Resistor Sorter 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

Joseph Laux
Fall, 2019

Technical Project Team Members
Robyn Guarriello
Joseph Laux
Kiri Nicholson

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

Resistor Sorter System

Statement of work:

Robyn Guarriello:

Robyn's main contributions to the project included programming the MSP430 and designing the CAD models for the physical structure. The MSP430 was programmed to control two stepper motors - one that would turn a slide to face the correct sorting bowl and one that would then turn a paddle to knock the resistor down the slide into the bowl. In order to do this, Robyn created an algorithm that would initialize all values for the motors, receive Bluetooth input, calculate the shortest distance for the slide to reach its destination, turn the first stepper motor, and then push the resistor down the slide. This was implemented in C using Code Composer Studio and flashed onto the MSP using a Spi-By-Wire interface. In addition to working with the MSP, Robyn was responsible for designing and modeling the physical structure of the system. The online CAD tool OnShape was used to create the design, which was then 3D printed and constructed. In order for the system to work properly, there were a few design constraints that Robyn had to take into consideration when designing the structure. First, the turntable and slide had to be as light as possible so as not to weigh down the stepper motor. Next, the PCB had to be able to fit inside of the base, but the base could not be too large such that the slide would not be able to turn. Finally, the stepper motor wires needed purposefully placed holes so they could spin freely without damaging the PCB.

Kiri Nicholson:

Kiri's main contributions to the project included work with the Android application. The team made the decision to use an open-source application that used a camera and Computer Vision to "read" a resistor. Kiri was responsible for downloading the application and getting it to work on two Android devices. Once the code was downloaded to a local machine, it needed to be edited (mainly gradle files) and built properly so that it would deploy successfully on the cell-phones. Once the application was up and running on the phones, some preliminary testing was done to check the accuracy of the resistor reading. Then, Kiri began working on adding Bluetooth capabilities to the application. Kiri first did research into Bluetooth communication protocol and Android application basics as this was an area that none of the team members had experience in. The development of the Bluetooth capabilities included a lot of trial and error, and progressed from the new Bluetooth code throwing errors upon building to the code being able to be deployed on the phone. Though the Bluetooth communication ability did not end up working by the demo day, all Bluetooth communication development work was Kiri's responsibility. In addition, Kiri took responsibility for the ordering of all parts through the ECE process that the team needed (other parts were purchased individually by other team members).

Joseph Laux:

Joseph’s main contributions included designing the schematic and the layout for the PCB, as well as soldering the physical components into the final PCB. The electronic schematic was designed first using Multisim and the board layout was completed using Ultiboard. In the schematic, multiple components such as a TI MSP430G2553, ULN2003 stepper motor drivers, and a Bluetooth HC05 module were integrated into the circuit. The voltages and currents for each of these components were crucial, so Joseph was tasked with analyzing the respective datasheets for the components in order to ensure the power limitations were being met. Additionally, Joseph was responsible for implementing the Spy-Bi-Wire MSP430 Debugger connector onto the PCB. This component was crucial for debugging and ultimately flashing the code onto the board itself in order for the project to operate without the use of an MSP430 launchpad. Additionally, small design decisions such as the placement of bypass capacitors being directly adjacent to all of the integrated circuit (IC) chips on the PCB were made. Ultimately, these capacitors improved the performance and reliability of the main components on the board and provided stability in the final product’s execution. Finally, the copper tracks of the PCB were carefully designed. Critical areas of the board, such as the power supplies of the ICs had thicker copper tracks to ensure the reliability of the connection.

Table of Contents

Contents

Table of Figures

Abstract

The idea behind the Band Pass Filters Resistor Sorter was to provide a way to quickly organize piles of resistors. Often when constructing circuits, resistors can be misplaced or mixed in with resistors of other values, which can lead to the wrong value resistor being used in the circuit. This can be detrimental to the accuracy of the circuit or even the ability for the circuit to work at all. The goal for the Resistor Sorter was to function as a semi-autonomous machine that uses computer vision to determine the resistance by looking at the colored bands on the outside of the resistor. A student-provided phone would use an open-source but capstone-team-modified Android application to implement computer vision. The application would recognize the resistance and send this data point over a bluetooth connection to a HC-05 Bluetooth module interfaced with the MSP430 microprocessor. The microprocessor would then compute a precise angular distance for the stepper motor to turn in order to line up with the correct sorting bin for the resistor to fall into. The stepper motor would turn the precise angular distance and a second stepper motor would turn a paddle that would knock the resistor down into the correct bin.

Background

The team considered many possible projects for our capstone before we ultimately selected this one. As a team of three people, we first knew that we wanted to scope this project well so that we could easily accomplish it by the semester’s end. We decided fairly early in the process that we wanted to build some sort of electro-mechanical system, perhaps one that could performs some sort of menial task. We then landed upon the idea of building a system that sorted something. We ultimately chose the resistor sorter system as it was a well-scoped and fun idea that incorporated aspects of our educations as Computer Engineering students. We are often met with messy piles of resistors that are not sorted easily. This project allowed for a simple and fun solution to this problem. We were wary of “biting off more than we could chew” in terms of the mechanical part of this project, so we made an adjustment to utilize Computer Vision to sort the resistors instead of a electro-mechanical system with an arm that would use an ohmmeter system to measure the resistance.

In our search for a capstone project topic, we took to the Internet to look for inspiration. Once we had settled on developing a sorting mechanism of some kind, and then further narrowed this to a resistor sorter, we drew ideas and tips from several sources. We found a proposal for a resistor sorting system at Swarthmore College [1]. This system was far more mechanical than ours, however, and it did not employ computer vision, but measured its resistance directly. One resistor sorting system we found used a hopper and a PIC processor to feed resistors into its sorting machine [2]. Yet another (still under construction according to its website) used a voltage divider to measure the resistance and a belt drive in order to sort the resistors [3].

We were eventually led to a webpage that featured an open-source Android application that had logic implemented that used Computer Vision to allow us to sort the resistors based on their bands, and not measuring their actual resistance [4] [5]. The spin that we wanted to put on this project was the use this existing app, add Bluetooth capabilities to it, and allow a physical apparatus to do the rest of the sorting. We drew inspiration from many of the sources noted above, but the combination of Computer Vision, Bluetooth, and the mechatronic system was novel to our knowledge.

Our previous coursework came into play in several ways as we worked on this capstone. Basic electrical engineering design skills came in handy as we perused data sheets and other information to decide on the proper components for our design. These skills were also needed for the design and development of our Printed Circuit Board (PCB). These skills were acquired in classes like Fun I-III. We also used skills acquired in our Intro to Embedded programming class, such as knowledge of the MSP430, our chosen microprocessor, and operational knowledge of working with the stepper motors that rotated our table and resistor-pushing mechanism on our project. Some team members also acquired basic 3D modeling skills in classes such as Introduction to Engineering that came in handy in our design and printing of our mechanical parts of our project. As far as Bluetooth communication protocol and Android mobile application programming, courses such as Networks, Advanced Software Development, Software Development Methods, and Operating Systems came in handy for their tangential and background knowledge that they provided.

Constraints

Design Constraints

We had a few key constraints that we considered in our development of our project. We knew that the timeline for the project was going to be the biggest constraint that we had to consider. From the beginning, scoping our project in a way that made it possible for us to finish it in a semester was of the utmost importance to us. In tandem with considering our timeline, we also had to consider our limitations in knowledge and what we could feasibly master in a semester. Because of this, we chose to use the MSP430 and stepper motors for a large part of our project. Parts such as stepper motors, MSP430s, and the 3D printed parts we designed were all able to be acquired within the timeframe we had, so these were considerations when choosing these parts. When working with the Android application, these concerns were some of the main ones that led us to choosing this open-source application. Then, since the application was primarily written in Java and meant to run on an Android device, these choices were effectively made for us in terms of coding development.

In addition, there were a few design constraints related to the physical structure of the system. Foremost, the stepper motor that was used, a 28BYJ-48 Stepper Motor, is rated for 300 gf.cm pull in torque and 5V DC rated voltage. Because of the limitations of the physical stepper motor, the weight of the turntable and slide that it was responsible for turning had to be minimized to ensure accurate stepping. This meant that the material used to construct the structure had to be chosen carefully so that size was not sacrificed. Additionally, because the stepper motors are wired, the distance from the PCB that stepper motors could be placed was a limitation. Holes had to be placed precisely in the base to ensure that wires could reach the PCB and would not get tangled while the structure was spinning.

The MSP430G2553 microprocessor used also presented some constraints. The 20 pin MSP that was used had only 16 I/O pins. Thus, the number of components that were controlled by or connected to the MSP were limited. Additionally, the MSP430 has a 16-bit RISC CPU. However, because the computational power that the project required was not significant, the CPU did not present too much of a constraint for this particular project.

Economic and Cost Constraints

An obvious consideration in the development of our project was making sure that it was not exorbitantly priced. As the entire point of our project was to develop a resistor sorting system, it is conceivable that this would primarily be used in an academic lab setting or by hobbyists. Both of these stakeholder groups are parties that primarily work under tight or limited budgets. Therefore, cost was a consideration when looking into our design. Thankfully, many of our tools used (Software Development IDEs and 3D modeling tools) were free or previously acquired for other academic purposes. We had to order many electronic parts, as well as send our PCB out to be manufactured, but these came to an overall low cost. A further breakdown can be seen in our cost spreadsheet in Appendix A. We were not put into a position where we had to make a decision about acquiring an individual high-cost part for our system. In addition, we never intended for our capstone project to develop a system that would ultimately be sold or mass-produced, so keep costs low was not a 'dire need.' While economic and cost constraints were considered, they were not a make-or-break issue given the scope of our project.

External Standards

In our capstone proposal document, we outlined a number of external standards that we would be following in the design and development of our capstone. As we noted there, Bluetooth devices operate in the 2.4GHz band of the unlicensed ISM frequency band [6]. The IEEE Standard that comes into play here is the Bluetooth Standard, or 802.15.1 [7]. By using a module built to communicate using the Bluetooth protocol and by using the Bluetooth capabilities of the Android platform, we ensured that we were in compliance with the IEEE Bluetooth Standards.

In addition, since we used a PCB in our project, IPC Standards for PCBs were used [8]. Since we designed our PCB in Multisim and sent it to a PCB manufacturer, we can be sure that all standards were followed. Some examples include IPC-2581 that specifies a specific format that PCB files and information must be sent in (from us to the PCB manufacturer) and IPC-4101C that specifies material that the PCB may be made out of [8].

Tools Employed

We used a variety of different engineering tools in order to complete this project. First off, when developing the Android application, we started by using Visual Code Studio but quickly transitioned to using Android Studio, as this offered a variety of Android mobile app specific features. (For instance, it made it much easier to compile, build, and deploy the code with little know-how.) Android Studio had never been used before by the team, however, and there was definitely a learning curve involved. This tool in particular is very powerful, but can be overwhelming to a novice. The open-source app was originally downloaded off of GitHub, and then it was made to run on two Android devices.

To develop the MSP430 code that controlled the receiving Bluetooth communication as well as the stepper motors, Code Composer Studio was used to develop the code. We used Code Composer Studio in our Introduction to Embedded Computing course in Fall 2018, so thankfully using this tool required very little brush-up. The printing of the 3D parts for our design required that we use and develop CAD models. To do so, we used two different 3D modeling programs: Free CAD and OnShape. Neither had been used by the team members before, but were similar to other 3D modeling tools employed by the team in past coursework.

Ethical, Social, and Economic Concerns

Environmental Impact

The environmental impact of any design and subsequent product is rarely completely negligible, but our team feels that our design has very little environmental impact. Our system has the ability to be powered by 3 rechargeable D batteries. Choosing to use rechargeable batteries significantly cut down on the potential waste produced by our design, as we have the ability to continue to reuse the same batteries as opposed to having to dispose of a set every time they die. Other environmental considerations include the heat generated by the electronics as our systems runs, as well as power consumed to produce the structure. Both of these concerns are negligible, however. Overall, our system does not have much of an environmental impact. It is small enough that its footprint is also negligible in its physical environment.

Sustainability

The 3D structure itself is made of plastic, as it was 3D printed, which is obviously not the most sustainable material to use, as it is slow to break down or lacks the ability to be recycled easily. If our system were to ever move into a stage where it was going to be mass produced, other materials such as wood or a more sustainable type of plastic may be considered in order to cut down on the consumption of hard plastic.

In addition the materials used to print and assemble the PCB (including the electronics soldered on) were of standard choice and are not especially sustainable.

As touched on above, the choice to use rechargeable batteries has helped ensure that our design is a sustainable one. The system itself is relatively low-power, and does not pose an issue as a large power draw.

Health and Safety

Our system does not pose a major health or safety threat to the public. Some minor risks are involved with its use including but not limited to the following: (1) Minor burns or shocks when handling the electric components if something is mishandled (2) Occasionally a malfunction in the system will allow for a resistor to be “flung” at a person as a projectile, but this is not a major concern due to the mass and size of the resistor (3) Contact with hot surfaces in the event of a short circuit or overheating malfunction.

Overall, our system presents a negligible health and safety risk. However, other minor factors may be considered such as possible eye strain from use of the resistor scanning application and the possibility of the a glass sorting bowl falling and shattering.

Manufacturability

It was never our intention to design our system in a way that would lend itself to be manufactured at a later time. The scope of our project was always a simple device that might be useful in an academic or hobbyist setting. Therefore, we gave very little weight to the issue of manufacturability when we were designing our system.

For example, our system is made of many different, small pieces made of different material that were assembled together by precise human effort. Our structure was 3D printed and then hot glued together, and the PCB was soldered and connected through wiring to a proto board. If we were making an effort to consider manufacturability, a better design choice would’ve been to design the 3D structure in a way that allowed it to be printed and assembled in fewer steps. We also would’ve looked more closely at how we were using our space as we designed, since form factor is a big consideration in the world of manufacturing. However, since this was not a major consideration, little weight was given to these options.

