Temperature Checkpoint System

A Technical Report submitted to the Department of Electrical and Computer Engineering

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Matthew Bain

Spring, 2021

Technical Project Team Members

Andy Hui

Amanda Rein

Jack Shefer

Greg Vavoso

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Harry S. Powell, Department of Electrical and Computer Engineering

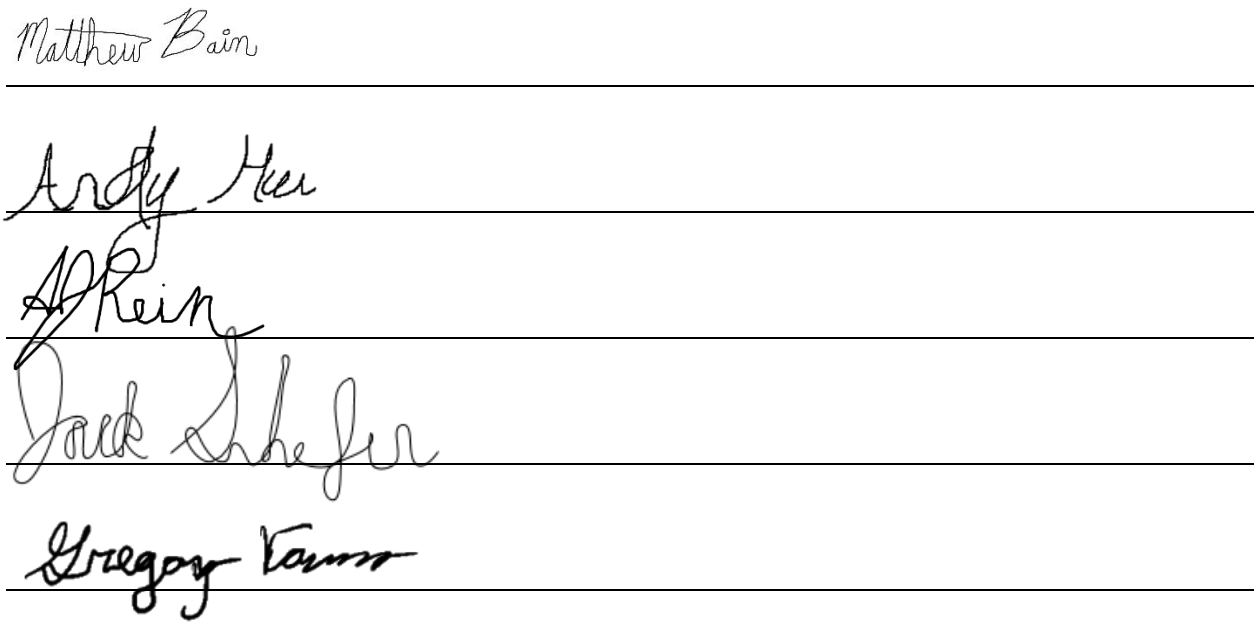
Temperature Checkpoint System

Matt Bain, Andy Hui, Amanda Rein, Jack Schefer, and Greg Vavoso

December 12, 2020

Capstone Design ECE 4440 / ECE4991

Signatures



Statement of work:

Matt Bain

During the course of the semester, I was primarily involved with the hardware aspects of our system. I created the circuit schematics after we determined parts as a team. I was also the lead on PCB design including both individual part footprint creation and full PCB design and integration. Each team member picked a part of the project to be the conceptual expert on, and I was the lead on the locking mechanism of our project. Therefore, I determined the electric strike and high-side switch used to control the locking mechanism. Once the board was designed and shipped, I wrote the embedded code to control and test the locking mechanism. Through the final weeks of the semester, I was a jack of all trades and assisted with any testing, late-stage design changes, or debugging efforts required to finish the project on time.

Andy Hui

My contributions to the project were spread throughout the project. Primarily, I worked on the orchestration, testing, and verification of the motion and temperature sensor. Due to temperature sensor shortages nationwide, I was in charge of communicating with sensor experts to ensure maximum temperature sensor efficiency. I was also responsible for the mechanical design including CAD modeling and final mechanical assembly. Towards the end of the semester I went to the lab often and helped with hardware assembly and debugging as well as contributed in component soldering. I also aided my team members in other areas of the project. I followed and assisted Matt's great work in the PCB design, aided in hardware testing, and contributed to our team's report writing.

Amanda Rein

I was the primary team member responsible for selecting and researching a motion detector that would integrate well with our system. I spent a significant amount of time educating myself on various motion sensing options before selecting the PIR sensor that I ended up purchasing for the system. Additionally, I wrote the embedded code for interfacing with and testing our motion detector. Besides taking the lead on our motion detector, I also did a significant amount of report writing. I've always been more skilled at high-level design decisions and drafting write-ups, so I found that I was able to contribute to the team best in those capacities.

I was present at most all of the team meetings, both in the lab and via Zoom, and jumped on any task that needed extra attention. I sometimes spent my time aiding Greg and Jack in debugging or writing code tests, and other times I spent soldering or helping Matt with testing various hardware components. As such, I was a floater on the system during assembly and testing, which seemed to serve the team best overall.

Jack Schefer

Throughout the project, I took primary responsibility for the Raspberry Pi server which included receiving Bluetooth data, writing it to a database, and exposing a usable web interface. I selected the software stack to use on the Raspberry Pi and then implemented all of the required code. I also took a lead role in determining the packet structure of our Bluetooth messages.

I also tried to help my teammates throughout the project. Towards the beginning of the semester, I had a larger role in creating PCB footprints for our components until Matt fully took that responsibility and I started checking over his work. I spent a lot of time with Greg debugging our use of the temperature sensor's I2C interface. Lastly, while Andy soldered most of the board together I did a few components and made sure things were in the right place.

Greg Vavoso

My primary responsibility in the project was the embedded system. I researched and implemented a real-time operating system on our MSP432 microcontroller. This involved creating tasks for each subsystem, such as the temperature sensor, and allowing them to work together to meet our full system functionality. With the responsibility of the embedded system came the responsibility of integrating each subsystem into the overall system. This included understanding the communication protocols and defining the microcontroller's software and hardware functionality for supporting this communication.

In addition to the embedded system, I also took on some smaller roles such as managing the budget, helping Andy with physical design, organizing part orders, assisting Jack with Bluetooth communication, and offering other general support.

Contents

Table of Figures

Abstract

This project's final deliverable is an automated health screening door connecting temperature measurements to the door lock. When a prospective entrant approaches a door outfitted with this system, they will place their wrist under a mounted box, triggering a motion detector and awakening a non-contact temperature sensor. The temperature sensor will take a temperature reading from the person's wrist, and that reading will be interpreted within a MSP432 microcontroller to determine whether to unlock the door. If the temperature is above a healthy range, the MSP432 will communicate to the door that it should lock and prevent that person's entry to the space until they can return with a normal temperature reading. At the same time that the MSP432 makes the locking decision, the temperature reading is sent to a Raspberry Pi server, which stores all temperature readings in a database. This database is utilized to render a web dashboard for stakeholders to this door locking system. Anyone interested in seeing the number of people allowed to (or prevented from) entering the space, or other statistics on door usage, may find this information on the website.

Background

The COVID-19 pandemic has wreaked havoc on the global economy. Commercial businesses, such as retail, restaurant establishments, and entertainment venues, have been forced to reopen to avoid bankruptcy, despite the serious health risks. Consumers also itch to return to a sense of normalcy, leave their homes in which they have been confined for months and support local businesses. Educational institutions are also reopening their doors amidst the pandemic, as students desire to remain on-track in their education. While reopening the economy, supporting local businesses, and keeping children in classrooms are all noble pursuits, they have created a dire need for ways to discern individual's health before they potentially infect the others in these spaces.

Ideally, individuals would be tested for COVID-19 before they are permitted to enter a shopping mall, restaurant, gym, movie theater, or classroom, but this is not possible for multiple reasons. For starters, there are no existing testing options that can return results in a short enough time to make this feasible. Secondly, there has been a shortage of COVID tests worldwide, making it incredibly difficult and expensive for corporations or institutions to acquire even enough tests to give students before they return to a residential school, for example. The University of Virginia, a prominent and powerful institution in the United States, is not even able to test students more than a few times per semester unless the student requests a test with a credible case for being exposed to the virus.

For these reasons, many businesses and schools have resorted to more generic screening metrics to gauge a person's overall health. A multitude of businesses, such as doctors offices and gyms, have defaulted to measuring the temperature of individuals seeking entry. Body temperature is thought to be correlated with contraction of COVID-19 because fevers are one of the major symptoms of the virus. This requires the business to staff at least one employee at the entryway to manually screen temperatures. Not only is this a financial concern; this poses a

health risk for the person who must interact with potential entrants, getting close enough to take their temperatures. Even with non-contact devices, they would need to be relatively close to the entrant for an accurate scan. The Absolute Chicanery Capstone team saw the inefficiency of this system that had been adopted by many and wondered why an automated option hadn't filled this gap in the market.

After some research on prior art, it is clear that the systems currently automating temperature scans before entry are very expensive. In fact, Advanced Imaging estimates that the average temperature scanning kiosk costs between $2,000 to $7,000 [1]. A primary example of one of these expensive kiosks is Meridian's Personal Management Kiosk, which starts at $2,895. This kiosk, like most others of its kind, includes facial recognition technology so that the temperature readings can be associated with a person's identity [2]. While this level of identifiability may have its proper use cases, it is a very controversial technology. Many people are uncomfortable with the idea of this information being collected about them without consent.

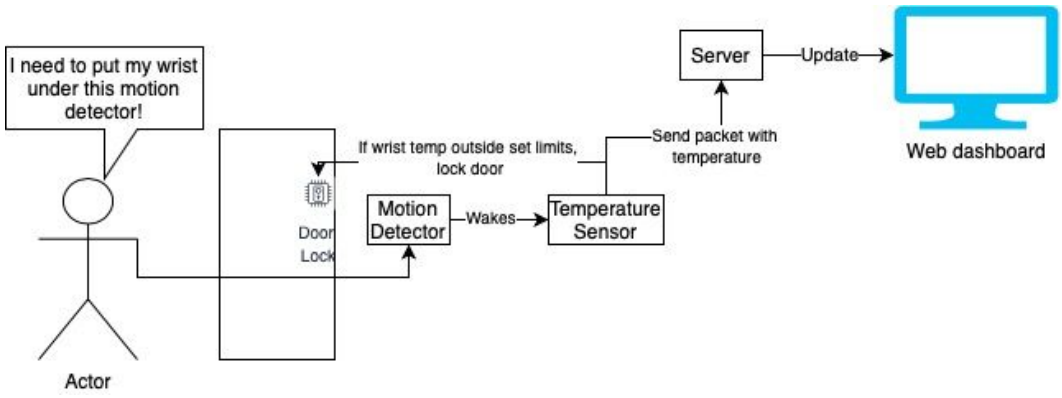


Figure 1. System Flow Diagram from User Perspective

To address this gap in the market, the team presents a system to automate the temperature screening process, preventing individuals with higher-than-average temperatures from entering a space without privacy concerns involving storing personal data. This system also promises to be cost-effective when produced at scale, in comparison with competitors on the market today.

Constraints

Design Constraints

This project was under a considerable time constraint, as it had to be completed in one semester. The team had under four months to complete the design, from start to finish, while also juggling other classes and responsibilities as full-time students.

The Capstone projects were required to include either a myRIO or MSP43X microcontroller, limiting the CPU capabilities for the system. Additionally, each project had to include a custom PCB design, created via Multisim and Ultiboard. The manufacturing

constraints on these boards included that they could be no more than 30 square inches in size, with a maximum of 50 drill holes per square inch and 10 millimeter drill hole size. The manufacturers were also unable to drill any slats or non-circular holes into the PCB, which affected which components were ultimately selected. Additionally, any lines or traces had to be 5 millimeters in thickness.

The team was also constrained by a lack of a crimper; WWW Technologies (a company with which the team worked to solder the smaller components onto the PCB) did not possess one, and the costs of purchasing one were far out of the budget.

There were constraints involved with utilizing a Raspberry Pi for the server, as well. The Raspberry Pi has 1 GB of RAM and a 32 GB disk, constraining how much data the server can store at any given time.

There were additional constraints associated with creating the housing for this system, as well. 3D print designing required precision down to the millimeter for component through-holes; our team was forced to opt out of printing screw mounts into the design since printing errors were too large for our precise dimensions. Additionally, closing the unit atop the shell required additional hours of printing time as well as support material so we had to add acrylic sheets on top to finish the assembly.

Economic and Cost Constraints

This Capstone project was capped to a maximum budget of $500. This was the primary cost constraint for the design. While this seems like an ample budget, it did not permit for purchasing a highly accurate, medical-grade non-contact temperature sensor. This caused issues with inaccurate temperature readings during testing.

If this product were taken into high volume production, many costs would be eliminated. Around $90 was used on testing and revisions. This included multiple board revisions and part revisions.

Another significant cost came from using development boards. For example, the MSP432 LaunchPad board was approximately $24, while the MCU chip itself would be less than $5 in volume. A cheaper chip could also have been used to further reduce costs. The Raspberry Pi 3B+ ($35) could similarly be removed from the system and replaced by a computer or server already present in the building.

External Standards

There are a few key external standards that this project takes into account. Firstly, safety standards for this project include FCC regulation of radio frequency devices. This project contains both intentional and unintentional radiators. The intentional radiators consist of the selected Bluetooth module and the Raspberry Pi. Both these have been approved by the FCC for use in short range communications contexts [3]. The rest of the circuitry and digital devices are categorized as unintentional radiators. We have aimed to keep these devices outside of the 30 MHz - 960 MHz frequency range so they are exempt under 47 CFR 15.101 (b) [4]. The MSP and various circuitry will stay below this range while the Raspberry Pi is clocked above it.

Fire safety is something that should be considered whenever implementing a locking mechanism on an outward facing door to a building. Safety is an important consideration when designing doors that are in route of egress. Every three years, the Virginia Uniform Statewide Building Code (VUSBC) adapts the International Building Code (IBC) updated by the International Building Code Council. Section 1010.0.9.3 of the IBC states that:

> "Where electrical systems that monitor or record egress activity are incorporated, the locking system shall comply with Section 1010.1.9.7, 1010.1.9.8, 1010.1.9.9, 1010.1.9.10 or 1010.1.9.11 or shall be readily openable from the egress side without the use of a key or special knowledge or effort." [5]

The VUSBC alters this requirement by stating the following [6]:

"Change Item 2 of Section 1010.1.9.3 of the IBC to read: 2. In buildings in occupancy Groups B, F, M and S, the main exterior door or doors are permitted to be equipped with key-operated locking devices from the egress side provided:

1. The locking device is readily distinguishable as locked.
2. A readily visible durable sign is posted on the egress side on or adjacent to the door stating: THIS DOOR TO REMAIN UNLOCKED WHEN THIS SPACE IS OCCUPIED. The sign shall be in letters one inch(25 mm) high on a contrasting background.
3. The use of the key-operated locking device is revokable by the building official for due cause."

