound trapped in the enclosure to excite a resonance that allows the speaker to output deeper pitches. The resonance frequency is determined by the passive radiator's mass. Passive radiators are commonly used in home stereo speakers, subwoofer cabinets and car audio speaker systems [66].

4.8 Power

This section details the research that went into determining how to provide power to our device. This includes considerations such as the power source, transforming and AC-to-DC conversion, power regulation, filtering, and low power modes.

4.8.1 Power Supply

To power the Medspencer, we use standard city power. In the United States, the AC power standard is 120 V and 60 Hz. To power a device using this AC power, first the power must be scaled down to safer levels using a transformer. Then we must convert the AC power to DC power. Finally, the voltage and current levels must be scaled down according to the ratings of each component of the Medspencer.

To build an AC to DC converter, first it requires a transformer to decrease the AC voltage amplitude. Then the secondary side of the transformer is connected to a full wave rectifier circuit, which makes the entire voltage waveform positive. A capacitor is used to filter the voltage waveform and convert the voltage to a constant DC output. The capacitor filtering accomplishes this because as the AC input voltage decreases below its maximum value, the capacitor discharges, which keeps the output voltage at a roughly constant value. A ripple voltage can be observed in the output voltage waveform. The transformer-based AC to DC circuit is shown below, and it includes the transformer, full wave rectifier, and filtering capacitor [37, 59].

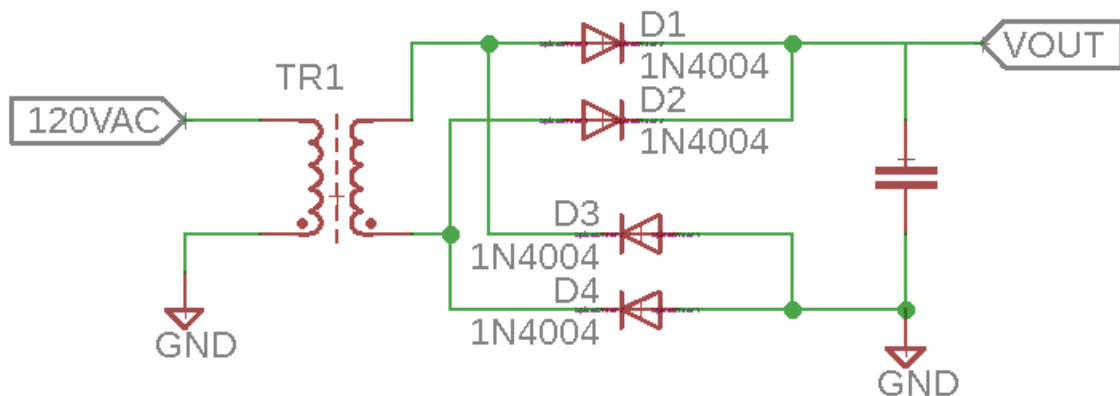


Figure 16. Filtered full wave rectifier

4.8.2 Power Regulation and Filtering

In order to power each individual component of the Medspencer, the voltages and currents must be scaled down to match the ratings of the components. We used voltage regulators to scale down the voltage, to avoid damaging any of our hardware components.

To limit the voltage input to components, we purchased voltage regulators, such as switching regulators or low-dropout (LDO) voltage regulators. Voltage regulators can be placed in parallel to the load, connected to the power supply and ground. This keeps the load input voltage regulated to safe operating levels.

The power input to each component should be filtered of noise. To filter out noise, capacitors can be placed in parallel to a load; these are called decoupling capacitors or bypass capacitors. Decoupling capacitors suppress high-frequency noise in the power supply, which may cause damage to sensitive hardware components. The current-voltage relation for a capacitor is $I(t) = C \, dV(t) / dt$. This equation shows that for constant voltage DC operation, there is zero current, and the capacitor acts like an open circuit. For high frequency power supply, the capacitor acts like a short circuit. Thus, using a decoupling capacitor allows high frequency noise to be shorted to ground, while the DC voltage supply still goes to the load.

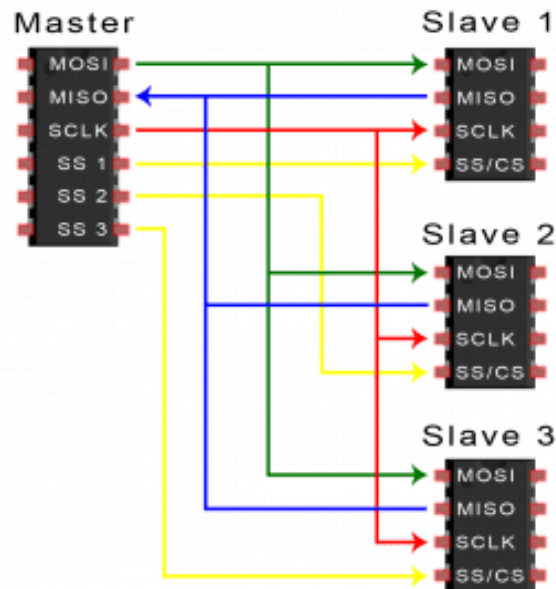
4.9 Communication Protocols

This section discusses the different communication protocols that may be used for the Medspencer and compares them. The relevant communication protocols that we will discuss in this section include SPI, UART, and I2C.

First, communication channels can be simplex, half duplex, or full duplex. Simplex communication channels can only send information in one direction. Half duplex communication channels may transmit data in both directions, however it can only communicate in one direction at a time. Full duplex communication channels may transmit data in both directions simultaneously [65]. Data can be transmitted either in parallel or serial form. For parallel communication, bits are sent at the same time through separate wire connections. For serial communication, the data bits are sent serially through a single wire [18].

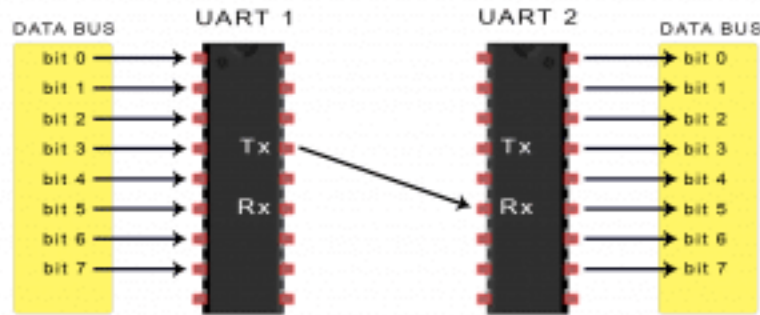
SPI stands for Serial Peripheral Interface, and it is synchronous and full duplex. Unlike UART or I2C communication, SPI allows for data to be transmitted continuously, as data is not sent in packets. SPI communication also allows for a single master to control multiple slave systems. The speed of data transfer and sampling is determined the clock frequency, which is configured by the master system. The master can communicate with multiple slaves by setting the slave select pins. Slaves can be wired in parallel (if there are

multiple slave select pins), or they can be daisy-chained (if only one slave select pin is available). The connections for parallel slaves are shown below in Figure 17. Data is transferred serially through MOSI (master output/slave input) and MISO (master input/slave output) lines. In total, four connections are required: clock, slave select, MOSI, and MISO. Disadvantages of SPI include that there is no acknowledgement that data was received, or error checking (like with a parity bit) [18].

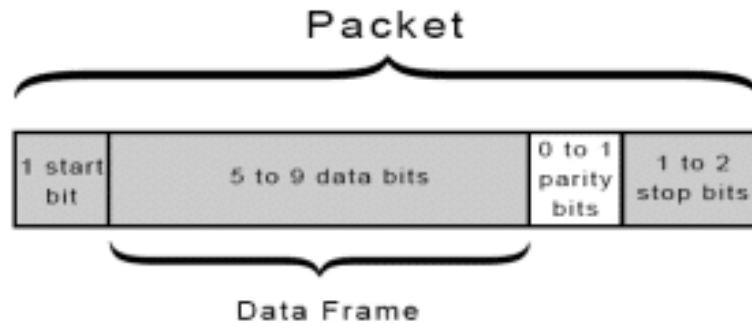


*Figure 17. SPI communication using multiple slaves connected in parallel [18]
Courtesy of Circuit Basics*

UART stands for Universal Asynchronous Receiver/Transmitter, and consists of a physical circuit connection for a microcontroller or IC. UART communication is asynchronous and full duplex, and can transmit and receive serial data. Two UARTs communicate directly between each other using just two wires. The TXD pin of one UART is connected to the RXD pin of the other UART, as shown below in Figure 18. First the transmitting UART converts parallel data into serial form, then transmits it to the receiving UART, as shown below in Figure 18. The transmitting UART adds start, parity, and stop bits to the transmitted data packet, which define the beginning and end and allow for error checking. The organization of the data packet is shown in Figure 19. The receiving UART reads the incoming bits at the Baud rate frequency; both UARTs must operate at the same Baud rate [20].

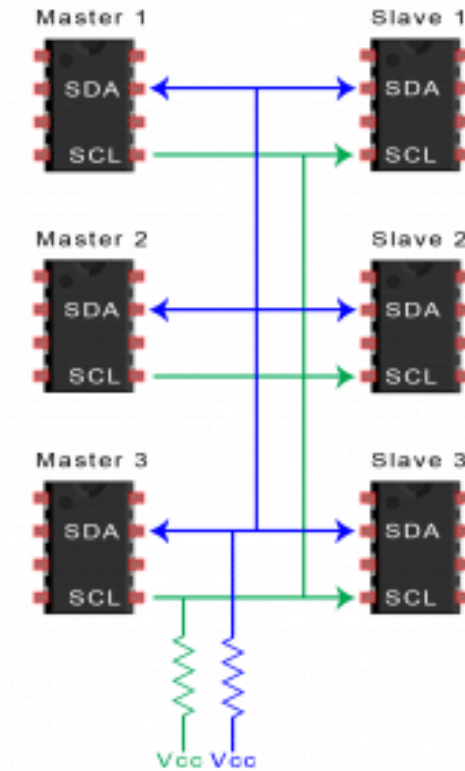


*Figure 18. UART communication connections
Courtesy of Circuit Basics [20]*

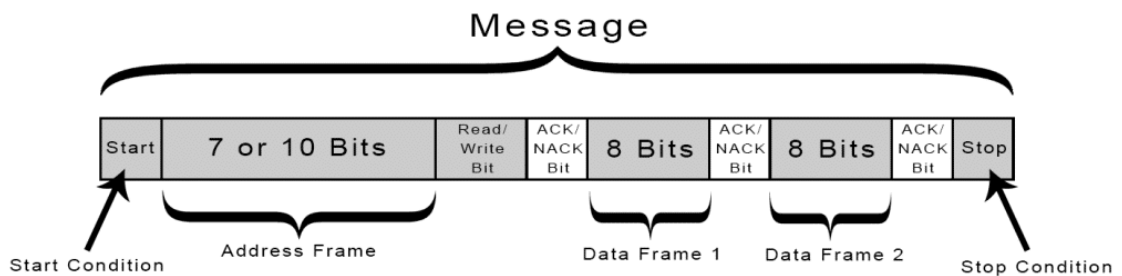


*Figure 19. Transfer of data packets using UART communication
Courtesy of Circuit Basics [20]*

I2C stands for Inter Integrated Circuits, and it is synchronous, half duplex, and allows for multiple masters and multiple slaves. Similar to UART, I2C only requires two wires to transmit data between devices. One line allows the master and slave to send and receive serial communication. The other line carries the clock signal, which is controlled by the master. The connections required for multiple masters and multiple slaves is shown in Figure 20. Data is transferred in messages, and they include start and stop conditions as well as acknowledge bits. The address frame specifies the slave to be communicated with; each slave compares the address sent from the master to its own address, and will return an acknowledge if it matches. The overall organization of a data message for I2C is shown in Figure 21. An advantage of I2C communication is that it supports multiple masters and slaves, and the acknowledge bit confirms whether frames are transferred correctly. However, I2C communication has a slower data transfer rate than SPI communication, and requires more complicated hardware implementation [19].



*Figure 20. I2C communication connections using multiple masters and multiple slaves
Courtesy of Circuit Basics [19]*



*Figure 21. Data message organization for I2C communication
Courtesy of Circuit Basics [19]*

5.0 Project Hardware and Software Design Details

This chapter discusses the design details of our Medspencer device. The first half discusses the hardware design of the Medspencer and provides details on the component choices and designed circuit schematics. When looking into specific hardware components, we considered the project requirements, the available technologies, and the pros and cons of all the available products for purchase. The second half of the chapter discusses the software choices and designs.

5.1 Hardware Design

Section 5.1 details the physical hardware components, interfaces, and electrical circuit connections that will be used to build the Medspencer. The first section presents an overarching view of all the hardware components used for the Medspencer. The specific hardware components that we have chosen and their characteristics and ratings are presented, as well as detailed circuit schematics that will interface the power supply, MCU, and peripherals.

5.1.1 Hardware Block Diagram

For the senior design project, the first task of our team was to write an initial “Divide and Conquer” document. In this document, we included a hardware block diagram that illustrates the general design plan ideas and necessary hardware components. We split up the project based on all of our physical hardware components, in order to ensure that each team member contributed equally.

For this project, the hardware components that are required include the microcontroller, WVGA touchscreen display, Wi-Fi module, fingerprint scanner, speaker, and motors. The components are displayed below in the hardware block diagram in Figure 22. The components are color coded according to which team member the responsibility is assigned to. In addition, the progress status is indicated within each block.

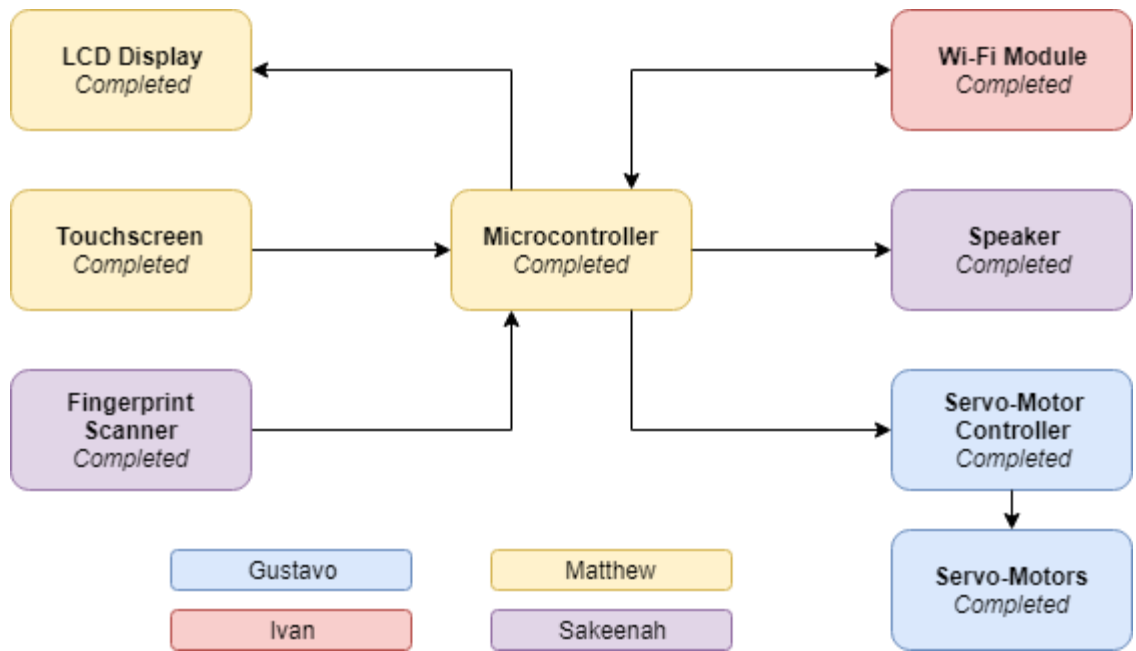


Figure 22. Hardware block diagram showing team member responsibilities

Laying out this initial block diagram allowed us to create a good starting point for conducting research and designing the Medspencer. After further research and testing, the design changed and matured in order to facilitate the effective and timely completion of the project. The modified hardware block diagram is displayed below in Figure 23.

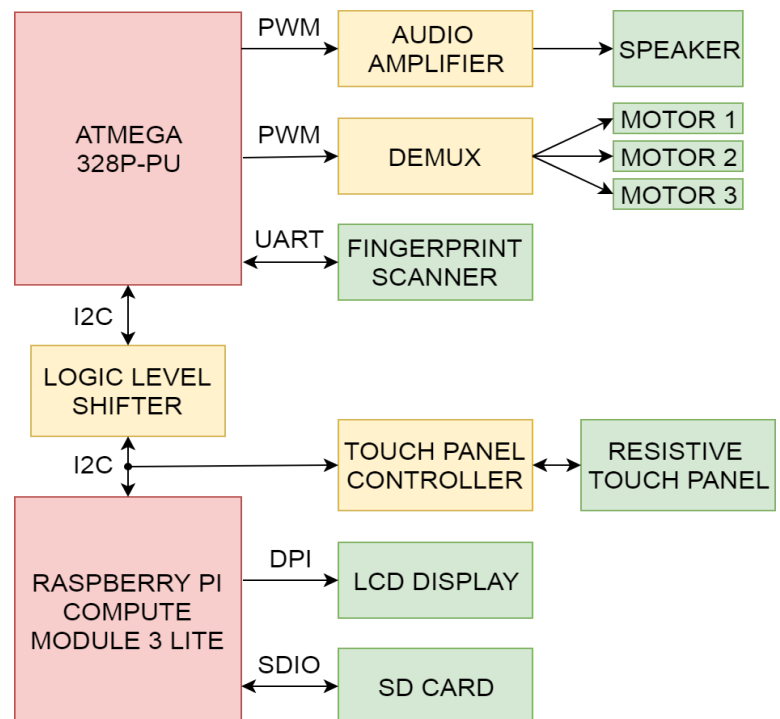


Figure 23. Hardware block diagram

The hardware block diagram in Figure 23 is more detailed and shows the final choices for the main control units and the methods to control each of the peripherals. It also illustrates the communication/signal protocol used for each peripheral. As shown in the diagram, the Compute Module 3 Lite (CM3L) is the main control unit. The CM3L directly controls the LCD display via Display Parallel Interface (DPI); the resistive touch panel via the AR1021 touch panel controller, which communicates using I2C; and the ATMEGA328P microcontroller, which also communicates using I2C. Since the CM3L and the ATMEGA328P use different voltage levels, a bidirectional logic level shifter is required to avoid damaging the CM3L. The ATMEGA328P directly controls the fingerprint scanner through UART communication, and sends pulse-width-modulated (PWM) signals to the speaker. For the dispensing mechanism, the ATMEGA328P sends a PWM signal and an address to a demultiplexer, which then selects the appropriate motor to receive the PWM signal and rotate, thereby dispensing the correct medication.

5.1.2 Hardware Design Overview

A prototype for the Medspencer has been completed as a requirement of Senior Design 2. Throughout the report the different components and parts regarding hardware and software were mentioned and explained in order to give more detailed explanation of the project functionality. The purpose of this section is to assemble all these components and get a clear image of how the Medspencer works. From the electronic inner components to the outer enclosure, this subsection will list the characteristics for which these parts were selected and how they help to achieve the objective in a better way.

For this project, the main components required include the Pi CM3L, ATMEGA328P microcontroller, LCD display, resistive touch panel, fingerprint scanner, speaker, and dispensing mechanism. Additional components include an SD card reader, a touch panel controller IC, an audio amplifier for the speaker, a demultiplexer to control the motors that make up the dispensing mechanism. To communicate between the microcontrollers and our various peripherals, the interfaces we utilized include Inter-Integrated Circuit (I2C), Universal Asynchronous Receiver/Transmitter (UART), Display Parallel Interface (DPI), Secure Digital Input Output (SDIO), and Pulse Width Modulation (PWM).

Our final design uses two PCBs. The main PCB has the Pi CM3L, ATMEGA328P, and all the peripherals and their corresponding circuitry. The second PCB has all of the voltage regulation circuits and accepts 20V DC power. The PCBs are connected using a ribbon cable. Figure 24 shows the final schematic for the first PCB, and Figure 25 shows the final PCB layout.