Ethical Issues

With any automation of any task, this of course brings about some ethical questions and concerns about taking work away from human individuals that were previously performing this task. As mentioned many times above, however, the scope of this project was to build a simple system that may be useful in an academic lab or in a hobby setting. Therefore, it is unlikely that prior to the use of our system it was any one human individual’s job to sort resistors. As the final product stands on ECE Demo Day, the system still requires a fair amount of human interaction, (scanning or maybe reading the resistor, placing it on the slide, punching in the Bluetooth communication) so it is safe to say that taking away jobs is not something we have to be concerned with when it comes to our system.

The use of a camera on a phone to collect data, while maybe not the greatest concern here, since the app just looks for centroids of different colors to read the bands, is a major ethical issue that warrants discussion [5]. To our knowledge and using our foresight, its use in our system will not present any adverse privacy effects to users, but this area of technical development is certainly fraught with ethical questions, and is worth mentioning here.

Intellectual Property Issues

It was never a goal of our team to develop a system that we could patent, and because of this, patentability was an issue that was given very little thought throughout the design and construction of our system. Especially given the inspiration we drew from similar projects we viewed online (noted above), and out use of open-source code that we did not originally develop, it seems a poor choice to pursue a patent. However, a patent search is still a useful learning experience and it is worth doing to take a closer look at previous work done in this area. Below are three patents we discovered after doing a preliminary patent search with terms related to our project.

This patent is for “Computer vision and machine learning software for grading and sorting plants” and has a few interesting claims that related to our project [9]. Its first and independent claim is:

> *A method to recognize and classify a bare-root plant on a surface, comprising the steps of: receiving from an imaging device a continuous output of raw image data of several bare-root plants passing through a field of view of the imaging device, wherein each of the several bare-root plants are arbitrarily disposed on the surface and positioned between the surface and the imaging device; wherein the surface is horizontal and each of the several bare-root plants is lying flat on the surface; and wherein an orientation of each of the several bare-root plants on the surface is not uniform; identifying a single bare-root plant in the raw image data by detecting and extracting a region in the raw image data corresponding to the single plant; classifying each pixel of the bare-root plant identified in the raw image data to form a classified bare-root image based on trained parameters, comprising: generating a vector of scores for each pixel of the classified bare-root image; and identifying a plurality of sub-parts of the bare-root plant based at least on the vector of scores and the trained parameters; evaluating the classified bare-root image based on trained features to assign the bare-root plant to a configured category; and sorting the bare-root plant based on the assigned configured category as it moves from the field of view, using a sorting device in communication with a vision system.* [9]

A few of the dependent claims, that come after this one reference the use of color masking and manipulation in order to recognize the plant [9]. “7. The method according to claim 6, wherein the step of selecting the color of the surface to form the background color further comprises a step of segregating the foreground color and the background color based on a hue threshold to create a mask”[9]. This is similar, though we don’t think too close, to how the resistor sorter code operates. The code used by the team reads bands and uses centroids to translate bands into a cohesive resistor value. The process did seem reminiscent to what is being claimed here, however.

This patent is for a “Coin recognition system and method” that also uses image recognition [10]. What is most similar here though, is its “binning” of coins as described here “7. The method of claim 5, wherein the bin map indicates files or folders in which ones of the second set of coin images are to be stored” in this dependent claim [10]. Though the binning does not seem to be physical, but rather in some sort of file system, it’s still worth mentioning as small objects are being sorted via image into “bins” much like our project.

This patent also sorts coins, and is filed as “Digital image coin discrimination for use with consumer-operated kiosks and the like” [11]. As noted in the abstract of this patent, “The methods can include recognizing colors and/or reflectivity of the coin using, for example, pixel thresholding algorithms. The methods can further include adding a line to an image of the coin, and measuring angles between the line and the edges on the coin” [11]. This is similar to our resistor reading application logic, as it also uses color recognition to “read” the resistance of a particular resistor.

Detailed Technical Description of Project

The resistor sorter is a system that allows users to quickly sort rogue resistors that are not in their correct spot. The resistor is first scanned using a computer vision application that tells the user what the value of the resistor is by reading the colored bands that denote the value. Once the resistors value is determined, the value is communicated to an HC-05 Bluetooth module that is interfaced with an MSP430. The MSP430 processes this value, calculates the angular distance it must turn a stepper motor in order to face the correct sorting bowl and does so, and promptly pushes the resistor down a slide and into the correct bowl. The system can be programmed to sort six resistor values at a time and will sort any other values into a “reject” bowl. The system can then be reprogrammed to sort a different set of values, and the previous reject resistors can then be sorted.

The resistor sorter used the following components in order to accomplish its goals:

- Two 28BYJ-48 – 5V Stepper Motors [12]
- Two ULN2003 Stepper Motor Drivers [13]
- MSP430G2553 [14]
- HC-05 Bluetooth Module [15]
- Spy Bi-Wire (2 Wire JTAG) for MSP430 (Provided, not linked)
- Moto G2 Android Phone (Student Provided, not linked)
- Three 9V Battery Clips [16]
- Seven test points
- 3.3V Regulator [17]
- Two 5V Regulators [18]
- PLA Filament
- 7x2 Male connector pins [19]
- 6x1 Male connector pin [20]
- Connector Socket 16 pin (SMD) [21]
- Three 9V Batteries [22]

- Connector Socket 20 pin (MSP) [23]

The block diagram for the system can be seen in Figure 01 below. This diagram illustrates the connections between each component of the system and how the resistor goes from being read by the application to being sorted into the correct bowl. First, the android phone uses the resistor sorting application, which uses the phones camera to “see” the resistor, to find the value of the current resistor. The Bluetooth terminal application, which is also on the phone, then communicates with the HC-05 Bluetooth module through a Bluetooth connection. The HC-05 module then sends the received value through its pin connections to the MSP430, which is powered by a 9V battery. The MSP430 processes the value it received, calculated the shortest distance to the correct sorting bowl and then, through pin connections to two stepper motor drivers, sends the correct signals to turn the motors the correct distance at the correct times. The two motors are each powered by a 9V battery as well. Finally, the motors have physical connections to the slide that is responsible for delivering the resistor to the correct bowl. When the motors turn, the slide also turns, thus positioning the resistor correctly for release into the corresponding sorting bowl. A simplified depiction of this can be seen in Figure 02, which illustrates how each submodule completes its tasks and communicates with the next submodule..

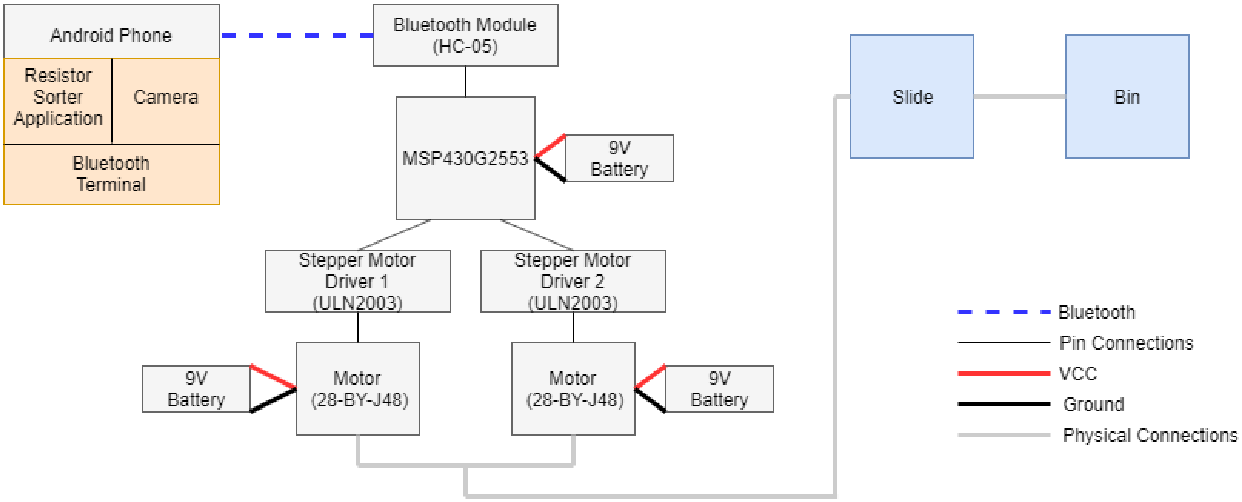


Figure 01 Block Diagram

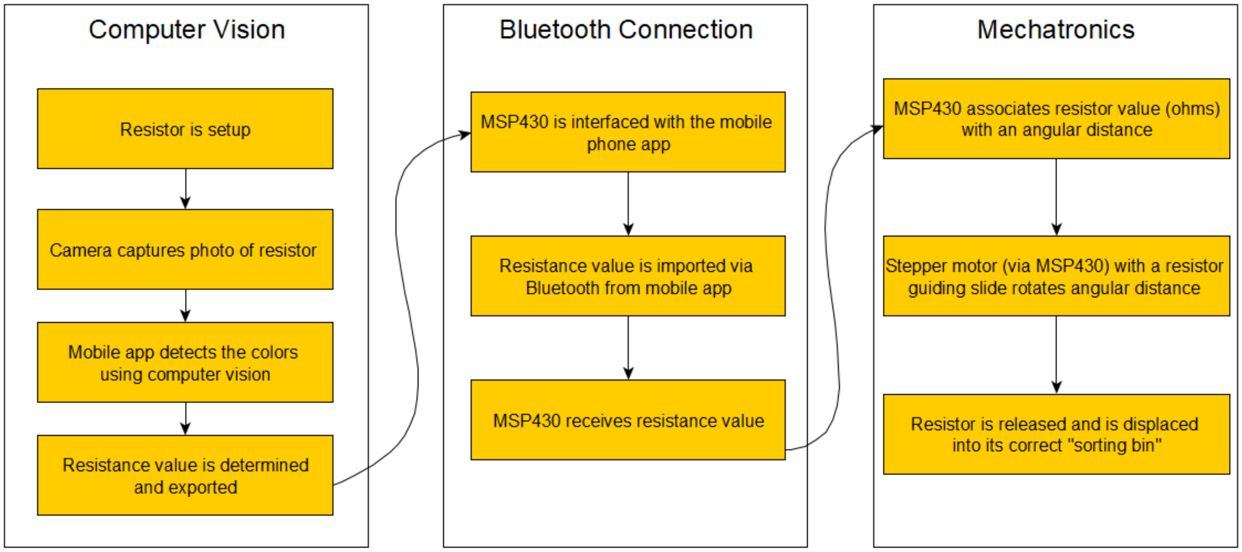


Figure 02 Submodule Block Diagram

The MSP430 code was written in C using Code Composer Studio and followed the algorithm outlined in Figure 03 below. There were four main actions that the MSP was responsible for handling. First, the code had to initialize all constant values and pin modes. This included the number of steps the stepper motors took for a full rotation - 4096 - as well as the pin numbers that the motors were connected to. These pins were 6-9 for motor one and 10-13 for motor two, all of which were set to output mode. The positions of the sorting bowls also had to be initialized at the beginning of the program, along with the resistor values that the user desired to be sorted. Once the initialization was complete, the program waited for a value to be passed through a Bluetooth connection. When a value was received, the MSP parsed it into an integer value which could then be used to calculate the shortest angular distance to the corresponding sorting bowl. If the received value was one that was to be sorted, the correct sorting bowl position was used to calculate the distance that the motor needed to turn. If the resistor was a reject, the program used the position of the reject bowl to calculate the number of steps to turn. This function is shown in Figure 04 below. The MSP would then turn the first stepper motor the calculated number of steps. After waiting 500ms, the pushResistor function, shown in Figure 05, would be called to turn the second stepper motor in order to push the resistor down the slide. The first stepper motor would then turn back to face position 0 after a 1 second delay in preparation for the next resistor value to be received over the Bluetooth connection.