To satisfy these conditions for our project, we procured a fail safe electric strike. The strike allows the door latch to be released if pulled from inside the building, but it controls the door latch if attempted to be pulled from the outside of the building. Since this is "readily openable" from the inside, or "egress" side, the door meets inspection [7]. To increase the fire safety of the system, the fail safe strike unlocks when there is a loss of power - allowing emergency personnel into the building. Also, the high side switch is configured to only lock the door when the microcontroller is supplying a signal- otherwise if the microcontroller is faulty, the system remains unlocked even if the building's power is operational [8].

Aside from safety standards, extra care must be taken when dealing with health data and medical records. While this device does not fall under the jurisdiction of the Health Insurance Portability and Accountability Act (HIPAA), the US Department of Health and Human Services (HHS) guidelines for HIPAA compliance were useful when evaluating the ethical consequences of device installation [9].

Additionally, standards exist for mechanical assembly of devices like this system. Particularly, under IPC standards [10], we classify our project as a Class 1 General Electronic Product where function is the primary goal. Further, we achieved a Level C design producibility where significant manual assembly is tolerated. This is an area that would require improvement if the device were to be sold commercially.

Finally, this project will rely on standardized patterns for both internal and external communication. Internally, communication between the MSP and the Raspberry Pi will use Bluetooth [11] technology due to its compatibility with the Raspberry Pi and it's low energy requirements. The Bluetooth specification is maintained by the Bluetooth Special Interest Group (SIG). Each door-mounted MSP device will use a Bluetooth component (controlled via UART [12]) to send data to the Raspberry Pi hub. Then, interaction with the Pi will be done over IEEE 802.11 WiFi [13] because of its wireless nature and use in the HTTP and SSH protocols that follow. Management and development for the Pi will be accomplished using SSH [14] for security and wireless access. Management and access to the temperature sensor will also rely on the I2C communication standard [12]. Externally, users will consume temperature data using the HTTP 1.1 protocol [15] for access from any available device and browser on the network. Use of these communication standards will ensure interoperability of hardware components (Bluetooth emitters, receivers) as well as software tools (SSH clients, browsers).

Tools Employed

The primary computer aided design for this project was done with the National Instruments suite of Multisim 14 [16] and Ultiboard [17]. These tools allowed rapid schematic creation and PCB footprint design. Additionally, the plastic housing to contain our device was designed with AutoDesk AutoCAD 2021 [18].

Assembly and testing took place in UVA's NI Lab which provided various additional resources. The Virtual Benches and accompanying digital multimeters and logic analyzers were critical for powering and testing our prototypes. The soldering equipment was important for final assembly of our PCB. Lastly, UVA's mechanical engineering department provided access to 3D printing capabilities that ultimately built the plastic housing for our device.

Software creation came with it's own set of tools. Throughout the project, software was tracked in a Git [19] repository for version control and easy code sharing among team members. The embedded programming made use of Texas Instruments' Code Composer Studio [20] development environment and accompanying compiler [21]. During debugging, PuTTY [22] was

used to read from the microcontroller's serial port. To access the Raspberry Pi during development, OpenSSH [14] provided remote terminal access.

The final software also relied on some important libraries. The embedded code made extensive use of the Texas Instruments DriverLib [23] suite and the FreeRTOS [24] real-time operating system. The Raspberry Pi relied heavily on the Raspberry Pi OS [25] operating system as well as the PySerial library [26], the SQLite database system [27], the Flask web framework [28], and the Apache2 web server [29].

Ethical, Social, and Economic Concerns

Environmental Impact

The system manages environmental impact by reducing overall energy usage with the incorporation of a motion detector. While the motion detector was not a necessary component to the system, it makes the system far more energy-efficient because it avoids the need for the temperature sensor to be constantly scanning for temperature readings. The motion detector will only wake the temperature sensor once motion is detected, reducing the temperature sensor's energy consumption. That being said, the system does consume some level of energy constantly, as the motion detector must be able to detect motion at all times.

Significant portions of the project can be disassembled and re-used in other contexts later on. Components like the Raspberry Pi, the MSP432 microcontroller, and the digital sensors can all be unplugged with relative ease for use in future Capstone projects or in other academic pursuits. Similarly, if one of these parts breaks, it can be removed from the system and replaced by an identical but functional part; that way, the entire system does not need to be thrown away. For that broken part, electronics recycling is freely available in the Charlottesville area through the McIntire Recycling Center, for example [30]. This is more complicated for components soldered to our PCB, but these can still be disassembled and re-used to a certain extent.

Sustainability

The automated health screening door lock device is a highly sustainable system as its individual components can easily be replaceable or updated while still maintaining a consistent and effective result. The motion detector allows the other components of the system to stay dormant until usage is required meaning overall longevity of the system is high. All the power in our project comes from electrical wall outlets, both on the microcontroller and the server side, so there is no reliance on disposable batteries.

In terms of the electrical components, the main concern is the constant power consumption of the motion detector. However, the passive infrared sensor is very inexpensive and requires low power so it won't wear out easily. The Bluetooth module also poses a sustainability question because of the range of communication. The HC-05 chip used for our

Bluetooth communication, though, exceeds expectations and allows for 3 Mbps of modulation with a complete 2.4GHz radio transceiver and baseband. Additionally, the module auto-reconnects within 30 minutes when disconnected from out of range connections.

Finally, the mechanical casing of the system is highly durable and can maintain drops due to the lightweight of the device. The main shell is formed from 3D printed thermo-plastic filament, which is sustainable but causes a waste issue due to its un-recyclability [31]. In addition, the clear 5 mm acrylic sheet enclosing the system is also durable but also imposes the same issue on un-recyclability similar to the thermo-plastic, which often goes straight to landfills [32]. Finally, initial prototypes failed to protect the internal components from rain and other weather conditions, which presents problems for system durability but could be further developed before production.

Health and Safety

The design for this system prioritized health and safety, largely by ensuring that the door locking mechanism follows fire and safety regulations. The lock we chose is specially shaped to allow the strike to leave if the change in the latch comes from the inside [7]. Therefore, regardless of power availability, a user can exit the building. The strike needs power supplied continuously to remain locked, and the high side switch adds redundancy in case the system fails but the building's power is still working. The high side switch controlling the lock also has a fault detection functionality that allows different errors to be detected, reported, and fixed in a timely manner [8]. While this default unlocked state would mean that people with fevers may be allowed to enter the space if the system is not functioning properly, this would just be similar to the system not having been present rather than posing a risk to those who could be trapped inside.

Another relevant safety consideration is the spread of contagious disease, which the system hopes to decrease. The system was designed to be hands-free and even make it unnecessary for users with fevers to touch the doorknob. When a fever is detected, the LEDs facing out of the housing indicate that the temperature was invalid and that the door remained locked. As such, potentially contagious users never need to make physical contact with the device or the door to prevent the spread of germs.

There is a health risk to collect an artificially high temperature reading, as it can cause anxiety for the user to see that their temperatures may be out of range when they are actually healthy. This is a risk that is worth the reward of a system that maintains the health of those inside, though, especially since the risk is mitigated when the user can wait a moment until they have cooled off (i.e. if they approached the system after jogging or partaking in an activity that spiked their body temperature) and then try again to enter. There is no limit to how many times a user may request entry to the space via the system.

Manufacturability

Manufacturability is an important consideration for any potential consumer product. During prototyping however, our emphasis was focused on completing the functionality of the device under the given time constraints which is why we aimed for an IPC Level C design producibility where manual assembly is permitted. That being said, many of the PCB components could be adjusted for more manufacturing-friendly surface mounts and configurations. Additionally the MSP432 Launchpad could be swapped out for the surface-mount package, or even a less powerful microcontroller. The external casing would be quite amenable to manufacturing at high volume by switching from 3D printed plastic to injection molding or other techniques. The limiting factor for manufacturing efficiency would likely be the manual placement of the sensors and the installation of the wiring between them.

Ethical Issues

Unlike some of its competitors mentioned previously, this system does not utilize facial recognition technology, nor does it collect any personal information on the individuals associated with each temperature scan. As such, the primary ethical concerns in storing user data are negligible. There do exist corner cases for identifiability, however. For example, if timestamps shown on the system's dashboard could be correlated with entrance times observed by a bystander or video camera. These considerations may be mitigated in the future by restricting dashboard access or implementing an encryption scheme.

Additionally, it is possible that the placement of the wrist scanner disadvantages certain people in the population. For example, a person without arms or a toddler that cannot reach the temperature scanner may have trouble gaining entrance to the space.

Another potential area for concern is how skin pigmentation affects temperature measurement reliability in our device. Producing a system which disproportionately allows members of certain ethnicities to enter a building over other ethnicities would present serious ethical barriers to the success of the project.

Intellectual Property Issues

Patent US10692596B2 introduces a health kiosk that includes a "physiological measurement apparatus connected to the computing unit". This kiosk includes a weight scale, blood pressure monitor, and a heart rate monitor. That being said, some of them allow for the addition of blood pressure cuffs, EKGs, ECGs, blood glucose measurers, and thermometers. A web server stores the measurements taken by the kiosk, then can transmit them on the network as a URL or HTTP cookie of an HTTP request. The kiosk includes a screen for more user

interaction/communication, such as an LCD that is a touch screen. It also may include keyboard, mouse, microphone, or other external tools that the user may utilize to communicate with the kiosk. Some versions may include technology for automatic user identification, such as a camera for facial recognition or a fingerprint scanner [33].

Patent US20180158555A1 is a patent for another medical kiosk, though this one focuses on tele-medicine and check-in for medical appointments or prescription pick-up. The kiosk includes video conferencing capabilities for tele-medicine appointments, and the kiosk includes user input that "can include one or more components selected from the group consisting of a key pad for identification and/or data entry, … motion sensor, sound sensor temperature sensor, …" [34].

Patent US9256719B2 is a patent for a smart mirror that includes an infrared camera to record body temperature, among other sensors/devices collecting biometric data. This system also includes facial recognition software [35].

Since all of these patents do not include a door locking mechanism controlled by the temperature readings, Absolute Chicanery's Capstone deliverable is patentable. Additionally, the lack of any cameras or other ways of gathering user identities makes the system stand out from existing patents.

Detailed Technical Description of Project

Overall System

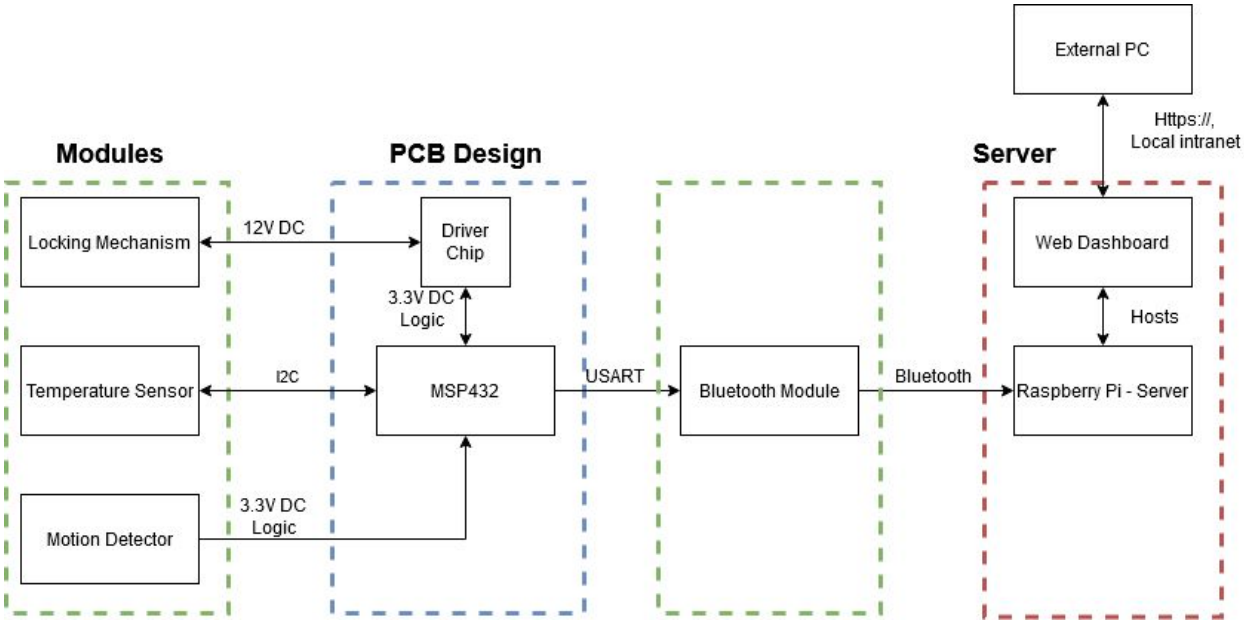


Figure 2. High level system architecture.

There are 7 main areas of the project: the power supply, the door lock, motion detector, temperature sensor, Bluetooth module, embedded system, and server/web dashboard. Figure 2 shows how the different systems interact as well as the expected communication signals used on each connection. The different modules in green interact with the MSP432 microcontroller in blue and the server architecture in red. These colors provide conceptual distinction among subsystems. The MSP432 acts as a central hub that controls the logic and communication of all modules. The MSP432 sends data to the Bluetooth module in order to communicate with the raspberry pi server and display the dashboard.