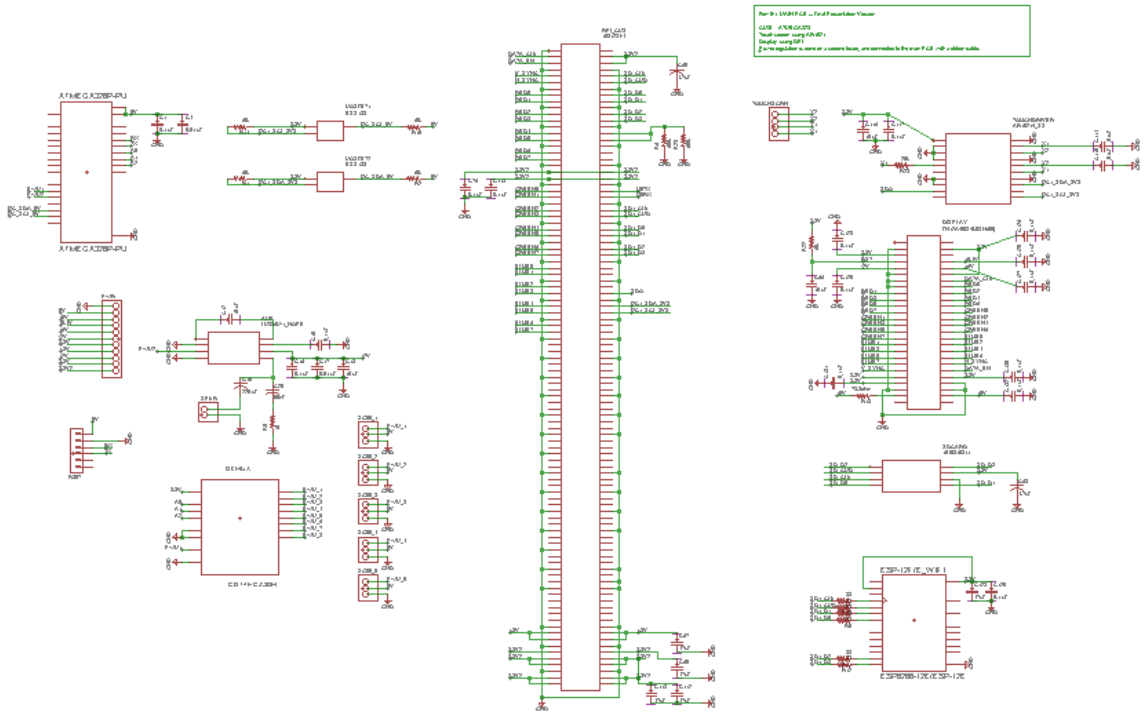


Figure 24. Main PCB schematic

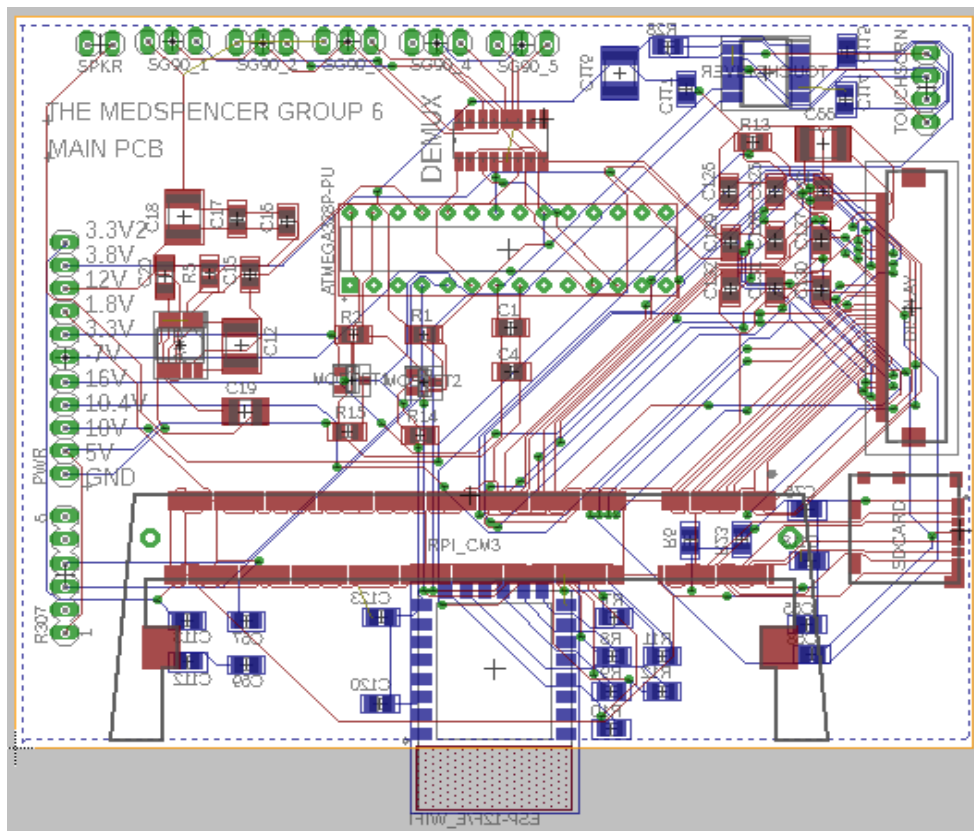


Figure 25. Main PCB layout

Each of the hardware components and their required external circuitry are discussed in detail within this chapter. The voltage regulation PCB is shown and discussed later on in this chapter.

5.1.3 Microcontroller Design

For the purposes of our project, we have chosen to utilize the ATMEGA328P microcontroller in conjunction with the Raspberry Pi Compute Module 3 Lite (CM3L). We also considered using the PIC32MZ DA microcontroller by Microchip. Table 4 compares the qualities, requirements, and capabilities of the three different processing units we considered.

Table 4. Processor Comparison

	PIC32MZ DA	Pi CM3L	ATMEGA328P
COST	\$20.63	\$25.00	\$2.15
PROGRAM MEMORY	2M bytes	SD card (8GB)	32K bytes
DATA MEMORY	640K bytes	1G byte	2K bytes
I/O PIN	176	200	23
CLOCK RATE	200MHz	1.2GHz	20MHz
POWER	1.8V and 3.3V	1.8V and 3.3V	5V

Originally, we chose the Microchip PIC32MZ DA as the central processor of our project. The main draw to this microcontroller was its graphics capabilities: it had a hardware graphics processing unit and LCD controller, which supported the screen size we had chosen for this project. It was also accompanied by a software framework for developing graphical applications. And like all microcontrollers it had numerous hardware peripherals that we would also need. However, after initial testing, the PIC32MZ DA was found to be less capable than advertised. The development framework and documentation for the graphics hardware were poorly developed and often contained unfinished or broken code, which made developing graphics applications difficult. It was also new, having been released last year, which made finding resources for it difficult. With time running out, we made the decision to move our graphics functionality to a different processor: the Raspberry Pi Compute Module 3 Lite (CM3L).

The CM3L is a system-on-module, which has a 1.2GHz Broadcom processor and 1 gigabyte of memory. It supports the Linux operating system, has a number of video controllers, and is more than powerful enough to run our user interface software. However, the CM3L only replaced the graphics portion of our controller needs.

With the graphics being managed by a different processor, the speed, cost, and memory of the PIC32MZ DA was unnecessary for simply controlling peripheral devices. Keeping in mind cost and development time, we chose to replace the controller functionality with an ATMEGA328P. This is a microcontroller we had experience working with, and its low

cost meant that we could easily have multiple prototypes and development environments active at once.

The CM3L, as shown in Figure 26 below, is effectively a stripped-down version of the Raspberry Pi 3. This device is a PCB that contains nothing but a Broadcom processor, a 1 GB bank of DDR2 memory, and a few regulation circuits. It has 200 I/O pins, 54 of which are general purpose, and a number of peripherals. The ones used in this project are an I2C communication bus, for interfacing with the ATmega controller; a Display Parallel Interface (DPI), used to control the LCD screen; and two SDIO interfaces, to which the SD card non-volatile memory and the ESP-12F Wifi module are connected. It runs Linux, which facilitates our user interface software, as well as any future expansions in functionality. The operating voltages are 1.8 and 3.3 V [89].



*Figure 26. Raspberry Pi Compute Module 3 Lite (CM3L) [89]
Courtesy of Raspberry Pi*

The schematic for the CM3L is shown in Figure 27. The necessary connections include the I2C bus to communicate with the ATMEGA328P and the AR1021 touch screen controller, SDIO interfaces for the SD card and ESP-12F Wi-fi module, and Display Parallel Interface (DPI) for the LCD display, as well as connections to power and ground, decoupling capacitors, and pull-down resistors as recommended by the datasheet [89].

The necessary connections for the ATMEGA328P microcontroller include PWM signal connections for the motors (PWM1) and speaker (PWM2), UART connections for the fingerprint module, address bits for the demultiplexer that controls the motors, and SDA/SCL connections for the I2C bus. The schematic in Figure 29 below illustrates the necessary connections for the ATMEGA328P.

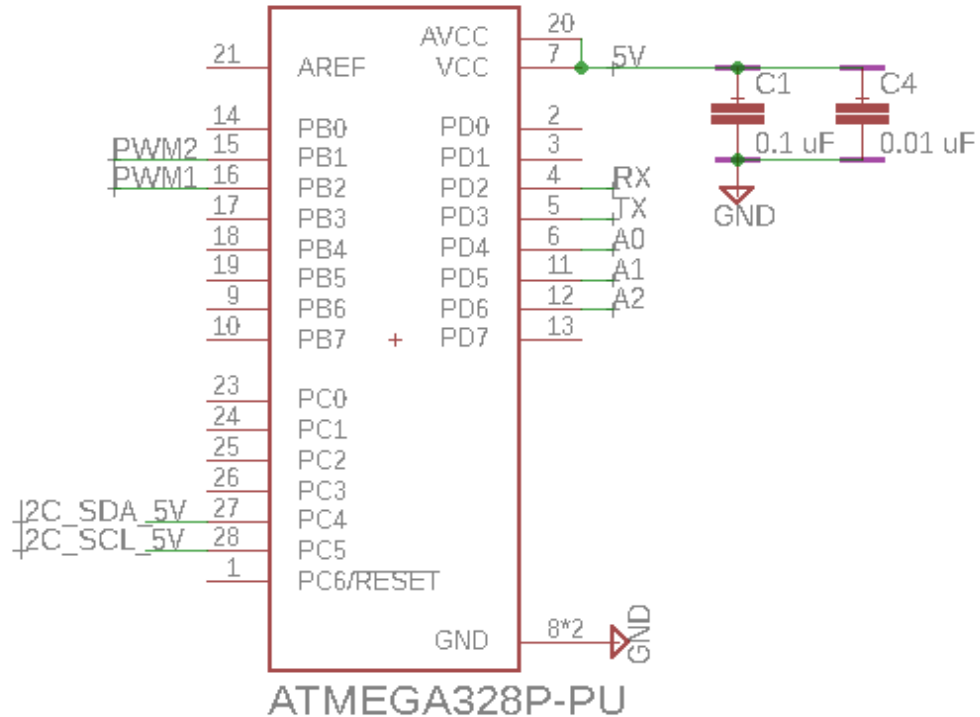


Figure 29. ATMEGA328P-PU schematic

As already mentioned, the CM3L and ATMEGA328P communicate over I2C. Because the CM3L and ATMEGA328 have different operating voltages, special care must be taken where they communicate. To avoid damaging the Compute Module with the ATMEga's 5V signals, a bidirectional logic level shifting circuit was needed [92]; this circuit is shown in Figure 30.

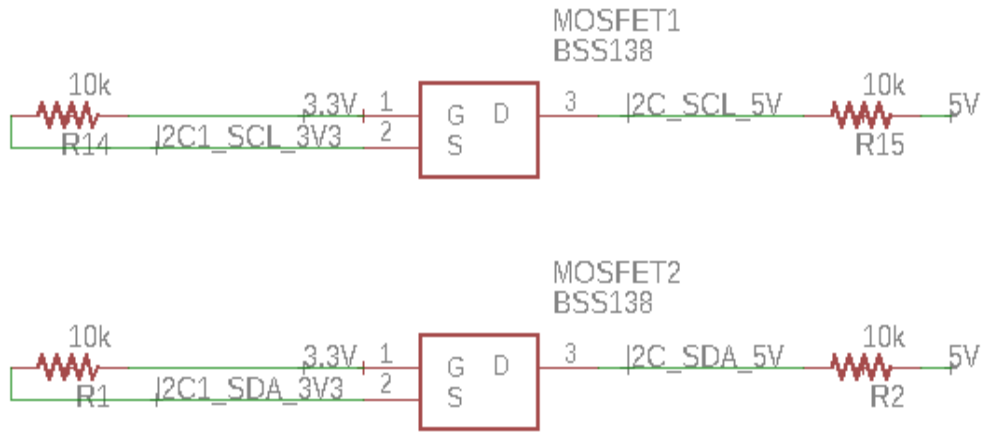


Figure 30. Bidirectional logic level shifting circuit

This circuit requires a MOSFET transistor with a very low threshold voltage, as it must be able to activate when 3.3V is applied to the gate. The BSS138 MOSFET was chosen for this application. The I2C communication bus standards require a pullup resistor on the bus so the signal is never left floating, and those resistors are incorporated into both sides of the design of the level shifter. This circuit was used on both wires of the I2C bus.

5.1.4 Wi-Fi Module

The Wi-Fi module chosen for this project is the ESP-12F module by Espressif, which utilizes the ESP8266 IC. The ESP-12F module is shown in Figure 31 below. The ESP8266 integrates an enhanced version of the Tensilica L106 32-bit RISC processor, which uses low power consumption and reaches a maximum clock speed of 160 MHz. The Real-Time Operating System (RTOS) and Wi-Fi stack allow 80% of the processing power to be available for user application programming and development [93].



*Figure 31. ESP-12F Wi-fi module [93]
Courtesy of Espressif*

We utilized the SDIO interface to communicate between the ESP-12F and the Raspberry Pi CM3L [94]. Figure 32 below shows the schematic for the ESP-12F, with all of the necessary connections to power and interface the module.

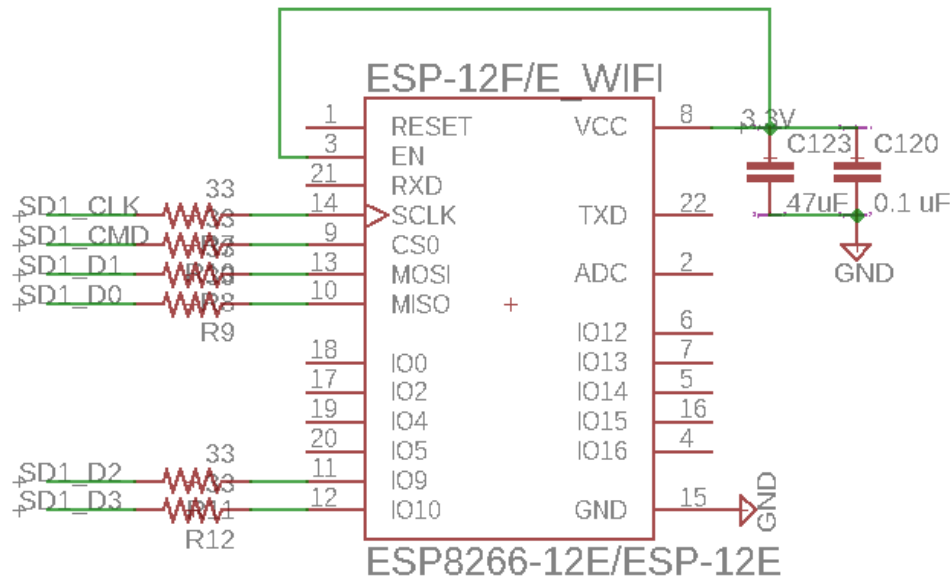


Figure 32. ESP-12F schematic

The operating voltage for this module is 2.5-3.6V. To power the Wi-Fi module, 3.3V is supplied by the LP5912-3.3 LDO regulator.

5.1.5 Display Design

The display we utilized is the Innolux AT070TN90 LCD Display. The LCD display we chose to use is a WVGA display with 800x480 resolution and 24-bit color. Each color, red, blue, and green, is controlled with 8 bits. The display is controlled directly by the Pi using the Display Parallel Interface (DPI). The alternative option to control the LCD display is by using an HDMI cable. This option requires more hardware circuitry and power overhead, however it's more flexible. For our project applications, which would be one self-contained manufactured product, we don't need the flexibility to plug in/out the screen, so we decided it's more efficient to use the parallel data out.

The screen requires a number of signal and power lines to operate correctly. The GLCD controller on the microprocessor provides direct pinouts for the screen enable (DE), V-Sync (VS), H-Sync (HS), clock (DCLK), and color data (R<7:0>, G<7:0>, B<7:0>). Additionally, there are some data lines that will have to be hardwired. V_{COM} should receive 3.8V (typ). MODE should be pulled high to enable horizontal and vertical synchronization. The L/R and U/D should be high and low, respectively, to indicate up to down and left to right. RESET should be normally be pulled high, but if pulled low will cause the display to enter its reset state. The datasheet suggests attaching to an RC reset circuit [14].

Table 5. Pin Assignment [14]

Pinout	Symbol	Power/Input	Description
5,36,38,48	GND	P	Power ground
6,46	V _{COM}	I	Common voltage
7	DV _{DD}	P	Power for digital circuit
8	MODE	I	DE/Sync mode select
9	DE	I	Data input enable
10	VS	I	Vertical sync input
11	HS	I	Horizontal sync input
12-19	B<7:0>	I	Blue data
20-27	G<7:0>	I	Green data
28-35	R<7:0>	I	Red data
37	DCLK	I	Sample clock
39	L/R	I	Left/right selection
40	U/D	I	Up/down selection
41	V _{GH}	P	Gate ON voltage
42	V _{GL}	P	Gate OFF voltage
43	AV _{DD}	P	Power for Analog circuit
44	RESET	I	Global reset pin
47	DITHB	I	Dithering function

To interface with the LCD display, we utilized the Display Parallel Interface (DPI) on the Raspberry Pi CM3L. The schematic for the LCD display connections is shown below. (Note that the pin numbers are reversed from the numbers in the datasheet's pin assignment table [14]). The connector we used is the FH12A-50S-0.5SH 50-pin FPC/FFC connector manufactured by Hirose, as recommended by the datasheet [14].

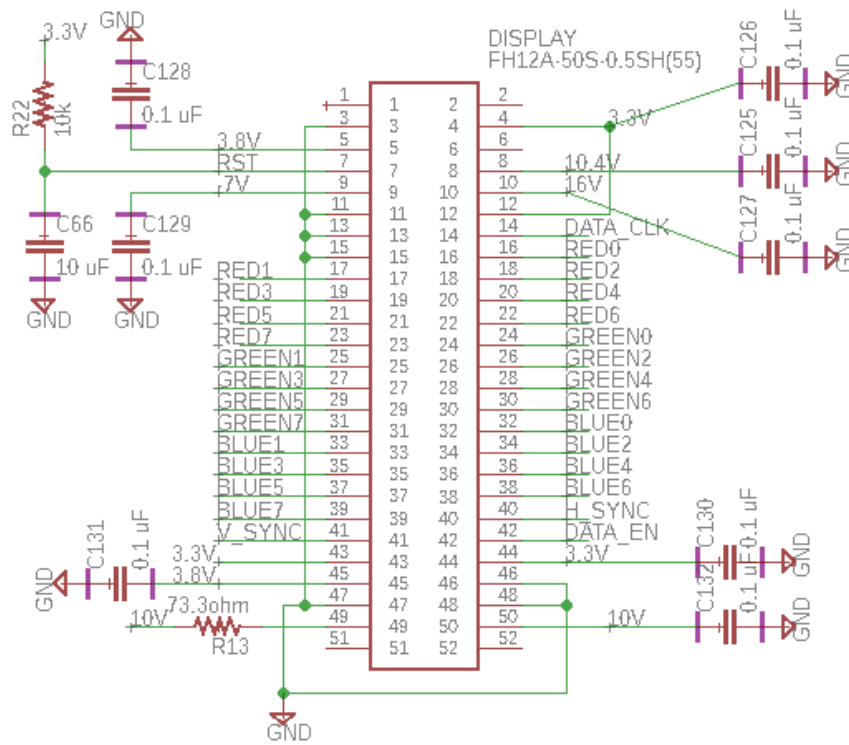


Figure 33. LCD display schematic

To power the LCD display, six different supply voltages are required [14]. These are discussed in detail later on in this chapter.

The touch panel utilizes the AR1021 resistive touch screen controller, which communicates to the CM3L using I2C. The AR1021 is connected to the X1, X2, Y1, and Y2 electrodes on the resistive touch panel. This controller processes the touch data and delivers calibrated touch coordinates to the host MCU. The operating voltage is 2.5-5V [90]. 3.3V is supplied to the AR1021. The schematic for the AR1021 is shown below.

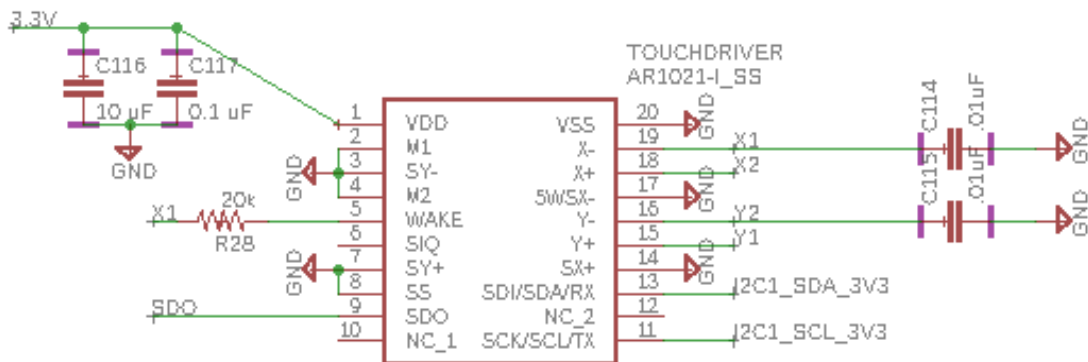


Figure 34. AR1021 schematic

5.1.6 Fingerprint Scanner

The fingerprint scanner that we utilized for this project is the R307 Fingerprint Identification Module by Hangzhou GROW [70]. This fingerprint scanner utilizes an optical sensor to scan the fingerprint and create a digital image. A picture of the R307 fingerprint module is displayed in Figure 34. The specifications and ratings for the R307 are summarized in Table 7 [70].