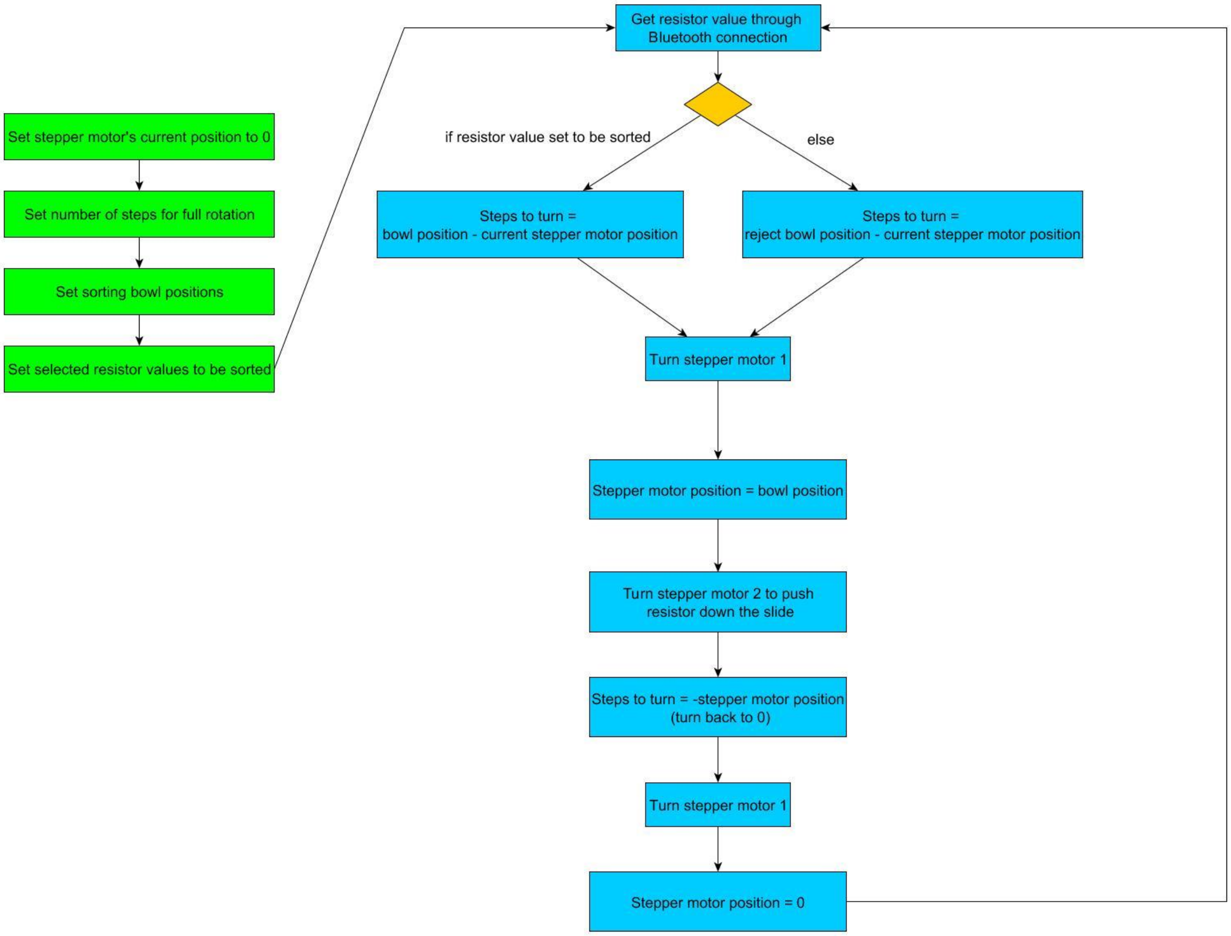
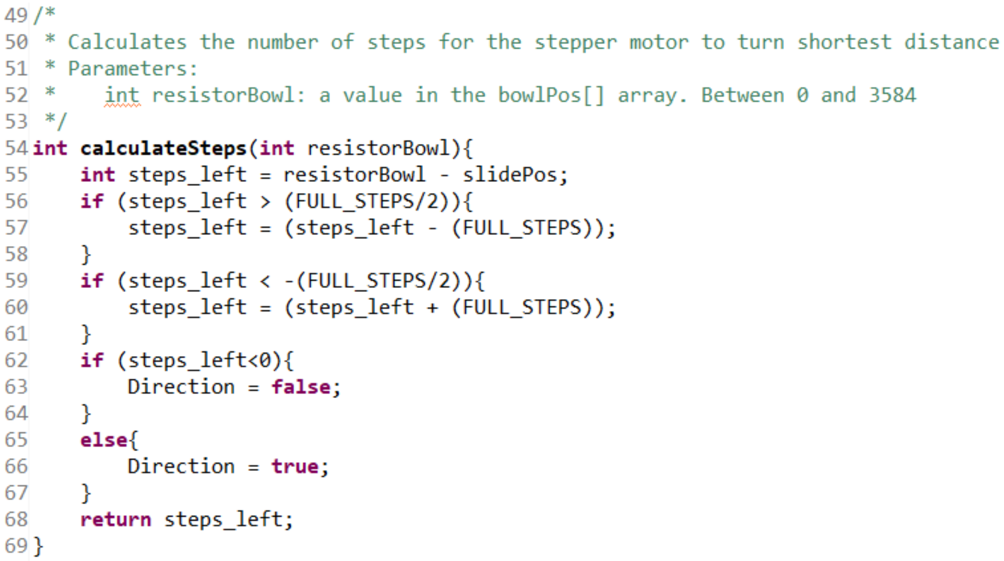
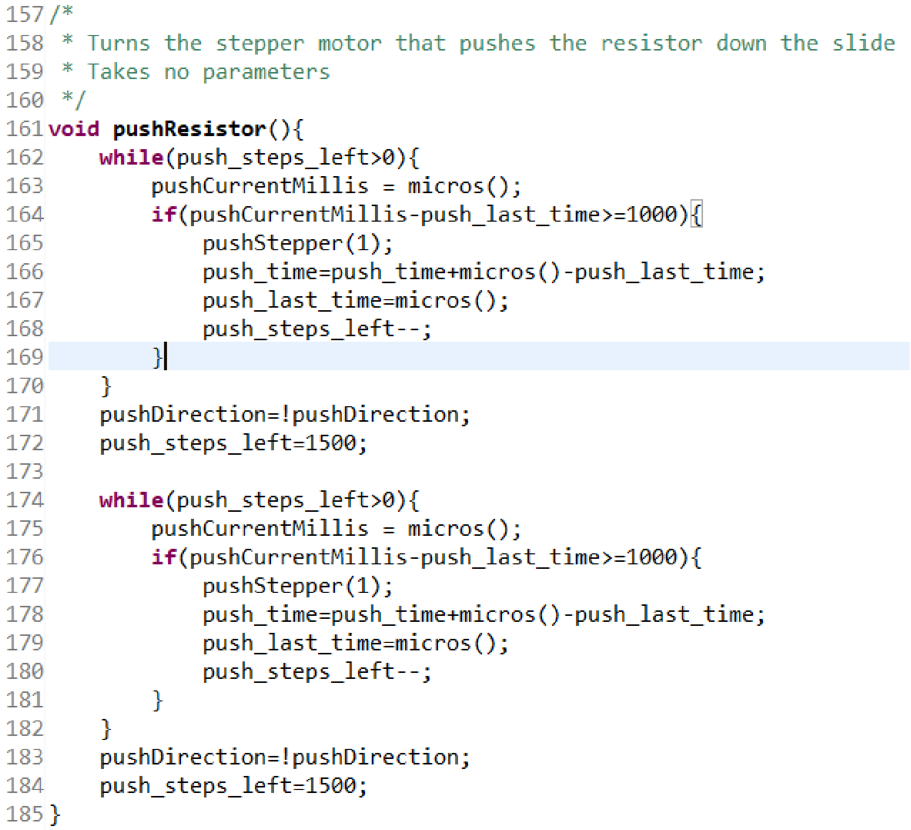


Figure 03 MSP430 Code Algorithm



```
/*
 * Calculates the number of steps for the stepper motor to turn shortest distance
 * Parameters:
 *    int resistorBowl: a value in the bowlPos[] array. Between 0 and 3584
 */
int calculateSteps(int resistorBowl){
    int steps_left = resistorBowl - slidePos;
    if (steps_left > (FULL_STEPS/2)){
        steps_left = (steps_left - (FULL_STEPS));
    }
    if (steps_left < -(FULL_STEPS/2)){
        steps_left = (steps_left + (FULL_STEPS));
    }
    if (steps_left<0){
        Direction = false;
    }
    else{
        Direction = true;
    }
    return steps_left;
}
```

Figure 04 MSP430 calculateSteps Function



```
/*
 * Turns the stepper motor that pushes the resistor down the slide
 * Takes no parameters
 */
void pushResistor(){
    while(push_steps_left>0){
        pushCurrentMillis = micros();
        if(pushCurrentMillis-push_last_time>=1000){
            pushStepper(1);
            push_time=push_time+micros()-push_last_time;
            push_last_time=micros();
            push_steps_left--;
        }
    }
    pushDirection=!pushDirection;
    push_steps_left=1500;

    while(push_steps_left>0){
        pushCurrentMillis = micros();
        if(pushCurrentMillis-push_last_time>=1000){
            pushStepper(1);
            push_time=push_time+micros()-push_last_time;
            push_last_time=micros();
            push_steps_left--;
        }
    }
    pushDirection=!pushDirection;
    push_steps_left=1500;
}
```

Figure 05 MSP430 pushResistor Function

The physical structure of the system was modeled using OnShape, a free online CAD software, and can be seen in full in Figure 06 below. There were four main parts to the physical structure: the base, the slide, the paddle, and the turntable.

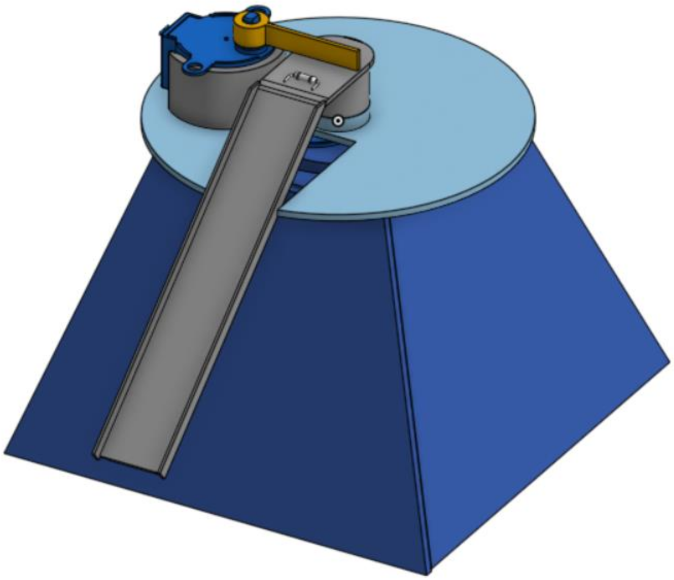


Figure 06 Full CAD Model

Each part played an important role in the final mechanical workings of the system. The base, which is the flat-topped pyramid structure and shown in Figure 07 below, was responsible for housing the PCB as well as holding the weight of all the other parts, including both stepper motors. The pyramid shape was chosen for one main reason; the slide had to be able to spin freely in any direction without impedance from the base structure. This meant the base had to fit completely under the turntable on the top while the bottom was wide enough to fit the full PCB and all wiring underneath. Because the PCB was rectangular, the team felt it made more sense to have a rectangular base than a cylindrical one. Another main design component of the base is the holes that appear on the top panel. The round hole with a rectangular back is for the first stepper motor to fit in to. Placing the stepper motor in this hole ensures it remains balanced even as it is spinning and that it is always in the correct position. The 'C' shaped hole is for the second stepper motor's wires. Because the second stepper motor is positioned outside of the base, its wiring had to have a way to connect to the PCB inside of the base. The 'C' shape allowed the wires to move as the motor turned without harming any of the connections. Finally, there is a hole on one side of the base for the battery clips to be accessible from outside of the base. This allows for ease of use when plugging and unplugging the system. The final design decision that was made for the base was to print each side separately. Not only did this allow for quicker 3D printing times, but allowed for only one side to be reprinted if an error occurred, as opposed to reprinting the entire base each time there was a mishap.

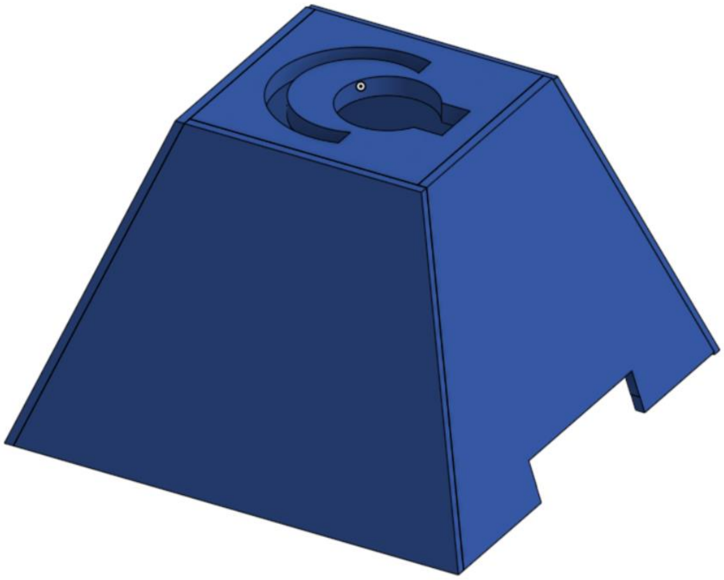


Figure 07 Base CAD Model

The next notable parts are the turntable and slide, shown in Figure 08 and 09 below. The turntable was designed with two main elements in mind: weight and size. The turntable had to be as light as possible so as not to impede the ability of the stepper motor to turn. Any excess weight on the motor leads to decreased performance - something that the team looked to avoid. However, the table also had to be large enough such that the second stepper motor could sit on it in order to push the resistor down the slide. To achieve these objectives, the table was made as slim as possible and just wide enough to fit the second stepper motor. There are a few other designed aspects to the table. First, the rectangular cutout is there so that the slide does not collide with the table. The hole in the center of the table is where the stepper motor connects, and the hole on the side is an indicator of where the second stepper motor should sit.

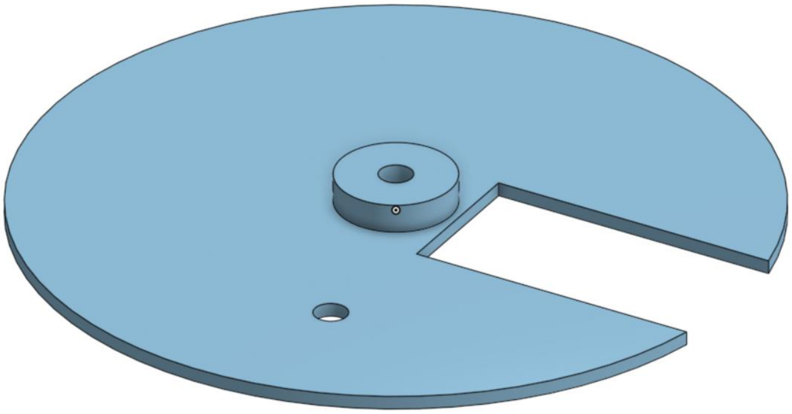


Figure 08 Turntable CAD Model

The slide was designed, again, to be as light as possible so as not to impede performance while being wide enough for a resistor to fit down the slide easily. Additionally, side guards were added to keep the resistor straight as it was falling down the slide. This helped to protect against resistors falling off of the slide before they reached the sorting bowl.