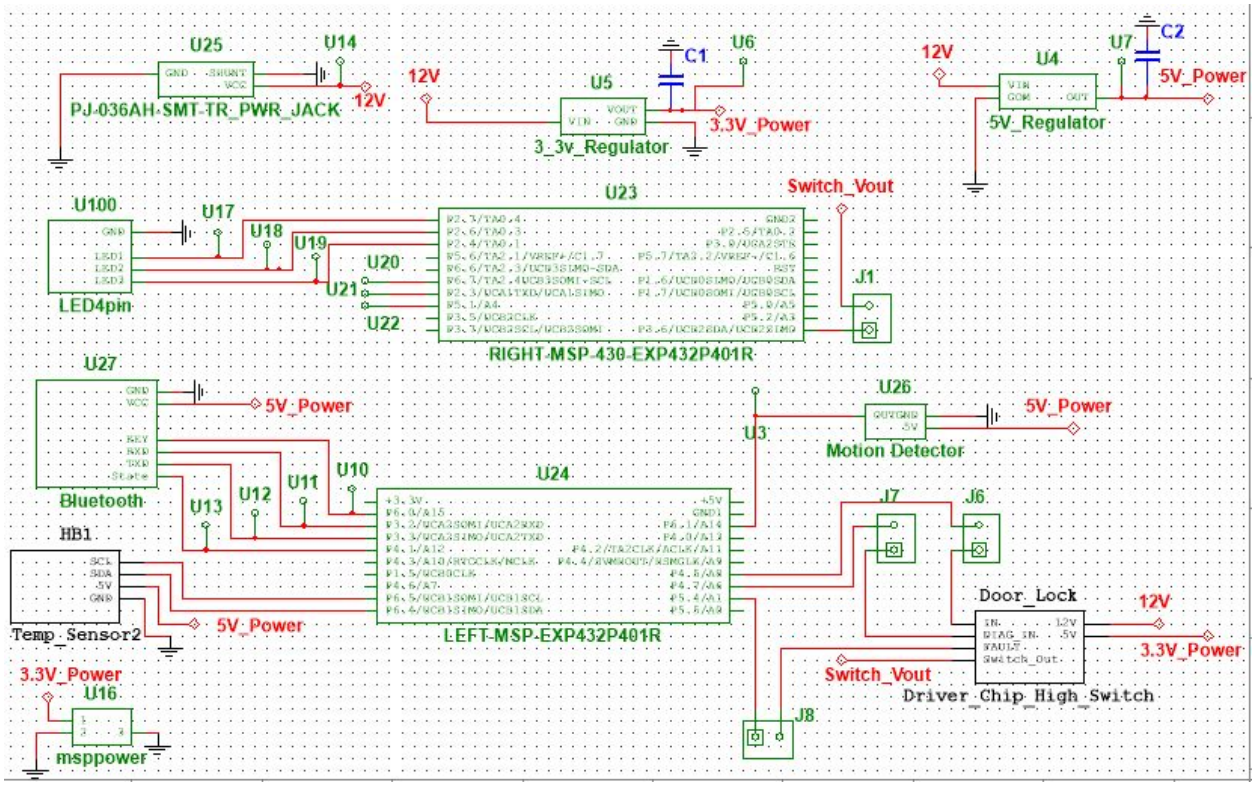


Figure 3. Top level schematic diagram.

Figure 3, provides a more detailed schematic description of how the subsystems managed by the microcontroller interact with one another, including test points and MSP432 pin assignments. The power supply subsystem runs along the top of the schematic while the connections to the MSP432 run around the outside. The LED board is connected to the right hand side of the MSP432, while the other modules connect to the left hand side, for organization. The Bluetooth module and motion detector connections are represented by MOLEX on-board connectors, as well as the lock and temperature sensors once inside their respective hierarchical blocks. The two hierarchical blocks are examined more closely in their sections, but the main connections are shown in Figure 3.

Power Supply

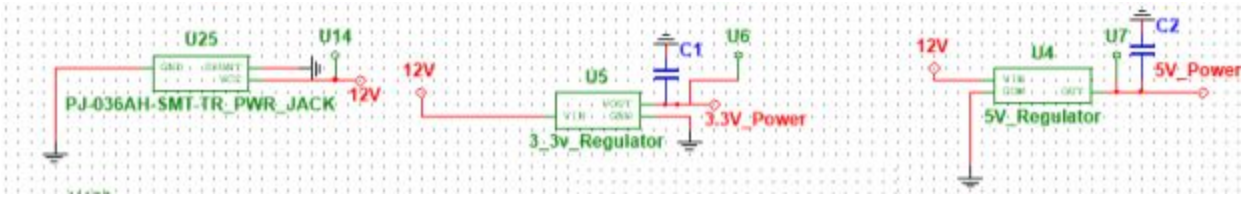


Figure 4. Schematics for power supply subsystem.

Different subsystems in this project have different power needs, adding complexity to how we must power the device. The lock strike itself requires 12 V when locked, which led us down the path of powering the device from the wall outlet rather than using a battery. A 12 V power adapter for a laptop computer was used to plug into the wall and convert to DC. This adapter was plugged into our board via a surface-mount power jack. Two linear voltage regulators were used to step the 12 V supply down to both 5 V and 3.3 V for different components in the system. The Bluetooth module, temperature sensor, and motion detector were all powered from the 5 V supply whereas the microcontroller itself used the 3.3 V supply.

Current requirements also play a role in the power supply design. The Bluetooth module has a maximum current requirement of 30 mA and the temperature sensor has a maximum current draw of 16 mA [36]. The motion detector has the least amount of current draw with a 3 mA maximum [37]. The microcontroller was allocated 75 mA based on the recommendations form its data sheet, and finally the door lock and driver chip use a combined 285 mA of current at maximum [38]. Therefore, the regulators need to be able to handle these maximum current draws. The 5 V regulator is rated for a typical load of 150 mA whereas our system only needs 68 mA of current, total [39]. The 3.3 V regulator can handle 250 mA where the system only needs 91 mA [40]. The off- board LEDs were powered directly by the microcontroller's digital pins since their current requirements of 2 mA were low enough [41]. In total, the system needs about 410 mA of current to operate effectively. The DC power connector we used can handle 5 A, so it is more than sufficient [42].

Door Lock

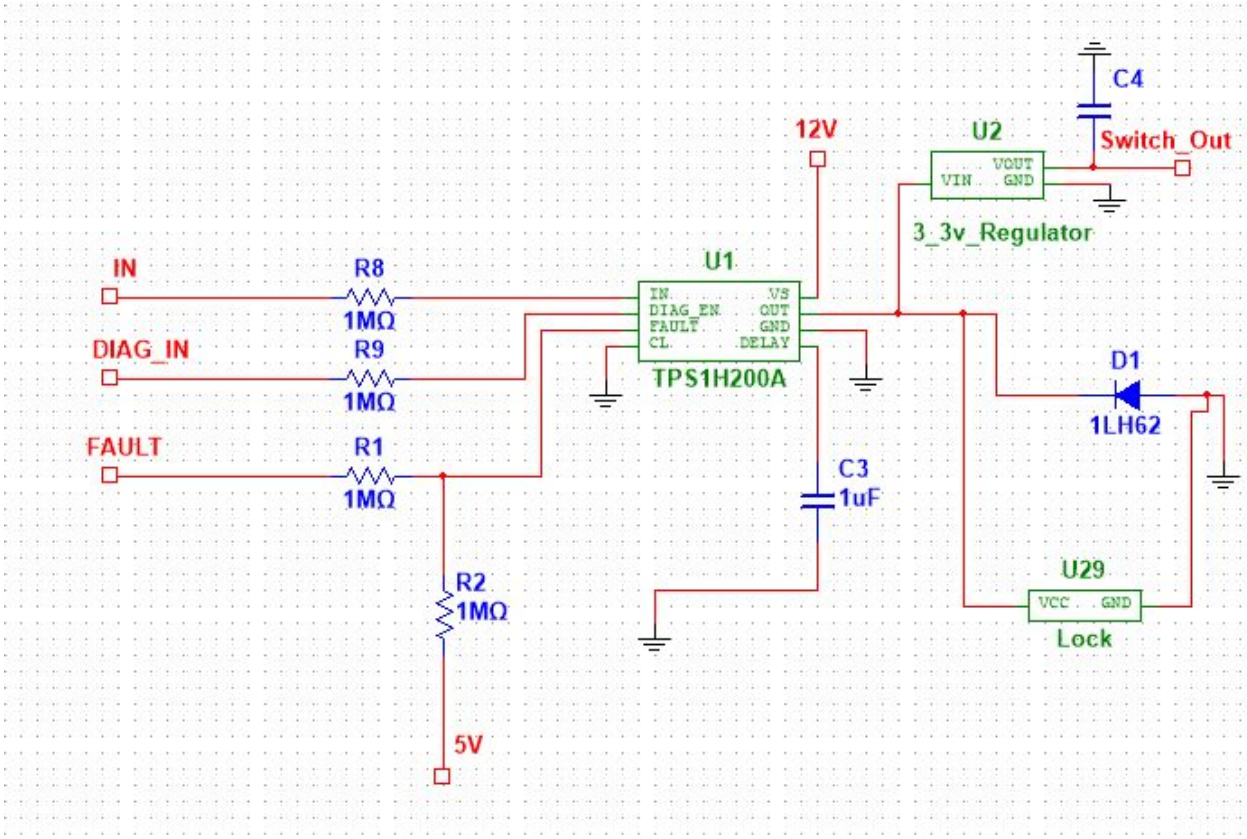


Figure 5. Switch and door locking subsystem schematic.

The locking mechanism for the system is a fail safe lock designed for general use. It is controlled by a chip on the custom manufactured PCB that intercepts the 12 V power supply and connects through the 12 V VCC connection to the strike. This can be seen in the schematic in Figure 5 and the data flow diagram in Figure 6**.** The chip is controlled by the MSP432 through the IN and DIAG EN 3.3 V digital logic controls. When IN is high, current from the 12 V supply is connected to the strike, locking the door. When IN is low, the 12 V power supply is disconnected from the strike, and the strike unlocks. The DIAG EN control enables fault protection on the chip and diagnostic information to be relayed to the microcontroller through the FAULT pin, which can pinpoint a fault according to a truth table.

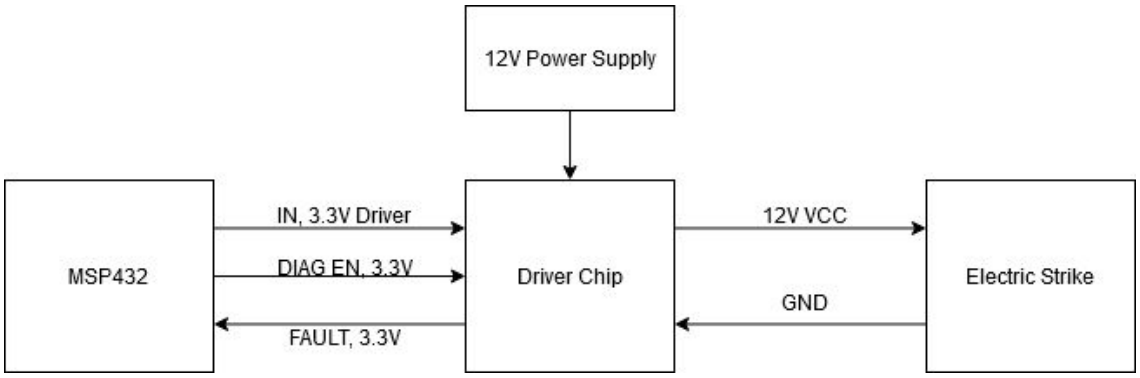


Figure 6. Component interaction in the door lock subsystem

Motion Detector

The motion sensor removes the need for our temperature sensor to continually collect temperature readings, which would require an unnecessary amount of power. The motion sensor also ensures that the system only collects temperatures to send to the web dashboard and utilizes it for deciding whether or not to lock the door *when someone has approached*. It would be problematic for both power consumption and system usability if the device constantly sent ambient temperature readings over Bluetooth to the server.

We decided to utilize a Passive Infrared (PIR) motion sensor, as it will allow us to constantly sense motion in a low-power manner. PIRs are composed of pyroelectric sensors that detect movement via elevated levels of infrared radiation. The sensor is split into two halves (technically, it is composed of two individual sensors). If both sides detect the same amount of infrared, it is considered idle. Then, when anything warm (i.e. a human wrist) passes the sensor, it will detect the movement as an imbalance in infrared readings between the two halves.

As for the components that make up this device, there is a JFET transistor at the core, which has low noise and built-in buffering for high impedance. Atop this JFET is a lens, a very low-cost plastic dome that drastically increases the detection area. Using optics concepts, this lens condenses a large field of view into a small one, which is ideal for the tiny sensor that takes the infrared readings. The lens is split up into sections, each of which is a Fresnel lens (a type of lens that is made up of concentric rings to concentrate into a thin beam).

The PIR has 3 digital pins: power, ground, signal. The PIR has digital output (high or low voltage readings; high when motion is detected). The power is 3-5 V DC, though 5 V is recommended and what we used in the system. When the PIR detects motion, the signal pin goes to “high”, outputting 3.3 V. The PIR used in this project is from Adafruit and has a trimpot that allows users to easily adjust the sensitivity. The sensitivity range for this device is up to 6 meters (20 feet), with a 110-70 degree range of detection. This will adequately cover anyone who is

coming up to the door to scan their wrist temperature. There is also a trimpot to adjust how long the output stays high after a motion detection occurs (10-20 second range).

In our project, the signal pin is repeatedly polled when the device is idle. When the signal pin goes high and motion is detected, processing flow is yielded to the temperature sensor to take and compute temperature measurements. This represents the rising edge of the signal pin but we also take advantage of the falling edge as it transitions back to ground. This is the point where the system is ready for another scan so the door is re-locked and the readiness is indicated on the LEDs.

Temperature Sensor

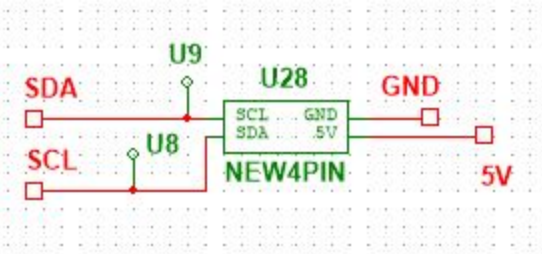


Figure 7. Temperature Sensor Circuit Schematic Pins

Our device incorporated the MLX90614 infrared temperature sensor manufactured by Melexis. Specifically, the temperature sensor was purchased through a third-party vendor which came as a GY-906 module chip. This contactless temperature sensor module uses infrared rays to measure the human body temperature by converting the signal from the sensor to a digital value and communicating to the microcontroller using an I2C protocol. Specifically, the contactless temperature measures the intensity of the emitted infrared energy radiated from a living being, which is directly proportional to the object's temperature as stated in the Stefan-Boltzmann law. The overall device consists of two chips integrated into a single sensor – one sensing the IR temperature and the processing unit converts that signal to a digital value.

There are three main functions that allow the temperature sensor to operate. First is signal processing. The device is controlled by an internal state machine, which controls the measurements and calculations of the object and ambient temperatures. Second is amplification. A low noise low offset amplifier with programmable gain is implemented for amplification of the IR sensor voltage. Third is the SMBus 2 Wire Protocol. The module comes with digital SCL and SDA pins for use with SMBus. These pins, shown in Figure 7 above, are decoded by an integrated circuit on the chip and allow the microcontroller to alter the state of the sensor and request temperature readings.

This device can output a temperature using either PWM output or SMBus (a protocol that extends I2C). Our system uses the temperature sensor as the slave in a master-slave I2C configuration, while the MSP432 microcontroller acts as the controlling device. When a read command is sent over the shared bus, the temperature sensor responds with a 15 bit temperature value that can be converted to a Kelvin temperature using division in the microcontroller logic. Figure 8 illustrates the I2C communication that takes place for a read command as observed with

a VirtualBench Logic Analyzer. The red labels indicate information written to the bus by the microcontroller whereas the yellow labels indicate data written by the temperature sensor. Table 1. Example calculation of Fahrenheit temperature based on sensor response. below illustrates the steps taken to decode this sensor response to a Fahrenheit temperature.