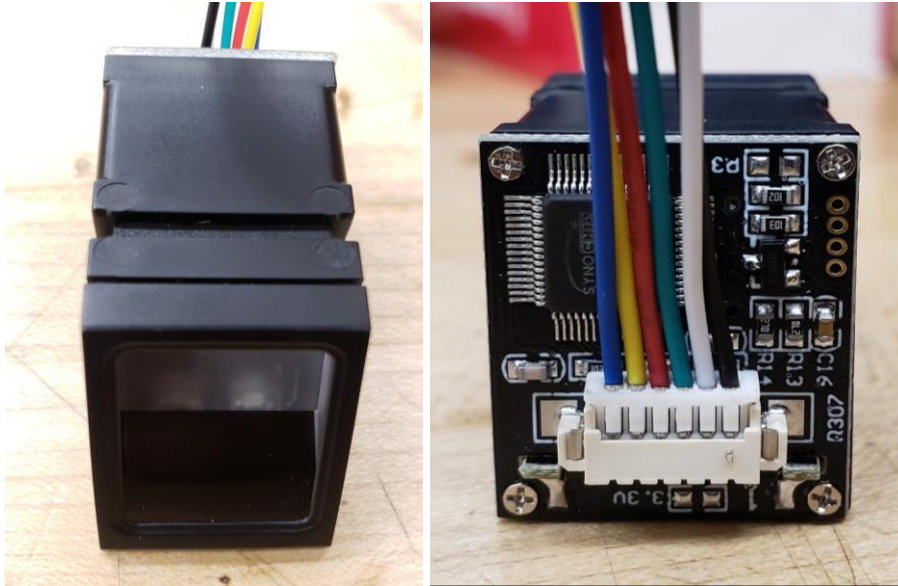


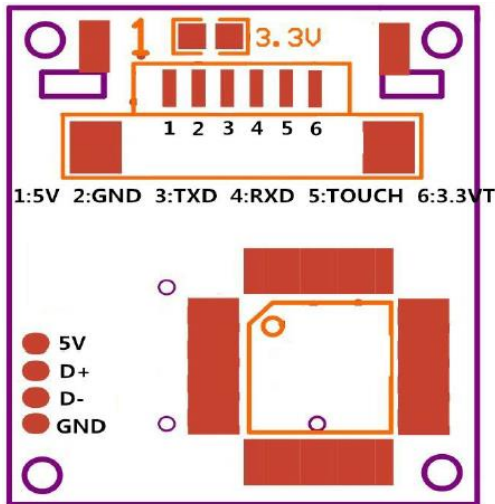
Figure 35. Picture of R307 fingerprint module

Table 6. R307 Fingerprint Module Specifications and Ratings [70]

Power input	Working current	Finger detection power	Baud rate	Image acquiring time	Average search time	Template size
4.2-6 V or 3.3 V	50 mA, <75-80 mA	3.3-5 V (5 μ A)	9600*N bps, N=1~12 (default N=6)	<0.5 sec	<1 sec (1:1000)	512 bytes

The fingerprint module comes with a scanner, detection section, and four pins for connections, which include power supply, ground, RXD, and TXD. The module requires 200 msec for initialization, during which it cannot accept commands. Fingerprint processing involves two parts, fingerprint enrollment and fingerprint matching. When enrolling a fingerprint, the user must enter the fingerprint two times. The system will process the two finger images, then generate and store a template of the finger. When matching, the user enters the fingerprint and the system will generate a template of the finger and compare it with the templates in the fingerprint library. The matching mode can

be either 1:1 or 1:N. For 1:1 matching, the fingerprint module will compare the live finger with a specific template designated in the module. Meanwhile for 1:N matching, or searching, the system will search the entire fingerprint library for the matching finger. The system will return either a success or failure, depending on whether a match is found [70].



*Figure 36. Exterior interface of the R307 fingerprint module [70]
Courtesy of Hangzhou GROW*

The utilized hardware interface is UART. The R307 uses serial interfaces and TTL logic levels. Via the serial interface, the module may communicate with an MCU of 3.3 V or 5 V power. The hardware interface of the R307 fingerprint module is shown in Figure 35. The module may be powered using either 3.3 V or 5 V (4.2-6 V); to use 3.3 V operation, short the connection between the two contacts located next to the label “3.3 V” [42, 70]. The module has a working supply current of 50 mA (typical), or <75-80 mA. The module may communicate with an MCU of 3.3 or 5 V via serial interface. To connect the module to the MCU, the R307’s pin 3 (TXD) connects to the MCU’s receiving pin, and the R307’s pin 4 (RXD) connects to the MCU’s transmitting pin [70]. We interfaced the R307 with the ATMEGA328P, which operates on 5 V. A 5 V power supply was connected to the R307.

5.1.7 Dispensing Mechanism

To facilitate the dispensing mechanism of the Medspencer, individual servo motors are used to dispense medicine from each medicine vial. For our prototype, we utilized 5 servo motors, which are accessed by the ATMEGA328P microcontroller via the CD74HC238M demultiplexer shown in the next figure. Using a demultiplexer, the amount of servo motors and medicine vials could be expanded in the future. The demultiplexer uses 3.3V power supply and requires 3 address bits to select which motor to access [91]. A PWM signal is

sent to the demultiplexer and relayed to the correct motor, in order to dispense the appropriate medicine.



*Figure 37. CD74HC238 3-to-8 Demultiplexer [91]
Courtesy of Texas Instruments*

We chose to use SG90 servo motors shown in the figure below. These little devices are small enough to fit within the Medspencer enclosure. For the purpose of this project, we created a filter that attaches to a motor and allows a pill to be dispensed when the motor is rotated. By default this servo motor can rotate 180 degrees maximum [95]. The ATMEGA328P microcontroller controls the servo motors using pulse width modulation (PWM). The operating voltage of the SG90 servo motor is between 4.8V to 6V, and the typical current draw is 220 mA. The LMR23615 step-down regulator was used to supply a regulated 5V to the servo motors.



Figure 38. SG90 servo motor

The schematic for the dispensing mechanism is shown in the figure below. As the schematic shows, 5V is supplied to each motor and 3.3V is supplied to the demultiplexer. The ATMEGA328P sends 3 address bits and a PWM signal to the demultiplexer, which relays the PWM signal to the motor at the correct address.

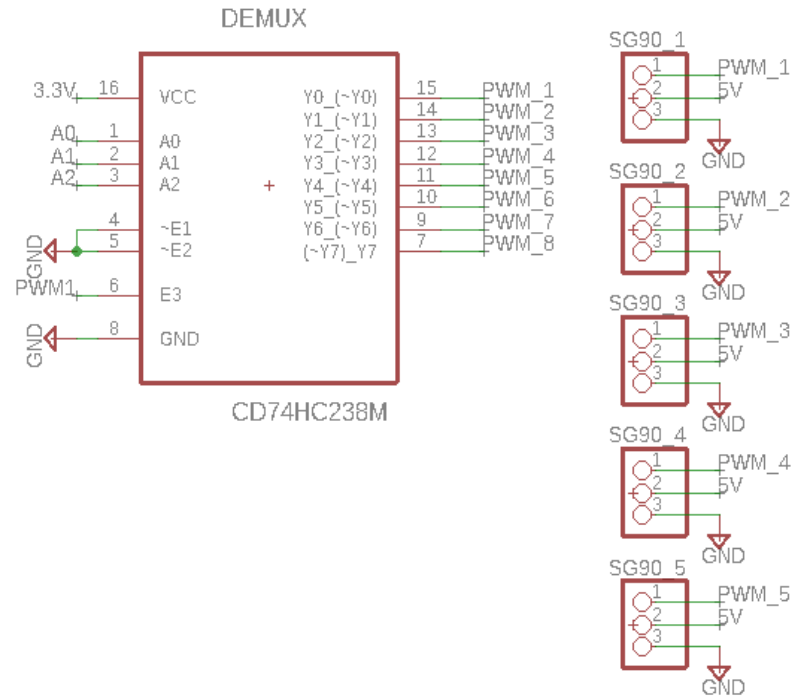


Figure 39. Dispensing mechanism schematic

5.1.8 Speaker

In order to notify the patient when it is time to take their dose of medicine, we utilized a small speaker. When it is time for a scheduled dose, the speaker plays a simple sound to alert the patient. The speaker we chose is the CMS-40504N-L152, a 2W, 4Ω speaker by CUI, Inc., shown below. Its resonant frequency is 500 Hz [96]. A PWM signal is required to sound the speaker, which is generated by the ATMEGA328P.



Figure 40. 2W, 4Ω speaker

The LM386 audio amplifier was utilized to amplify the PWM signal. This amplifier has a programmable gain that can be set from 20-200 [97]. The schematic for the LM386 and speaker is shown in the figure below.

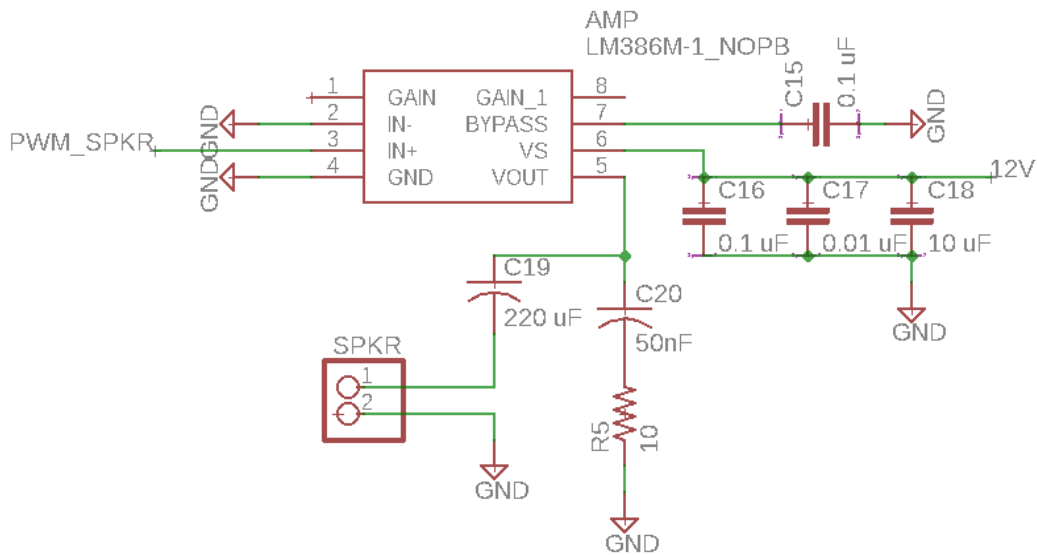


Figure 41. Audio amplifier schematic

Leaving pins 1 and 8 unconnected results in the default gain of 20. The audio amplifier is supplied 12V, regulated by the LM5009 switching regulator.

5.1.9 Power Supply

To power the Medspencer, we used standard U.S. domestic power. We purchased an AC to DC adapter and a DC power jack to mount on the PCB. The AC adapter we utilized supplies 20V and 3.25A. The DC plugs are inner positive and outer negative. The DC power jack has three pins, as shown below in the figure below. Pin 1 is connected to positive power, and pin 2 is connected to ground. Pin 3 can be connected to a battery. When the plug is inserted, the connection between pins 3 and 2 breaks. We chose not to use this backup power feature, so we left pin 3 unconnected grounded so it is permanently connected to pin 2.



*Figure 42. 3-pin DC power barrel connector [32]
Courtesy of Digi-Key*

The AC adapter converts the AC power to a 20 V, 3.25 A DC output. This corresponds to a power of 64 W, according to the relation $P = I \times V$. However, each component used for this project has different voltage and current ratings. We will use voltage regulators to scale down the voltage, to avoid damaging any of our hardware components.

5.1.10 Power Regulation and Filtering

The main DC power supply for the device is 20V and 3.25A, after undergoing transformation and AC-DC conversion. The power ratings of all the hardware components used for this project are summarized in Table 7 below. To provide the correct voltages and currents to each component, we must use voltage regulators to supply each power requirement.

Table 7. Power Requirements

COMPONENT	V _{SUPP} (V)	I _{SUPP} (mA)
ATMEGA328P-PU	5	4
Raspberry Pi CM3L	3.3	250
	1.8	250
Bidirectional Logic Level Converter	5	22
	3.3	
104031-0811 SD Card Reader	3.3	500
AT070TN90 LCD Display	3.3	10
	10.4	50
	10.0	135
	16.0	1
	-7.0	1
	3.8	10
AR1021 Resistive Touch Screen Controller	3.3	17
CD74HC238M Demultiplexer	3.3	50
SG90 Servo motors	5	220
LM386 Audio Amplifier	12	4
R307 Fingerprint Scanner	5	50
ESP-12F Wi-fi Module	3.3	80

We carefully considered the required input and output power for each component, and any specific datasheet recommendations (such as whether we should use an LDO regulator, such as for 3.3 and 1.8V power supplies for the processor). Then we chose appropriate voltage regulators and designed the circuits around them. We used TI's Webench Power Designer as an aid for the power regulation design [98]. Webench helped us consider efficiency, board size, and availability, helping us to choose regulators that fit our specs. Webench also helped us decide how to cascade the regulator circuits. We made sure to choose regulators with at least 80% efficiency, considering the input and output power for each regulator. The voltage regulation devices we chose are summarized in Table 8 below.

Table 8. Voltage Regulation Devices

DEVICE	V_{IN,MAX} (V)	V_{OUT} (V)	I_{OUT,MAX} (A)
LP5912-3.3 LDO	6.5	3.3	0.5
LMR23615 regulator	36	5	1.5
LMR23610 regulator	36	3.8, 10	1
TPS62745 regulator	10	1.8	0.3
LM43601 regulator	36	7	1
ADM8660 inverter	7	-7	0.1
LM5165 regulator	65	16, 10.4	0.15
LM5009 regulator	95	12	0.15

Since we require 10 power regulation modules, we decided to place these on a second PCB. The schematic in the figure below shows all the power regulators we utilized for our final design. Some of the regulators are fixed value, some are adjustable, some programmable. The required circuitry includes resistors to set the desired output voltage, decoupling capacitors, and additional components for switching power supplies.

Figure 44 shows the final PCB layout for the voltage regulation PCB. It's attached to our main PCB with a ribbon cable, and receives power through a jack that can be plugged in the wall. Schematic capture and PCB layout was done using Eagle [35].

The Pi CM3L requires 1.8V, which is supplied by the TPS62745 step-down converter. This regulator's output is programmable through its output voltage selection pins. It supplies up to 300 mA [99]. The schematic is shown in Figure 45.

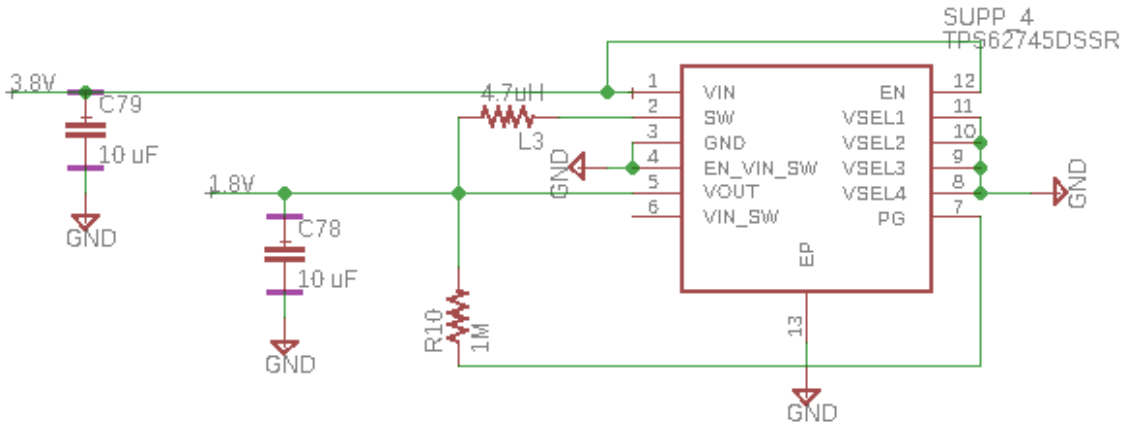


Figure 45. 1.8V regulation schematic

The LCD display requires -7V for the gate-off voltage. The LM43601 step-down voltage converter regulates +7V. The adjustable regulator's circuit was designed using the following equation:

$$R_{FBB} = V_{FB} R_{FBT} / (V_{OUT} - V_{FB})$$

The LM43601 supplies up to 1 A [100]. The ADM8660 inverts the voltage from +7V to -7V, and outputs up to 100 mA [101]. The schematic for the -7V supply is shown in Figure 46.

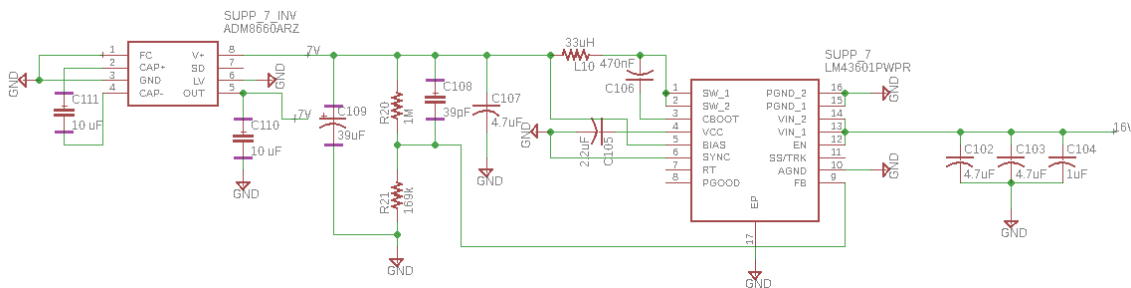


Figure 46. -7V regulation schematic

The LMR23610 simple switcher is used to supply 10V and 3.8V to the LCD display. To adjust the output, the following design equation was used:

$$R_{FBT} = (V_{OUT} - V_{ref}) R_{FBB} / V_{ref}$$

It can supply up to 1 A [102]. The schematics are shown in Figures 47 and 48.

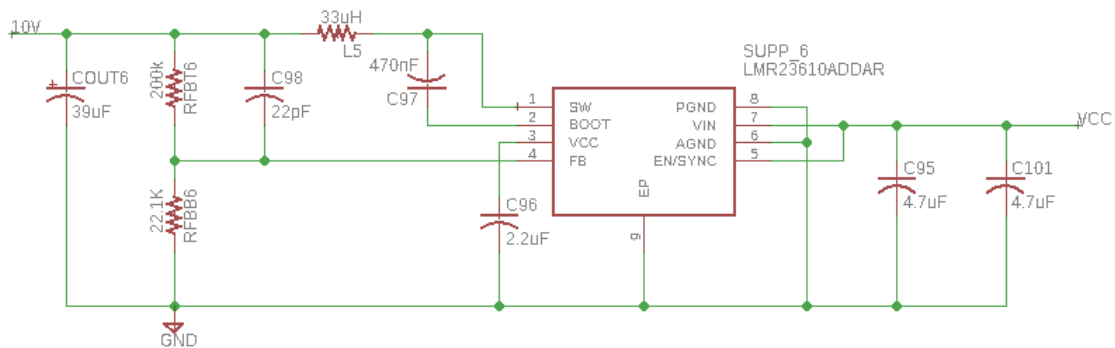


Figure 47. 10V regulation schematic

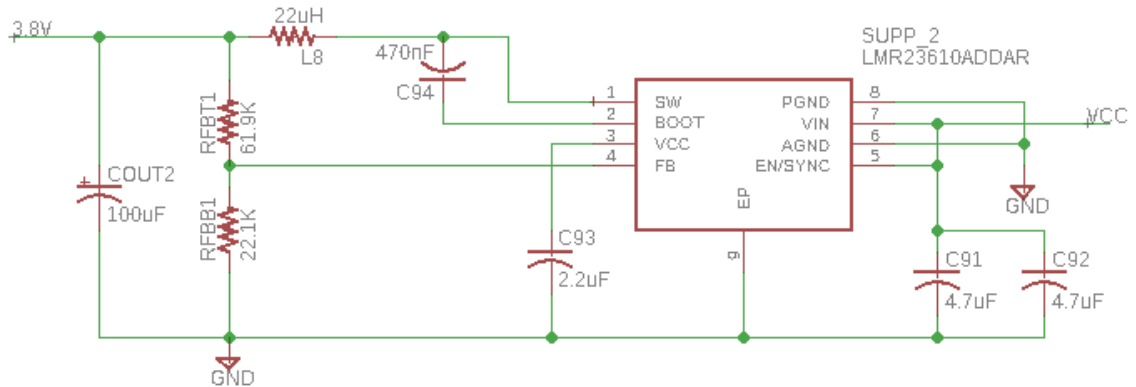


Figure 48. 3.8V regulation schematic

The 5V power supply for the ATMEGA328P microcontroller and peripherals is supplied by the LMR23615 synchronous step-down converter. The necessary design equation is:

$$R_{FBT} = (V_{OUT} - V_{ref}) R_{FBB} / V_{ref}$$

Up to 1.5A can be supplied [103]. The schematic is shown in Figure 49.

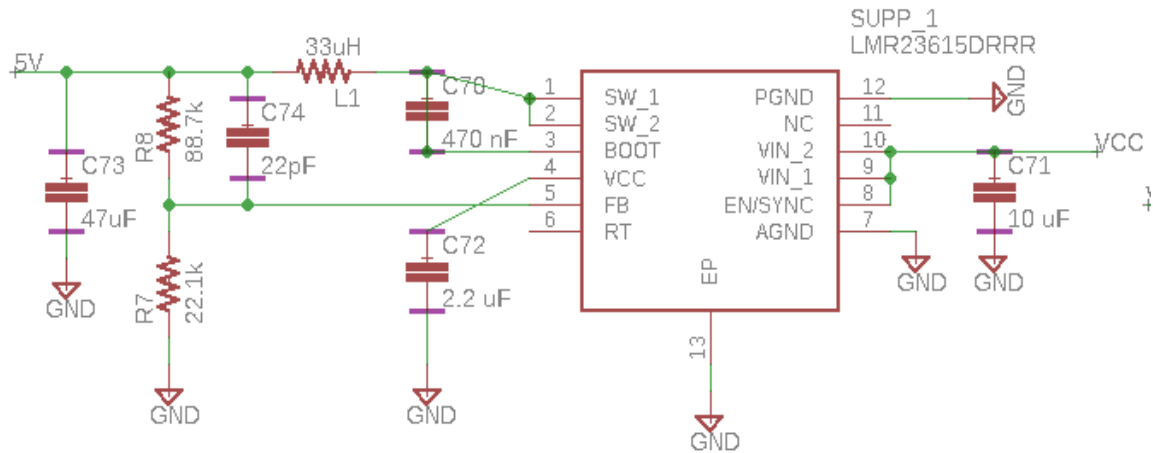


Figure 49. 5V regulation schematic

The LM5165 synchronous buck converter is used to supply 16V and 10.4V to the LCD display. The necessary design equation is:

$$R_{FB2} = V_{FB}R_{FB1} / (V_{OUT} - V_{FB})$$

It can supply up to 150 mA [104]. The schematics are shown in Figures 50 and 51.