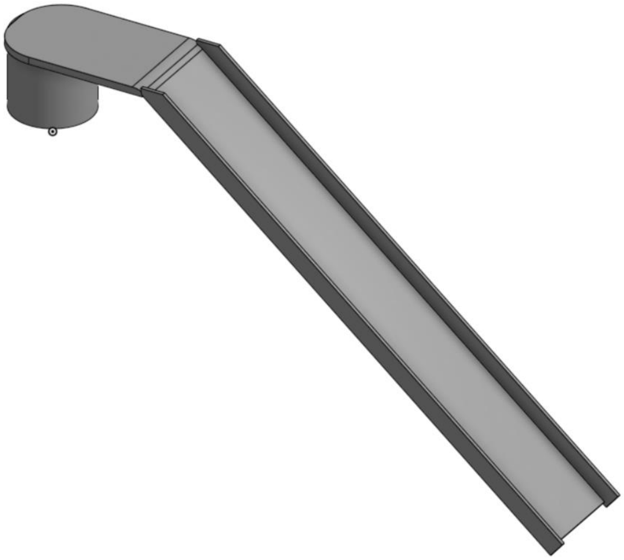


Figure 09 Slide CAD Model

Finally, the model included the paddle to knock resistors down the slide and a stepper motor holder for the second motor, shown in Figures 10 and 11 respectively. The main purpose of the paddle was to snap onto the second stepper motor and push the resistor down the slide. Because of this, a few design decisions were made. First, the hole seen in the cylindrical part of the paddle is where it connects to the motor. The rectangular part is what ultimately pushes the resistor. This part had to be long enough that it reached the entire length of the resistor so that it was not pushed at an angle.

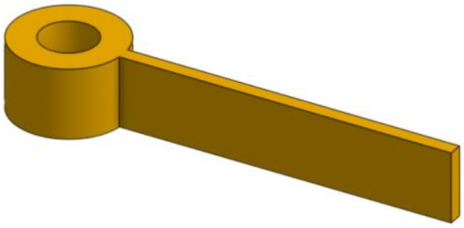


Figure 10 Paddle CAD Model

The stepper motor holder was added to ensure that the second stepper motor was always positioned correctly. Originally, the second motor was going to be glued to the turntable to keep it in place. However, if the motor were to burn out, it would be very difficult to replace with that design. Thus, the design decision to add a motor holder that would not only keep the motor in the right position but also allow the motor to be removed when necessary was made. This holder sits on the side of the turntable.

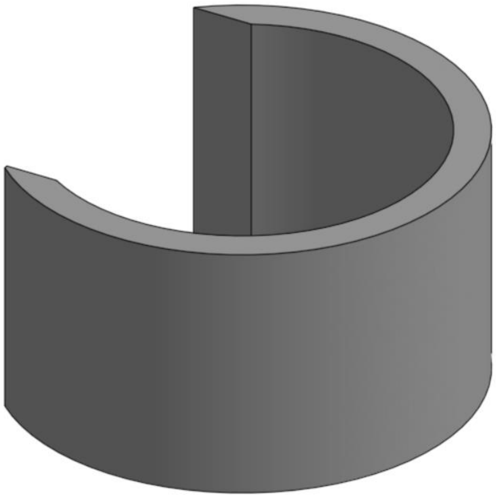


Figure 11 Motor Holder CAD Model

Figure 12 below displays the full Multisim Schematic of the electronics of the resistor sorter system.

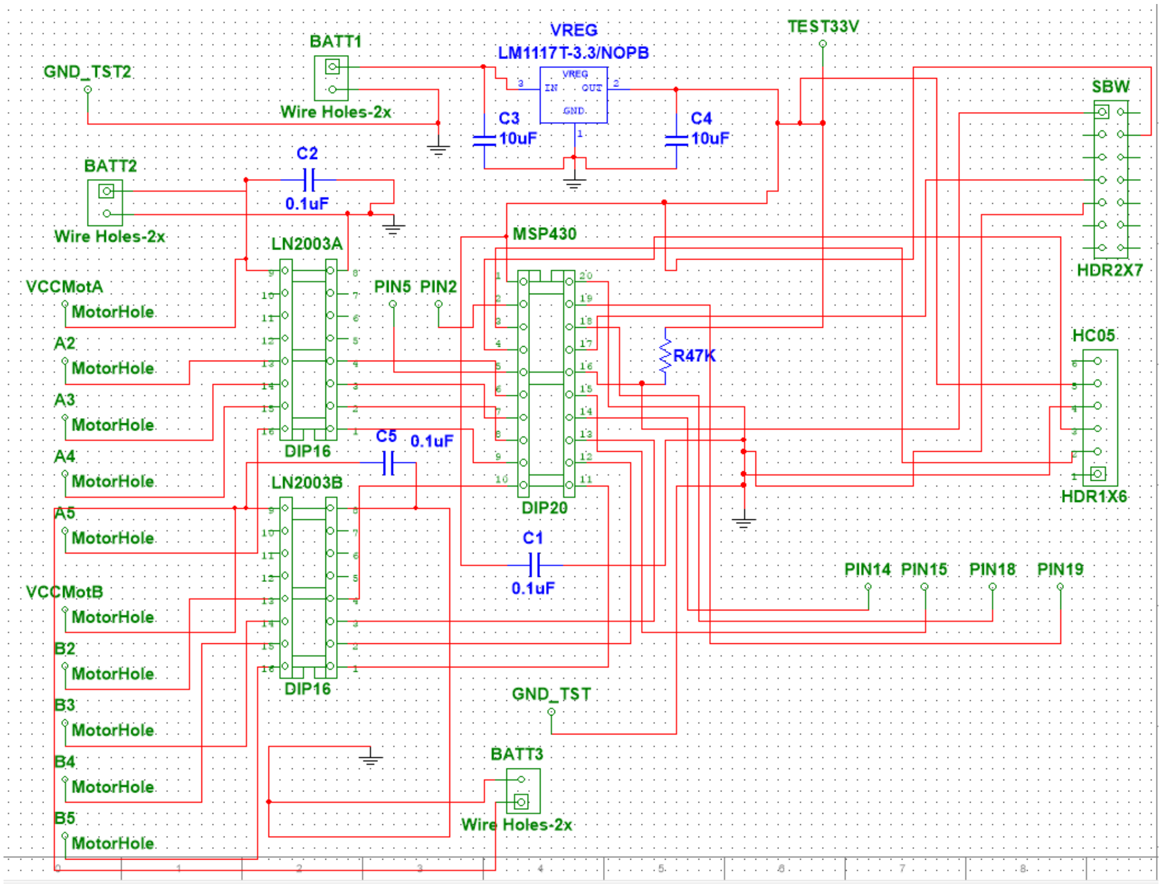


Figure 12 Full Multisim Schematic

The overview of the schematic details all of the major components and how they connect to one another. For example, the connections between the stepper motor drivers and the MSP430 and the HC05 Bluetooth module is apparent. Additionally, unused pins from the MSP430 were mapped onto the schematic and ultimately onto the PCB just in case they were needed in the future. A breakdown of each part of the board is shown below.

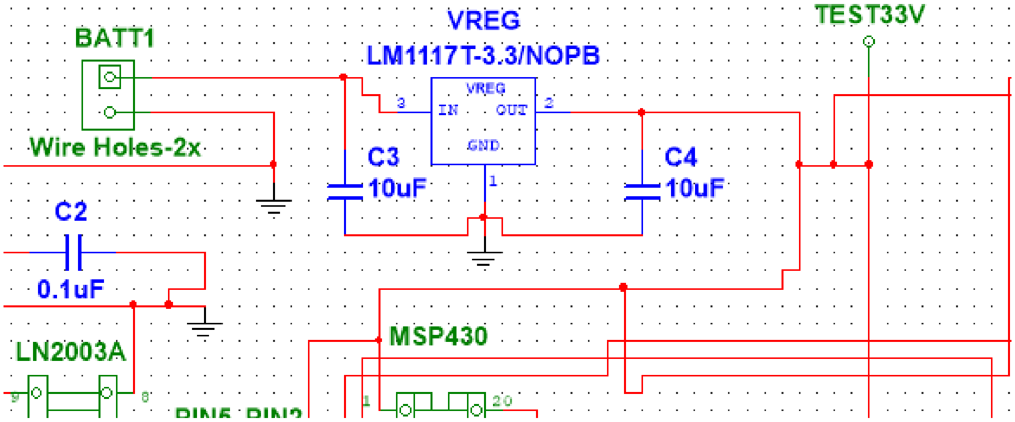


Figure 13 MSP430 Power Schematic

Above in Figure 13, the first battery (BATT1) is shown connected to a LM117T-3.3 voltage regulator. This regulator steps the 9V battery voltage down to 3.3 V for the MSP430 and the HC-05 Bluetooth module to operate. Capacitors C3 and C4 are present as bypass, decoupling capacitors for the regulator. These help ensure that the output is stabilized and consistent. Finally, as shown in the 3.3 V voltage regulator in the figure above, pin 1 is connected to ground, pin 2 is the output voltage (3.3 V) that ultimately connects to pin 1 of the MSP430 (VCC) and the TEST33V pin, and pin 3 is the input voltage (9 volts from the battery).

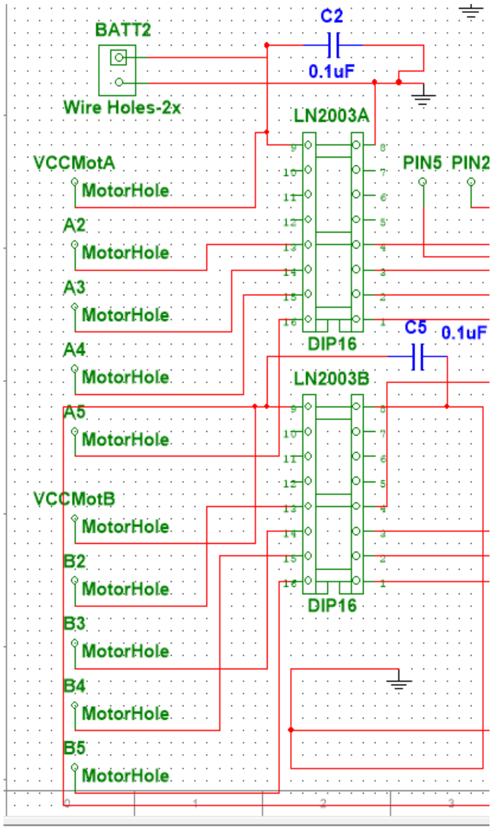


Figure 14 Stepper Motor Drivers Schematic

The figure above (Figure 14) illustrates the schematic for the stepper motor drivers. Since the final, physical 3D design utilizes a turntable and a lever/paddle that pushes each resistor down into its respective sorting bowl, two stepper motors are needed (and hence two stepper motor drivers). In the figure above, LN2003A represents the first stepper motor driver and LN2003B represents the second stepper motor driver. The connections made in the schematic were based solely on the datasheets of the component and both of the two stepper motor drivers are connected in the same way as the other.

As shown in the figure above (Figure 14), pin 9 of each of the stepper motor drivers is connected to the power supply. There is a small caveat to this connection, however. The stepper motor drivers are rated for 5 volts input voltage. The figure above and the schematic that produced the PCB left out a 5 volt, voltage regulator for the input voltage to the stepper motor driver. This was simply an oversight and was ultimately corrected for the final implementation of the project but is not seen in the figure above. Next, pin 8 of each driver was the ground pin connection, per the data sheet for the component. Finally, the remaining four wires from the five wire stepper motor are connected to pins 13,14,15,and 16. The respective connections to the MSP430 are on pins 1,2,3, and 4 of each stepper motor driver. The remaining pins on the LN2003 stepper motor drivers are unused. Finally, two bypass capacitors are used (one for each driver) in order to stabilize the input voltage with a constant value and prevent the concern of failure of the device.