Figure 8. Screen capture of I2C communication with temperature sensor.

High Data Byte	39h = 57
Low Data Byte	F5h = 245
Full Temperature Payload	39F5h = 14,837
Decoded Temperature, Kelvin	14837 / 50 K = 296.74 K
Decoded Temperature, Fahrenheit	75.46 F

Table 1. Example calculation of Fahrenheit temperature based on sensor response.

There are many key specifications that make this temperature sensor unique. The ambient temperature range for detection is between -40℃ to 125℃ whereas the object temperature range is -70℃ to 382.2℃. The field of view of the sensor lens sits at 90 degrees and the measurement resolution to determine accuracy is at 0.02℃. In terms of power, the operating voltage is at 53.3 V after using a voltage regulator from the 12 V AC adapter input, and the supply current is at 1.5 mA. Finally, it is important to note that the Melexis datasheet indicates that the ideal distance between sensor and object is between two and five centimeters.

As will be discussed later on, we had issues with the accuracy of the sensor likely relating to the large field of view. While ambient room temperatures were always measured accurately,

the temperature varied drastically based on how far away the wrist was from the sensor. A measurement taken with a wrist 1 cm from the sensor could be significantly different from a measurement taken with a wrist 5 cm away.

Bluetooth Module

Our selected Bluetooth module exposes a UART interface which can send bytes one-by-one to paired devices. Once a 15-bit Fahrenheit temperature has been decoded from the temperature sensor's encoding, a two-byte Bluetooth packet is constructed with a re-encoded lower resolution temperature. The first byte contains metadata and error correction information and the second byte contains the temperature payload. Temperatures are encoded as a 7-bit unsigned integer representing the number of tenths of a degree over 95 ℉. Thus, in 7 bits we can encode temperatures from 95.0 - 107.6 degrees Fahrenheit in degradations of 0.1. Table 2, below, specifies the formal packet structure.

Byte 1: Header								
Bit #	7	6	5	4	3	2	1	0
Use	0	Lock Fault		Allow	-	Checksum		
Byte 2: Data								
Bit #	7	6	5	4	3	2	1	0
Use	1	Encoded Temperature						

Table 2. Bluetooth Packet Structure.

The most significant bit of each byte is used as a sequence number, differentiating header bytes (0) from data bytes (1). The "Lock Fault" bits store an encoding of the state of the lock. There are three potential failure cases that the driver chip can output, or these could be zero to indicate no errors. The "Allow" bit is set to 1 if the door was unlocked as a result of this reading and set to 0 if it remained locked. Bit 3 of the header byte is unused for now and always set to 0. The "Checksum" bits store the number of bits set to 1 in the entire packet, modulo 8. This can be used to detect if there were errors in transmission, acting as an additional layer of reliability on top of the L2CAP Bluetooth protocol. The "Encoded Temperature" bits store the unsigned integer described above. To reconstruct the temperature, the following formula can be applied:

$$Temperature\ (F) = 95 + 0.1 * Encoded$$

These Bluetooth packets are then sent to the server's Bluetooth receiver, where they are persisted for future use.

Embedded Programming

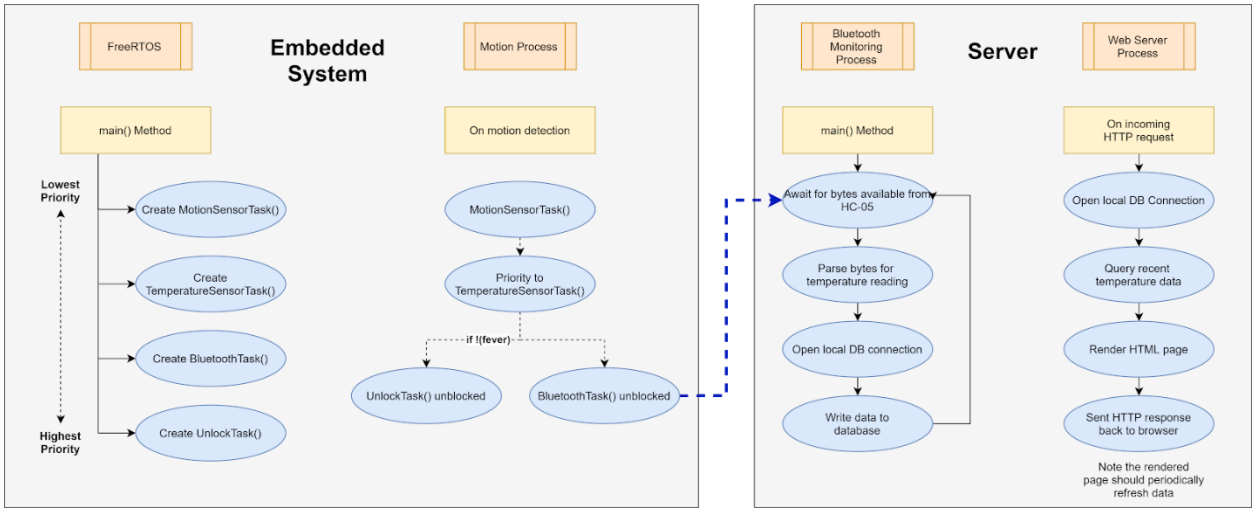


Figure 9. Embedded System diagram

The MSP432 uses a real-time operating system to manage the timing between different subsystems. The FreeRTOS kernel was chosen for its ease of use and reliability, as well as being distributed under the MIT open source license [24]. The FreeRTOS kernel was chosen over the TI-RTOS for its portability and level of documentation, allowing for other microcontrollers to be used should this project continue.

The FreeRTOS kernel uses the term "task" to define any code to be scheduled. The architecture of this project was to assign each subsystem a task. The following tasks were created: Bluetooth task, Lock task, Temperature task, and Motion task. FreeRTOS uses a priority based preemptive scheduling technique. In this project, semaphores were used to control flow from task to task.

Generally, the flow from task to task is as follows: motion, temperature, lock, then Bluetooth. The motion sensor task runs periodically until motion is detected. Once motion is detected, a semaphore is released to allow the temperature task to run. The temperature task then collects 128 temperature samples. The MSP432 uses its I2C module to send a request to access the temperature sensor's RAM, which contains an encoded Kelvin object temperature. After successfully reading a temperature, the temperature task releases semaphores for both the lock task in and the Bluetooth task, in that order. The lock task determines whether or not to toggle the locking mechanism. The Bluetooth task assembles a 2-byte packet and sends it wirelessly to the Raspberry Pi. Each of these tasks takes their semaphore to indicate that they have handled the current door-entrance attempt. Once all tasks have completed, the RTOS returns to execution of

the lowest priority task, motion detection. From here, the system is ready for a new temperature scan.

Another strategy employed in the design of the embedded system was to begin with "driver" code. Each subsystem would have a driver source file associated with it for use as a small test program for the subsystem's functionality. For example, "TemperatureSensorDriver.c" was the driver for the temperature sensor subsystem and repeatedly took temperature measurements. This file could be compiled without the use of the RTOS, which was useful for isolated testing. The strategy was to design driver files and test each subsystem before bringing each of the driver functions together in the RTOS code. This also allowed isolated testing of the RTOS as well.

Server & Web Dashboard

The Raspberry Pi server had two primary functions: to receive temperature data from the microcontroller and then to evidence that data in a usable dashboard. Figure 10 summarizes the high-level architecture of the server. The Bluetooth monitoring process and the web server process were both implemented by the team whereas a database library (SQLite) was used as-is in the middle. This was an important design decision because it allowed the Bluetooth code and the web dashboard code to be developed independently and tested separately with the use of a dummy database file.

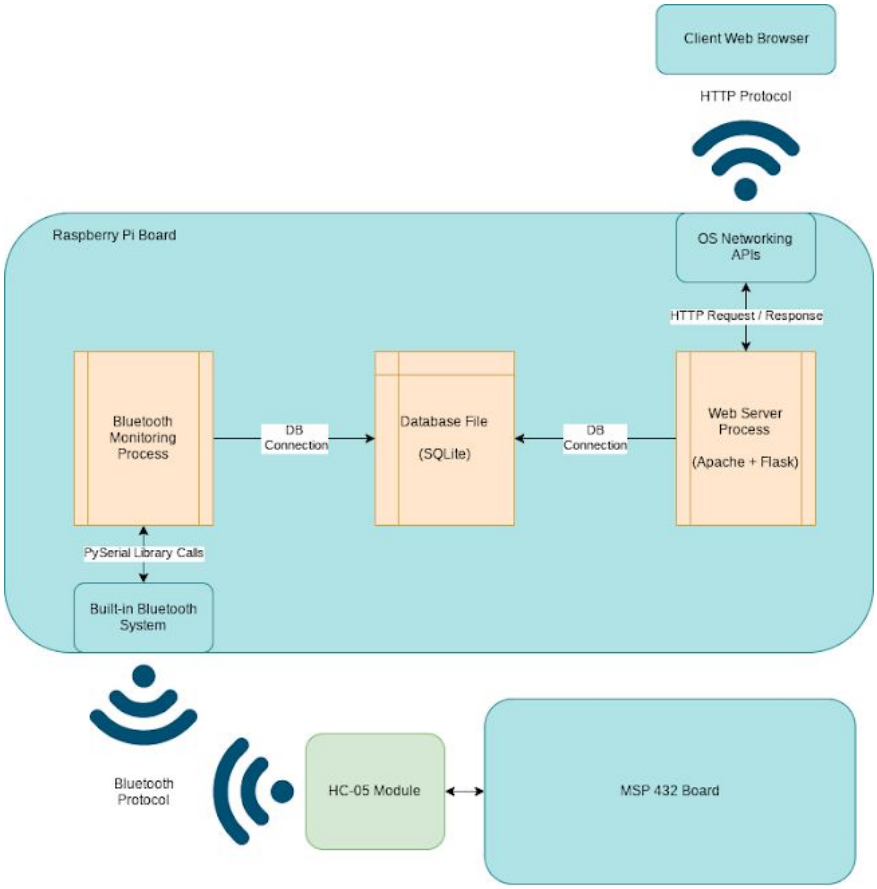


Figure 10. Raspberry Pi Architecture

To get the Bluetooth process working, it first had to be manually paired with the HC-05 on the microcontroller. Using the MAC address of the Bluetooth module, the Raspberry Pi paired and connected with the device. Then, once paired the device was available in the */dev/rfcomm0* file on the server's filesystem. Once paired and available, our code could automatically listen for packets and decode them using a finite state machine. When a complete transmission occurred as defined in our packet structure, the resulting information was written to the database.

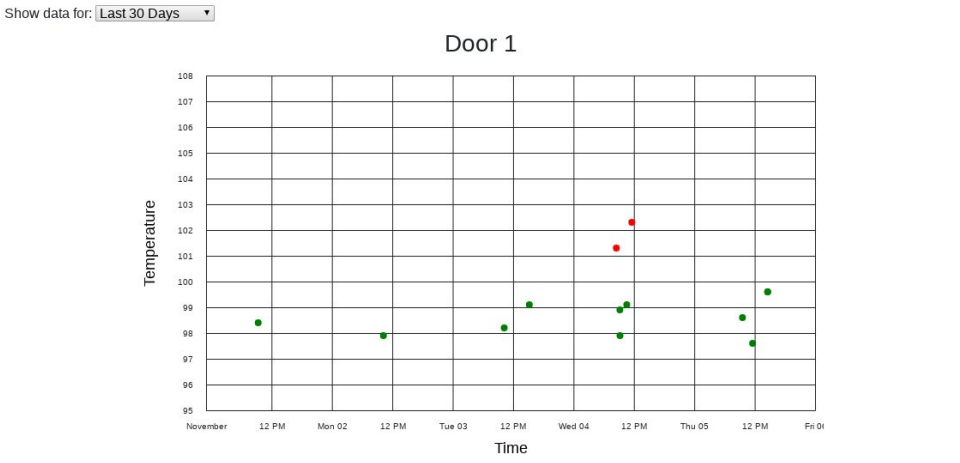
The database had the table schema shown in Table 3. For each temperature reading received, a row was written to this database table with all fields populated.

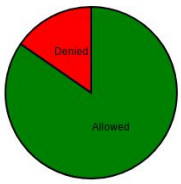
Attribute	Type	Description
door_id	string	A human-readable indicator of which door the message came from. This is included for easy extension of the project to multiple doors reporting to the same server, but was only ever set to “Door 0” in our initial implementation.
fault	integer	A representation of the fault signal that the locking mechanism’s driver chip reports. By looking at this integer, one can discern between potential errors like “no load connected” or “insufficient power”.
temperature	float	The actual temperature scan reported by the microcontroller. A fahrenheit measurement between 95 and 107.6.
decision	string	Restricted to the strings “ALLOW_THROUGH” and “LOCK_OUT”, an indication of what the microcontroller decided to do for this temperature scan. This is useful in case the temperature cutoff becomes configurable in the future.
timestamp	timestamp	The time that this scan occurred.

Table 3. Server database schema.

Separate from this data reception step was the web server and dashboard rendering process. A small website was written using the Flask framework to show the dashboard on incoming HTTP requests. Figure 11 shows a portion of the final dashboard configured to show the last 30 days worth of data. Here, the temperature scans are plotted for the user to see and then some sample statistics are calculated. Following these statistics is a table enumerating all of the scans (though the image is cut off to just show the first one). This dashboard refreshes its data automatically every 2 seconds to provide immediate feedback when new scans are made. The timespan to view data for is configurable on the dashboard, but restricted to the choices “Last Hour”, “Last Six Hours”, “Last Day”, “Last Week”, “Last two Weeks”, and “Last 30 Days” for security purposes.

This dashboard is available via the Raspberry Pi’s IP address to all devices on the local area network. This was accomplished using the Apache2 web server library. Apache2 was configured to point all incoming network traffic on port 80 (the HTTP port) to forward to the Flask application. Using Apache2 also provided benefits like reliability and constant availability whenever the Pi was powered on.





Statistic	Allowed	Denied	All
Count	11	2	13
Average	98.63	101.80	99.12
Max	99.6	102.3	102.3
Stdev	0.69	0.71	1.36

Decision	Fault	Temperature	Timestamp
ALLOW_THROUGH	0	99.6	2020-11-05 14:34:52

Figure 11. Final Web Dashboard.