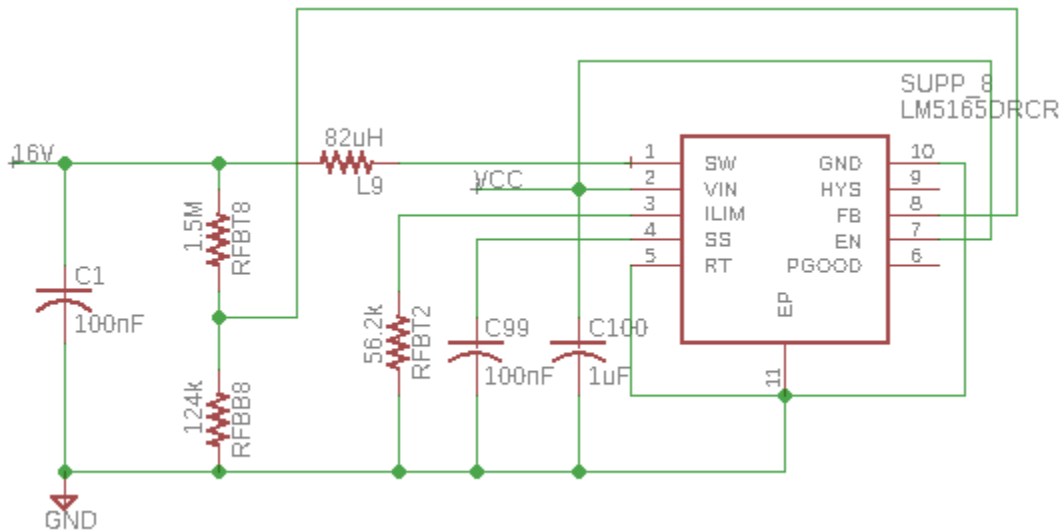


Figure 50. 16V regulation schematic

communication, peripheral control, and other aspects of the software are examined and explained in detail. This chapter should result in a broad overview of every functionality of the Medspencer's software.

5.2.1 ATmega328P Software

The software of the ATmega328p consists of a main loop, two interrupt functions, and a few buffers. After initialization, the software goes into the main loop, which reads commands from the command buffer and executes them. If the command buffer is empty, then the loop waits for the next command. When an I2C write event occurs, the bytes from the transmission are saved to the command buffer. The first byte is the command, and if there is a second byte, it is saved to the buffer as a parameter. When an I2C Read event occurs, the next byte in the Return Buffer is transmitted to the master. If the Return buffer is empty, a NO_DATA flag is transferred instead, telling the master that the data it's looking for isn't ready yet. Once a command is read in the main loop, one of the four functions is executed.

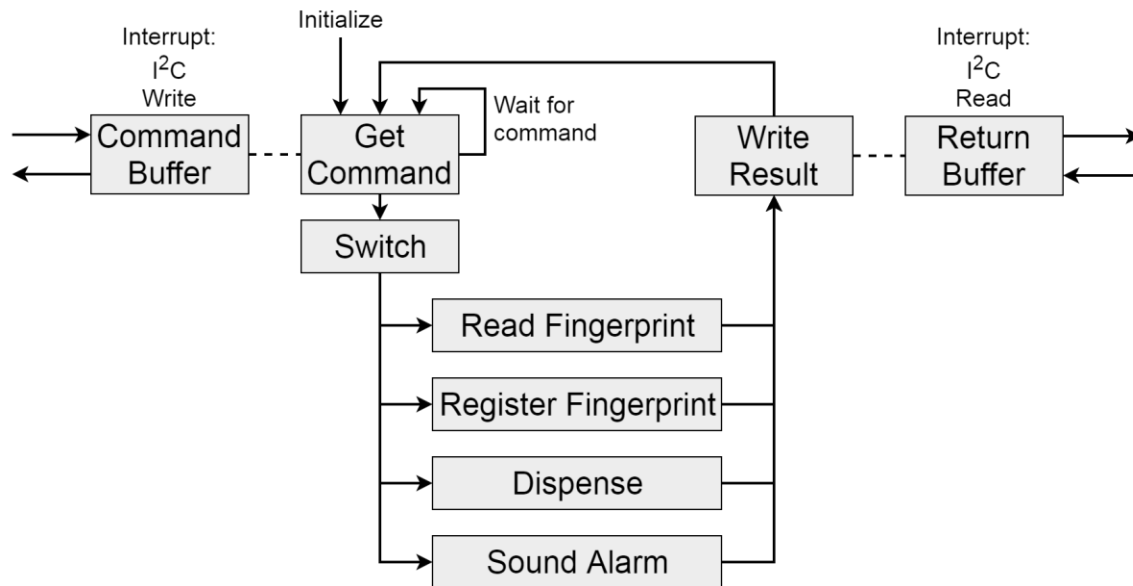


Figure 53. ATmega328P Software Diagram

To scan a fingerprint, the ATmega sends various instructions to the Fingerprint module over a serial connection: capture an image, convert it to a model, and searching it against previously saved models. The fingerprint module then sends back a packet of data containing the model index number. Registering a fingerprint is similar: capture two images of the same finger, create a model out of those, and store the model at the specified index. To recap: the ATmega is receiving bytes from the I2C which tell it what to do, then it sends commands to the fingerprint module to tell it what to do, and when data is returned from the fingerprint module, it is send back to the I2C master where the GUI program is running. The other two commands are simpler, as they don't involve another layer of communication. The dispense command is accompanied by an index number. The servo-motors are connected to the ATmega through a demultiplexer, so this number controls the

demux so the ATmega can be connected to the correct motor. The motors themselves are controlled by a pulse width signal, which is generated by the ATmega. This signal tells the motor to rotate fully counterclockwise, then back to where it began, to dispense a pill. The final command the ATmega performs is to generate another pulse width signal to sound the alarm. In the current iteration of the project, this signal simply consists of a few notes played in sequence.

5.2.2 User Interface

The user interface is the primary way for users to interact with the system. Built on the foundation of the LCD and touchscreen, the UI gives the user information, while also providing a way to receive information in return. The user interface system is made up of “frames”, segments of code that contain graphical and functional information. Each frame contains a “schema”, code to construct the graphics of that frame, as well as “events”, code that is triggered by specific occurrences on the frame.

Frames are triggered within the event codes of other frame. This calls the frame system to open the new frame on top of the current one, running the schema code immediately and loading the event code for running during the next frame cycle. Frames can have parameters for input, like a string for a title, and output, like a reference to save a result to.

A frame’s schema need not overwrite the entire screen. For instance, a keyboard page may only cover the bottom of the screen, leaving the top free to display. The approach to user interface design using frame makes the software very modular, as any frame needed for a specific purpose can be used wherever that purpose is needed.

Most frames has some return state in their event code, where their intent will be fulfilled and the system will go back to the previous frame. For example, when viewing a patient’s schedule, the administrator may wish to add a new medicine. Once that intent has been recognized (presumably through an “Add Medicine” button on the patient’s frame), the `add_medicine` frame will be called. This frame will pass the patient’s ID and a reference to save the medicine’s ID into. Then, the new frame will create a new medicine, add it to the list of medicines (thereby generating a new ID), add it to the patient’s list (using the passed-in patient ID), and return to the previous frame once finished. The patient’s schedule should now have a new medicine in it.

Since frames can call other frames, this creates a hierarchy of nested frame calls, allowing for a user to always have the ability to navigate back to the root frame. This root frame is the initial part of the system, as long no other frame leads to it. The root frame will be the default “idle” state, waiting for a timer event or screen interaction. The timer event system will be handled separately from the frame system.

Adding more than one medication to each patient will be an added feature for the future Medspencer 2.0.

5.2.3 SQLite

The Medspencer's chosen data storage solution is a SQL database engine called SQLite. SQLite is an open-source, serverless database system that is linked into the program at compile-time. Its resources are accessed through function calls instead of requests sent to alternate threads. All data, as well as the tables referencing it, are saved in one cross-platform file, which has an upper size limit of 140 terabytes, yet the library itself can be compiled down to nearly 500 kB. For the Medspencer, the library will be linked into the program code, while the database file itself will have to live onboard an SD card.

As the SQLite website stresses multiple times, most applications are served with the full amalgamation file that can be compiled on any Windows or Unix system without changes. However, highly tuned and specialized applications may wish to customize their compilation, perhaps to change the multi-threading system, to reconfigure the memory allocator, or to modify the virtual file system. Many of these changes can be done by giving specific commands at compile-time.

Most applications of SQLite work fine in its default configuration, and even more should be served by compile-time config options. However, if a serious rework is necessary, such as porting the library to a new operating system, new code must be written. For proper functionality, the application must provide a working multi-threading system, a working memory allocation system, and a working virtual file system implementation. All of these things can be provided in a single auxiliary C code file then linked with the full amalgamation file to provide the application with a working SQLite build.

In conclusion, it is entirely within the realm of possibility to use an implementation of SQLite in an embedded project. However, it may take extensive experimentation and rewriting to get it to function.

After some new development and changes to previous device changes, it was concluded that the Raspberry Pi compute module would be the best selected microcontroller which runs on linux base software. Linux is compatible to Python IDE which can run in parallel with SQLite to add, update, delete and select patient's records. With this code the python structure was created to build the frame system described in section 5.2.12.

5.2.4 Wi-Fi Communication

In the Wireless frequency inference network processor subsystem includes a dedicated ARM MCU to completely offload the host MCU along with an 802.11 b/g/n radio, baseband, and MAC with a powerful crypto engine for a fast, secure WLAN and Internet connections with 256-bit encryption. The CC3220x devices support station, AP, and Wi-Fi Direct modes. The device also supports WPA2 personal and enterprise security and WPS 2.0. The Wi-Fi network processor includes an embedded IPv6, IPv4 TCP/IP stack. [25] The wireless frequency inference network processor subsystem has an enhances the security capabilities available for development of IoT devices, while completely offloading

these activities from the MCU to the networking subsystem. There is also AES (WPA2-PSK), TKIP (WPA-PSK) and WEP for personal standards.

In order to operate these feature and requirement the software that are requirement is the SimpleLink™ Wi-Fi® Radio Testing Tool is a Windows-based software tool for RF evaluation and testing of SimpleLink Wi-Fi CC3220 designs during development and certification. The tool enables low-level radio testing capabilities by manually setting the radio into transmit or receive modes. Using the tool requires familiarity and knowledge of radio circuit theory and radio test methods. Created for the Internet of Things (IoT), the SimpleLink Wi-Fi CC3220 family of devices includes on-chip Wi-Fi, Internet, and robust security protocols with no prior Wi-Fi experience needed for faster development.

The Image Creator is a web application which is used to create a programming image; it can also write the programming image into the SimpleLink CC3220 devices. The programming image is a file which contains the SimpleLink device configurations and files required for the operation of the device. For the SimpleLink CC3220 wireless microcontroller, the Image Creator can also include the host application file. A new SimpleLink device should first be programmed by a programming image. The image, created by the Image Creator, can be programmed onto the device as part of the production procedure or when in development stage.

And the CC3220 SDK contains drivers, many sample applications for Wi-Fi features and Internet, as well as documentation needed to use the CC3220 Internet-on-a-chip solution. This SDK can be used with TI's MSP432P401R LaunchPad™ development kit, or with the SimpleLink Studio, a PC tool that allows MCU development with CC3220. All sample applications in the SDK are supported on TI's MSP432P401R ultra-low-power MCUs with Code Composer Studio™ IDE and TIRTOS. In addition, many of the applications support IAR [25].

After further research and design changes, we decided to instead utilize the ESP-12F Wi-fi module, which uses the ESP8266EX IoT chip by Espressif. Due to time constraints, this section of the project could not be completed. However, on the python code there is a user interface section so that the developers have access for future development.

5.2.5 Fingerprint Scanner

As mentioned before, the fingerprint module can facilitate 1:1 matching or 1:N matching. For 1:N matching, or searching, the module will search the whole fingerprint library for a matching fingerprint. The module will return either a success or failure. When enrolling a new fingerprint, the user must enter their fingerprint two times. The system will then process the two time fingerprint images, generate and store a template for the finger [70].

The fingerprint module has built-in nonvolatile flash memory, which holds the fingerprint library and the notepad. The user's notepad consists of 512 bytes (16 pages x 32 bytes) of memory set aside in the flash, and it can be read and written to by the user. The fingerprint template library is stored within the flash, and the library's capacity is 1000 fingerprint templates. Fingerprint templates are stored in sequential order. Users can access fingerprint

templates in the library by template number. An image buffer and two 512 byte character file buffers are located in the RAM. The buffers store images, character files, and template files. The image format is 256 x 288 pixels. The user can read and write to any of the buffers. Contents of the buffers will be lost when the module is powered off [70, 79].

For the R307 fingerprint module, we will utilize UART communication protocols. The serial communication is semi-duplex and asynchronous, with an adjustable Baud rate (default 57600 bps). The transferring frame format is 10 bits, and consists of a low-level starting bit, 8 bits of data (with the LSB transmitted first), and an ending bit [70].

At startup, the system checks whether a handshaking password is required. If the default setting is active, then the module does not require password verification and will enter into normal operation module. The password length is four bytes and the default value is 0FFH. If the password was modified using *SetPwd*, then the handshaking password must be verified before the module will enter into normal operation mode [70].

The system parameters can be modified or accessed with the instructions *SetSysPara*. When parameters are modified, the new configuration is recorded into flash, so they system will run with the new configuration at the next startup. To modify a parameter, specify the Parameter number and then specify the modification. The Baud rate is controlled via Parameter number 4, and can be an integer from $N = 1-12$; the corresponding Baud rate is $9600 \times N$ bps. Parameter number 5 indicates the security level, and controls the matching threshold value for fingerprint searching and matching. Parameter number 6 indicates the data package length, and controls the max length of the transferring data package when communicating with the MCU. Its value can be 0, 1, 2, or 3, corresponding to 32, 64, 128, or 256 bytes. The system status register can be read using the instruction *ReadSysPara* to view the current operation status of the module [70].

When communicating, the transferring and receiving of command/data/result are all wrapped in a data package format, as shown in the figure below. Commands are sent from the MCU to the fingerprint module, and the module acknowledges commands. Within the data package, the **Header** is 2 bytes and it signifies the start of the package; it has a fixed value of 0xEF01. Each module has an identifying address. The data package indicates the address with **Addr**; the module only responds to data packages whose address item value matches its identifying address. Addr's size is 4 bytes, and the value can be modified by the command *SetAddr* (default Addr value is 0xFFFFFFFF). The high byte is transferred first. The **Package identifier**'s size is one byte, and it can signify the command packet (value = 01H), data packet (value = 02H), acknowledge packet (value = 07H), or end of data packet (value = 08H). The **Package length**'s size is 2 bytes, and it refers to the length of the Package content plus Checksum; the maximum length is 256 bytes. The high byte is transferred first. The **Package content** can hold commands, data (such as a fingerprint character value or a fingerprint template), command parameters, or acknowledge result. The **Checksum** is 2 bytes, and equals the arithmetic sum of the Package identifier, Package length, and Package content. The high byte is transferred first. Command packages can be sent from the MCU to the fingerprint module, and the module will send back an acknowledge packet with a confirmation code and the returned parameter [70, 79].

Header (2 bytes)	Addr (4 bytes)	Package Identifier (1 byte)	Package length (2 bytes)	Package content (max 254 bytes)	Checksum (2 bytes)
---------------------	-------------------	-----------------------------------	--------------------------------	---------------------------------------	-----------------------

Figure 54. Data package format of R307 fingerprint module

Upon receipt of commands, the module will report the execution status and results to the MCU through an acknowledge packet. The acknowledge packet includes a one byte confirmation code, and may also include a returned parameter. There could also be a following data packet [70].

The R30X series provides 23 instructions to realize fingerprint authentication functions. However, the most important functions that we will utilize are: collecting a fingerprint image; generating a character file from that image; generating a template for the character file; store the template in the library; and searching the library [70, 79]. The package formats for commands and acknowledgements for the basic instructions we will use are summarized in Tables 9 and 10. The instructions are described in detail below.

In this paragraph, we discuss the relevant fingerprint processing instructions. The data package formats for these instructions are summarized in Table 9. To collect the finger image, we use the command *GenImg*. It involves detecting the finger, storing the detected finger image in the Image buffer, and returning a confirmation code. The confirmation code reports whether the finger collection was a success or failure, whether there was an error when receiving package, or whether the module can't detect the finger. If there is no finger, then the module will indicate that it can't detect a finger. To generate a character file from an image, we will use the command *Img2Tz*. This command generates a character file from the original finger image in the Image buffer and stores the file in character file buffer 1 or 2. The required input parameter is the buffer ID (1 B), which indicates the character file buffer; the value can be 1h (buffer 1) or 2h (buffer 2). The returned confirmation code indicates whether the character file generation is complete, there was an error receiving the package, or whether it failed to generate a character file, due to the over-disorderly fingerprint image; small size of fingerprint image; or lack of valid primary image. *RegModel* is the instruction to generate a template. It combines the character file information from character file buffers 1 and 2 to generate a template, which is then stored back into the character file buffers. The returned confirmation code indicates whether the operation was a success, there was an error receiving the package, or it failed to combine the character files, because the character files do not belong to one finger. The *Store* instruction is to store a template. This instruction stores the template in either character file buffer 1 or 2 into the designated location in the fingerprint library. The input parameters include the buffer ID (1 B) and page ID (2 B). Page ID indicates the fingerprint library location, and the high byte is transferred first. The confirmation code indicates whether storage was a success, there was an error receiving the package, the page ID location is beyond the library's size, or there was an error writing to memory. The instruction *LoadChar* reads a template from the fingerprint library. This involves loading a template from the specified location in the flash library into a character file buffer. The input parameters are the buffer ID and page ID. The confirmation code indicates whether the

load was a success, there was an error when receiving package, error reading the template, or page ID is beyond the library's size. The instruction *DeletChar* is to delete a template. The input parameters are page ID and N, which indicates the number of templates to be deleted. The confirmation code indicates whether the deletion was a success, there was an error receiving the package, or it failed to delete the templates. The instruction *Empty* is used to empty the entire fingerprint library. The confirmation code indicates whether it was a success, there was an error receiving the package, or it failed to clear the library. The instruction *Match* carries out precise 1:1 matching of two fingerprint templates that are held in the character file buffers. The return parameters include a confirmation code and a matching score (2 B). The confirmation code indicates whether the templates match, there was an error receiving the package, or the templates don't match. The instruction *Search* searches the fingerprint library. It searches the whole library for a template that matches the template in character file buffer 1 or 2. The input parameters are buffer ID, Startpg (2 B, the start address to start searching), and Pagenum (2 B, searching numbers). The return parameters are the confirmation code, matching score, and page ID (indicating the matching template's location). The confirmation code indicates whether a matching finger was found, there was an error receiving the package, or no matching finger was found (the page ID and matching score are both 0). Other available instructions include *UpImage* to upload an image from the image buffer to the MCU; *DownImage* to download an image from the MCU to the image buffer; *UpChar* to upload a template from a character file buffer to the MCU; *DownChar* to download a template from the MCU to a character file buffer; *WriteNotepad* for the MCU to write data to the notepad in module's flash; and *ReadNotepad* to read from the notepad in module's flash [70].

Table 9. R307 Package Formats for Fingerprint Processing Instructions

Command Name	Header (2 B)	Module address (4 B)	Package identifier (1 B)	Package length (2 B)	Instruction/ Confirmation code (1 B)	Param	Check sum (2 B)
GenImg	0xEF01	XXXX	01H (cmd)	03H	01H	-	05H
	0xEF01	XXXX	07H (ackn)	03H	XXH: 00H (success); 03H (failure); 01H (error); 02H (can't detect)	-	Sum
Img2Tz	0xEF01	XXXX	01H (cmd)	04H	02H	Buffer ID: 1h (buff1); 2h (buff2)	Sum
	0xEF01	XXXX	07H (ackn)	03H	XXH: 00H (complete); 01H (error); 06H/07H/15H (fail);	-	Sum
RegModel	0xEF01	XXXX	01H (cmd)	03H	05H	-	09H
	0xEF01	XXXX	07H (ackn)	03H	XXH: 00H (success); 01H (error); 0aH (fail)	-	Sum

Store	0xEF01	XXXX	01H (cmd)	06H	06H	Buffer ID Page ID	Sum
	0xEF01	XXXX	07H (ackn)	03H	XXH: 00H (success); 01H (error); 0bH (bad pageID); 18H (error)	-	Sum
LoadChar	0xEF01	XXXX	01H (cmd)	06H	07H	Buffer ID Page ID	Sum
	0xEF01	XXXX	07H (ackn)	03H	XXH: 00H (success); 01H (error); 0cH (error); 0bH (bad pageID)	-	Sum
DeletChar	0xEF01	XXXX	01H (cmd)	07H	0cH	Page ID N	Sum
	0xEF01	XXXX	07H (ackn)	03H	XXH: 00H (success); 01H (error); 10H (fail)	-	Sum
Empty	0xEF01	XXXX	01H (cmd)	03H	0dH	-	0011H
	0xEF01	XXXX	07H (ackn)	03H	XXH: 00H (success); 01H (error); 11H (fail)	-	Sum
Match	0xEF01	XXXX	01H (cmd)	03H	03H	-	07H
	0xEF01	XXXX	07H (ackn)	05H	XXH: 00H (match); 01H (error); 08H (don't match)	Matching score	Sum
Search	0xEF01	XXXX	01H (cmd)	08H	04H	Buffer ID Startpg Pagenum	Sum
	0xEF01	XXXX	07H (ackn)	07H	XXH: 00H (found); 01H (error); 09H (not found)	PageID Matching score	Sum

There are several system-related instructions we may use, and the corresponding data package formats are summarized in Table 10. To set and verify the module's handshaking password, the commands are *VfyPwd* and *SetPwd*. For our project, we probably won't use this function. *SetAddr* is used to set the module address. The input parameter is *NewAddr* (4 B), which specifies the new module address. The confirmation code indicates whether the operation is completed or there was an error. *SetSysPara* allows one to set the module's parameters. The input parameter is *ParaNum* (1 B, indicates the parameter number), and contents (1 B, indicates new parameter value). The confirmation code indicates whether the operation is completed, there was an error, or register number was invalid. *Control* allows the user to specify whether the UART port I "on" or "off". The input parameter is *Ctrl* (1 B).