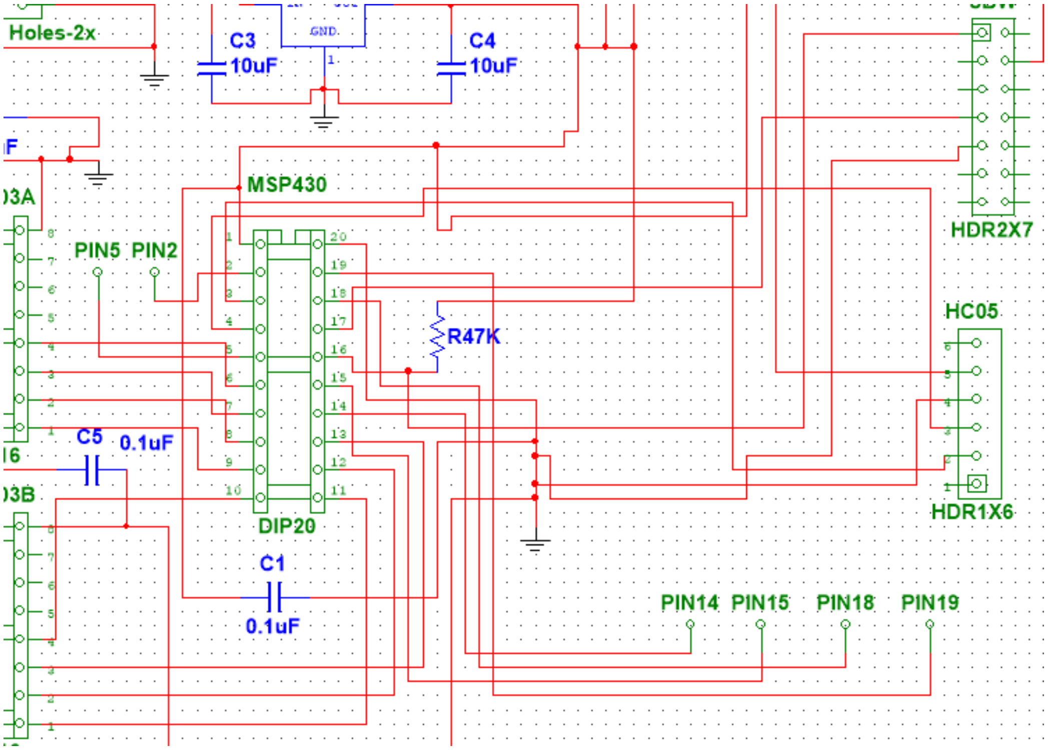
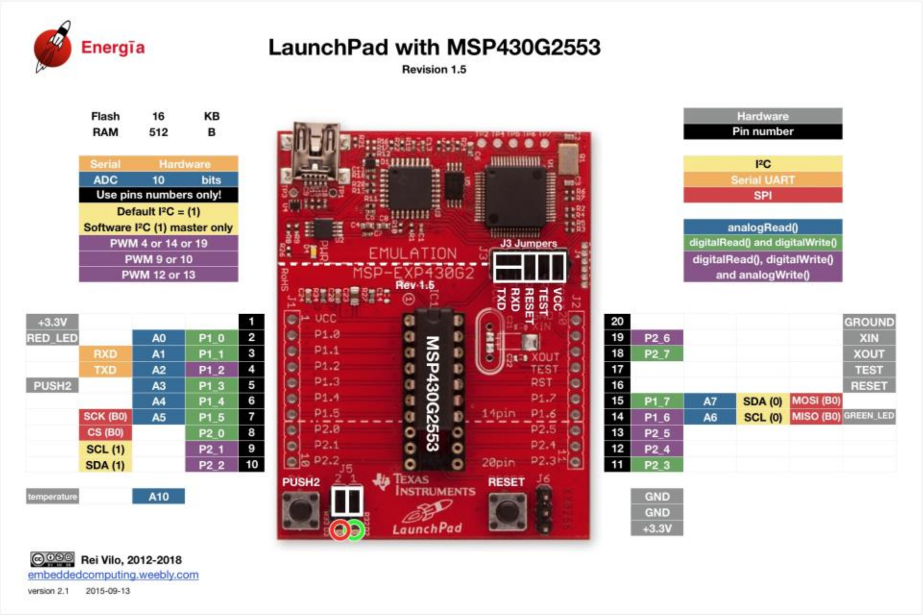


Figure 15 MSP430, HC05 Schematic



LaunchPad with MSP430G2553

Revision 1.5

Flash	16	KB
RAM	512	B

Serial	Hardware	
ADC	10	bits

Use pins numbers only!

Default I²C = (1)

Software I²C (1) master only

PWM 4 or 14 or 19

PWM 9 or 10

PWM 12 or 13

+3.3V				1
RED_LED		A0	P1_0	2
	RXD	A1	P1_1	3
	TXD	A2	P1_2	4
PUSH2		A3	P1_3	5
		A4	P1_4	6
	SCK (B0)	A5	P1_5	7
	CS (B0)		P2_0	8
	SCL (1)		P2_1	9
	SDA (1)		P2_2	10
temperature		A10		

Hardware

Pin number

I²C

Serial UART

SPI

analogRead()

digitalRead() and digitalWrite()

digitalRead(), digitalWrite() and analogWrite()

20					GROUND
19	P2_6				XIN
18	P2_7				XOUT
17					TEST
16					RESET
15	P1_7	A7	SDA (0)	MOSI (B0)	
14	P1_6	A6	SCL (0)	MISO (B0)	GREEN_LED
13	P2_5				
12	P2_4				
11	P2_3				
	GND				
	GND				
	+3.3V				

Rei Vilo, 2012-2018
embeddedcomputing.weebly.com
version 2.1 2015-09-13

Figure 16 MSP430G2553 Pin Layout [24]

Finally, the main logic controller, the MSP430G2553 schematic is shown above (Figure 15). Additionally, the pin layout of the component with descriptions of each pin is shown for reference (Figure 16). First, pin 1 of the MSP430 is connected to 3.3V output from the voltage regulator. Pin 2 of the MSP430 is unused and is outputted to the board as a backup test pin. Pins 3 and 4 are used for the RXD and TXD. These pins connect to the TXD and RXD pins of the HC-05 Bluetooth module respectively. The TXD pin of the HC-05 connects to the RXD pin of the MSP430 (pin 3) and the RXD pin of the HC-05 connects to the TXD pin of the MSP430 (pin 4). The pin layout for the HC-05 is shown below for reference in Figure 17. Pin 5 of the MSP430 was also unused and outputted to the board as a backup test pin. Pins 6,7,8, and 9 are used as general purpose pins and connect the first LN2003 stepper motor driver to the board. These pins are all physically next to each other and were simply chosen as the pins to be used for organizational purposes. Likewise, pins 10,11,12, and 13 of the MSP430 were connected to the second LN2003 stepper motor driver. Pins 14 and 15 were unused in the project (in the same regard as pin 2 and pin 5). Pin 16 is the pin used for the Spy-Bi-Wire (SBW) connector on the PCB. This connection allows the MSP430 to be programmed, flashed, and debugged without the use of a launchpad. Pin 16 first connects to a 47k ohm pull up resistor that is connected to the voltage supply and pin of the SBW connector itself. Pin 17 is connected to to SBW connector as well as the “Test” pin. Essentially, a “2 wire JTAG” connection is used for the SBW connector from the MSP430. More information on the layout and schematic of this can be seen below in Figure 19. Finally, pins 18 and 19 were unused and pin 20 was connected to ground.

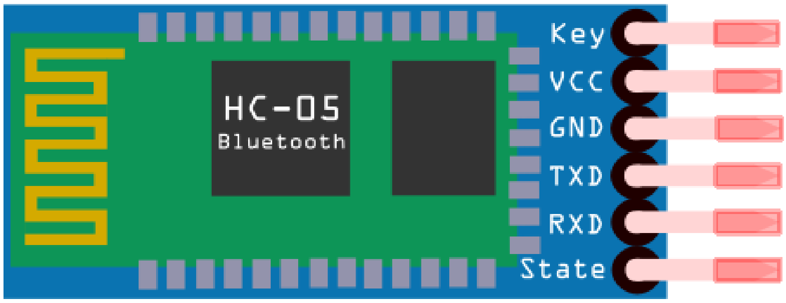


Figure 17 Pin Layout of HC-05 Module [25]

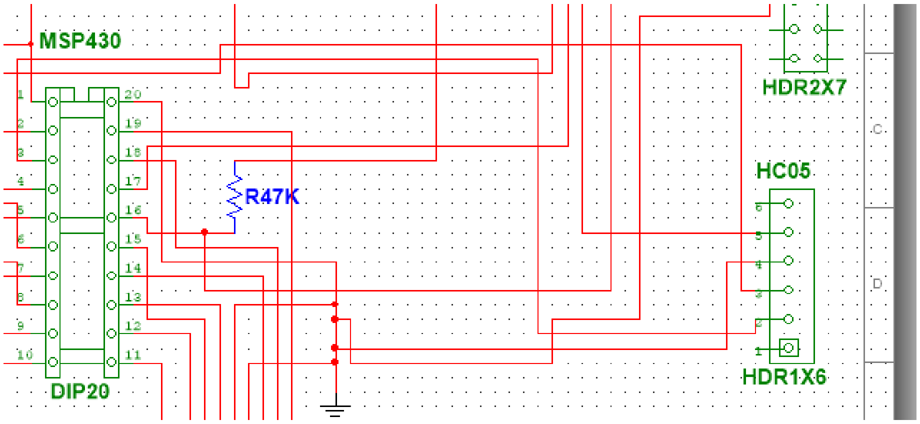
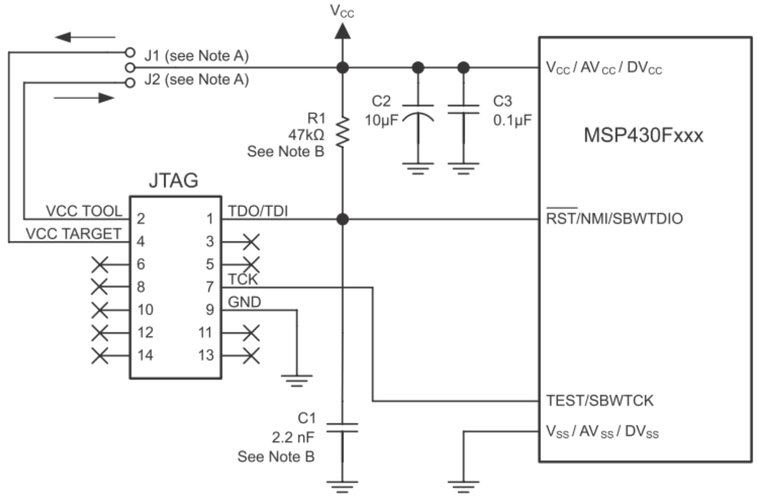


Figure 18 Schematic of HC-05 Module

The HC05 Bluetooth module has 6 total output pins. The first and last pins on the module are Key and State pins, which are unused in the operation of this project. The VCC pin connects to the same VCC as the MSP430 (3.3 volts). The ground pin connects to the main ground connection on the PCB, and the RXD and TXD pins connect to the TXD and RXD pins on the MSP430 as mentioned previously.



A Make connection J1 if a local target power supply is used, or make connection J2 if the target is powered from the debug or programming adapter.

B The device $\overline{RST}$/NMI/SBWTDIO pin is used in 2-wire mode for bidirectional communication with the device during JTAG access, and any capacitance that is attached to this signal may affect the ability to establish a connection with the device. The upper limit for C1 is 2.2 nF when using current TI tools.

Figure 19 Spy-Bi-Wire Schematic [26]

Finally, the last major component on the PCB is the Spy-Bi-Wire connector. This connector is a 14 pin socket that allows the TI JTAG USB interface to debug the embedded code on the MSP430 without a Launchpad. In this case, pin 1 of the connector was connected to the reset pin of the MSP430, pin 4 was connected to the 3.3 V VCC voltage source, pin 7 was connected to the TEST/SBWTCK pin of the MSP430, and pin 9 was connected to ground. The connections were based on the schematic in Figure 19 above.

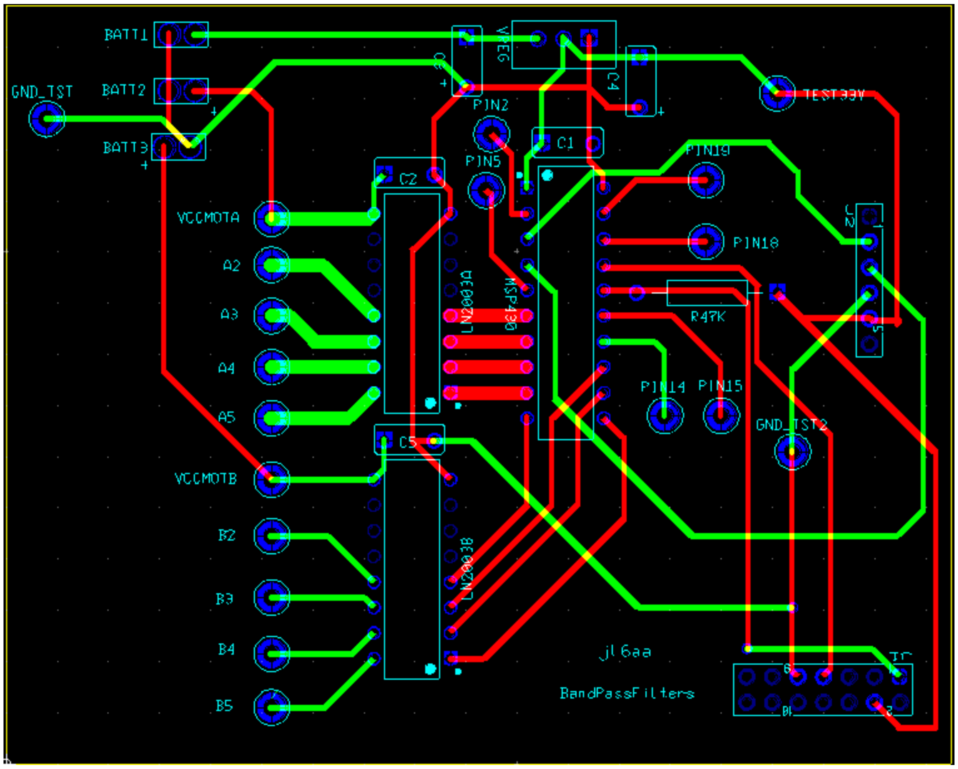


Figure 20 Ultiboard PCB Layout

Shown above in Figure 20 is the full PCB layout produced in National Instruments Ultiboard.

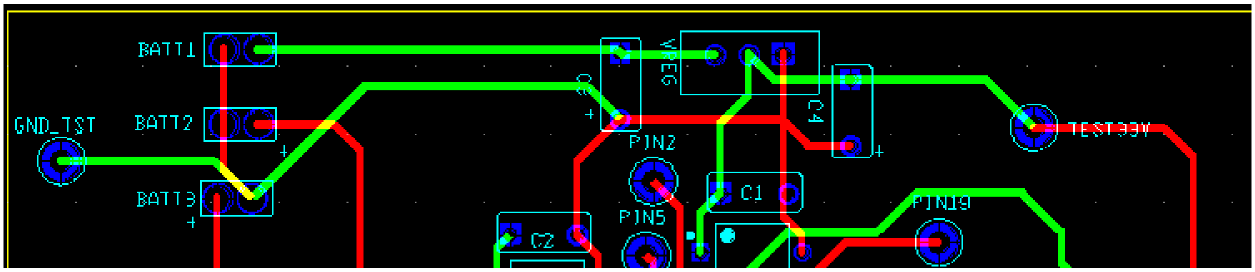


Figure 21 Power Section of PCB Layout

Figure 21 above details the specific layout of the power section of the PCB layout. On the left side of the above figure, the three batteries connections are visible. These were placed adjacent to each other primarily for convenience purposes. In the center of Figure 21 above, the 3.3 volt voltage regulator is seen with two capacitors on either side of it. These act as bypass/decoupling capacitors, the close proximity of these to the voltage regulator itself allow for better, more stable performance of the regulator.