System Assembly

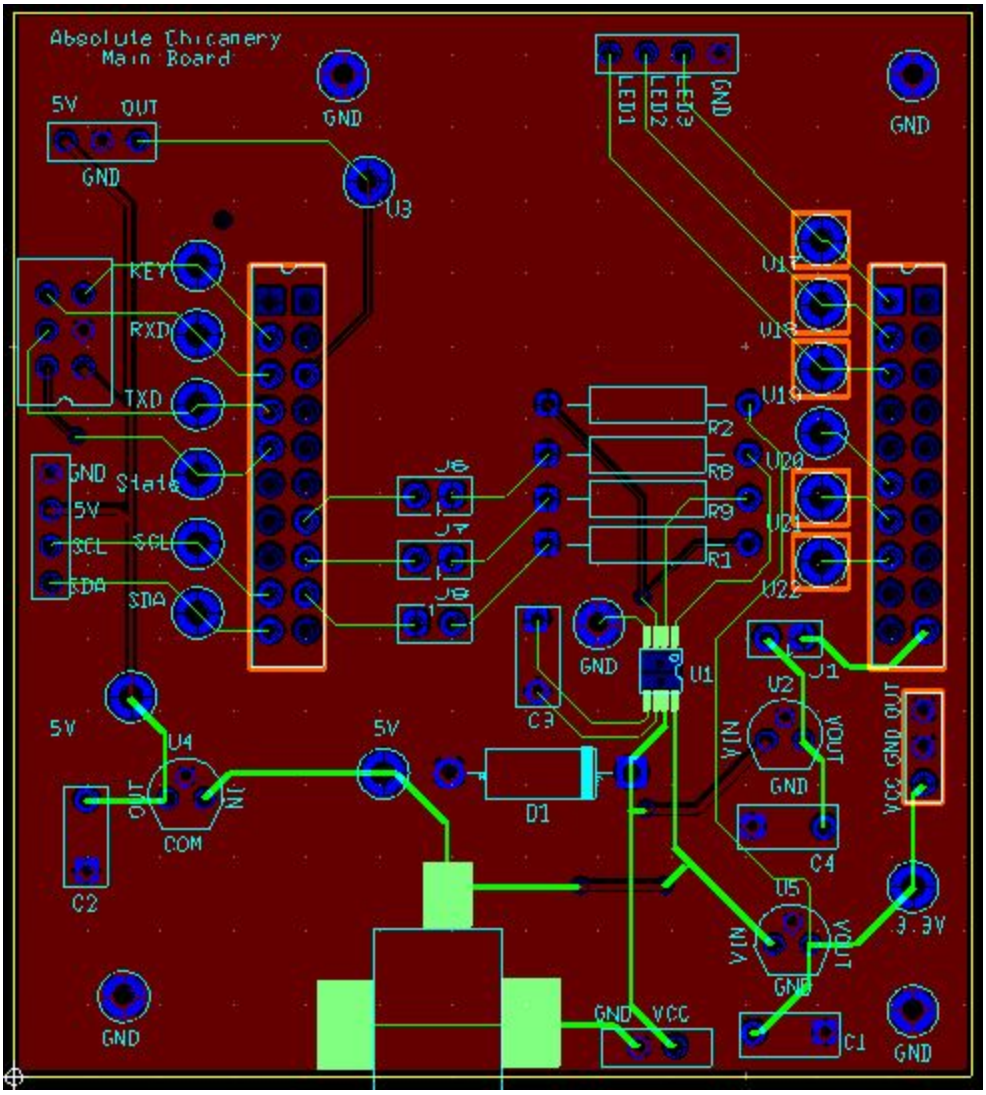


Figure 12. Final PCB layout for MSP432 header board.

Pictured in Figure 12 is the final main PCB revision used for the project. The board went through 3 major revisions, two of which were fabricated and tested. Each component on the board, barring standard components, was created in Ultiboard using the footprint creation wizard. The board is organized by power supply requirements. The left side of the board is for components that require the 5 V power supply. The components consist of the MOLEX connections to the temperature sensor, bluetooth module, and motion detector. The right hand side of the board is for the components that require 3.3 V and 12 V. The components that requires 3.3 V is the microcontroller, while the components that require 1 2V are the driver chip

high side switch, MOLEX electric strike connection, and the two 3.3 V regulators used for the microcontroller and to step down the VOUT of the switch to implement fault logic.

The top of the board has a MOLEX connector that leads to the LED sideboard, shown in Figure 13. This board is separate from the main board in order to allow for flexibility in the location of the LEDs when placed in the housing. These LEDs convey to the user if they should wait, enter, or were denied. It consists of low power LEDs and drain resistors to set the current flowing through them.

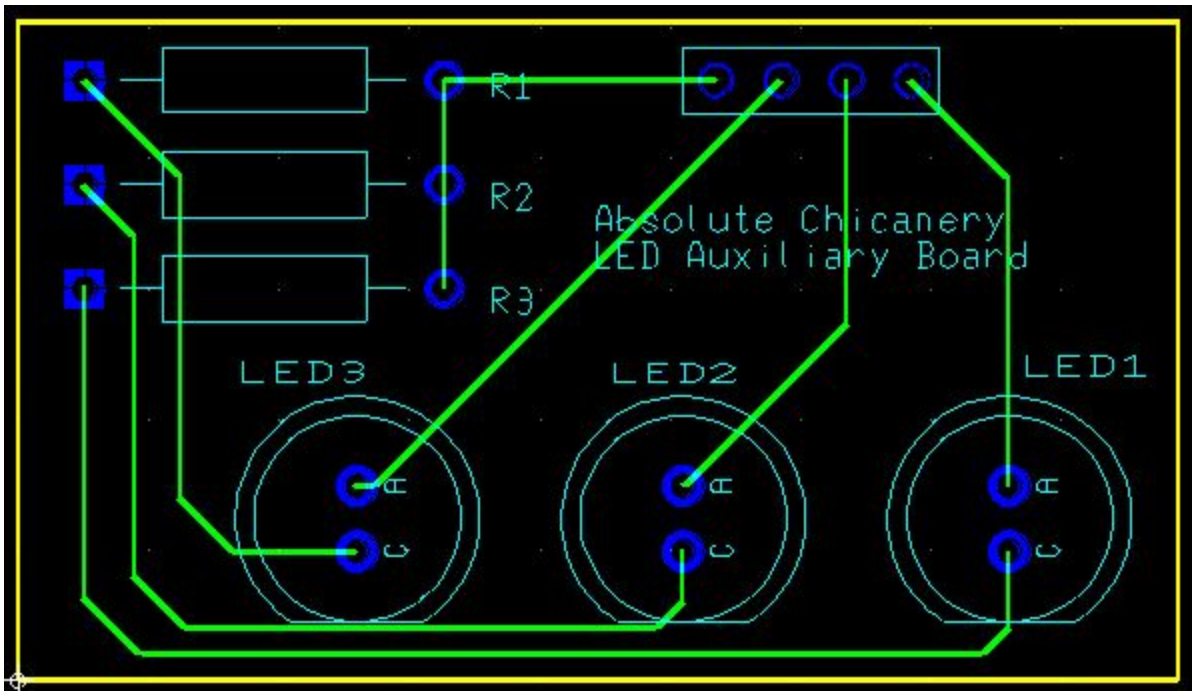


Figure 13. PCB layout for LED sideboard.

The design for the housing unit is shown from an aerial view in Figure 14. From left to right, the design features 3 holes for the power supply, temperature sensor, and motion detector. The LED panel faces outward through three holes cut post fabrication on the right hand panel. The unit is covered with a piece of clear acrylic to aid in debugging the system after it has been assembled. The microcontroller is mounted on the back panel of the housing and the Bluetooth module is to be attached on either side of the vertical piece of the system. We chose an “L” shaped design to clarify wrist placement and keep ambient temperature constant across the temperature sensor by blocking sunlight. This housing is intended to mount on a door frame at slightly above waist level, though this was never fully achieved during testing due to concerns about needing to further test the device outside of the lab. The fully assembled housing used during testing is shown in Figure 15.

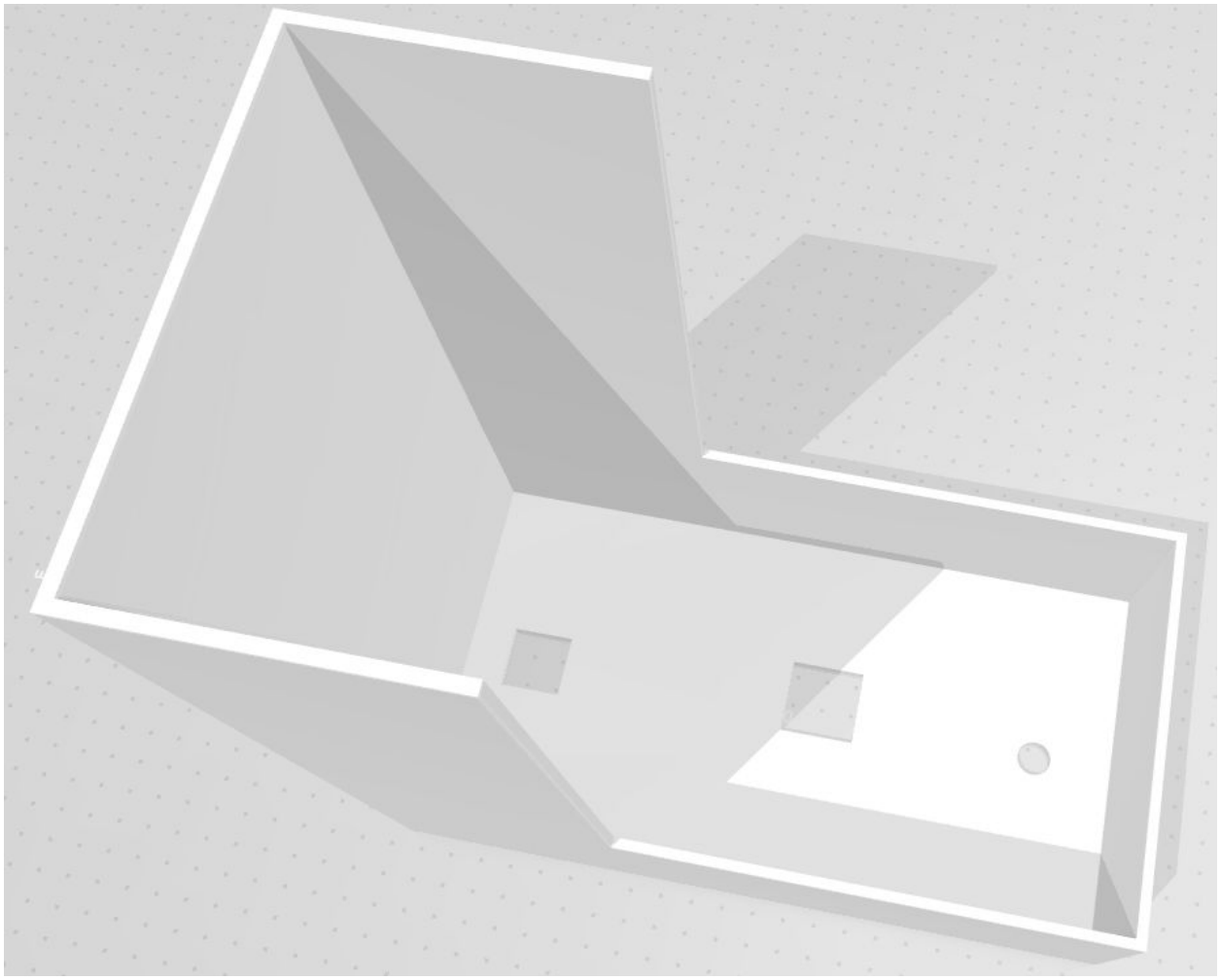


Figure 14. CAD model of mechanical housing.

Figure 15. Assembled system prototype, from above (left) and below (right).

Project Time Line

Figure 16 and Figure 17 show the team's proposed timeline for execution and what ended up actually happening, respectively. At the beginning of the semester, greater parallelism was achieved as part determination, schematic and footprint design, and embedded coding could all happen independently of one another for each subsystem. As the semester went on, the more complex components like the temperature sensor and Bluetooth module took more time to get working independently when compared to the motion detector and lock. While we initially aimed to have the entire system prototyped on the breadboard by October 15, this did not end up happening until the first week of November. Final PCB design and mechanical assembly thus were pushed back to the last week before Thanksgiving as we had to wait for some last-minute parts to arrive. Thankfully, due to thorough prototyping on the breadboard the system worked as anticipated when fully installed with little need for adjustment.

Figure 16. Proposed Project GANTT Chart

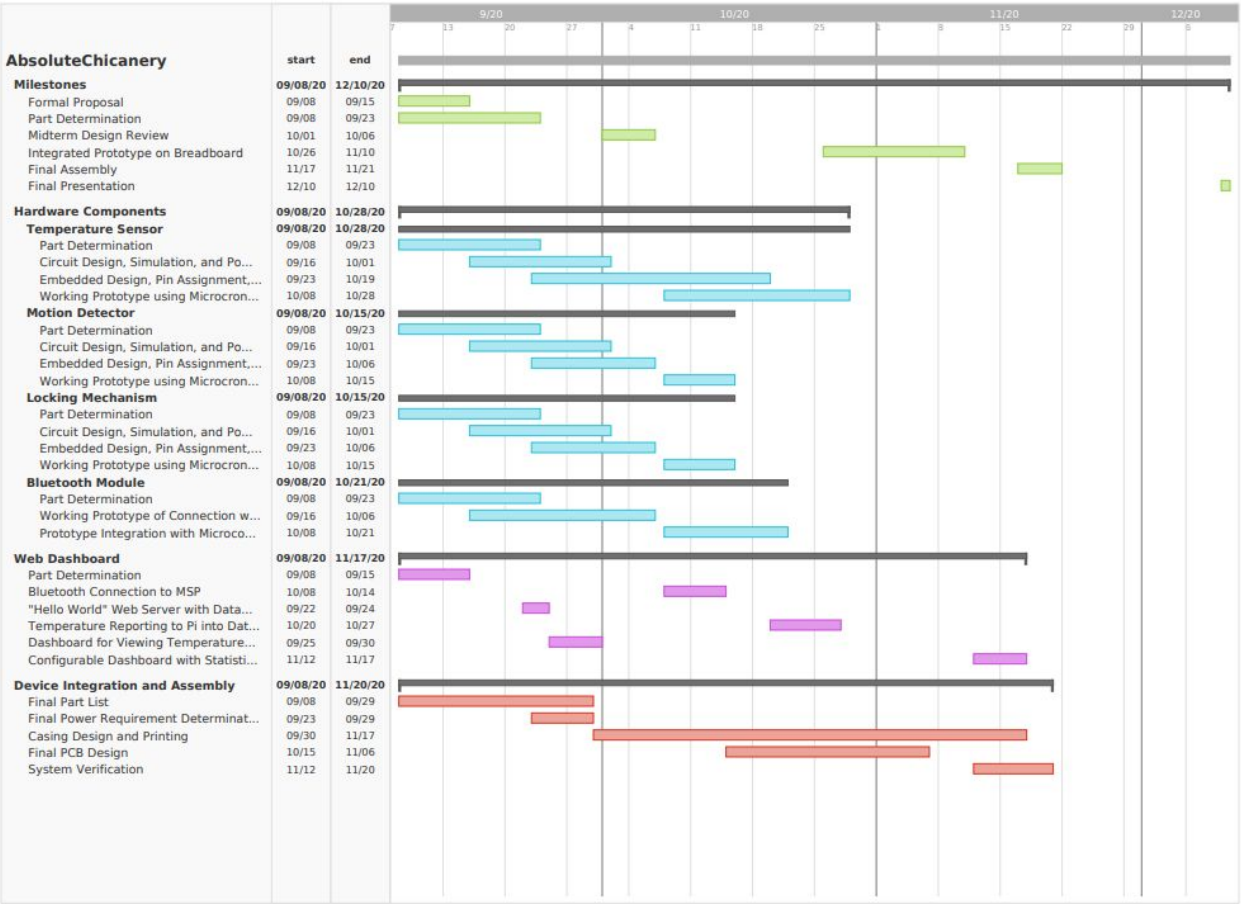


Figure 17. Final Project GANTT Chart

Test Plan

To minimize the need for adjustment when integrating the components together, a separate test plan was created for each subsystem in the project. This way, subsystems could be verified independently and any issues occurring after assembly must be due to the connections and interactions with one another.