Table 10. R307 Package Formats for System-related Instructions

Command Name	Header (2 B)	Module address (4 B)	Package identifier (1 B)	Package length (2 B)	Instruction/ Confirmation code (1 B)	Param	Check sum (2 B)
SetAddr	0xEF01	XXXX (Old)	01H (cmd)	07H	15H	NewAddr	Sum
	0xEF01	XXXX	07H (ackn)	07H	XXH: 00H (complete); 01H (error)	-	Sum
SetSys Para	0xEF01	XXXX	01H (cmd)	05H	0eH	ParaNum: 4/5/6 Contents	Sum
	0xEF01	XXXX	07H (ackn)	03H	XXH: 00H (complete); 01H (error); 1aH (wrong reg)	-	Sum
Control	0xEF01	XXXX	01H (cmd)	04H	17H	Ctrl: 0 (off); 1 (on)	Sum
	0xEF01	XXXX	07H (ackn)	03H	XXH: 00H (complete); 01H (error); 1dH (fail)	-	Sum

5.2.6 Patient Identification & Adding/Removing a Patient

To maintain and management patient’s information or identity to the medication and medication schedule, they data is stored in a database which will be easy access for the MCU to process and transfer. In the requirements to use SQLite database is a 32 bit processor or higher and an SD micro Card with about 4 GB or storage capacity. The database will contain a unique identification alphanumeric sequence or in medical terms a Chart number. This unique identification code number protects the patient’s information. The database has a table for storing the medication name (band name not the generic name), dosing, where would the medication be taken, how many times or at what time, and how long will they be taking it. This database will also record and store the data so weekly will send a report to the doctor by email.

The method to add a patient is to first turn on the device, the administrator or caretaker will scan there fingerprint, given him or her access to add a new patient. In this section, the caretaker will first enter in the chart number, the first and last name of the patient, the Date of Birth of the patient, and cell phone of the patient.

The method to remove a patient is first the caretaker must be login to his or her account by using their fingerprint. They will select the patient that they need to delete and push the delete button.

5.2.7 Prescription Parameters and Schedules

From the conversation with Dr. Jacobs, the EMR (Electronic Medical Records) has a very unique schedule system for the doctors, as shown below in Figure 55. This figure shows that any user can select what is described on a prescription bottle into the Medspencer.

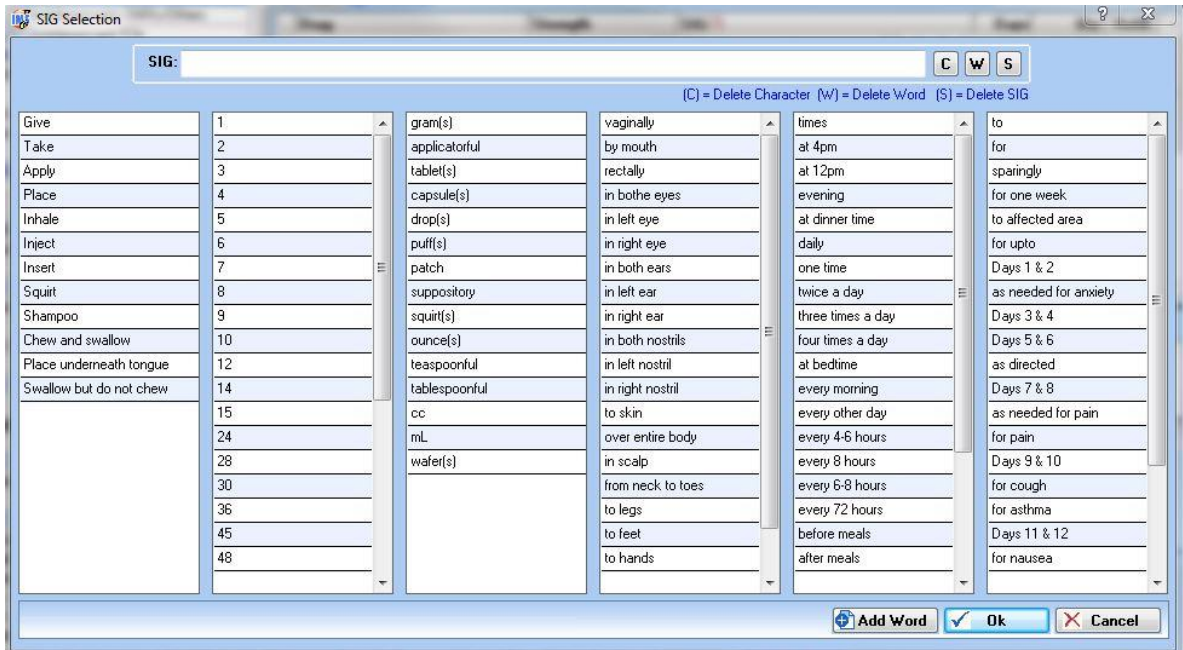


Figure 55. Schedule Selection Screen

The Figure 56 is a prescription label sample on a prescription bottle, and using the simple guide in Figure 55 a caretaker or a reliable family member can select the exact directions that the patient has been prescribed.

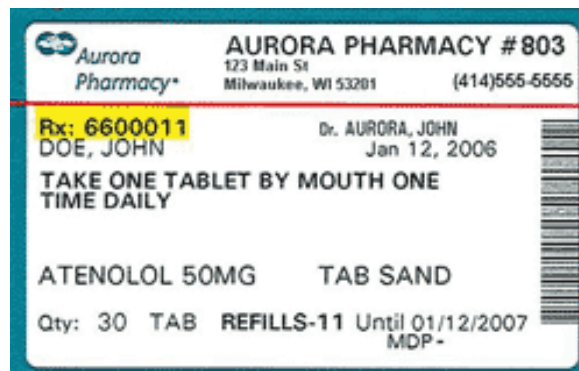


Figure 56. Sample prescription label

5.2.8 As-needed Medication Requests

In order to follow the as needed request or as needed for pain, specific requirement and specification must be followed in the FDA and doctor's recommended plans. However, doctor's do have a tendency to go off label to benefit the patient's health. This is where the benefit outweighs the risk even though the FDA states not to do so because of complications and serious results.

5.2.9 Fixed-schedule Medicine Notifications and Requests

Once the medication has been assigned to the container, the user or caretaker will follow the Prescription parameters and Schedule section to enter how many times, starting at what time and how many pills to take a each moment. In the research, the Food and Drug Administration standards varies between medications and dosage of the medications. They state to take your medication at the same time every day, Tie taking your medications with a daily routine like brushing your teeth or getting ready for bed. Before choosing mealtime for your routine, check if your medication should be taken on a full or empty stomach, keep a "medicine calendar" with your pill bottles and note each time you take a dose, Use a pill container. Some types have sections for multiple doses at different times, such as morning, lunch, evening, and night, when using a pill container, refill it at the same time each week. For example, every Sunday morning after breakfast, purchase timer caps for your pill bottles and set them to go off when your next dose is due. Some pill boxes also have timer functions, when travelling, be certain to bring enough of your medication, plus a few days extra, in case your return is delayed, if you're flying, keep your medication in your carry-on bag to avoid lost luggage. Temperatures inside the cargo hold could damage your medication [85].

In the project the device would reduce the reminding but sounding alarms and other sounds on when to take the medication, the display will show if the patient has taken the medication or not. The alarm will sound off every 5 mins for 90 mins until the patient push the button and the medication can be dispensed. If the patient does not press the button after 90 mins the medication will not be dispense and will be recorded and send an email to the doctor's office.

The fixed schedule will be an increments of morning at 0800, lunch at 1200, evening at 1500, and night at 1900. However, these time will change depending on the sleep pattern of the patient and their work schedule. If the patient request a medication the display will advise them what is the medication that they are requesting. The database will contain the information when the medication can be dispensed after a limited interval. The minimum interval is an hour to 24 hours due to the medication complexity and chemical structure.

5.2.10 Rescheduling Doses

In the rescheduling dosing section, the caretaker will remove the container where the medication is store to show the medical profession that the patient is taking or not taking the medication and the weekly reports that are send to the office for the visit. If the medical

profession, decide to refill the medication they will need to take the container to the pharmacist to get it replaced and refill with the new medication or the old medication. Once that process is done the caretaker will re-enter the direction of the same or new medication in the slot where the container will be stored in. This will notify the pharmacist and physician of the updated change and the slot is a clean and new slot.

5.2.11 Recording events in Schedule & Summary Reports

The Wi-fi module will submit a report every week showing the Chart number of the patient, date of birth, medication name, taken at what time and date, and the amount taken at that time and date. This report will automatically be sent to the email of the office so when the patient goes to the office visit they will not need to take more things than just themselves and any other problems.

In a side note, this report can be uploaded to the application of the caretaker to receive alerts and problems that they need to monitor and inform the doctor or the office staff.

Due to the Wi-fi module not being completed this section of the project was also put on hold for future development and configuration. However, the python code also has a section on creating pdf when Medspencer 2.0 comes around.

5.2.12 Python Structure

The software of the Medspencer is broken into two parts: the software running on the Raspberry Pi CM3L, and the software on the ATmega328P microcontroller. The CM3L contains the code for the components that the user interacts with. It controls the display, reacts to the touch screen. The ATmega328P manages the peripherals, meaning it controls the servo motors, speaker, and fingerprint scanner. This means that the ATmega328P does not react to user input on its own; it receives commands from the CM3L. This communication between the two processors defines the project functionality.

The most critical part of the CM3L software is the frame system. Each frame consists of both a display object and event code. The display portion of a frame contains an arrangement of text, buttons, and other interactable components for the touchscreen. The event code is what reacts to user actions, and triggers changes in the system state, such as sending a command to the ATmega328P or moving to a new frame.

The first frame that is loaded is the Time frame, which simply shows the time. This is the beginning of the frame tree, where a user is presented with the opportunity to scan their fingerprint. If the user is registered as a patient, they enter the Patient frame, and if they are recognized as an admin (or 'caretaker'), they are taken to the administrator frame. If their print is not recognized, they are refused entry. This process is shown in the next figure.

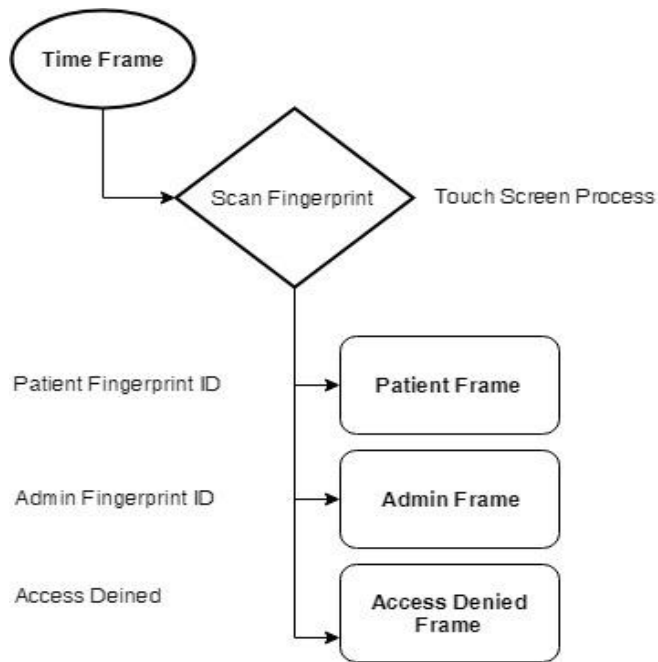


Figure 57. Fingerprint Frame

After the user scans their fingerprint, the system identifies them and moves them to the corresponding frame. In the Patient frame, the user will have access to their demographics, which contains the user’s name, date of birth, address with zip code, phone number and email. From here, the patient can select a medication to dispense.

If the medication is listed as “As Needed”, then it is available at any time. However, some medications have a regulated dose, so they are only available according to a schedule. On this page, patients can either choose a medicine to dispense, or return to the Time frame. The structure of the Patient frame is shown below in the figure below.

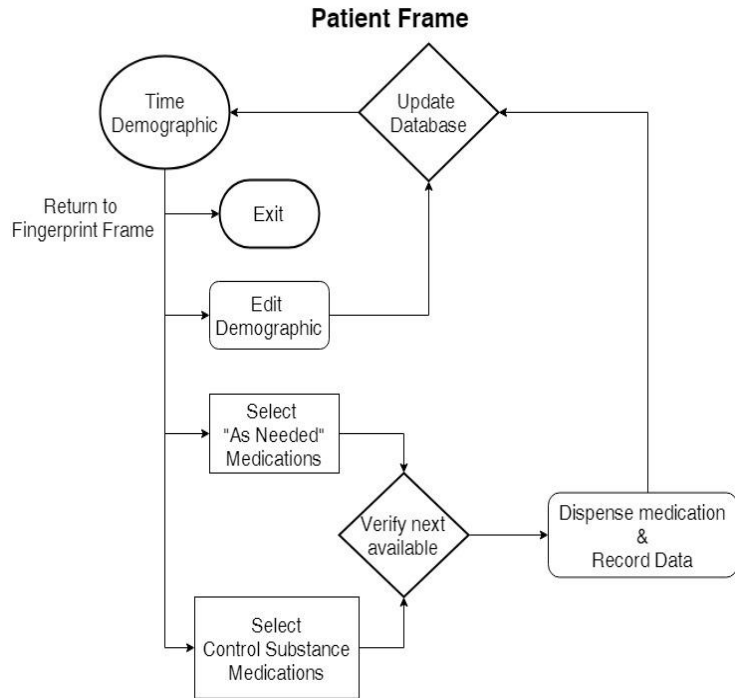


Figure 58. Patient Frame

If a user’s fingerprint is registered as an administrator, they are taken to the Admin frame upon scanning. In the Admin frame, as seen in the figure below, options exist for viewing the admin’s and the patients’ demographics, much like the patient; the important difference is that administrators are allowed to edit this information. Admins can view, update, add, and delete patients, as well as the patients' demographics and medication prescriptions. Administrators also manage prescribing doctors’ profiles. These profiles contain demographic information, much like patient and admin profiles.

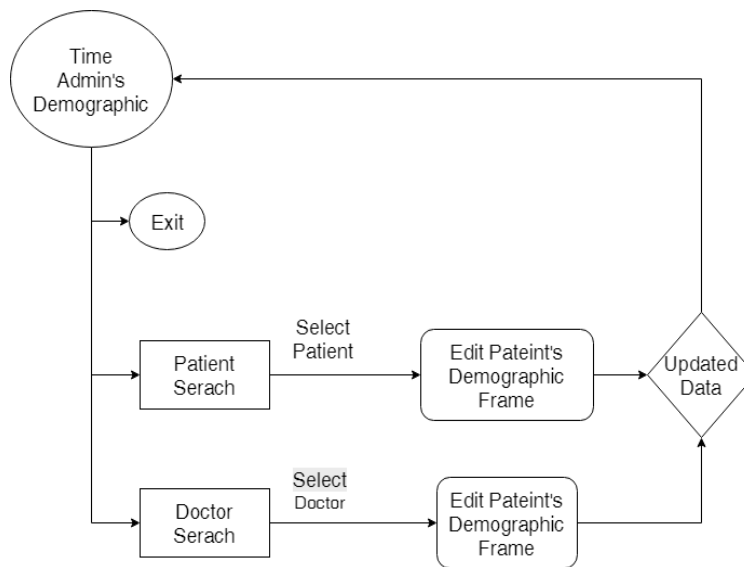


Figure 59. Administrator Frame

All user data is contained in an SQL relational database. This database maps patient, administrator, and doctor data to the frame when it is opened, and it also contains the fingerprint IDs of the patients and administrator. For every patient, the database will contain a unique alphanumeric sequence; a chart number, in medical terms. This will be used to identify and protect the patient's information. For every prescription, the database will record the medication name (brand name, not generic), dosage, timing restrictions, and number of doses.

6.0 Prototype Construction

This chapter details the first steps that have been taken to construct a prototype. Not much has been purchased yet, and even less has been rigorously tested, but this is the roadmap for how the Medspencer will eventually be assembled.

6.1 Parts Acquisition and Bill of Materials

The table below summarizes all of the prices for all of our hardware and electrical components used in the final product; a more detailed budget table is in chapter 8. While looking and comparing for the different necessary components, we noticed that there are a wide variety of option and most of the time is the same product with different price or from a different country. Discussing with the group, it is obvious that one goal is to save money while getting the parts, but we have to consider if the provider and cost are worth the wait because most of the cheaper components can take up to 15 business days, so when considering time constraints it may be worth it to pay extra for shipping or for a different vendor.

Table 11. Final Budget

COMPONENT	COST	DISTRIBUTOR
Display AT070TN90	\$20.00	Alibaba
Microcontroller ATMEGA328P-PU	\$2.15	Mouser
Raspberry Pi CM3 Lite	\$25	ALLIED/Element14
Servo Motors SG90 x5	\$8.85	Amazon
Touch panel	\$14.99	Alibaba
Fingerprint Reader R307	\$9.90	AliExpress (HZGROW)
ESP-12F Wi-fi module	\$3.05	Banggood
Electrical components	\$50.11	Mouser
Voltage regulators	\$30.42	Mouser
PCBs	\$40.00	JLCPCB
Speaker	\$2.06	Mouser
Connectors/jacks	\$27.44	Mouser/ALLIED
RPI CM3L Dev Kit (R&D)	\$200	Element14

The total sum for the final budget was \$440.92. However, \$221.95 of that was pure research and development costs for the CM3L development kit and the Arduino UNO. After subtracting this sum, the manufacturing cost came out to be \$218.97.

6.2 Printed Circuit Board

Our final design utilized two PCBs, as discussed previously in chapter 5. Our final PCBs utilized small surface mount components to minimize board area. We utilized Eagle to design out schematics and PCB layouts [35]. Eagle has many footprint libraries already pre-installed but for purpose of our project, we needed to download more. We downloaded footprint libraries from SnapEDA [107] and SamacSys [108].

The PCB vendor we chose was JLCPCB. They have a lead time of 6-7 days and charge \$2-5 for 10 PCBs and \$18 for shipping [106]. For our various electrical components, we mainly utilized Mouser, which has a short lead time depending on how much you pay for shipping.

6.3 Prototype Constraints

Regarding the prototype, we have focused on separating the whole project in different modules. As recommended by the tutors, divide and conquer, the key is to be sure that every module works perfect independently. To understand how every module works in essential, that way the complete device can easily be assembled. While running some tests, we had to get different microprocessors that runs on different software.

Our first prototype was extremely limited. We are trying to accomplish that the dispenser will be able to read any fingerprint and to dispense. For the fingerprint reader, we were connecting it to an Arduino microprocessor and run it through its respective software. As the Arduino has enough connection slots to connect more components, we attached a servo motor to another port. We use the Arduino microprocessor because is easy to use and to program, but this will not be used in the main project. The purpose of using a microprocessor to run the first steps of the Medapencer is to use this connections as a template for the future main board.

It is software concern to program the frequency of the servo motors. We have to assign different ports to the different servo motors. For purpose of this project, the main board will not have more than 6 servo motor connections. As the MedSpencer will have the chance to interact with the patient through a touchscreen display, the main board will have a data bus connection.

Before we build the box for the dispenser, we have to make sure that all modules and components work fine. As it is mentioned in section 5.1.2, the measures for the box will be 20"x10"x24". Those measures are tentative because we may need more room for the holders for the medication capsules and for the main PCB.

We have thought many ways how to dispense the medicine and how it will be the most effective. After reviewing many other prototypes and also many patents that concern to our

project, the most simple way is to place one servo motor at the bottom of each medication container that works as a filter.

One difference between this project and the existing products is the ability to display time, patient's name and type of medication dispensed. The Medspencer will count with a 7" touchscreen display that enables not just see the information mentioned before, it will also help the administrator to input new data or to edit the existing one. For the prototype, we need only to display the patient's information so as the time. An alert display will also be programmed to show when the schedule medication matches the time.

6.4 Issues and Challenges

This section outlines various problems that the team faced during the prototyping phase of Senior Design 2. Device problems fall into one of two categories: hardware or software.

6.4.1 Hardware Issues

Some of the general challenges we faced while assembling the hardware for this project includes accidentally shorting connections. This could happen if wires touch when prototyping on a breadboard, and if accidental bridges occur when soldering the PCB. We took care to examine the hardware circuitry before providing power, to ensure this kind of mistake did not occur.

Another challenge we faced was how to meet the numerous power requirements for our LCD display, and how to narrow down which products would be best to use for our applications. We used Webench as an aid for to help us choose the most efficient parts in terms of power and board size, and to help us design the required circuitry. Webench was extremely helpful in narrowing down which parts to use.

Another issue we faced with our first PCB revision: we realized the LCD backlight doesn't have any internal resistance, so we blew out the regulator for that 10V power supply. To compensate for that, we checked the required voltage and current in the datasheet, and used Ohm's law to calculate the required resistance to limit that current [53].

A big challenge we faced with hardware was the need to update PCB schematics and designs whenever we made a design change. The biggest design change we made during Senior Design 2 was changing our processing unit choice from the PIC32MZ DA microcontroller, to using the Pi CM3L in conjunction with the ATMEGA328P microcontroller. This required extensive modifications and additions to the PCB schematics and designs, and ordering new PCBs also takes time as JLCPCB has a lead time of 6-7 days (which is actually a very short time when compared to other PCB vendors). Because of the extensive changes and lead time, changing the hardware designs really pushed out time constraints.

6.4.2 Software Issues

The original plan for this project was to use the PIC32MZ DA as our central processing unit. The PIC32MZ DA utilizes the MPLAB Harmony Framework to program it. MPLAB Harmony Framework had poor documentation, contained unfinished code and libraries, and little online community or resources. Our solution was to switch over to the Pi. Then we no longer needed PIC32 processing power, so used ATMEGA as the embedded controller for peripherals.

We also had software issues when using I2C communication. The Pi CM3L was programmed using Python. Initially, Python 3 was used to program the GUI (Graphical User Interface). However, I2C communication is only supported in Python 2. This meant we had to revert back to using Python 2 and rework the GUI.

7.0 Prototype Testing

This chapter discusses how we tested our various components during our prototyping phase. This involved testing physical hardware components and circuits to ensure that we use appropriate operating levels, and testing the designed software implementation. It is important to test each component to ensure that the device will work properly and accurately and none of the components will get damaged. Each component will be tested individually to make sure that nothing is overlooked.

7.1 Hardware Testing

In this section, we detail the procedures that we used to test our various hardware components for this Medspencer project. We tested to ensure that our design circuits and interconnections are valid implementations. This involves making sure that the operating current and voltage levels are appropriate. The operating current and voltage levels are specified in the data sheets for each individual hardware component. The circuits we have designed in order to connect our components to power supply and to each other are specified in Chapter 5. We used the resources available at the electronics labs at the University of Central Florida in order to assemble and test our components. Resources we used for testing include DC voltage supply, digital multimeters, oscilloscopes, breadboards, wires and through-hole components, and so on.