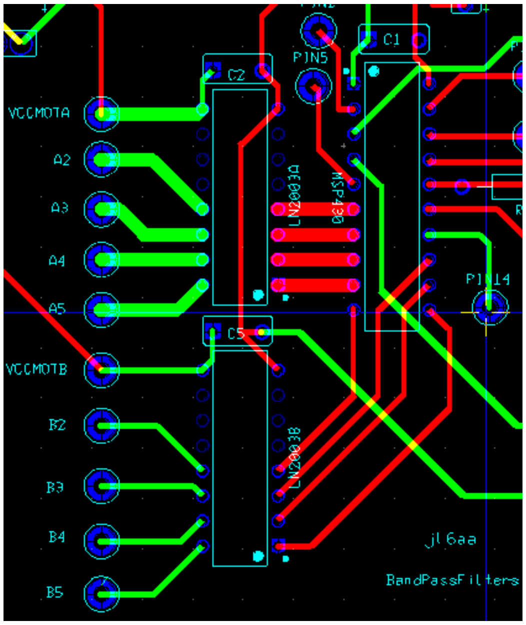


Figure 22 Motor Drivers and MSP430 PCB Layout

Figure 22 above shows the most detailed portion of the PCB layout. This section contains the connections between the stepper motor drivers and the MSP430. The connections are made based on the Multisim schematic and the copper tracks on the board were made as wide as possible given the space constraints. Additionally, C1, C2, and C5 are bypass capacitors placed as close as possible to their respective ICs in order to stabilize their operation.

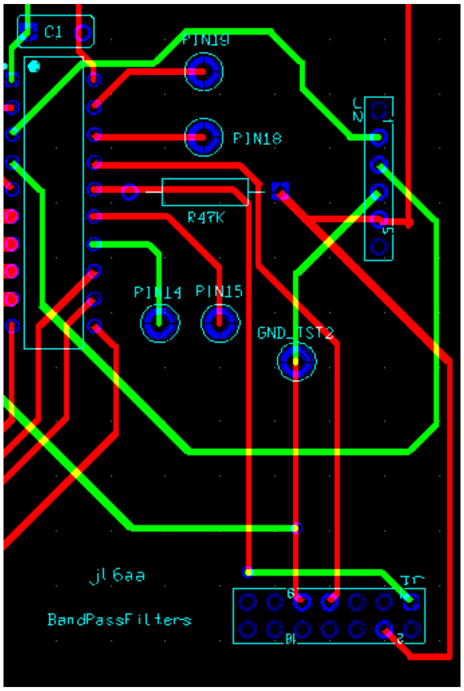


Figure 23 Spy-Bi-Wre and HC05 PCB Layout

The final section of the PCB layout contains the Spy-Bi-Wire connector and the HC-05 module. The connections made for these components are not necessarily unique, but it should be noted that through hole vias were used with the connections near the Spy-Bi-Wire connector (located at the bottom right corner of Figure 23 above). These allow for an easier mapping of the copper tracks while avoiding conflicts with overlapping tracks. Finally, the placement of the Spy-Bi-Wire connector itself is significant to some degree as it is located far from the other components. This ultimately allows for the connection to be done easily without other components obstructing the area.

The last subsection of the project to be discussed is the Computer Vision application. As mentioned above several times, the team found this open-source Android application code on the Internet when searching for ideas, tips, and inspiration for our capstone project. Once the team persused it, it was determined that the source code was primarily written in Java, and the app utilized the OpenCV (CV being the abbreviation for "Computer Vision") library in order to use the camera to "read" the resistor value. Though none of the team members had experience in mobile application programming, the team decided using this application with its strong base code was the best idea.

Once the application code was downloaded from the GitHub and unzipped, it was time to decide which IDE would be used to develop the application further. Originally, Visual Studio Code was picked due to a team member's suggestion. After looking into it further, though, it was discovered that a lot of the Android application specific functions (building, deploying) were much easier in an IDE like Android Studio, so the IDE choice was swapped to Android Studio.

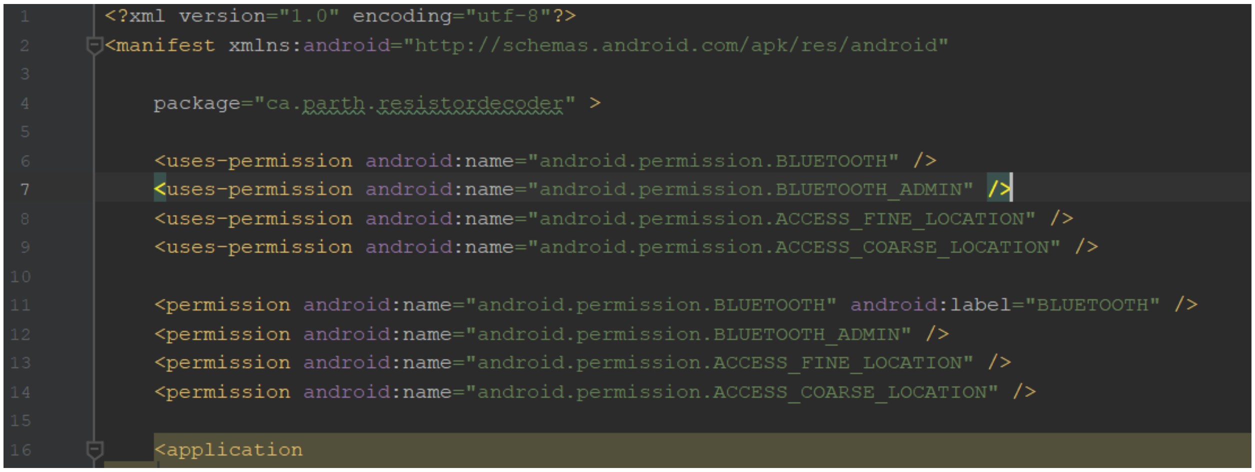
Once the project was being developed in Android Studio, it was time to try and deploy the application on an Android device. A team member provided an older model Android phone used previously by that team member (at no cost) to the project. Once the phone was plugged into the computer via USB, some settings had to be checked/adjusted in the Devices Manager of the computer to allow for upload and debugging on the device. Additionally, on the phone itself, some settings had to be checked/adjusted in order for the phone to allow an application to be uploaded, and for USB debugging to be enabled. These steps were quite easy to execute, and information on how to do this was readily available on the Internet.

After setting up the device, it can be seen under the "Devices" drop-down in Android Studio on the toolbar next to the "Run" button. Additionally, a configuration must be added for the application so that the application can be deployed on the selected device. Once a configuration is added, it will be seen under the "Configurations" drop-down in Android Studio on the toolbar next to the "Devices" drop-down.

Once the devices were all set up, some work had to be done with the gradle files in order to get the correct build behavior. This required lots of research and troubleshooting, as the team member programming had no experience with mobile application deployment. After some work debugging and updating, the application was finally able to be successfully loaded onto an Android device. The team additionally loaded the application onto another team member's newer phone. These devices (running the applications) were tested preliminarily to determine the accuracy of the resistor scanner. While not 100% accurate, the application was somewhat reliable, and the team made plans to tinker with the recognition logic to better these results.

The application uses OpenCV library to “read” the resistors. Scalars of the color bounds for each color possibly found in a resistor are set up. The image is captured through the phone’s camera (in real-time). Contours of the color bands are found, as well as the x-coordinates of their centroids. From there, some more complicated logic takes these centroids (color, x-coordinate) and translates this information into a resistor value. The value is then displayed on the screen of the phone.

The next big hurdle was implementing Bluetooth on in this application. Lots of preliminary research was done on how to best implement Bluetooth with an Android mobile application. After much delaying, due to the delay in the arrival of our HC-05 module, work was finally started on the Bluetooth capabilities. First, Bluetooth permissions must be declared in the application’s manifest file. These permissions should be declared outside the <application section like so:



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"

    package="ca.parth.resistordecoder" >

    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

    <permission android:name="android.permission.BLUETOOTH" android:label="BLUETOOTH" />
    <permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