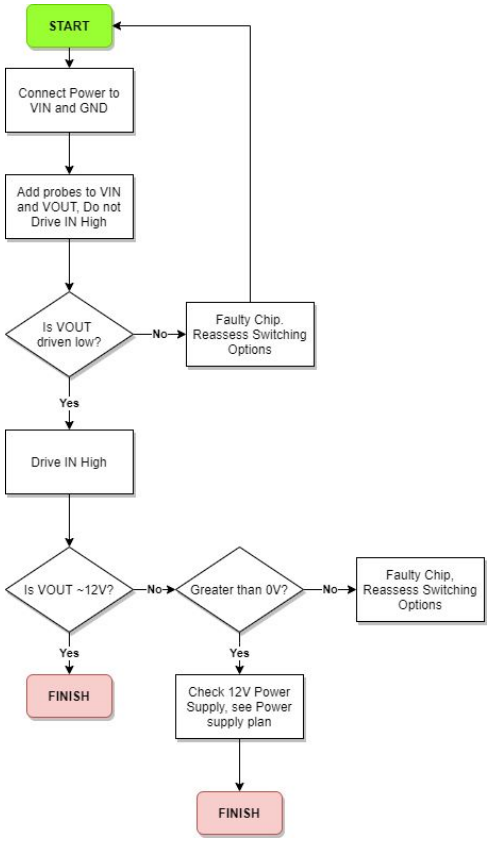


Figure 18. Driver chip test plan.

The test plan for the driver chip is shown in Figure 18. The chip has 3 inputs: the VCC to be switched on/off, the in control, and a diagnostic enable that allows fault protection to be enabled and diagnosed. The outputs to check functionality are OUT and FAULT. First, the chip is wired with 12 V in VIN and all other pins driven low or grounded. VOUT should be low at this stage of testing. If VOUT is low, then IN should be driven high. When IN is driven high, VOUT should switch to 12 V. If the above conditions are met, then the chip is operating adequately, and testing can continue.

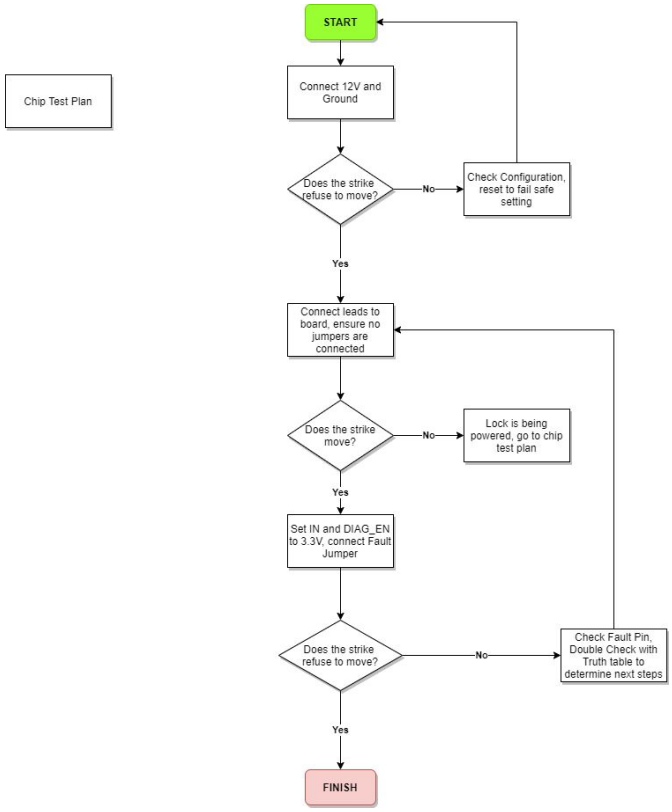


Figure 19. Locking mechanism test plan

The electronic strike only has two inputs, VIN and GND. The latch on the strike can be considered an output. Figure 19 begins by connecting VIN to a power supply and GND to the GND of the power supply. Since the strike is classified as fail safe, it should be locked when voltage is applied. Therefore, the lock should not move when 12 V is applied to VIN. Next, power should be removed from VIN and the strike should be wired to the board to ensure the traces from the driver chip are behaving properly. No power should be applied, and the strike should move. Lastly, the driver chip should be triggered to activate the strike, and it should be locked.

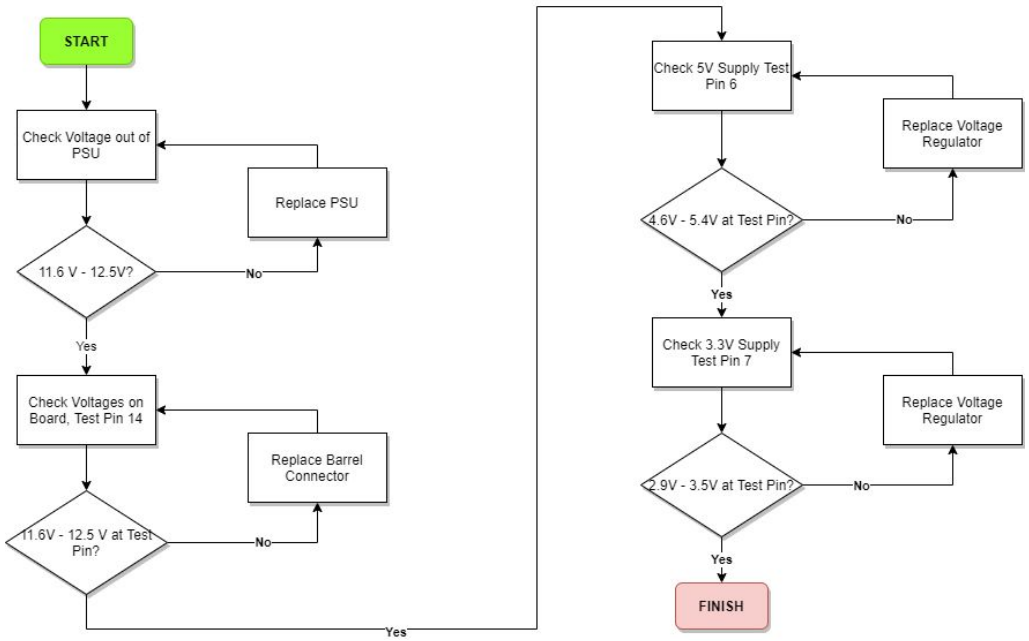


Figure 20. Power test plan

The power supply for the entire system starts with a 12 V power supply unit (PSU) from a wall outlet. This voltage is used to control the door lock, and is stepped down with linear voltage regulators to 3.3 V and 5 V for the microcontroller and the peripherals, respectively. The PSU voltage should first be confirmed to be within 0.5 V of 12 V. Then, the PSU should be inserted in the barrel connector on the board and the voltage verified at a test point. Then, the linear voltage regulators should be verified on a breadboard. Once the regulators are confirmed to be working, they should be added one at time to the board and verified on a test point.

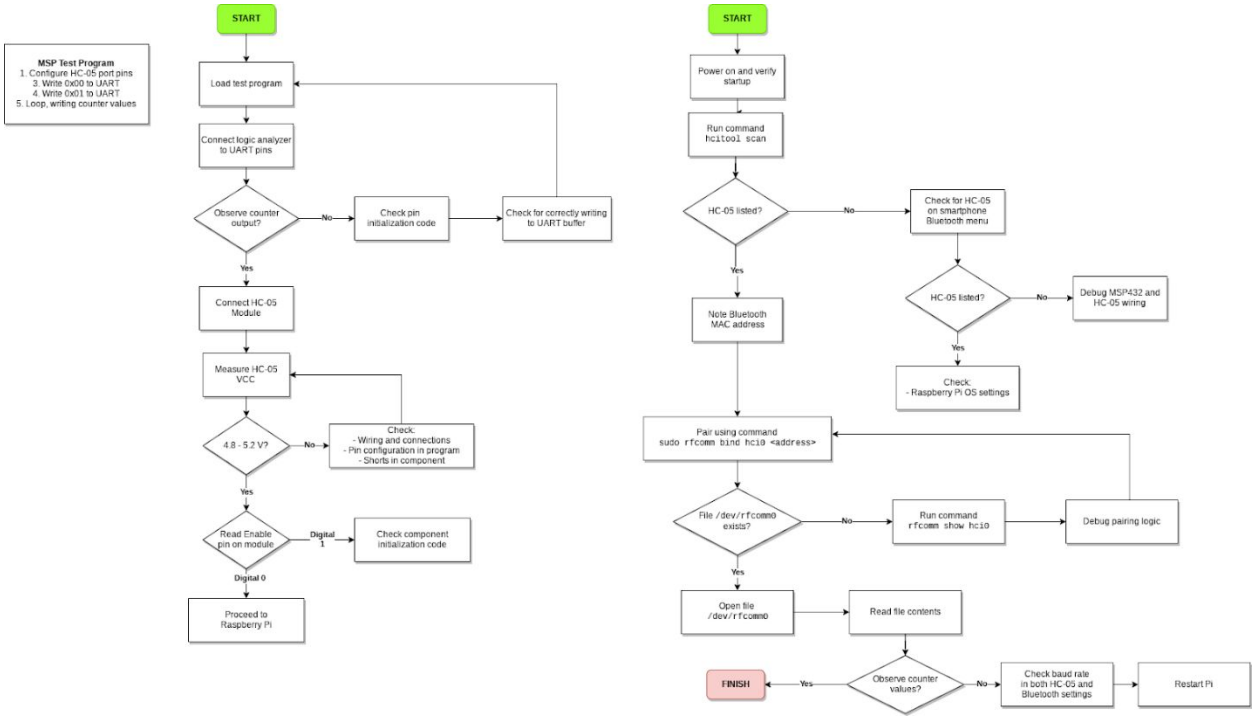


Figure 21. Bluetooth test plan, HC-05 side

Figure 21 and Figure 22, below, comprise the testing of the Bluetooth connection on both the microcontroller and the server, respectively. The HC-05 is initially loaded with a test program that repeatedly writes an incrementing counter to the UART interface. This can be read from the logic analyzer on the VirtualBench to make sure the embedded code uses the interface correctly. After plugging in the HC-05, we can make sure that VCC power is not distorted and that the code properly configures each pin on the module. On the server side, the tester must get a remote terminal on the Pi to run shell commands. First, the tester must scan for the HC-05 module, pair with it, and bind to it (the first two of these are only required on first setup). Once connected, the tester can examine the corresponding file on the device to see the Bluetooth counter values. The exact commands to run and files to examine are listed in the boxes of the test plan. Packet encoding and decoding were tested separately with unit tests in code, so they were not included in this test plan for the sake of simplicity.