7.1.1 Breadboard and Development Board

In order to have a final PCB either designed or already manufactured, we first need to draw a schematic specifying all its components. A rough circuitry drawn on paper can be the first step of designing the project. The purpose of this step is to move from a schematic, pass through breadboard implementation and to get a final PCB layout. This steps will help us to simulate and get experimental values that may not be got during theoretical calculations. To design on breadboard is an important procedure because we may encounter glitches or unexpected situations like sudden drops of voltage or a component required more voltage to power.

For the initial breadboard testing, we utilized development boards and breakout boards for our processing units and for the Wi-fi module, and we purchased through-hole components for the other electrical components we needed to test. To design the external circuitry, we referenced the datasheets and online resources. Testing using the breadboards helped us verify the functionality of the designed circuitry.

7.1.2 PCB Testing

This section goes into how we tested the PCB to ensure that each component should receive the correct input voltage and current. This is important so as to ensure that the components work to their product specifications, and to avoid damaging any of the components.

In order to test the PCB, we included test points in the PCB design. The test points allowed us to verify whether each mounted piece is receiving the correct voltage and current levels. To design a PCB test point, we placed an open connector hole with a positive and negative terminal on the PCB. The test point sites dedicated pads where we put pin headers. Test points provide the tester the ability to easily connect to locations where tests may be needed. These are advantageous, as opposed to probing the solder locations. Solder locations are not ideal for testing purposes due to their variable shape and contact surface.

For each test point, we verified whether the correct DC voltage and current levels are passing through them. The desired levels depend on which component is supposed to be supplied power. First, we checked that the circuit is receiving the correct voltage level. To do so, we measured the voltage using a digital multimeter set to voltage range. In the case that PCB testing shows voltage levels that are not desired, then troubleshooting may be required. There may be insecure connections or soldering mistakes, or there could be some components that have issues. This is assuming that the circuit layout was already tested and verified on a breadboard, so that should not be the issue when it comes to PCB testing.

7.1.3 Testing Procedure

Running the tests for the Medspencer involved from testing of each of the components individually with their own microcontroller (Arduino, MSP430) to assemble them together in one customized PCB. As many of this components only required lines of codes and a proper connection to the microcontroller, this was just the easy part of it. Many tutorials are available on Youtube about how the connections should be made in order to run or improve the performance of the component.

For a complete view of the design of the main PCB, the group had to identify the inputs of the dispenser and how they will trigger the different procedures of it. One of the Medspencer's inputs is the fingerprint and it can trigger the patients procedure or the administrator procedure. The first test to be done was that the fingerprint reader not just only identifies but to store in the memory and assign a different roll. As the fingerprint is a R307 model, it has a capacity for 1000 fingerprints, so there is plenty of memory for patients, caretakers and administrators. The software section will describe with details the codes used to store and assign roles for each one of the people involved.

For the servo motors, they were tested first individually on an Arduino microprocessor. The code was not that complicated because with the proper library and understanding, it can run with no problems. The servo motor SG90 came with three different size arms, but for our purposes, we have to design our own filter.

Once the servos were tested, they were assigned to one individual port. It is a code concern that each port will be assigned to each patient. While coding and assigning the patients to each medication container the programmer should also consider the dosage of each specific medication to each specific patient. That task can be accomplished by programming the cycle. The prototype only runs from reading the fingerprint of any patient and that triggers the servo motors. Later on the project was improved to recognize each fingerprint and only run the assigned servo motors.

The main purpose for the touchscreen display is for the administrator to input the patients data. The group considered to leave just a LED display and add a keyboard but that would add more power consumption to the project. There are many options on the market and all of them can be tested on an Arduino microcontroller, but this project will have its own software.

For the software testing, the computer engineers need to run many times the software they are creating with the components connected. Let us remember that the inputs for our projects are time (got from the Wi-Fi module) and fingerprint, so the software should be reading this components constantly. When the true event happens, time marches and fingerprint matches, the program should redirect the voltage to the respective servo.

Once the software is tested and certified that runs properly, it should be coded to work with a touchscreen display. The touchscreen display will only interact with the caretaker and the administrator. The extensive coding for the touchscreen usage will be for the administrator options because they are the ones who must enter the patients and medication info, set alarms and set medication slots into the memory.

7.1.4 Microcontroller Testing

The testing of the processing units was done in three phases: unit testing, integration testing, and system testing. Unit testing is done to ensure that each module of the project works independently. Modules will have their own host of test cases, engineered to rigorously test any expected input and output. Integration testing facilitates the combination of modules into a functioning whole, ensuring that proper functionality occurs at each step along the amalgamation process. Finally, system testing proves the complete functionality of the system: does this system do what it is supposed to? This testing should result in a complete product that is ready for the end user. There are two categories of modules that need to be tested as they relate to the microcontroller: software functionality and peripheral functionality.

Software functionality will be unit tested using software, obviously. This involves taking the written software module, providing it with test inputs, and checking whether the results meet expectations. If a particular code module does not give the expected results, then it is obvious where the problem is. The large system operations, the main program loop, touch system, page system, time event system, will have to be tested rigorously, as they are very important. However, smaller code modules will also have to be tested. Each page will have to be run to see if its contents resolve in the expected way. A number of different medicine schedule configurations will have to be run through the time event system to see how they react in that environment, to make sure a broad range of time events work properly. The touch system is rather simple on the software side, but it still needs to be rigorously tested.

Peripheral functionality unit testing, as it relates to the microcontroller, is focused around the interfaces between the microcontroller and the peripheral itself. Unit testing assumes the peripheral works properly, and tests to see how the microcontroller works with those inputs. Ensuring the software code interacts properly with the processing unit's

communication interfaces is important so the microcontroller can interact with the peripherals.

Once the unit testing finished, then the time for integration testing arrives. This is the process where fully tested modules are combined into larger working wholes, and this aggregation process must be tested at every step in the hierarchy.

Software module combinations are not much more difficult to test than their module counterparts. However, these tests usually require more testing data, and more analysis, to determine what the expected result would be. At this stage, the actual functionality of the program is starting to take shape, and this means the results may be related to more than just code considerations. At this stage, a page would be tested for how it relates to other pages: what data is being given to a new page from its parent, what data is the parent expecting to receive, how does the active page react to the touch system and the time event system, etc. These sorts of considerations affect every part of the software system, as many of these modules will have an effect on each other.

Table 12. Example integration test cases related to page interaction

Test Case Summary	Prerequisites	Test Procedure	Test Data	Expected Result	Test Environment
Page creation and data passing	An active page must call a new page and pass it data.	Make the current page call a new page, and pass it an integer to display.	A test integer.	The new page will display the passed integer.	Raspberry Pi CM3L
Page registering touch event	An active page must have an event that triggers from touch data. Touch data must be set.	Run the active page and see if it reacts to the touch data.	A set of coordinates to represent a touchscreen event.	The touch coordinates will trigger a specific page event.	Raspberry Pi CM3L
Page registering time event	An active page must have an event that triggers from a time event. A time event must be set.	Run the active page and see if it reacts to the time event.	A time event for the active page to react to.	The time event will trigger a specific page event.	Raspberry Pi CM3L

Integration testing the peripherals is where the real functionality starts to come together. Once all the different connecting systems are assembled, then the peripherals can be slotted into their appropriate spot in the system. The microcontroller constructs a command, which is sent to the UART, which is transmitted to the fingerprint scanner. All of those connections are being tested and solidified in this stage.

System testing is where the two categories of modules come together: does the software run the hardware? Here, the testing team will be asking questions like “Can a patient check their medicine level?” “What happens when a fingerprint isn’t recognized by the system?” “Does the onscreen keyboard properly relay text to the box it was intended to?” These sorts of tests not only check to see if the various modules work together, but if the product is in a usable state.

Table 13. Example system test cases

Test Case Summary	Prerequisites	Test Procedure	Test Data	Expected Result	Test Environment
Patient check medicine level	The database must contain a test patient and test medicine.	Using the touchscreen, the test patient must navigate to their medicine page and view its status.	The level of the test medicine .	The medicine's status page will display the correct level of medicine remaining.	Raspberry Pi CM3L
Fingerprint not recognized	The fingerprint library must be enabled, but not contain the test fingerprint.	The unrecognized finger must be placed on the fingerprint scanner.	An unrecognized fingerprint.	The system should refuse the unrecognized fingerprint and tell the user it is unrecognized.	ATMEGA 328P with fingerprint reader
Keyboard return text properly	An active page must contain a call to the keyboard, and then must display the return text.	The active page must initiate the call to the keyboard, and the test user must enter in a string of characters and return to the active page.	A test string to enter on the keyboard .	The test string will appear on the page that called the keyboard.	Raspberry Pi CM3L

The microcontroller testing will be occurring throughout much of the development process of the Medspencer as modules are completed, then as whole systems are assembled. Testing during development helps to keep things moving, and it also brings alert to problems early so they can be fixed early in the development cycle.

7.1.5 Touchscreen Testing

The touchscreen begins with the user. The user's finger, pen, or other pointing utensil touches the screen, triggering an electrical connection between two conductive layers. The electrodes on each side of each layer are wired to inputs and outputs on the microcontroller. Each of these interfaces needs to be tested. Some basic tests will be done to the screen with a multimeter while it is hooked up to a power supply, while more detailed tests are done when the touchscreen is wired to the microcontroller.

The first test cases will be focused on the microcontroller itself, with a voltage difference between the electrodes on one layer and a multimeter measuring the voltage on the other layer. Some of these tests might include "How much does the voltage change between set points?" "Does the voltage drop linearly across the entire screen, or are parts of the screen more resistant?" "How much force is required on a touch to trigger an electrical connection?" "How close to the source voltages does a touch at the edge of the screen measure?" All of these tests are designed to improve the team's knowledge about the screen, and give a benchmark for later tests that involve the microcontroller itself.

With the assurance that the electrical properties of the screen are correct, the testing phase can move on to integrating the touchscreen with the microcontroller. Simple test cases would involve ensuring electrical continuity between screen and microcontroller, and testing the microcontroller's ADC detection. Once everything is properly hooked up, then more complex tests can be performed. These new tests ask questions like "How quickly can the software read a finger touch?" "How accurate are the generated coordinates in relation to the touch location?" These tests depend not only on the electrical properties of the touchscreen, but also the functionality of the microcontroller's hardware, and the efficiency of the software code.

Table 14. Test cases relating to electrical properties of the touchscreen

Test Case Summary	Prerequisites	Test Procedure	Test Data	Test Environment
Voltage change between points	A voltage difference between the two electrodes of the touchscreen's top layer.	Touch the screen at the set locations. Measure the voltage of the bottom layer's electrodes.	Set voltages on the electrodes. Consistent points on the screen to touch. Same amount of force with each touch.	Touchscreen with power supply attached to top layer and multimeter attached to bottom layer.
Voltage rate of change	A voltage difference between the two electrodes of the touchscreen's top layer.	Touch the screen at a number of equally spaced points. Measure the voltage of the bottom layer's electrodes. Calculate the rate of change of voltage.	Set voltages on the electrodes. Consistent equally spaced points on the screen to touch. Same amount of force with each touch.	Touchscreen with power supply attached to top layer and multimeter attached to bottom layer.
Measuring the margins of the touchscreen	A voltage difference between the two electrodes of the touchscreen's top layer.	While measuring the voltage of the bottom layer's electrodes, see how close to the margins of the screen a touch will still make an electrical connection. Note the voltage at which it stops. Compare margin voltage with the rail voltages.	Set voltages on the electrodes. Same amount of force with each touch.	Touchscreen with power supply attached to top layer and multimeter attached to bottom layer.

As the hardware and software systems are designed, it is important to be constantly testing new setups. This ensures that proper data has been gathered for further development, and finds problems in finished components before other modules depend on them.

Table 15. Test cases relating to the software performance of the touchscreen

Test Case Summary	Prerequisites	Test Procedure	Test Data	Test Environment
Electrical continuity between touchscreen and microcontroller	A touchscreen and a microcontroller wired together.	Using a multimeter, test every connection for electrical continuity and excess resistance.	A reasonable resistance threshold to be under.	Touchscreen wired to Raspberry Pi CM3L, tested with a multimeter.
Touch response speed test	A touchscreen and a microcontroller wired together, with code to measure voltages with the ADC.	Measure the time it takes for a touch on the screen to result in a change in status.	Consistent points on the screen to touch.	Touchscreen wired to Raspberry Pi CM3L.
Touch coordinates accuracy test	A touchscreen and a microcontroller wired together, with code to draw a crosshair at the touch location.	Touch the screen and see where the crosshair appears.	A wide spread of touch locations to test.	Touchscreen wired to Raspberry Pi CM3L.

7.1.6 Display Testing

The display system involves two very different components: the microcontroller’s LCD controller and the screen itself, along with the wires that connect them. However, it would be difficult to test the LCD controller without an LCD to control, so testing will focus on the display’s hardware first, then move on to the controller’s software.

The LCD chosen for this project has a number of different requirements: specific power inputs, proper wiring designs, signal timing characteristics, etc. When designing the circuits for the screen, consideration must be taken for these factors. Power obviously is important, as too little would result in a non-functioning screen and too much would result in a broken screen, so a good test case for the screen could be “The analog power circuit will provide a voltage between 10.2 and 10.6 volts, and be able to source currents up to 50 mA.” Similar cases could be made for the other power inputs. It is important to remember that some wires on this screen are designed to source power and some are data communication lines. For instance, the screen has both a Power Ground pin and a Common Voltage pin. In some applications, it can be very important to keep your signal and power

circuits separated to eliminate noise, so one good test could be “The power lines do not create noise in the data lines.” These sorts of tests ensure that the circuits designed to power and run this LCD will fulfill the requirements set forth in the datasheet and will result in proper performance.

Once the LCD is wired up to a power source and to the microcontroller correctly, then it is time to work with the LCD controller in code. One of the most important things that allows the controller to properly communicate with the LCD is timing information. The LCD’s datasheet has multiple pages containing very specific timing instructions, like how fast to run the clock, how many pulses to have in between lines, when to trigger a new line or new column, etc. So, a great test case would be “Does the LCD controller’s timing information result in a perfectly displayed image?”

Table 16. Test cases relating to the LCD

Test Case Summary	Prerequisites	Test Procedure	Test Data	Test Environment
Proper power provided	A circuit has been constructed to provide power to the LCD.	Connect a multimeter to the power outputs of the circuit. Measure the voltage output and the max current it provides.	The output voltage should be between 10.2 and 10.6 V, and should be able to source current up to 50 mA.	Constructed power circuit measured via multimeter.
Noise insulation between signal and power lines	An LCD that is hooked up to power and data lines.	While the screen is running, use an oscilloscope to ensure there is no cross-contamination of noise in the data lines.	The data lines should be either 0 V or 3.6 V.	LCD wired to power source and microcontroller .
LCD Controller timing	An LCD that is connected to an LCD controller.	The LCD controller should send the screen image data. See if the screen displays the image correctly.	A test image.	LCD wired to power source and microcontroller .

7.1.7 Wi-Fi Module Testing

When starting to the configure a wireless frequency interference microcontroller, read from the user guide on the essential commands and parameters for the ESP8266EX Wi-fi chip [93]. Helpful resources to test the ESP-12F module with the ESP8266EX chip connected to the Raspberry Pi were found on Github and Osh Lab [94]. Examples on how to code the module with a Linux driver and on how to connect the hardware for the module were provided on these websites.

7.1.8 Fingerprint Module Testing

The fingerprint module must be tested to ensure that it undergoes proper operation according to the product specifications. The R307 fingerprint module may be powered using either 3.3 V or 4.2-6 V. The working current should be around 50 mA. The module may communicate with an MCU of 3.3 V or 5 V via UART.

To do the initial testing to verify the circuit design and user operation for the fingerprint module, the circuit was built on a breadboard and verified. The design circuit was tested to ensure that the output voltage and output current to the load match the desired values. The output voltage levels of the MCU's general input/output pins were tested to check that they output the appropriate voltage level of 5 V. Once the circuit design is verified on the breadboard, the circuit plan for the fingerprint module can be finalized for the PCB. Once the PCB is received, the voltage and current levels will again be tested to ensure that everything is connected properly and that the components all work properly.

Testing also involved sending basic commands from the MCU to the fingerprint module. This allowed us to verify the functionality of the fingerprint module. The basic commands used include system-related commands as well as fingerprint analysis related commands, which include collecting a fingerprint image, enrolling a new fingerprint, and searching the library for matching fingerprints. The instruction format is specified within the Software Design section of Chapter 5.

7.1.9 Power Supply Testing

The objective of power supply testing is to test our method of power supply for the system, and to test the power inputs going into each hardware component, in order to ensure that safe voltage and safe current levels are applied to the system and each of its hardware components.

Before we can test the overall power supply to the Medspencer system, we had to purchase an AC-to-DC converter that can convert U.S. standard domestic power. We also purchased a power jack that can be connected to our PCB. To test that the correct power is being supplied to the system, we connected the AC-to-DC converter to the PCB via the power jack. Then we measured the voltage supplied using a digital multimeter set to voltage

range. We ensured the voltage supply was 20V. We connected the multimeter to the PCB through one of the designed PCB test points. This allowed us to check whether the correct voltage is being supplied to the system.

We also needed to test the voltage and current levels being supplied to each individual component, such as the fingerprint module, touchscreen display, servo motors, Wi-Fi module, and speaker. Each component has specific circuits that were designed to convert the power levels to appropriate operating levels. These circuits are displayed in the Hardware Design section of Chapter 5. In order to test that the correct power level is applied across a component, first the circuits were constructed on a breadboard to simulate what the final implementation will look like. This physical test will require a breadboard and a DC power supply, as well as a digital multimeter to make measurements. The output of the circuit was tested using a digital multimeter set to voltage range, in order to ensure that the correct voltage level will be applied across the component. Then the load was connected on the breadboard. The voltage was applied and the current was measured with the digital multimeter in order to test that there is an appropriate operating current. In the case that measured voltage and current levels do not correspond to the desired levels, then it is possible that the circuit design may need to be altered, or a component is not working properly, or the connections are not secure. Troubleshooting may be required.

After the circuits were tested and verified on the breadboard, the PCB was designed using Eagle and then printed. After the PCB was printed and obtained, we had to solder on various components. The same tests were repeated utilizing the PCB test points in order to ensure that the correct voltage and current levels will be applied to each component. Utilizing the multiple test points on the PCB allowed us to reduce the total testing time for the PCB. It also makes troubleshooting easier, as it takes less time to identify the integrated circuit or component that has failed.

7.2 Software Testing

In this section, tests are outlined to ensure the quality of the Medspencer's software. These tests are either simulated in the development environment, or run as code on the Raspberry Pi compute module.

7.2.1 IDE Used

The environment used for testing the Python is Eclipse IDE Version Photon Release (4.8.0) Build ID: 20180619-1200. Afterward, more software was downloaded and install called PyDev for Eclipse and PyDev for Eclipse Developer Resources which contain the libraries for basic python. Furthermore, the original Python 27, Python 37, and Python IDE which is the Python Interpreters was installed for executing the python code.

7.2.2 Python Procedures, Simulation, and Physical Testing

In writing a code structure, the libraries are set at the top, global variables, follow by creating database tables and sample patients, functions that will be used inside the frames and then the “Home” frame or the initializing frame which create the layout of all the other frames. Follow by the individual frames and the main function which starts the process to open the initialize frame.

However, the structure of the frames will not allow the database to input data if there is no data to be put in. So a dummy patient was created in the database to allow the frames to be created and run through. Once the frame to scan the user fingerprint is input the data will change to the correct user and have access the necessity frames.

When simulating, each frame is for created and ran then adjusted and modified to follow the structure that was planned. After the simple structure was made per section of frame, it was later testing on the Raspberry Pi where it was discover the python3 works for Graphic User Interface (GUI) but won't work for communication using I2C or UART. However, python2 can communication with I2C and UART but have a different coding for GUI. So there was a try and exception clause create to run python3 and python2. So when it was simulated in the eclipse it will work but when it was in raspberry pi it can still work.

An example of the first frame is the Fingerprint Frame show in Figure 60. Nevertheless, the Home frame is the initialization frame but its purpose is to create a blueprint of the frame system.

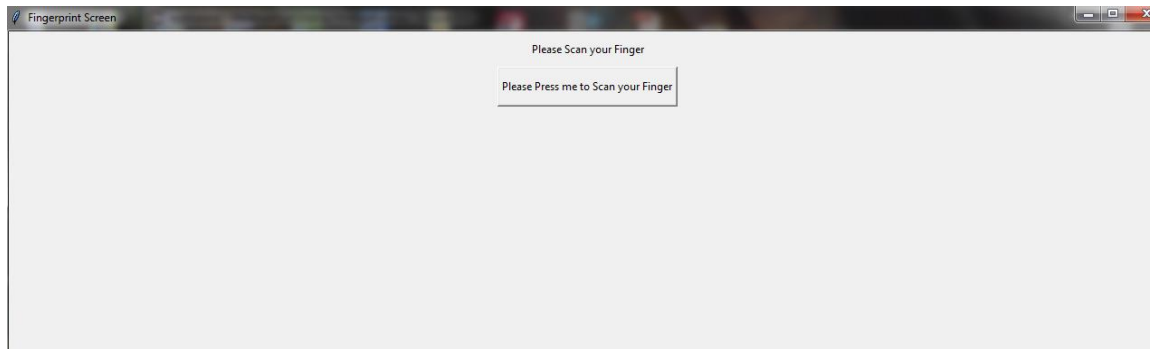


Figure 60. Fingerprint Frame

When the patient scans their fingerprint the patient screen will pop-up the patient's demographics and medication linked to the servo motors shown in Figure 61.

Patient Screen

QUIT

First Name	Sakeenah	Skittles-Red
Last Name	Khan	
Date of Birth	06062008	
Doctor	Fredesvinda Jacobs-Alvarez, Child, Adolescent and Adult Psychiatric	
Address	Unknown	
ZipCode	32832	
Phone Number	4075751176	
Email	Unknown	

Figure 61. Patient Screen

If the administrator or caretaker scans their fingerprint then a different set of information comes on for them shown in Figure 62.