    <application
```

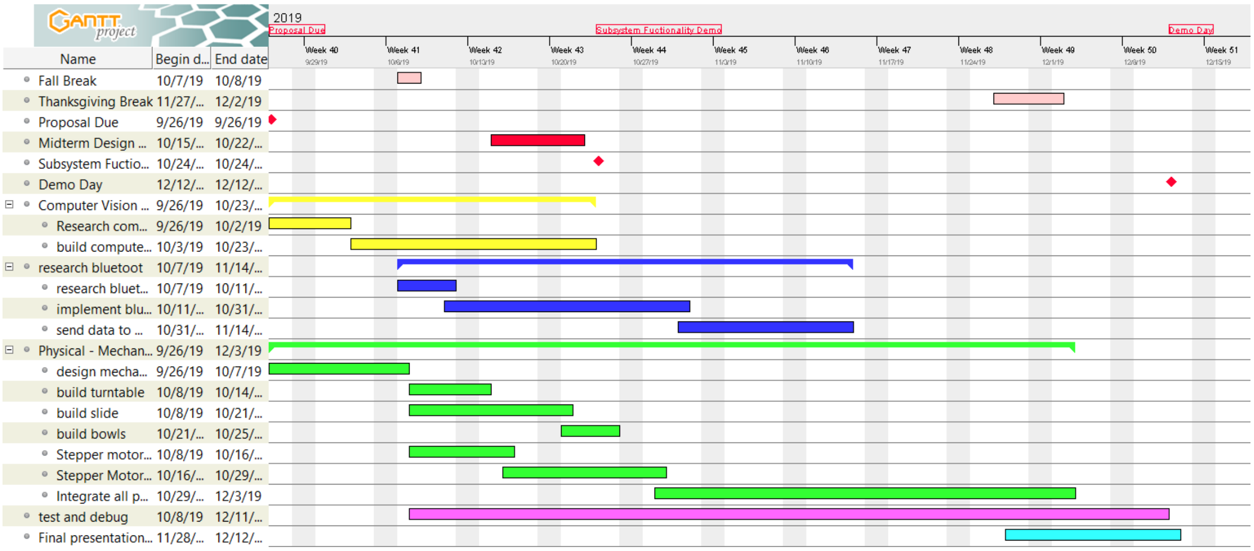
Figure 24: Android App Manifest File

Then, the actual Bluetooth code was started. First a Bluetooth Socket must be acquired, and a UUID constructed. The UUID is passed into the function createRfcommSocketToServiceRecord() argument that is called on a paired device in order to acquire this socket. Then, connect() must be called on the socket. Finally, an OutputStream() must be set up on the socket. Once this is done, the method write() should be able to write a byte [], and these values should be sent via a Bluetooth connection (via the socket) to the paired device. The resistor value was converted from an integer to a byte [] and this is the value that was passed to write().

As mentioned previously, the ability to send a value over Bluetooth was not achieved. Lots of time was spent adjusting, debugging, and paring down the code in an effort to get the value sent. First, many errors had to be fixed by importing some modules that the team was previously unaware of and by fixing example code snippets found in Internet Bluetooth tutorials. Code that allowed for the application to pair with a device inside the application itself was thrown-out in favor of connecting in the settings app before launch in order to get rid of unnecessary code that may have been causing a problem. Originally, the Bluetooth connection was set up as a server. This caused the application to crash when it was opened. When this was not working, the Bluetooth connection was set up as a client. This provided some headway, and the application crashed later into the process. Eventually, through much debugging, the application got to the point where it would scan and show a value on an object it thought was a resistor (though it was often a false positive) and would not crash. However, no value was sent, and errors/warnings continued to show-up at runtime in debug mode.

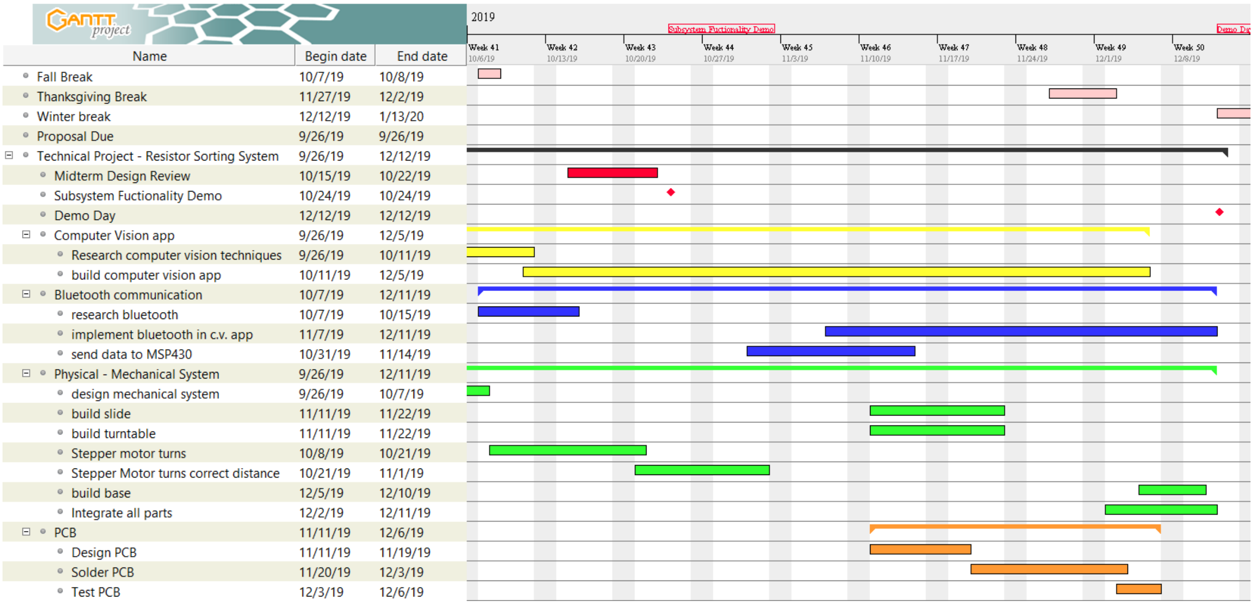
Project Time Line

The proposed timeline for the resistor sorting system can be seen in Figure 25 below. As opposed to our original timeline, our final timeline, seen in Figure 26 included extended time to build and debug the computer vision application and bluetooth communication. Additionally, building the physical system was pushed back from its originally planned date to allow for extra time to work on debugging the computer vision app, printed circuit board, and MSP430 code. Complications arose with the computer vision application and sending values over Bluetooth, which brought about a need to allow extra time for these tasks and, thus, less time for others. The final timeline also includes time spent designing, soldering, and testing the printed circuit board - tasks that were forgotten in the originally proposed timeline.



Name	Begin d...	End date
Fall Break	10/7/19	10/8/19
Thanksgiving Break	11/27/...	12/2/19
Proposal Due	9/26/19	9/26/19
Midterm Design ...	10/15/...	10/22/...
Subsystem Fuctio...	10/24/...	10/24/...
Demo Day	12/12/...	12/12/...
⊟ Computer Vision ...	9/26/19	10/23/...
Research com...	9/26/19	10/2/19
build compute...	10/3/19	10/23/...
⊟ research bluetoot	10/7/19	11/14/...
research bluet...	10/7/19	10/11/...
implement blu...	10/11/...	10/31/...
send data to ...	10/31/...	11/14/...
⊟ Physical - Mechan...	9/26/19	12/3/19
design mecha...	9/26/19	10/7/19
build turntable	10/8/19	10/14/...
build slide	10/8/19	10/21/...
build bowls	10/21/...	10/25/...
Stepper motor...	10/8/19	10/16/...
Stepper Motor...	10/16/...	10/29/...
Integrate all p...	10/29/...	12/3/19
test and debug	10/8/19	12/11/...
Final presentation...	11/28/...	12/12/...

Figure 25 Proposed Gantt Chart Timeline



Name	Begin date	End date
Fall Break	10/7/19	10/8/19
Thanksgiving Break	11/27/19	12/2/19
Winter break	12/12/19	1/13/20
Proposal Due	9/26/19	9/26/19
Technical Project - Resistor Sorting System	9/26/19	12/12/19
Midterm Design Review	10/15/19	10/22/19
Subsystem Fuctionality Demo	10/24/19	10/24/19
Demo Day	12/12/19	12/12/19
Computer Vision app	9/26/19	12/5/19
Research computer vision techniques	9/26/19	10/11/19
build computer vision app	10/11/19	12/5/19
Bluetooth communication	10/7/19	12/11/19
research bluetooth	10/7/19	10/15/19
implement bluetooth in c.v. app	11/7/19	12/11/19
send data to MSP430	10/31/19	11/14/19
Physical - Mechanical System	9/26/19	12/11/19
design mechanical system	9/26/19	10/7/19
build slide	11/11/19	11/22/19
build turntable	11/11/19	11/22/19
Stepper motor turns	10/8/19	10/21/19
Stepper Motor turns correct distance	10/21/19	11/1/19
build base	12/5/19	12/10/19
Integrate all parts	12/2/19	12/11/19
PCB	11/11/19	12/6/19
Design PCB	11/11/19	11/19/19
Solder PCB	11/20/19	12/3/19
Test PCB	12/3/19	12/6/19

Figure 26 Final Gantt Chart Timeline

For the most part, tasks were able to be completed in parallel. For example, work was done on the MSP430 and stepper motor code while work on the computer vision application was being done simultaneously. Bluetooth communication, the physical structure, and the printed circuit board were also worked on in parallel. However, some tasks needed to be done in serial. These tasks included designing, soldering, and then testing the PCB. In this case, each step had to be completed before the next could begin. Additionally, the slide and turntable were built before the base so that precise measurements could be taken to ensure that, when the base was 3D printed, it would fit as expected.

When designing, building, and testing the resistor sorting system, each team member had primary and secondary roles. The primary role for Robyn Guarriello was writing the MSP430 stepper motor code and designing the physical structure while her secondary role was working with the Bluetooth communication. Joseph Laux's primary role was designing, soldering, and testing the PCB and his secondary role was designing the physical structure. Finally, Kiri Nicholson's primary role was working with the computer vision application and sending values through Bluetooth communication.

Test Plan

By design, the resistor sorting system was divided into multiple testable sub modules. Each different part of the system could be tested and debugged individually, outlined in the proposed test plan in Figure 27 below. The sub modules included the MSP430 interfacing with two stepper motors and receiving Bluetooth communications, the computer vision application reading resistor values, and the computer vision application sending values over Bluetooth. First, the MSP430 was testable by itself by first coding in an array of resistor values to sort to ensure that it was able to turn the stepper motors the correct angular distance and push the resistor down the slide at the correct time. Once this was tested and confirmed to work, a Bluetooth terminal application was able to connect to the HC-05 module and send a value to trigger the stepper motors to turn. These steps were first done using a breadboard and Launchpad and then repeated using the soldered PCB that was designed for the system. Secondly, the computer vision application was tested for reading values of resistors and then tested for sending the value over Bluetooth.

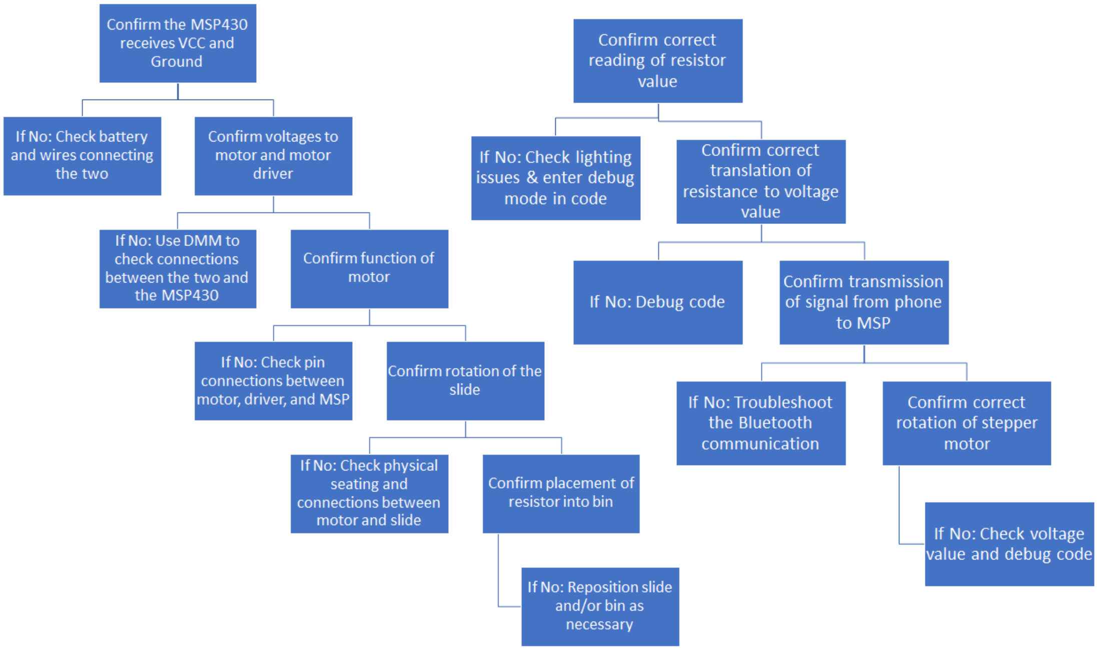
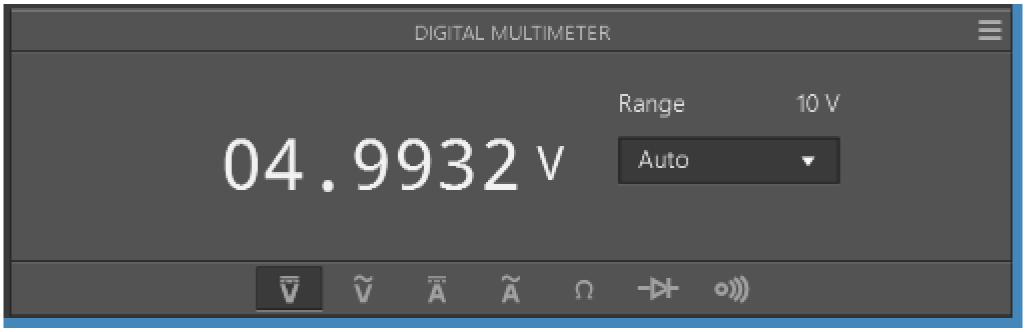


Figure 27 Proposed Test Plan

For the most part, the proposed test plan was followed as intended. The first aspect of the system that was tested was connections to VCC and ground on the PCB. There were three main points that needed to be tested. The first two VCC points that were checked were for the two stepper motors. Each stepper motor required a 5V input and, as seen in Figure 28 and 29 below, each stepper motor was successfully wired to get a 5V input [27][12]. The MSP430 required a 3.3V input so, likewise, the voltage input to the MSP was then confirmed to be 3.3V, as shown in Figure 30 [28][14].

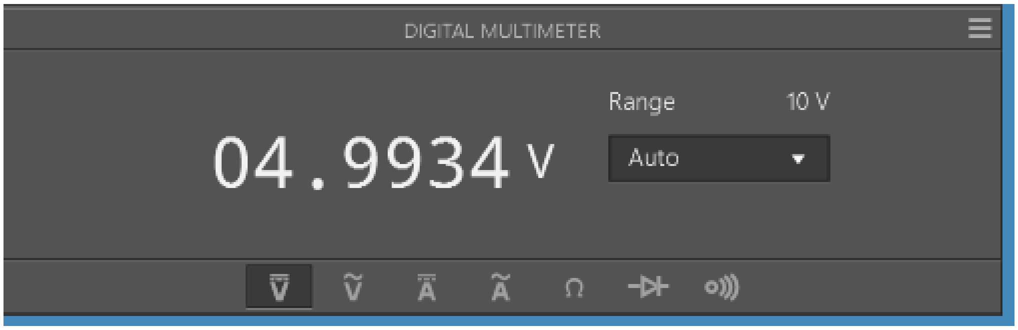


04.9932 V

Range 10 V

Auto

Figure 28 Stepper Motor 1 VCC

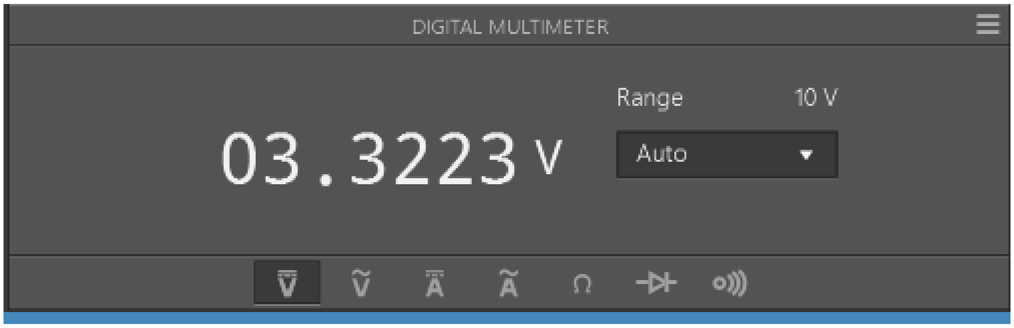


04.9934 V

Range 10 V

Auto

Figure 29 Stepper Motor 2 VCC



03.3223 V

Range 10 V

Auto

Figure 30 MSP430 VCC

Once the voltages were tested and confirmed to be correct, the performance of the code flashed to the MSP430 was tested. The 3D printed turntable, slide, and the second stepper motor were all placed on the first stepper motor and a resistor value was hard coded to ensure that the motor could handle the necessary weight and turn the precise distance necessary. This process was done for each bowl position to ensure that the angular rotation could be handled by the system. When testing this portion of the system, it was found that a 180 degree turn put too much stress on the wired connections. Because of this, a slight redesign was required. Instead of sorting seven resistor values with one reject bowl, the system would sort six resistor values and have one reject bowl. This allowed the system to avoid a 180 degree turn and increase in reliability.

Next, the second stepper motor was tested. Once the first stepper motor turned the structure to face the correct sorting bowl, the second stepper motor was to turn a paddle that would push the resistor down a slide and into the bowl. To test this, the same values used in the previous testing step were again hard coded. A resistor was placed at the top of the slide and once the system turned to face the correct bowl, the second stepper motor successfully pushed the resistor down the slide. This step worked very reliably and so no redesign was necessary.

The next subsystem that was tested was receiving a value over a Bluetooth connection. To test this, a Bluetooth terminal Android application was downloaded. This app allowed the team to send a value to the HC-05 module with ease. Once connected to the HC-05 module, a value was sent through the Bluetooth terminal application to the MSP430. This replaced the hardcoded values used in the previous steps. Once the value was sent, the system turned and pushed the resistor down the slide and into the correct sorting bowl just as it had been tested and confirmed to do previously. This confirmed that the system was able to establish a Bluetooth connection and receive any value sent over that connection.

The team was also able to test the computer vision application while the MSP430 testing was being completed, as they were separate sub modules. The first testing that occurred with the application was the accurate reading of resistor bands. When first testing this portion of the app, the team found that it was able to read bands relatively reliably. However, later in the semester, the app was tested in different ambient lighting due to the time of day. It was found that the app worked well in late-morning to early afternoon on a partly cloudy day, but was less reliable on cloudy days and during the nighttime. To mitigate this, different colored backgrounds were tested to see which would result in the most reliable reading. It was ultimately found that the color of the tables in the NI Lab (light gray, less shiny) gave the best results. This led to a slight redesign where a resistor would have to be read while laying on the table rather than when it is already placed on the slide.

The next part of the computer vision application that was tested was the Bluetooth communication. The first aspect of the Bluetooth communication was establishing a connection with the HC-05 module via the traditional settings app on the phone. The application was successful in creating this connection - which was seen by the blinking light on the HC-05 module. When the module is not connected, it would blink at a rapid pace. Once a connection was established, the light would blink at a slower frequency [29]. The next testing step was attempting to send a value from the app to the HC-05 over this connection. The team spent many hours attempting to fix this bug but, in the end, was unsuccessful. Thus, a final redesign was necessary. Instead of the application automatically sending the resistor value over Bluetooth, the resistor value was simply displayed on the screen and was manually sent to the HC-05 module using the previously mentioned downloaded Bluetooth terminal application.

Final Results