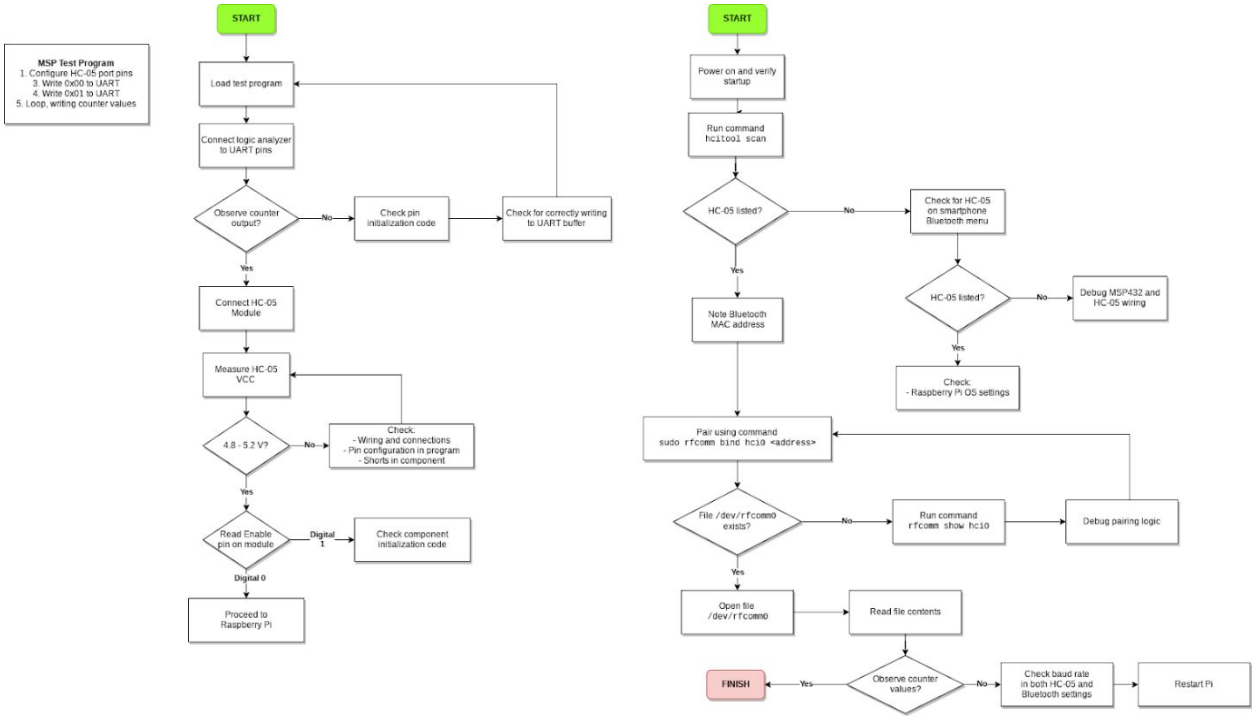


Figure 22. Bluetooth test plan, Raspberry Pi side

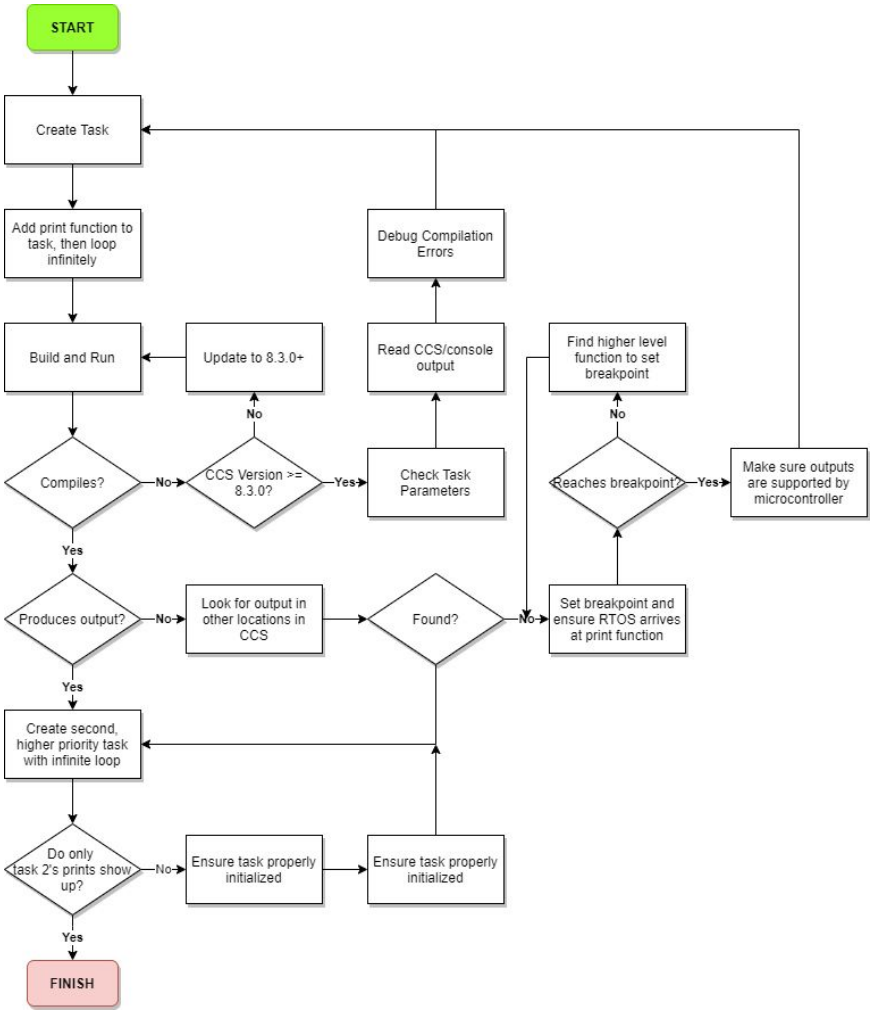


Figure 23. RTOS test plan

The RTOS testing was fairly seamless. The example projects provided by FreeRTOS greatly reduced the amount of debugging involved with compiling and designing for the MSP432 system. The largest struggle with the RTOS was managing different clock speeds for each subsystem. For example, the UART communication for the Bluetooth module operated at a different clock speed than the I2C bus.

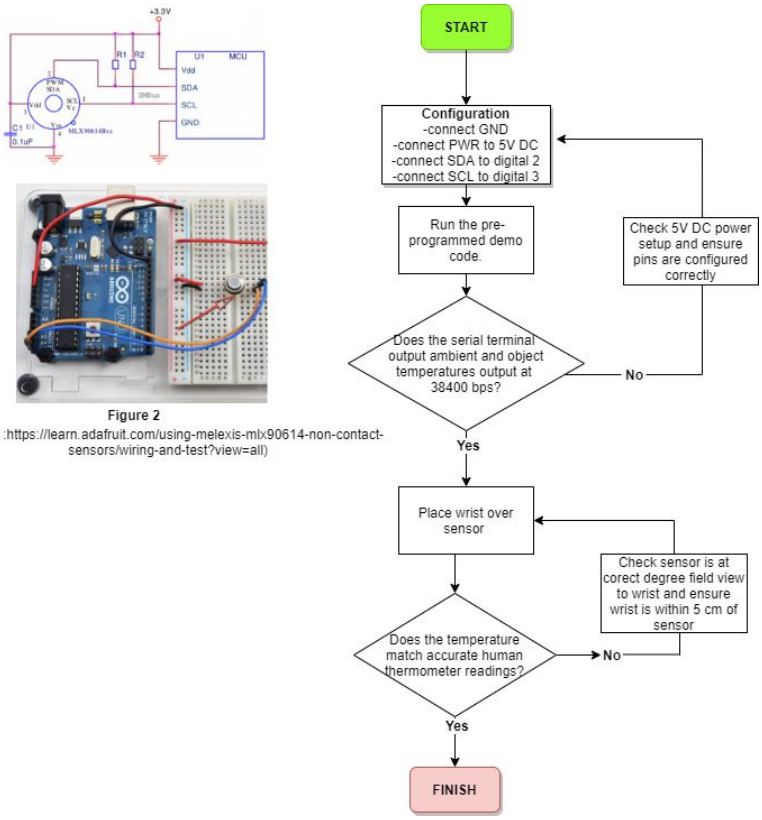


Figure 2
(Source :https://learn.adafruit.com/using-melexis-mlx90614-non-contact-sensors/wiring-and-test?view=all)

Figure 24. Temperature sensor test plan

Testing the temperature sensor module chip went as planned. Despite using the MSP432 microcontroller for the overall system, testing for this part was done using an Arduino microcontroller allowing our team to easily view serial terminal results with test code provided from Melexis Technologies. After wiring the temperature sensor to the Arduino and running the test code, we were able to get ambient and object temperature readings from the sensor shown on the serial terminal every few seconds, as expected, ensuring functionality.

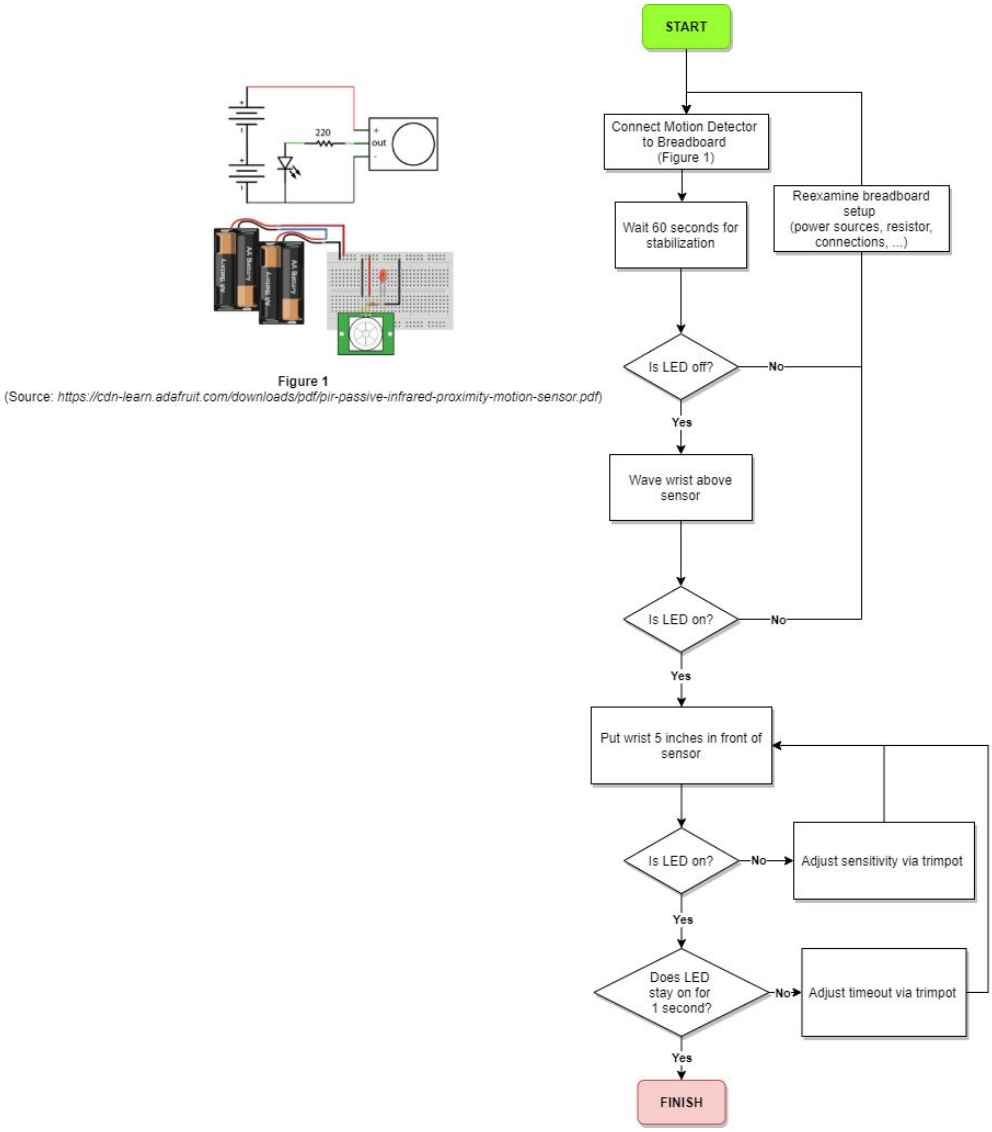


Figure 25. Motion sensor test plan

The test plan above highlights how the motion detector was tested. The test circuit shown in Figure 1 of the test plan itself comes from the datasheet, as well as the flow of steps taken to ensure that the motion detector is working properly. Since the manufacturers strongly recommend this version of a test plan, it was determined to be a trusted and comprehensive way to test this component [43].

Final Results

The final prototype of our device successfully takes non-contact temperature measurements, conditionally locks the door based on these measurements, and sends the data over Bluetooth to the Raspberry Pi to be shown on a web page. Table 4 represents the requirements for success outlined in the initial project proposal, though some features have been added and some changed slightly over the course of the semester.

For the locking mechanism section, we altered expected behavior so that the lock defaults to being locked until a valid temperature is scanned rather than defaulting to unlocked as specified in the table. This was a conscious change based on further thoughts on user interaction and fire safety. When a user scans their temperature, the lock reacts quickly, imperceptible to our group members during final testing. In the case of non-fever temperature the door unlocks, and after a reasonable amount of time afterwards (around 10-20 seconds) it re-locks. If power is lost, the door unlocks but when powered, the default state is locked. This represents full completion of the proposed functionality.

In the temperature sensor category, we have fulfilled almost all of the requirements for full credit except for that of displaying temperatures to users. Our temperature scans are non-contact using infrared, and are taken immediately once motion is detected, and are then sent to the server over Bluetooth. When motion is detected, 128 temperature scans are rapidly taken and averaged. Instead of displaying the raw temperature, door users are only shown an LED indicating the lock decision. Temperature scans can be taken back-to-back in a reasonable amount of time (the same 10-20 seconds), and one of the LED indicators is used to show when the system is ready for a new scan. As discussed with advisors and application engineers from the manufacturer throughout the semester, our temperature accuracy during testing has been noticeably erratic. We believe this is mostly due to the specific sensor we were able to purchase, since we had limited options available as a result of the pandemic. Even after averaging 128 samples, temperatures are sometimes erratic and heavily dependent on the wrist's distance to the sensor. In summary, this sensor would be unsuitable for use if going to market but we don't believe it should affect our outcome in this category. In the end, we achieved all requirements for full points except that we use LED indicators rather than displaying the raw temperature on an LCD screen.

For the dashboard, we believe we achieved full functionality. When accessed by any web browser on the local network, the page polls for data every two seconds to provide the experience of immediate updates. Temperatures over a user-configurable timespan are plotted using color to differentiate fevers from non-fevers. Statistics are calculated and shown to users about these temperatures as well as a log of all data. If the server goes down for any reason, no data would be lost since it is persisted on the server's filesystem, though no new data could be added and the website would not be available during downtime.

Points	Locking Mechanism	Temperature Sensor	Dashboard
2	• Locks immediately upon receiving temperature reading above the limit • Lock remains in locked position until receiving temperature in accepted range or within reasonable amount of time • Lock default (even with no power) is unlocked for safety	• Temperature readings are done via infrared (non-contact) • Temperatures taken as soon as new person steps up to device • Temperature readings are triggered by motion detector for energy-saving purposes • Temperatures readings are sent immediately to the server • Temperature readings are also displayed for users to see • Temperature readings can be taken back-to-back in reasonable amount of time	• Updates automatically as soon as new temperatures are taken • User-friendly UI including helpful graphs showing trends over time • Available as a website (not device-dependent) • Not fully dependent on server being connected to power + will not lose data if server loses power
1	• Locks a reasonable time after the person steps up to the door, the majority of the time • Lock default is still unlocked	• Temperature readings are not displayed to user specifically but displayed via red/green LED (red = temperature outside of normal range, green = temperature OK) • Temperature readings triggered by motion most times, but may take several seconds to recalibrate +	• Only available on 1 computer • Updates less frequently (i.e. once per day) with new readings • Contains 1 graph of temperatures taken over time

		redetect for new readings	
0	• Lock does not trigger even with temperature reading outside of the normal range	• Temperature readings not triggered with any motion • Temperatures taken are very inaccurate	• Not created

Table 4. Initially proposed project requirements and points.

While the presented project fulfills the requirements initially outlined, it is not ready for market adoption. As mentioned previously, the temperature sensor is not accurate enough for a commercial product. Another shortcoming is that the team eventually wanted to integrate multiple doors to the system. That is, multiple locks and multiple MSPs reporting data to the same server. While portions of the system were designed with this in mind (most of the server code is equipped to handle it), it quickly fell out of scope with time and budget constraints imposed this semester. We also fell short in our mechanical assembly and installation. Due to the team's limited experience in this area and time constraints in the last week of lab access, the final assembly was not as sturdy as initially expected. To fulfill manufacturing standards and generally keep the system functioning reliably, this would need to be significantly improved if commercialized.