Admin Page Screen

EXIT

First Name	Admin	Add Fingerprint
Last Name	Ivan	
Relationship	Caretaker	
Phone Number	Alvarez	
Email	4072263733	
		Edit Motor 1
		Edit Motor 2
		Edit Motor 3

Figure 62. Administrator or Caretaker

7.2.3 ATMEGA328P Programming Procedures

To develop software for the ATmega328P, we used the Arduino development environment. The Arduino environment consists of both hardware and software. The hardware side is comprised of a development board, which features a ATmega328P socket, pinouts for the chip, and a USB to Serial chip programmer. Software is written in the Arduino IDE, which provides an interface for writing C-like code that can use the built-in Arduino libraries. The Arduino development platform was designed for use by non-engineers and hobbyists, so it simplifies and abstracts away some of the underlying technical aspects of microcontroller development, but this does allow for faster and cheaper development.

The basis of the ATmega328 software is a main functionality loop, and two rotating buffers for incoming and outgoing I2C data. Development began on the main software loop, where the peripherals are controlled. A library for the fingerprint scanner was used to send commands to it over serial UART, so the challenge with that subsystem was related to which commands to send and in what order, with an eye on reliability in case of communication issues. Both the speaker and servos are sent signals from the ATmega328's pulse width modulators, but the way they are controlled is different. To generate a tone on the speaker, the frequency of the note desired is initialized in an array, then played for a specific amount of time. With an array of note frequencies and note durations, the alarm function simply plays all in order then returns. The dispense function transmits a pulse width signal that varies from 0 degrees to 180 degrees and back again. The dispense command also includes a byte that tells which motor to control, which is broken down into its binary representation and used to tell the demux which output to enable. There were problems with this, of course, but the Arduino IDE is easy to debug and resources are plentiful. Basically, I did a thing, then if it didn't do the thing, I had to fix the thing, then it did the thing.

Once this functionality was designed and working, then we moved on to the I2C communication. There were a number of problems with the I2C communication throughout this project, from hardware issues with wire crosstalk to software issues with timing. However, the ability to send individual bytes from the Compute Module helped development. After basic communication was established from the ATmega328 to the CM3L, further steps were taken to make the system more resilient to communication problems. Transmission buffers were added to ensure multiple communications would not overlap each other and cause unintended functionality, and to allow the CM3L to retrieve data for every command it sent. Once the I2C communication was added, a configuration flag was added to allow the ATmega328 to function with a serial console to a computer, or to read from the transmit buffer.

8.0 Administrative Content

This chapter details the administrative content to facilitate organizing and planning out the Medspencer project. This includes sections such as a milestones discussion, division of labor, and budget and finance discussion. The milestones discussion details the various milestones we outlined for the completion of the Medspencer project. Breaking the project down into smaller tasks made working on the project more manageable. We also set goals for when each milestone should be completed. The division of labor section discusses which members take responsibility for the various components of the project. The individual costs for different components and the overall budget are also discussed. Additionally, a section on project operation is included. This section functions as the owner's manual for the Medspencer product.

8.1 Milestone Discussion

Discussing the different approaches of how the project would be idealized and created, the group has created a list of milestones with their respective dates to complete.

The initial report: Divide and Conquer was due January 28 and its content was a brief description of the project including a table where describes the cost of all components. The House of quality in which the project showed the relation of the market features with the technical features and how they will be improved. That table showed that in order to expand our project to handle more patients it will require the expansion of dimension and a more powerful microprocessor for all information.

An interview with Dr. Fredesvinda Alvarez, our sponsor, was set to February 24 in order to have a better understanding of this medical field and how our product can get into the market and how it will impact with the improvements and updates discussed. That interview set the confidence of the sponsor in our group. As the interview kept going and all the questions and answers from both parts were discussed, there was also room for suggestions that the group kindly considered knowing the limitation of time.

The table of contents was due March 20 and it gave us a broad view of what our research should aim. The table of contents shows all the topics and sub-topics of the project. It can be considered as a checklist of all the documentation needed and all the steps taken from planning, research and development of the project. The fact that a topic, out of the group's area of expertise, is mentioned on the table of contents is a clear message that more of a research there will be interviews.

A first draft was due for the Senior Design 1 document for April 9 with half of the final 120 pages to deliver. This is to show that all research and test was properly documented and considered under the proper standards. As the project was divided into different modules, both majors chose the topics to describe in a very detailed way. While researching, different standards and previous work were found, so the project was constantly evolving in order to get a better impact on the market. The research guided the group through to a considerably large number of components that may work better than the

ones mentioned on the *Divide and Conquer* document. In here, the group is taking in consideration the delivery time vs the effective work vs the cost of each component needed.

The final document was due April 26 and then a properly documented, researched and cited project is expected. The 120 page document is not just limited to research and theory but to test and a final schematic. The final schematic is the final diagram of how the project will work and will include the final PCB connected through certain electronic components to the different modules (fingerprint reader, servo motors, touchscreen display).

For the final presentation of the project the following semester Summer 2018, the project was running properly without any help from a pre-assembled microcontroller and also with a proper enclosure that get the attention of the judges. The final documentation by then should be edited with all new tests and new data acquired while building the project. This final document will have more explanation about how the different modules work for the entire project but now using the experience of running the tests rather than theory.

For Summer 2018, the final critical design review presentation and project demonstration took place on July 24. The 8 page conference paper was due a week prior. The corrected 120 page Senior Design document and website were due on August 30.

Table 17 shows the milestone chart for our Senior Design project. It includes milestones for both Senior Design 1 and 2. It details the start and end dates for each milestone, and the status of each milestone. As the Senior Design project was completed by the end of the Senior Design 2 semester, each of the milestones is marked as completed.

Table 17. Milestone chart

Number	Task	Start	End	Status	Responsible
Senior Design I					
1	Ideas	1/15/2018	1/17/2018	Completed	Group 6
2	Project Selection & Role Assignments	1/15/2018	1/24/2018	Completed	Group 6
3	Initial Document – Divide & Conquer	1/18/2018	1/28/2018	Completed	Group 6
4	Research with Psychiatrist and Pharmacist	2/3/2018	2/24/2018	Completed	Group 6
5	Table of Contents	3/10/2018	3/20/2018	Completed	Group 6
6	First Draft	3/15/2018	4/1/2018	Completed	Group 6
7	Final Document	3/20/2018	4/26/2018	Completed	Group 6
8	Schematics and Dispenser design	4/1/2018	4/26/2018	Completed	Group 6
9	Recording & Data abstraction (Order and test parts)	3/18/2018	4/26/2018	Completed	Group 6
Senior Design II					
10	Order & Test Parts	5/1/2018	5/15/2018	Completed	Group 6
11	Design Prototype & PCB Layout	5/15/2018	7/17/2018	Completed	Group 6
12	Build Prototype	5/15/2018	6/15/2018	Completed	Group 6
13	Testing & Redesign	6/15/2018	7/15/2018	Completed	Group 6
14	Finalize Prototype	7/15/2018	7/24/2018	Completed	Group 6
15	Final Report	7/15/2018	7/24/2018	Completed	Group 6
16	Final Presentation	7/15/2018	7/24/2018	Completed	Group 6

8.2 Division of Labor

The product is too broad that involves more than electrical and computer engineering. The materials for the enclosure and the medication containers are topics for the chemical and industrial engineers. As our group is made of two electrical engineers, one computer and one double major, they focused on certain modules that concern their area of expertise.

For the electrical engineers, their area involves from the soldering standards to the schematics. Everything that relates connections, components, digital diagrams are part of the electrical duties. Regarding the standards, they are relevant when manufacturing the main board and running the tests. For the computer engineers, their area involves programming language standards and features for microprocessors. They have to consider which microprocessor is powerful enough to handle the whole program and what instructions take less cycles to run and make an effective program.

A flow chart is essential for both majors because it gives a very descriptive narration of the sequence and conditions of the project. According to this “map” either electrical and computer engineers can follow and have an idea of which components and lines of codes to use. While looking for which components to get for the project, the electricals should consider the power usage and input voltage. Having knowledge of that information, a more precise and accurate schematic can be drawn in which all components are labeled and calculations are close to practice measurements. The map can also help computer engineers to improve the code and get a better time response.

The figure below shows us the main responsibilities of the project and it is divided according to the major (electrical engineering, computer engineering) and also with a distinctive color the mutual responsibilities. Those responsibilities are only the ones considered before starting the project because when the project has started and the assembly and programming begins, many other tasks should come up. That is why the mutual responsibility of research is imperative. An extensive research within the field can either clarify or question more project needs.

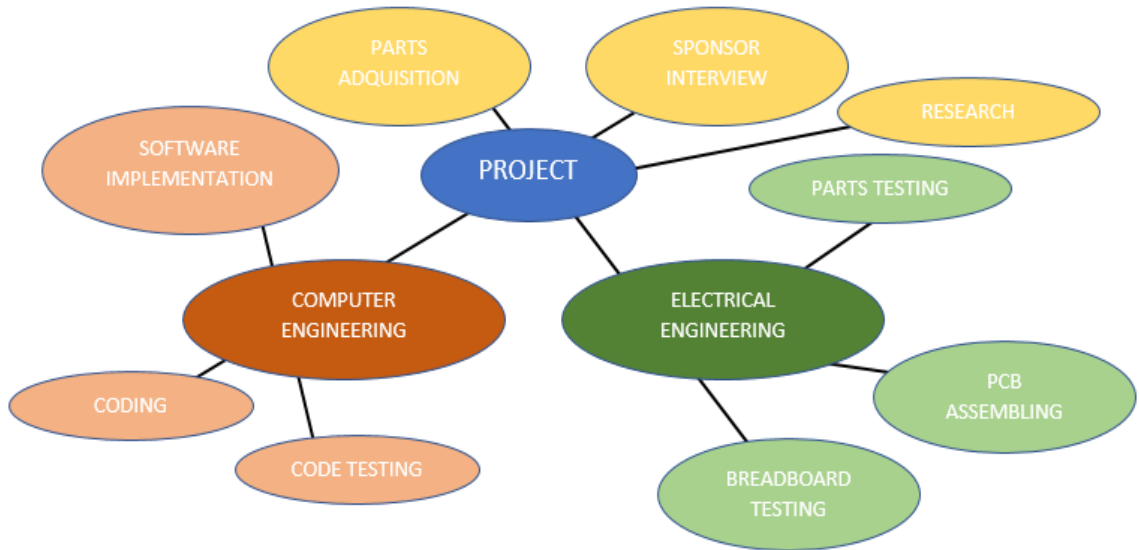


Figure 63. Division of labor

There may be some responsibilities that does not fall in anyone’s hands, like the PCB printing, because we hired a company that will provide us three boards to avoid a higher probability of glitches. In the event of glitching, breaking, losing or wasting the boards, there are tutorials online that explain how to create a PCB with the proper tools and equipment. In that case, the responsibility of creating a new PCB would fall upon the electrical engineers. That would be the scenario and responsibilities if the purpose is to avoid the waiting or to avoid the overspending. An overnight request will affect the products price and also that would mean a lack of responsibility from the group part to the sponsor.

When the time of assemble and program the project begins, each individual member should be responsible of their own tools, computer software and computer good performance. It may sound that it has nothing to do with the project but it is a good sign and a good review to the sponsor that a prepared and organize team is taking care of a project that has the potential to leave a mark in the industry. For instance, it is recommended that the group should not rely on a single computer, but to share the documentation and have the same access to software.

That the project is divided for a faster compliance, does not mean that the responsibilities are individual, it means that every person is focusing more on a specific topic but also helping to the partners.

The testing part can be considered as a play-around or get-to-know stage because all members are testing each module and pushing them to the limits. For computer engineers, they are changing codes, adding instruction, program in different languages, getting translators from assembly language to high level language. For electrical engineers, the test can be from making the servo motors to rotate and check all connections for each component.

When running some tests for the main board, many glitches can show up and sometimes the fastest solution is to hard-wire ports. While soldering, an excess of tin lead can create a short circuit; meanwhile assembling all components together, the voltage given to one of those may not be enough to power them up. The values for the voltages and power will be different from the theoretical and sometimes the arrangements made on the schematic have to be redistributed while building the final product.

The table below shows how the responsibilities were split up and assigned to different team members. P indicates primary responsibility, while S indicates secondary responsibility. Putting a person in charge of each task ensured that someone would be checking up on that task's progress, even in other members pitched in to help complete it.

Table 18. Work Distribution

	<u>Power Regulation</u>	<u>Controller Circuitry</u>	<u>Peripheral Circuitry</u>	<u>Embedded Development</u>	<u>Communication Interfacing</u>	<u>Python Development</u>
Gustavo	P		P	S		
Ivan					S	P
Matthew		S		P	P	
Sakeenah	P	P	P			

8.3 Budget and Finance Discussion

The project was financed by Dr. Fredesvinda Jacobs, head of Esperanza Behavioral Health and Services in Orlando. As this project will have a good impact at the health industry, Dr. Alvarez will predict that this project will have a good demand on years to come. As the project did not exceed the cost of \$450, the group paid for the testing and prototype stage and the sponsor returned all investment when the final product was finished. Table 31 shows all the components necessary and researched throughout many websites. After considering the reviews, the cost and the performance, the following were the parts chosen to be part of the project.

Table 19. Updated Budget List

COMPONENT	COST	DISTRIBUTOR
Display AT070TN90	\$20.00	Alibaba
Microcontroller ATMEGA328P-PU	\$2.15	Mouser
Raspberry Pi CM3 Lite	\$25	ALLIED/Element14
CM3L Conector	\$20	ALLIED
Servo Motors SG90 x5	\$8.85	Amazon
Touchscreen	\$14.99	Alibaba
Fingerprint Reader R307	\$9.90	AliExpress (HZGROW)
ESP-12F Wi-fi module	\$3.05	Banggood
Capacitors	\$25.05	Mouser
Resistors	\$19.01	Mouser
Inductors	\$3.18	Mouser
Diodes	\$0.80	Mouser
Voltage regulators	\$30.42	Mouser
PCBs	\$25.00	JLCPCB
Speaker	\$2.06	Mouser
SD card slot	\$1.75	Mouser
Amplifier LM386	\$0.88	Mouser
Demultiplexer CD74HC238M	\$0.65	Mouser
DC power jack PJ-067B	\$2.50	Mouser
50p TTL interface connector	\$3.19	Mouser
BSS138 MOSFETs x2	\$0.54	Mouser
RPI CM3L Dev Kit (R&D)	\$200	Element14
Arduino UNO R3 (R&D)	\$21.95	Amazon
TOTAL	\$440.92	

The total cost was \$440.92. Disregarding the development board prices, the manufacturing cost was \$218.97. The manufacturing cost was within our goal of less than \$300. There are many other elements that are not listed and were acquired on the run while building and testing the prototype. The shipping costs are not included in this budget.

For the prototype building and testing, the group acquired the Arduino, the servo motors and the fingerprint reader. Those were the cheaper components to get and also the ones for which the tests were easy to run. The final PCB was purchased at the end of July 2018 because the schematics for it should run a perfect simulation. The component that cost more is the microcontroller PIC32MZ because this is the responsible to run the whole project with a very small time delay.

As a group we have discussed during our meetings the prospective components and where to get them for accommodate price. The necessary components can be get from aliexpress, as mentioned on the Table 30, but with the restriction that they take at least 15 business days to get to the United States from China and India. Even though the prices are affordable, we still need to discuss if the components will be used or they can be replaced or something else.

Another topic that involves financing but is different from sharing expenses, is how easy can people get benefit from this project. Regarding to how the patient may get this sort of service, an interview with Xavier Ortega, medical visitor from the company Prime-Care Home-Care located in Saint Petersburg, that sends nurses, therapist and social workers to visit patients, check for their compliance and report to the physician in charge. A key topic mentioned during the interview was how these services are covered, they work with Medicare, so the patient have to either pay as low as he or she can or nothing. This service is pretty similar to what the final product is going to offer. Considering this company and job, it gives us an idea that the product may be eligible for being covered by the insurances companies and let it be introduced to the market.

8.4 Project Operation

The Medspencer is a home appliance to be used by multiple members of a household. There are two types of users that may access the Medspencer: the administrator/caretaker, and the patients. The Medspencer has a user-friendly GUI that utilizes a touch screen display, and the options shown on the display are self-explanatory.

To use the Medspencer, first a user must scan their fingerprint. The Medspencer will then identify the user as either the administrator or as a patient and will pull up the appropriate account. (If the fingerprint is not recorded in the Medspencer database, then the user will be denied access to the device.)

When the administrator accesses the Medspencer, they have the ability to add, delete, and modify all of the patient accounts as well as the medications stored within the device. Patient data they may modify or add includes the patient's demographics (such as name, fingerprint, address, etc.), contact info for the primary care provider, accessible

medications, and medication schedules. The administrator can also access the medication containers to add, remove, or refill medications.

Once the patients and medication schedules are programmed into the Medspencer, the Medspencer will sound an alarm whenever it is time for a patient to take a dose. When the alarm sounds, the patient should approach the device and scan their fingerprint. Once the fingerprint is scanned and the patient is identified, the patient's account will pull up. The patient may view their patient and medication data; however, they may not edit this data. If it is time to take a scheduled dose of medication, then the Medspencer will dispense the appropriate medication. The patient may also request any as-needed medications that are prescribed to them. When a patient requests an as-needed medication, the Medspencer will check whether enough time has passed since the last request; then it will dispense the medication if it is safe to do so.

Whenever a patient takes a dose of medication, the Medspencer records the event in the patient schedule. Every week, the records in the schedule are compiled into a Compliance Report. The report is emailed to the primary care provider at the end of each week.

9.0 Conclusion

For our senior design project, we decided to create the Medspencer. The main goal of the Medspencer is to increase medication compliance in patients, by providing automated support and reminders at home. The Medspencer organizes the patient's various medications, schedules the doses, and reminds the patient when it is time to take a dose. The Medspencer should also reschedule doses in the case of a missed dose and send reports on the patient's medication compliance to the PCP.

During our first semester of senior design, the team was tasked with designing the Medspencer and planning its construction. The first step in this process was conducting extensive research. We researched similar products on the market and met with medical professionals in order to get a better idea of what patients and medical professionals need, and what is currently available to them. This helped us solidify our overall goals for the Medspencer and what features to include to best meet those goals. Then we researched existing technologies that we could use to create the Medspencer. To do so, we considered the features we want to include in the Medspencer and researched the different technologies available in order to provide those features. Then we went into the design phase. This involved figuring out the specs we need and choosing physical hardware components to purchase that meet those specs. We planned the hardware and circuit design and the interconnections between the MCU, power supply, and various hardware components. We also planned the software design for the Medspencer system. Lastly, we laid out the prototype construction plans and testing procedures we plan to go through as we verify our designs to create the final Medspencer product.

During our second semester of senior design, the team had to actually construct the Medspencer and present a working prototype by the end of the semester. The planning and research we did in senior design 1 was helpful in providing a starting point for senior design 2. Initially, we bought the various hardware peripherals and components we needed to use and tested them using breadboards, through-hole components, and testing components (such as DC power supply, digital multimeters, and oscilloscopes) at the UCF electronics labs. After testing and verifying the functionality of our hardware components and their external circuitry, we designed the PCB schematics and board layouts using Eagle. After further research, testing with the components and PCB, and several design choice changes, the PCB schematics and board layouts were revised. By the end of July, the team presented a critical design review and demonstration of the working prototype to a panel of faculty judges. The team also corrected the 120 page senior design document and wrote an 8 page conference paper, and constructed a website to document and showcase the Medspencer project.