Costs

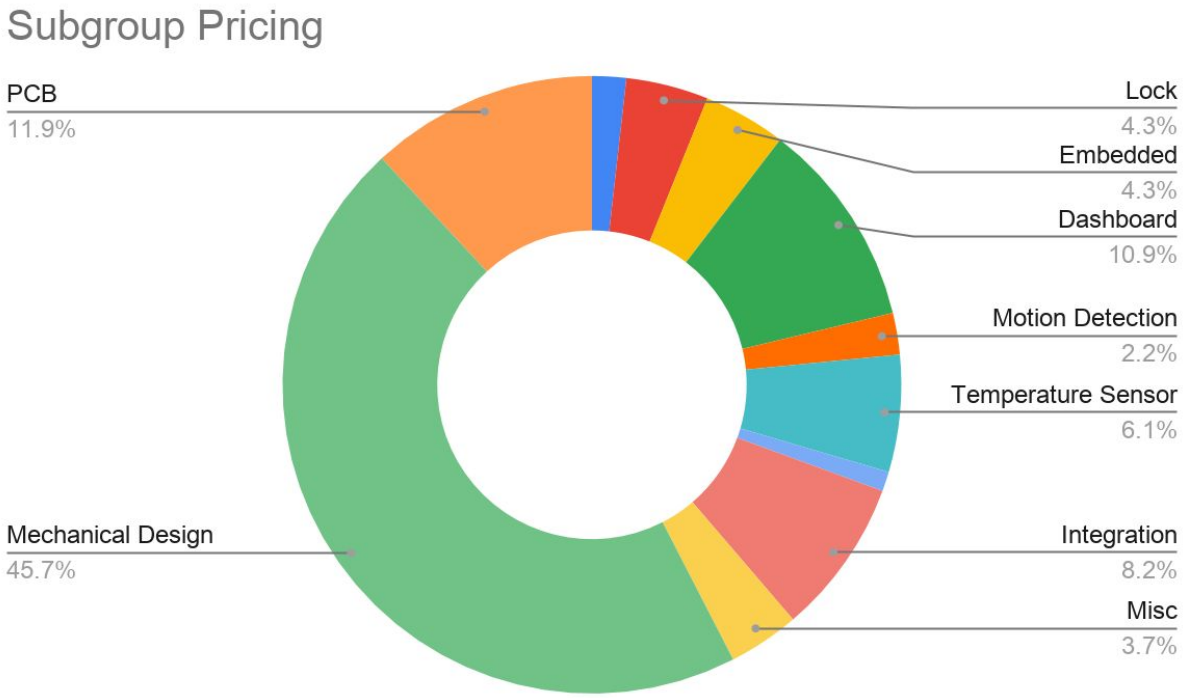


Figure 26. Budget subgroup breakdown

Each budget item was broken down into a subsystem category. The mechanical design was the most expensive, which included costs associated with 3D printing, acrylic parts, and other fasteners. The PCB was the next most expensive subgroup. This included the cost of our PCB as well as a revised version. The dashboard included the Raspberry Pi 3B+ as well as SD cards and other necessary peripherals. Most of the expense from the Embedded system came from the MSP432 LaunchPad. The integration subgroup included parts that were necessary to physically connect different subsystems, such as connectors, wires, and crimps. The overall cost of the project was $556.20.

If this project were taken into high volume manufacturing with 10,000 units, we would expect part prices to change from approximately $316 to $189. The 3D printing, which costed around $240 could be replaced with injection molding or some cheaper technique, which would be in the range of $15-20 for the price of plastic. Therefore, $200 would be a reasonable device manufacturing cost to aim for in high volume production.

Future Work

There are several major areas in which this system could be improved given more financial resources and time. The temperature sensor that has been utilized in this system should

be swapped out with a far more accurate sensor, perhaps with a lens that increases precision as well.

Future versions of this system should also enhance user experience. This could include replacing the three LED indicators with an LCD screen that instructed a first-time user on how to interact with the system, provided the user with their own temperature reading rather than a binary good-or-bad, and gave the owner of the system an idea as to the system's health.

A major concern with this system is how easily it may be rendered useless by user misuse, whether a user props the door open or just allows others to enter under his/her temperature scan. This was not a problem that the team could tackle in such a short timeframe, but it may be possible to incorporate a door alarm for the door being open for too long (to discourage propping the door open), a weight sensor to discern how many people had crossed the threshold per temperature scan, or perhaps a switch from the non-contact wrist temperature scanner to an infrared camera mounted on the door frame that would scan users' foreheads. These ideas, especially the lattermost, would likely have some ethical implications that would need to be explored.

A final extension that could make the project more usable would be allowing multiple door-mounted locks to report data to the same server. This would allow building owners to monitor entrant temperatures for all access points on the same dashboard. Much of the server code was designed with this extension in mind, but it would have required multiple copies of the PCB and microcontroller which we couldn't justify from a budget perspective.

References

[1] *Temperature Kiosks: What Are They and How Much Do They Cost?,* 2020.

[2] *Temperature Screening Kiosk | Personnel Management.*

[3] *Grant of Equipment Authorization,* 2018.

[4] *Equipment Authorization of Unintentional Radiators,* 2017.

[5] I. B. C. Council, International Building Code, Country Club Hills, IL: International'Code Council, Inc, 2017.

[6] D. H. C. D. Division of Building and F. Regulation, 2015 Virginia Construction Code Part 1 of the Virginia Uniform Statewide Building Code, State Building Codes Office, Richmond, VA, 2018.

[7] *Stainless Steel Electric Power Door Lock Access Control Fail Safe NO Narrow-type 12V DC.*

[8] *TPS1H200A-Q1 40-V 200-mΩ Single-Channel Smart High-Side Switch,* Texas Instruments, 2019.

[9] *Guidance Regarding Methods for De-identification of Protected Health Information in Accordance with the Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule.*

[10] *IPC Checklist for Producing Rigid Printed Board Assemblies,* Association Connecting Electronics Industry, 2019.

[11] *Bluetooth Core Specification,* Core Specification Working Group, 2019.
[12] *I2C-bus specification and user manual,* NXP, 2014.
[13] *IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,* IEEE, 2016.
[14] *OpenSSH,* 2020.
[15] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach and T. Berners-Lee, *Hypertext Transfer Protocol – HTTP/1.1,* IETF, 1999.
[16] *NI Multisim,* National Instruments.
[17] *Ultiboard.*
[18] *AutoCAD Overview.*
[19] *Git.*
[20] *Code Composer Studio (CCS) Integrated Development Environment (IDE).*
[21] *Code Composer Studio (CCS),* Texas Instruments.
[22] S. Tatham, *PuTTY Use Manual.*
[23] *MSP430 Optimizing C/C++ Compiler v20.8.0.STS,* Texas Instruments, 2020.
[24] *FreeRTOS.*
[25] *Raspberry Pi OS,* Raspberry Pi Foundation.
[26] *PySerial.*
[27] G. Haring, *sqlite3.*
[28] *Flask.*
[29] *Apache HTTP Server,* Apache Software Foundation.
[30] *Recycling FAWs.*
[31] M. Griffin, *Is PLA Recyclable?,* 2020.
[32] *How Do You Recycle Acrylic Resin?,* 2012.
[33] "Health kiosk". 12 2018.
[34] "Medical kiosk and method of use". 1 2018.
[35] "Multi-biometric enrollment kiosk including biometric enrollment and verification, face recognition and fingerprint matching systems". 5 2012.
[36] *HC-05 Bluetooth to Serial Port Module,* ITead Studio, 2010.
[37] *Pyroelectric Passive Infrared Sensor,* Adafruit.
[38] *MSP432P401R, MSP432P401M SimpleLink™ Mixed-Signal Microcontrollers,* Texas Instruments, 2019.
[39] *TL750L Low-Dropout Voltage Regulators,* Texas Instruments, 2009.
[40] *L4931 Very low drop voltage regulators with inhibit,* ST Microelectornics, 2017.
[41] *WP7113LGD T-1 3/4 (5mm) Solid State Lamp,* Kingbright, 2019.
[42] *PJ-036AH-SMT-TR DC Power Jack,* CUI Devices, 2019.
[43] *PIR Motion Sensor.*
[44] T. Ylonen, *The Secure Shell (SSH) Protocol Architecture,* C. Lonvick, Ed., IETF, 2006.
[45] M. Valentine, D. C. J. Bihm, L. Wolf, H. E. Hoyme, P. A. May, D. Buckley, W. Kalberg and O. A. Abdul-Rahman, "Computer-Aided Recognition of Facial Attributes for Fetal Alcohol Spectrum Disorders," *Pediatrics,* vol. 140, p. e20162028, 12 2017.

[46] D. Stauss, M. Rogers and M. Herr, *U.S. Privacy Law Implications with the Use of No-Contact Temperature Taking Devices,* 2020.
[47] D. Siegmund, S. Dev, B. Fu, D. Scheller and A. Braun, "A Look at Feet: Recognizing Tailgating via Capacitive Sensing," in *Distributed, Ambient and Pervasive Interactions: Technologies and Contexts*, vol. 10922, N. Streitz and S. Konomi, Eds., Cham, Springer International Publishing, 2018, p. 139–151.
[48] A. Sharma, *Motion Detector Using MSP430 Launchpad and PIR Sensor,* 2019.
[49] J. Morrissey, "Fighting the Coronavirus With Innovative Tech," *The New York Times,* 6 2020.
[50] E. Maras, *Temperature check kiosks ready to tackle COVID-19,* 2020.
[51] M. Kreiger and J. M. Pearce, "Environmental Life Cycle Analysis of Distributed Three-Dimensional Printing and Conventional Manufacturing of Polymer Products," *ACS Sustainable Chemistry & Engineering,* vol. 1, p. 1511–1519, 12 2013.
[52] H.-Y. Kang and J. M. Schoenung, "Electronic waste recycling: A review of U.S. infrastructure and technology options," *Elsevier,* vol. 45, p. 368–400, 12 2005.
[53] L. Hodges, *Ultrasonic and Passive Infrared Sensor Integration for Dual Technology User Detection Sensors.*
[54] E. Grodzinksy and M. Sund-Levander, "Understanding Fever and Body Temperature," p. 23–25, 8 2019.
[55] L. Greene, "Decoded Fail Safe vs Fail Secure - When and Where," p. 3.
[56] S. Graham, *Samuel Pierpont Langley,* 2000.
[57] S. Gong, C. C. Loy and T. Xiang, "Security and Surveillance," p. 18.
[58] A. Elejalde-Ruiz, "If you get sick with COVID-19, is your employer liable? As businesses prepare to reopen, worker safety is a priority.," *Chicago Tribune,* 5 2020.
[59] CDC, *Coronavirus Disease 2019 (COVID-19),* 2020.
[60] *WP7113LYD T-1 3/4 (5mm) Solid State Lamp,* Kingbright, 2019.
[61] *WP7113LID T-1 3/4 (5mm) Solid State Lamp,* Kingbright, 2019.
[62] *What Is Multisim.*
[63] *Validity of Wrist and Forehead Temperature in Temperature Screening in the General Population During the Outbreak of 2019 Novel Coronavirus: a prospective real-world study.*
[64] *Universal Asynchronous Receiver/Transmitter (UART),* Texas Instruments, 2010.
[65] *Ultiboard,* National Instruments.
[66] "Symptoms of Coronavirus (COVID-19)," p. 1.
[67] *Product Compliance and Safety.*
[68] "Pawl & solenoid locking mechanism". Patent US5887467A, 3 1999.
[69] *MLX90614 Family Datasheet Single and Dual Zone Infra Red Thermometer in TO-39,* Melexis, 2019.
[70] *Low-noise and long-range PIR sensor conditioner circuit with MSP430TM smart analog combo,* 2019.
[71] *How to use a high current solenoid with Arduino,* learnelectronics.
[72] *GitKraken,* Git.
[73] *DriverLib,* Texas Instruments.

[74] *COVIDWISE,* 2020.
[75] CDC, *COVID-19 Cases, Deaths, and Trends in the US | CDC COVID Data Tracker,* 2020.

Appendix

Table 5. Full list of project expenses.

Part	Quantity	Individual Cost	Shipping Est.	Total	Budget Rem.
HC-05 (x2)	1	$9.88	$0.00	$9.88	$490.12
Adafruit 1512	0	$15.95	$10.00	$0.00	$490.12
MSP-EXP432P401R	1	$23.99		$23.99	$466.13
Raspberry Pi 3B+	1	$35.00	$10.00	$45.00	$421.13
32GB SD Card	1	$8.49	$6.76	$15.25	$405.88

PIR Sensor	1	$12.09		$12.09	$393.79
Temp Sensor	1	$28.99	$5.00	$33.99	$359.80
Door Lock	1	$19.85		$19.85	$339.95
Power Barrel	2	$0.96		$1.92	$338.03
5V regulator	2	$0.79		$1.58	$336.45
3.3V Regulator	2	$0.65		$1.30	$335.15
Driver Chip	1	$1.56		$1.56	$333.59
Wire Connectors				$0.00	$333.59
2 Male	2	$0.26		$0.52	$333.07
2 Female	2	$0.40		$0.80	$332.27
3 Male	2	$0.28		$0.56	$331.71
3 Female	2	$0.45		$0.90	$330.81
4 Male	2	$0.29		$0.58	$330.23

4 Female	2	$0.48		$0.96	$329.27
6 Male	2	$0.32		$0.64	$328.63
6 Female	2	$0.57		$1.14	$327.49
MSP Power Header	2	$0.37		$0.74	$326.75
Test Pins BLK	10	$0.40		$4.00	$322.75
Test Pins WHT	15	$0.40		$6.00	$316.75
Test Pins Yellow	6	$0.40		$2.40	$314.35
2.2uF Caps	5	$0.50		$2.50	$311.52
NEW 3.3V Regulator just in case	2	$0.44		$0.88	$310.64
Shunts	3	$0.13		$0.39	$310.25
Shipping + Tax	1	$10.96		$10.96	$299.29
LED RED DIFFUSED T-1 3/4 T/H	2	$0.33		$0.66	$298.63
LED GREEN DIFFUSED T-1 3/4 T/H	2	$0.33		$0.66	$297.97
LED YELLOW DIFFUSED T-1 3/4 T/H	2	$0.33		$0.66	$297.31
CONN HEADER VERT 2POS	2	$1.11		$2.22	$295.09
CONN HEADER VERT 3POS 2.54MM	2	$1.40		$0.88	$294.21
CONN HEADER VERT 4POS 2.54MM	4	$1.45		$5.80	$288.41
CONN HEADER VERT 6POS 2.54MM	2	$1.37		$2.74	$285.67

CONN HOUSING 2POS .100" CRIMP	2	$0.44		$0.88	$284.79
CONN HOUSING 3POS .100" SINGLE	2	$0.42		$0.84	$283.95
CONN HOUSING 4POS .100" CRIMP	4	$0.44		$1.76	$282.19
CONN HOUSING 6POS .100" DUAL	2	$0.53		$1.06	$281.13
CONN SOCKET 22-24AWG CRIMP GOLD	25	$0.34		$8.50	$272.63
Shipping + tax	1	$9.65		$9.65	$262.98
Acrylic Sheet	1	$13.48		$13.48	$249.50
PCB Estimate	2	$33.00		$66.00	$183.50
3D Printing ESTIMATE	1	$240.00		$240.00	-$56.50