Appendix A: Citations

- [1] “16-Bit PIC Microcontroller Peripheral Integration.” Microchip Technology, Microchip Technology, May 2017, ww1.microchip.com/downloads/en/DeviceDoc/30010109D.pdf
- [2] “2N2102/2N2102A: NPN Silicon Transistor.” *Central Semiconductor Corp.*, Central Semiconductor Corp., www.mouser.com/datasheet/2/68/2n2102-42211.pdf
- [3] “32-Bit PIC and SAM Microcontrollers Peripheral Integration.” Microchip Technology, Microchip Technology, July 2017, ww1.microchip.com/downloads/en/DeviceDoc/60001455c.pdf
- [4] “3 Most Common PCB Assembly Defects.” *Optimum Design Associates*, blog.optimumdesign.com/3-most-common-pcb-assembly-defects
- [5] *Arduino Tutorial: Using a Servo SG90 with Arduino*. Dir. educ8s.tv. Perf. educ8s.tv. 2016. Youtube.
- [6] Association Connecting Electronic Industries. Requirements and Acceptance for Cable and Wire Harness Assemblies. Arlington Heights, Illinois, Mar. 2002.
- [7] Association Connecting Electronics Industries. Rework, Modification, and Repair of Electronics Assemblies. Bannockburn, Illinois, Nov. 2007.
- [8] Association Connecting Electronics Industries. *Acceptability of Printed Boards*. Northbrook, Nov. 1999. PDF.
- [9] Association Connecting Electronics Industries. *Acceptability Standard for Manufacture Inspection and Testing of Electronic Enclosures*. Bannockburn, Sept. 2013. PDF.
- [10] Association Connecting Electronics Industries. *Generic Standard on Printed Board Design*. Northbrook, Feb. 1998. PDF.
- [11] Association Connecting Electronics Industries. *Performance Test Methods and Qualification Requirements for Surface Mount Solder Attachments*. Bannockburn, Jan. 2002. PDF.
- [12] Association Connecting Electronics Industries. *Requirements for Soldered Electrical and Electronic Assemblies*. Northbrook, Feb. 2012. PDF.
- [13] Association Connecting Electronics Industries. *Specification for Base Materials for Rigid and Multilayer Printed Boards*. Bannockburn, Dec. 1997. PDF.
- [14] “AT070TN90 Specification.” *Panelook*, InnoLux Display Corporation, 2010, cdn-shop.adafruit.com/datasheets/AT070TN90.pdf
- [15] “Automated Medication Dispensers.” *MedReady Inc.*, MedReady Inc., 2014, www.medreadyinc.net
- [16] “Automated Medication Dispensing Service.” *Philips Lifeline*, Philips, 2016, <https://www.lifeline.philips.com/pill-dispenser/health-mdp.html>
- [17] “Automatic Pill Dispensers.” *e-Pill Medication Reminders*, e-Pill, LLC, 2018, www.epill.com/dispenser.html
- [18] “Basics of SPI Communication Protocol.” Circuit Basics, www.circuitbasics.com/basics-of-the-spi-communication-protocol
- [19] “Basics of the I2C Communication Protocol.” Circuit Basics, www.circuitbasics.com/basics-of-the-i2c-communication-protocol
- [20] “Basics of UART Communication.” Circuit Basics, www.circuitbasics.com/basics-uart-communication/

- [21] Boxall, Alistair B.A. “The Environmental Side Effects of Medication.” *US National Library of Medicine*, 5 Dec. 2014, www.ncbi.nlm.nih.gov/pmc/articles/PMC1299201/
- [22] Brown, Marie T, and Jennifer K Bussell. “Medication Adherence: WHO Cares?” *Mayo Clinic Proceedings*, Apr. 2011. *US National Library of Medicine*, <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3068890/>
- [23] “Brushless DC Motors vs. Servo Motors vs. Inverters.” *Oriental Motor*, ORIENTAL MOTOR USA CORP., 2017, www.orientalmotor.com/brushless-dc-motors-gear-motors/technology/brushless-dc-motors-servo-motors-inverter.html
- [24] Camara, Carmen, et al. “Security and Privacy Issues in Implantable Medical Devices: A Comprehensive Survey.” *ScienceDirect*, June 2015, www.sciencedirect.com/science/article/pii/S153204641500074X
- [25] “CC3220 SimpleLink Wi-Fi and IoT Solution a Single-Chip Wireless MCU Getting Started Guide.” Dec. 2017. PDF File.
- [26] “CC3220 SimpleLink Wi-Fi LaunchPad Development kit Hardware.” January 2018. PDF File.
- [27] “CC3220 SimpleLink Wi-Fi Wireless and Internet-of-Things Solution, a Single-Chip.” Dec. 5, 2017. PDF file.
- [28] “CC3220 SimpleLink Wi-Fi Wireless and Internet-of-Things Solution, a Single-Chip Wireless MCU.” Feb. 2017. PDF File.
- [29] “Clear Polycarbonate Tubing, 5/8’ ID, 3/4’ OD, 1/16’ Wall, 3’ Length.” Amazon, www.Amazon.com/dp/B000OMJ50K/ref=nav_timeline_asin?_encoding=UTF8&psc=1
- [30] “Comparison: Automated Medication Dispensers.” *The Senior List*, The Senior List, LLC, 18 Sept. 2017, www.theseniorlist.com/2017/09/automated-medication-dispensers/
- [31] Cua, Yvette M, and Sunil Kripalani. “Medication Use in the Transition from Hospital to Home.” *US National Library of Medicine*, 19 Feb. 2013, www.ncbi.nlm.nih.gov/pmc/articles/PMC3575742/
- [32] “CUI Inc. PJ-002B.” Digi-Key Electronics, 2018, www.digikey.sg/product-detail/en/cui-inc/PJ-002B/CP-002B-ND/96965
- [33] “Custom Builds Of SQLite or Porting SQLite To New Operating Systems.” SQLite, sqlite.org/custombuild.html
- [34] “DI-IPSC-81433A, SOFTWARE REQUIREMENTS SPECIFICATION.” *EverySpec*, NAVY/EC, Dec. 1999, everyspec.com/DATA-ITEM-DESC-DIDs/DI-IPSC/DI-IPSC-81433A_3709/
- [35] “Eagle.” *Eagle: PCB Design Made Easy*, 9.0.0, Autodesk, 2018, www.autodesk.com/products/eagle/overview
- [36] Eidson, John. “IEEE-1588 Standard for a Precision Clock.” *Agilent Technologies*, IEEE, Oct. 2005, www.nist.gov/sites/default/files/documents/el/isd/ieee/tutorial-basic.pdf
- [37] “Electronics II Laboratory Manual.” *Department of Electrical & Computer Engineering*, University of Central Florida, Jan. 2012, <http://www.ece.ucf.edu/files/labs/EEL%204309%20Jan%202012.pdf>
- [38] “e-Pill Med-O-Wheel SMART Portable Automatic Pill Dispenser.” *e-Pill Medication Reminders*, e-Pill, LLC, 2018, www.epill.com/medowheelsmart.html

- [39] “Fans Clogged by Dirt.” Dirty Computer, Feb. 2012, <http://www.dirtycomputer.com/?p=21>
- [40] “FDA Drug Safety Communication: Revised Recommendations for Celexa (Citalopram Hydrobromide) Related to a Potential Risk of Abnormal Heart Rhythms with High Doses.” *U.S. Food & Drug Administration*, 24 Aug. 2011, www.fda.gov/Drugs/DrugSafety/ucm297391.htm
- [41] “FH12 Series Datasheet.” Hirose Electric Co., Ltd, Apr. 2018, www.hirose.com/product/en/download_file/key_name/FH12/category/Catalog/doc_file_id/31648/
- [42] “Finger Print Sensor R307 (New R305).” *Sunrom Electronics/Technologies*, Sunrom Electronics & Sunrom Technologies, 2018, www.sunrom.com/p/finger-print-sensor-r307-new-r305
- [43] “FX10 Series Datasheet.” Hirose Electric Co., Ltd, Apr. 2018, www.hirose.com/product/en/download_file/key_name/FX10/category/Catalog/doc_file_id/31650/
- [44] Generation Robots, *Generation Robots*, generationrobots.com: <https://www.generationrobots.com/en/402044-esp8266-Wi-Fi-serial-module.html>
- [45] *Getting Started with the Fingerprint Sensor*. Dir. Adafruit Industries. Perf. Adafruit Industries. 2012. Youtube.
- [46] “Graphical and Segmented Display Solutions.” Microchip Technology, Mar. 2018, ww1.microchip.com/downloads/en/DeviceDoc/00001699C.pdf
- [47] Harris, Tom. “How Fingerprint Scanners Work.” *HowStuffWorks*, InfoSpace Holdings, LLC, Sept. 2002, computer.howstuffworks.com/fingerprint-scanner2.htm
- [48] Hedegaard, Holly, et al. “Drug Overdose Deaths in the United States, 1999–2016.” *Centers for Disease Control and Prevention*, US Department of Health and Services, Dec. 2017, www.cdc.gov/nchs/products/databriefs/db294.htm
- [49] “How Do Piezoelectric Buzzers Work?” Quora, 2018, www.quora.com/How-do-piezoelectric-buzzers-work
- [50] “IEEE Approves Standards for Data Encryption,” *IEEE Standards Association*, IEEE, Dec. 2007, <https://web.archive.org/web/20080203190557/http://standards.ieee.org/announcements/StdForEncryption.html>
- [51] “IEEE Standard for Software and System Test Documentation.” *IEEE Xplore*, IEEE, July 2008, ieeexplore.ieee.org/document/4578383/
- [52] “IEEE Standard for Information Technology-Systems Design-Software Design Descriptions.” *IEEE Xplore*, IEEE, July 2009, ieeexplore.ieee.org/document/5167255/
- [53] “INNOLUX AT070TN94 7.0 Inch TFT-LCD Panel.” *Amazon*, Innolux, www.Amazon.com/INNOLUX-AT070TN94-inch-TFT-LCD-Panel/dp/B0107O964E/ref=sr_1_3?ie=UTF8&qid=1524795809&sr=8-3&keywords=AT070TN94
- [54] Intel. (2009, January). IEEE std 1149.x and software debug. United States.
- [55] “ISO/IEEE 11073-10424:2016/Cor 1:2018 .” *International Organization for Standardization*, ISO, Jan. 2018, www.iso.org/standard/74911.html
- [56] Jayant. “Small Loudspeaker for Computer or Cell Phone.” *Circuit Digest*, circuitdigest.com/electronic-circuits/small-loudspeaker-circuit-diagram

- [57] “Jboss.org.” *Community Documentation*, 25 Apr. 2018, docs.jboss.org/jbossas/docs/Server_Configuration_Guide/4/html/J2EE_Declarative_Security_Overview-Enabling_Declarative_Security_in_JBoss.html
- [58] “JST - PH2.0 Interface Stereo Enclosed Speaker (2 PCS), 3 W, 8 Ohm. Applied to All Kinds of Audio Production, Raspberry Pi and Arduino DIY Projects.” *Amazon*, Amazon, www.Amazon.com/JST-Interface-Production-Raspberry-Projects/Dp/B0738NLFTG/Ref=sr_1_1?Ie=UTF8&Qid=1524793024&Sr=8-1&Keywords=CQRANQI0007US
- [59] Lady Ada. “Transformer-Based AC/DC Converters.” *Adafruit*, Adafruit, 27 Dec. 2017, learn.adafruit.com/power-supplies/transformer-based-ac-dc-converters
- [60] “Livi Features.” *Livi*, PharmRight Corporation & Livi, 2018, liviathome.com/features
- [61] “LM78XX/LM78XXA: 3-Terminal 1 A Positive Voltage Regulator.” *Fairchild*, Fairchild Semiconductor Corporation, Sept. 2014, www.mouser.com/ds/2/149/LM7812-461970.pdf
- [62] “Loudspeaker.” *Wikipedia*, Wikimedia Foundation, Apr. 2018, en.wikipedia.org/wiki/Loudspeaker
- [63] “Medication Adherence - Taking Your Meds as Directed.” *American Heart Association*, American Heart Association, Sept. 2016, www.heart.org/HEARTORG/Conditions/More/ConsumerHealthCare/Medication-Adherence---Taking-Your-Meds-as-Directed_UCM_453329_Article.jsp#.WuFemYjwbIU
- [64] “MPLAB Harmony Help.” Microchip Technology, Nov. 2016, ww1.microchip.com/downloads/en/DeviceDoc/MPLAB_Harmony_Help_v202.pdf
- [65] Orenda. “Introduction to Simplex, Half Duplex and Full Duplex.” *Medium*, July 2016, medium.com/@fiberstoreorenda/introduction-to-simplex-half-duplex-and-full-duplex-fbda8d591e3a
- [66] “Passive Radiator (Speaker).” *Wikipedia*, Wikimedia Foundation, Feb. 2018, [en.wikipedia.org/wiki/Passive_radiator_\(speaker\)](http://en.wikipedia.org/wiki/Passive_radiator_(speaker))
- [67] “PIC32MZ DA Family Starter Kit User’s Guide.” Microchip Technology, Apr. 2017, ww1.microchip.com/downloads/en/DeviceDoc/70005311A.pdf
- [68] “PIC32MZ Graphics (DA) Family Datasheet.” Microchip Technology, Oct. 2017, ww1.microchip.com/downloads/en/DeviceDoc/60001361F.pdf
- [69] “Piezoelectricity.” *Wikipedia*, Wikimedia Foundation, Apr. 2018, en.wikipedia.org/wiki/Piezoelectricity
- [70] “R307 Fingerprint Module User Manual.” Hangzhou Grow Technology Co., Ltd., Feb. 2011, <https://www.dropbox.com/sh/orprmb3bgb6lqb6/AACpiIXOF91R7-RQ9OkD4JXha?dl=0>
- [71] “REG1117/REG1117A: 800mA And 1A Low Dropout Positive Regulator.” *Burr-Brown Products*, Texas Instruments, July 2004, www.ti.com/lit/ds/symlink/reg1117.pdf
- [72] Saarni SI, Gylling HA. Evidence based medicine guidelines: a solution to rationing or politics disguised as science? *Journal of Medical Ethics* 2004;30:171-175.
- [73] Sabaté, Eduardo, editor. *Adherence to Long-Term Therapies: Evidence for Action*. World Health Organization, 2003, http://www.who.int/chp/knowledge/publications/adherence_full_report.pdf

- [74] “Section 22. 12-Bit High-Speed Successive Approximation Register (SAR) Analog-to-Digital Converter (ADC).” Microchip Technology, Oct. 2017, ww1.microchip.com/downloads/en/DeviceDoc/60001344D.pdf
- [75] “Section 54. Graphics LCD (GLCD) Controller.” Microchip Technology, July 2017, ww1.microchip.com/downloads/en/DeviceDoc/54_GLCD_60001379A.pdf
- [76] Services, U.S. Department of Health and Human. *Guidance for Industry Container Closure Systems for Packaging*. Rockville: Food and Drug Administration, 1999.
- [77] “SimpleLink™ CC3120, CC3220 Wi-Fi® Internet-on-a-Chip™ Solution Built-In Security Features.” Mar. 2017, pp. 1–32., www.ti.com/lit/an/swra509a/swra509a.pdf
- [78] “SimpleLink™ Wi-Fi® and Internet of Things CC3220 Programmer's Guide.” *Manual*, Feb. 2017, pp. 1–53., www.ti.com/lit/ug/swru464/swru464.pdf
- [79] Singh, Ganeev. “Fingerprint Detection Using Microcontroller.” EngineersGarage, 2012, www.engineersgarage.com/contribution/fingerprint-detection-using-microcontroller
- [80] “Technical Considerations for Additive Manufactured Medical Devices, Guidance for Industry and Food and Drug Administration Staff.” December 5, 2017. PDF file.
- [81] Thakkar, Danny. “Understanding Biometric Identification with Optical Fingerprint Scanners.” *Bayometric*, Bayometric, 2016, www.bayometric.com/biometric-identification-optical-fingerprint-scanners/
- [82] “The Difference Between an Illegal and a Controlled Substance.” *Foundation Recovery Network*, 24 Apr. 2018, www.foundationrecoverynetwork.com/the-difference-between-an-illegal-and-a-controlled-substance/
- [83] “Title 21 Code of Federal Regulations: Part 1304 - Records and Reports of Registrants.” *Diversions Control Division*, US Department of Justice, July 2003, www.deadiversion.usdoj.gov/21cfr/cfr/1304/1304_11.htm
- [84] “VSP-PTD-N: Technical Data Sheet.” *Vandal Stop Products*, Atlas American, hotelsinzanzibar.co/mla-format-works-cited-scarlet-letter/mla-format-works-cited-scarlet-letter-best-of-mla-format-citation-scarlet-letter-fresh-mla-format-samples/
- [85] “Why You Need to Take Your Medications as Prescribed or Instructed.” *U.S. Food & Drug Administration*, 16 Feb. 2016, www.fda.gov/Drugs/ResourcesForYou/SpecialFeatures/ucm485545.htm
- [86] “Ybee 10x Pcs SG90 Micro Servo Motor Mini SG90 9g Servo For RC Helicopter Airplane Car Boat Robot Controls.” Amazon, www.Amazon.com/Ybee-Micro-Helicopter-Airplane-controls/dp/B06WRS7PG8/ref=sr_1_2_sspa?ie=UTF8&qid=1524787921&sr=8-2-spons&keywords=SG90&psc=1
- [87] Zullig, Leah L. “Engaging Patients to Optimize Medication Adherence.” *NEJM Catalyst*, NEJM Group, May 2017, catalyst.nejm.org/optimize-patients-medication-adherence/
- [88] “ATMEGA328/P Datasheet.” Atmel, Nov. 2016, http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf
- [89] “Compute Module Datasheet.” Release 2. Raspberry Pi (Trading) Ltd., June 2018, https://www.raspberrypi.org/documentation/hardware/computemodule/datasheets/rpi_DATA_CM_2p0.pdf

- [90] “AR1000 Series Resistive Touch Screen Controller [Datasheet].” Microchip, 2016, <http://ww1.microchip.com/downloads/en/DeviceDoc/40001393C.pdf>
- [91] “CD74HC238 High-Speed CMOS Logic 3- to 8-Line Decoder [Datasheet].” Texas Instruments, Aug. 2004, <http://www.ti.com/lit/ds/symlink/cd74hc238.pdf>
- [92] “I2C bi-directional level shifter.” ARDUINO, 2018, <https://playground.arduino.cc/Main/I2CBi-directionalLevelShifter>
- [93] “ESP8266EX Datasheet.” Version 5.7. Espressif, Nov. 2017, https://www.mouser.com/pdfdocs/Espressif_ESP8266EX_Datasheet.pdf
- [94] J. Burkett. “ESP8266 Raspberry Pi GPIO Wifi.” Osh Lab, May 2016, <https://oshlab.com/esp8266-raspberry-pi-gpio-wifi/>
- [95] “Servo Motor SG90 Datasheet.” Tower Pro, http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf
- [96] “Model: CMS-40504N-L152; Description: Speaker [Datasheet].” CUI, Inc., June 2016, <https://www.mouser.com/datasheet/2/670/cms-40504n-1152-1311451.pdf>
- [97] “LM386 Low Voltage Audio Power Amplifier [Datasheet].” Texas Instruments, May 2017, <http://www.ti.com/lit/ds/symlink/lm386.pdf>
- [98] “WEBENCH Power Designer.” Texas Instruments, 2018, <http://www.ti.com/tools-software/design-center/webench-power-designer.html>
- [99] “TPS62745 Dual-cell Ultra Low IQ Step Down Converter for Low Power Wireless Applications [Datasheet].” Texas Instruments, June 2015, <http://www.ti.com/lit/ds/symlink/tps62745.pdf>
- [100] “LM43601 3.5-V to 36-V, 1-A Synchronous Step-Down Voltage Converter [Datasheet]” Texas Instruments, Jan. 2018, <http://www.ti.com/lit/ds/symlink/lm43601.pdf>
- [101] “ADM660/ADM8660 [Datasheet].” Rev. C. Analog Devices, 2011, http://www.analog.com/media/en/technical-documentation/datasheets/ADM660_8660.pdf
- [102] “LMR23610 SIMPLE SWITCHER® 36-V, 1-A Synchronous Step-Down Converter [Datasheet].” Texas Instruments, Feb. 2018, <http://www.ti.com/lit/ds/symlink/lmr23610.pdf>
- [103] “LMR23615 SIMPLE SWITCHER® 36-V, 1.5-A Synchronous Step-Down Converter [Datasheet].” Texas Instruments, Feb. 2018, <http://www.ti.com/lit/ds/symlink/lmr23615.pdf>
- [104] “LM5165 3-V to 65-V Input, 150-mA Synchronous Buck Converter With Ultra-Low IQ [Datasheet].” Texas Instruments, July 2017, <http://www.ti.com/lit/ds/symlink/lm5165.pdf>
- [105] “LM5009A 100-V, 150-mA Constant ON-Time Buck Switching Regulator [Datasheet].” Texas Instruments, Sept. 2016, <http://www.ti.com/lit/ds/symlink/lm5009a.pdf>
- [106] “JLCPCB.” Shenzhen JLC Electronics Co., Ltd., 2018, <https://jlcpcb.com/>
- [107] “SnapEDA.” SnapEDA, 2018, <https://www.snapeda.com/home/>
- [108] “Electronic Component Search Engine.” SamacSys, 2018, <https://componentsearchengine.com/index.html>

Appendix B: Copyright Permissions

Microchip

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

Texas Instruments

TI further grants to K-12 educational institutions, universities, and community colleges permission to download, reproduce, display, and distribute TI Services solely for use in the classroom, provided that such institutions identify TI as the source of TI Services and include the following credit line: “Courtesy of Texas Instruments.” Unauthorized use of any TI Service is expressly prohibited by law, and may result in civil and criminal penalties. This grant of permission terminates if you breach any provision in these Terms of Use or Service Terms. Upon termination, you agree to destroy any materials relating to TI Services.

Circuit Basics

Re: Circuit Basics - Web Enquiry



Scott C <circuitbasics@gmail.com>
Tue 4/24, 5:55 PM
Sakeenah Khan ↕

  Reply all | 


Flag for follow up. Start by Wednesday, April 25, 2018. Due by Wednesday, April 25, 2018.

Hi Sakeenah,

Sure, you can use any of the images on the site for your research report. If you could provide a link to the article where the image is located in your reference, that would be excellent. Good luck on your project!

Scott

e-pill

 Gmail

Permission to use e-pill image

Stefan Solvell <ssolvell@gmail.com>
To: sakeenahkhan@gmail.com

Medication Reminders | e-pill | CADEX | PuffMinder | ePill | <https://www.epill.com>

Dear Sakeenah,

Thank you for your e-mail.

You have our permission to use the image. Please complete the Photo Permission, sign it and fax it back to us.

Photo Permission

XXXXXX


The undersigned hereby grant(s) permission to (hereinafter referred to as the "Author"):

This permission is for Author's use of photographs taken by undersigned of:
images of e-pill® Medication Reminders copied from our Web site (www.epill.com).

The photographs may be used in XXXXX.

It is understood that the above grant of permission to Author shall in no way restrict republication by me or with my consent of the copyrighted photographs in other works.
The following photo credit line must be used by Author for each photo:

Courtesy e-pill® Medication Reminders
www.epill.com


1-800-540-0095


Stefan Solvell, President & Manager e-pill, LLC

Conditions accepted by:

Author: _____ Date: _____

(Please fax a signed copy to e-pill, LLC at Fax number: 781-235-3252)

Digi-key

 Gmail Sakeenah Khan <sakeenahkhan@gmail.com>

FW: Permission to use images Reference Id: 20180422-607908360496

Orders <Orders@digikey.com> Mon, Apr 23, 2018 at 8:26 AM
To: "sakeenahkhan@gmail.com" <sakeenahkhan@gmail.com>

-----Original Message-----

From: Kevin Brown [mailto:Kevin.Brown@digikey.com]
Sent: Monday, April 23, 2018 4:14:06 AM
To: Orders
Subject: Re: Permission to use images Reference Id: 20180422-607908360496

Sakeenah,
With appropriate attribution to Digi-Key for the images it is okay to use.
Thank you,
Kevin

Kevin Brown
Vice President, Global Brand Management
DIGI-KEY ELECTRONICS
701 Brooks Ave. South
Thief River Falls, MN 56701 USA
kevin.brown@digikey.com
+1 218 681 8000 x1010 (local)
+1 800 338 4105 x1010 (direct)

US Department of Health & Human Services

Image Galleries

Many HHS agencies maintain galleries of images that are in the public domain and can be used on your website or other communications materials. Visit the galleries at the links below or the directory of HHS Flickr accounts for more information on how images should be used and credited, if necessary.

Circuit Digest

Disclaimer

All the content published is sole copyright of CircuitDigest.com, if anyone is referencing the content published, reference should be cited. The content cannot be used for any commercial purposes without any prior permission from CircuitDigest.com team.

Information provided here is only for educational purpose. While we try to provide accurate information, we make no claims, promises, or guarantees about the accuracy or completeness of the same.