At the conclusion of the semester, the resistor sorting system was able to read the colored bands on a resistor using a computer vision Android application with some reliability. Though the application was able to connect to the HC-05 Bluetooth module, it could not successfully send a resistance value over the Bluetooth connection. However, the HC-05 Bluetooth module interfaced with the MSP430 was successful in receiving a value using a downloaded Bluetooth Terminal application, decoding it, and using it to compute the shortest angular distance in terms of steps that the stepper motor had to turn. The MSP was able to successfully turn the first stepper motor the correct angular distance and then turn the second stepper motor to move the paddle that pushed the resistor down the slide and into the bowl. The first motor would then turn back to position zero in preparation for the next resistor.

In the proposal for the project, four success criteria were outlined. These deliverables included a mobile app that is able to detect and calculate the resistance value of a resistor placed in front of the phone's camera, the ability to communicate the resistor value through a Bluetooth connection between the mobile application and the HC-05 module interfaced with the MSP430, the ability to turn a stepper motor through interfacing with an MSP430, and accurate and precise sorting of resistors based on their values. Based on these proposed success criteria, the project was successful in several ways. First, a mobile application that could detect and calculate the resistance of a given resistor based on the centroids found on the colored band was delivered.

The mobile application had significant difficulty recognizing some resistors, however. Ambient lighting level changes were identified to be a factor in this issue, though there could be others. Additionally, the MSP430 was successfully programmed to interface with two stepper motors and turn them, and the system could accurately and precisely sort these resistors based on the value passed to the MSP430. As for the Bluetooth connection defined in our success criteria, the team was partially successful in delivering. Though progress was made in developing the Bluetooth connection using the mobile application, it ultimately was not able to successfully send a value through this connection. The HC-05, however, was able to receive and decode a value. To complete this half of the Bluetooth communication, a Bluetooth Terminal Android application was used. Thus, values were still passed over Bluetooth, but the mobile application was not, in the end, successful in sending a value.

The grading defined in the proposal was as follows: a D should be awarded for one success criteria being met, a C should be awarded for two success criteria being met, a B should be awarded for three success criteria being met, and an A should be awarded for all four criteria being met.

Costs

A more detailed breakdown of our cost for our system can be seen in the Appendix A.

Overall, the cost for this system was quite low. It came to a total of $81.17 for the entire system. The most expensive components we had to purchase were the batteries and their charger, with the motors and the HC-05 module being high in cost as well.

If we were to manufacture in 10000 unit quantities, our costs would obviously change quite a bit. Using Digikey and Amazon to estimate costs if we bought components in that quality, our total cost per unit comes out to be $55.26 , not including assembly and labor costs. Our current design would lend itself well to automated equipment assembly, and would probably influence costs very little. The PCB and motor drivers need to be connected to batteries, and then the electronics must be stored inside the plastic base. Overall it’s not a complicated process, and we would expect this construction to cost very little.

Future Work

If this project were to be improved in the future, there are a few pitfalls that must be navigated carefully. First off, ambient lighting in a room, which is constantly changing due to the position of the sun during the day, the number of clouds in the sky, and the brightness of light bulbs, can greatly affect the reliability of computer vision. During testing, it was found the the computer vision application worked with a higher degree of reliability during the early morning and afternoon of a partly cloudy day, but decreased in reliability as the sun started to set. Attempts to diminish this variability - including the use of different backgrounds to balance out the light, turning on the phone flashlight to flood the image with light, and holding the camera at different angles - did not do much to help with the unreliability. Because of this, a great improvement for the project would be to include a controlled environment for the computer vision application to work with the same lighting conditions at all times.

Additionally, in its current state, the resistor sorter can only handle one resistor that is carefully placed on the slide at a time. For this system to be viable in a real lab environment, it should be able to handle piles of resistors at a time. Thus, expanding the system to include a funnel that a user could pour piles of resistors into would be very helpful. With this expansion, the system could separate single resistors from a large pile, automatically place it correctly on the slide, read its value, sort it, and then deposit the next resistor in the pile. Along with this addition, the ability to quickly change the values of resistors that are sorted through a user friendly interface on the Android application would allow for the system to be usable with a large number of resistor values, as opposed to just six at any given time.

Finally, a major area for improvement on this project is the Bluetooth sending ability of the mobile application. None of the team members had any experience in coding for Bluetooth on a mobile application, so significant progress was made with research and trial and error, but ultimately this ability was not delivered. In the end, the team member responsible underestimated the time needed to understand mobile application errors and the nuances of integrating Bluetooth communication protocol into an already existing open-source mobile application. If more time were available, additional resources for adding Bluetooth capabilities as well as debugging on a mobile platform may have been able to be sought out, aiding in the process of developing the Bluetooth capabilities further.

Overall, any future work on the resistor sorting system would likely be towards making the project viable for a lab environment, including making it more reliable and user friendly. With these additions and modifications, the system could feasibly be adopted in labs to keep the resistors organized in a quick and easy fashion.

References

[1]

D. Gentry and C. Sussman, “Cleaning Up the Labs Using a Resistor Sorter,” p. 7.

[2]

W. Al, “Sorting Resistors With 3D Printing And A PIC,” *Hackaday*, 30-Dec-2016. .

[3]

mikkelbg, “Resistor sorter,” *RobotShop Community*, 21-Dec-2008. [Online]. Available: https://www.robotshop.com/community/robots/show/resistor-sorter. [Accessed: 16-Dec-2019].

[4]

Resistor Scanner demonstration. 2017.

[5]

Parth, *thegouger/ResistorScanner*. 2019.

[6]

“Bluetooth Basics - learn.sparkfun.com.” [Online]. Available: https://learn.sparkfun.com/tutorials/bluetooth-basics. [Accessed: 16-Dec-2019].

[7]

“IEEE Std 802.15.1-2002, IEEE Standard for Information technology--Telecommunications and information exchange between systems--Local and metropolitan area networks--Specific requirements--Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer,” p. 1169.

[8]

Laura, “Guide to IPC Standards for PCBs,” *MCL*, 01-Apr-2019. .

[9]

D. A. LaRose *et al.*, “United States Patent: 9527115 - Computer vision and machine learning software for grading and sorting plants,” 9527115, 27-Dec-2016.

[10]

T. E. Rathjen, W. H. Lindhardt, I. I. I. Foell, and B. Lindhardt, “United States Patent: 10068405 - Coin recognition system and method,” 10068405, 04-Sep-2018.

[11]

S. Baltazor, “United States Patent: 9443367 - Digital image coin discrimination for use with consumer-operated kiosks and the like,” 9443367, 13-Sep-2016.

[12]

"MIKROE-1530 MikroElektronika | Motors, Solenoids, Driver Boards/Modules | DigiKey." [Online]. Available: https://www.digikey.com/products/en?mpart=MIKROE-1530&v=1471. [Accessed: 16-Dec-2019].

[13]

"ULN2003A STMicroelectronics | Discrete Semiconductor Products | DigiKey." [Online]. Available: https://www.digikey.com/product-detail/en/stmicroelectronics/ULN2003A/497-2344-5-ND/599603. [Accessed: 16-Dec-2019].

[14]

"MSP430G2553IN20 Texas Instruments | Integrated Circuits (ICs) | DigiKey." [Online]. Available: https://www.digikey.com/product-detail/en/texas-instruments/MSP430G2553IN20/296-28429-5-ND/2638885. [Accessed: 16-Dec-2019].

[15]

"Amazon.com: JBtek HC-05 Wireless Bluetooth Host Serial Transceiver Module Slave and Master RS232 For Arduino: Computers & Accessories." [Online]. Available: https://www.amazon.com/Wireless-Bluetooth-Serial-Transceiver-Arduino/dp/B00L083QAC. [Accessed: 16-Dec-2019].

[16]

"Amazon.com: 10 Pack 9V Battery Clip Connector, T Type Faux Leather Long Cable Housing Connection: Health & Personal Care." [Online]. Available: https://www.amazon.com/Battery-Connector-Leather-Housing-Connection/dp/B06X8YZJ64/ref=sr_1_3?keywords=9V+battery+clips+10+pack&qid=1576525934&sr=8-3. [Accessed: 16-Dec-2019].

[17]

"(6-Pcs) STMicroelectronics 3.3V 950mA, LD1117V33 Voltage Regulator, LD33V: Amazon.com: Industrial & Scientific." [Online]. Available: https://www.amazon.com/6-Pcs-STMicroelectronics-LD1117V33-Voltage-Regulator/dp/B01MQF7D9D/ref=sr_1_1?keywords=3.3V+regulator+6+pack&qid=1576526085&sr=8-1. [Accessed: 16-Dec-2019].

[18]

"(15 Pieces) MCIGICM IC 7805 Voltage Regulator, L7805 Linear Positive Fixed 1 Output 5V 1.5A: Amazon.com: Industrial & Scientific." [Online]. Available: https://www.amazon.com/MCIGICM-voltage-regulator-Regulator-Positive/dp/B07BDFMQF6/ref=sr_1_fkmr0_1?keywords=5v+regulator+ld1117v33&qid=1576526175&sr=8-1-fkmr0. [Accessed: 16-Dec-2019].

[19]
"PREC040SFAN-RC Sullins Connector Solutions | Connectors, Interconnects | DigiKey." [Online]. Available: https://www.digikey.com/product-detail/en/sullins-connector-solutions/PREC040SFAN-RC/S1212EC-40-ND/2774934. [Accessed: 16-Dec-2019].

[20]
"PPTC061LFBN-RC Sullins Connector Solutions | Connectors, Interconnects | DigiKey." [Online]. Available: https://www.digikey.com/product-detail/en/sullins-connector-solutions/PPTC061LFBN-RC/S7004-ND/810145. [Accessed: 16-Dec-2019].

[21]
"ED16DT On Shore Technology Inc. | Connectors, Interconnects | DigiKey." [Online]. Available: https://www.digikey.com/product-detail/en/on-shore-technology-inc/ED16DT/ED3046-5-ND/4147596. [Accessed: 16-Dec-2019].

[22]
"Amazon.com: 9V Rechargeable Batteries 800mAh 6 Pack with Keenstone 9 Volt Battery Charger: Home Audio & Theater." [Online]. Available: https://www.amazon.com/Keenstone-Rechargeable-Detector-Multimeter-etc%EF%BC%883-Pack%EF%BC%89/dp/B078JMHHF8/ref=sr_1_6?keywords=rechargeable+9v+battery&qid=1572554151&sr=8-6. [Accessed: 16-Dec-2019].

[23]
"1-2199298-6 TE Connectivity AMP Connectors | Connectors, Interconnects | DigiKey." [Online]. Available: https://www.digikey.com/product-detail/en/te-connectivity-amp/1-2199298-6/A120351-ND/5022043&?gclid=EAIaIQobChMIl_Thr63H5QIVJOeGCh2gTA38EAQYASABEgJAXPD_BwE. [Accessed: 16-Dec-2019].

[24]

"MSP430G2553 | MSP430G2x/i2x | MSP430 ultra-low-power MCUs | Description & parametrics." [Online]. Available: http://www.ti.com/product/MSP430G2553. [Accessed: 06-Dec-2019].

[25]

Sam, “How to Use HC-05 Bluetooth Modules - Tutorial,” Core Electronics, 07-Apr-2017. [Online]. Available: https://core-electronics.com.au/tutorials/how-to-use-bluetooth-modules.html. [Accessed: 16-Dec-2019].

[26]

“JTAG (MSP430),” JTAG (MSP430) - Texas Instruments Wiki. [Online]. Available: https://processors.wiki.ti.com/index.php/JTAG_(MSP430). [Accessed: 16-Dec-2019].

[27]

“step-motor-5v-28byj48-datasheet.pdf.” .

[28]

“msp430g2553.pdf.” .

[29]

“Sensors Modules Bluetooth Module Hc 05 | Sensors Modules.” [Online]. Available: https://www.electronicwings.com/sensors-modules/bluetooth-module-hc-05-. [Accessed: 16-Dec-2019].

[30]

Dejan, “Arduino and HC-05 Bluetooth Module Tutorial,” *HowToMechatronics*, 29-Feb-2016. .

[31]

Dejan, “How To Build Custom Android App for your Arduino Project using MIT App Inventor,” *HowToMechatronics*, 30-Mar-2016. .

Appendix

Section A - Spreadsheet

Name	**Quantity**	**Price per Unit**	**Total**	**10000 Units**
28BYJ-48 – 5V Stepper Motors	2	8.16	16.32	81600
ULN2003 Stepper Motor Drivers	2	0.54	1.08	3169.8
MSP430G2553	1	2.69	2.69	11843
HC-05 Bluetooth Module	1	8.49	8.49	84900
9V Battery Clips	3	0.318	3.18	9540
3.3V Regulator	1	1.07	6.4	10700
5V Regulators	2	0.466	6.99	9320
Male Connectors (14)	4	0.57	2.28	11648
Female Connectors (6x1)	1	0.52	0.52	2327
Connector Socket (SMD) 16 pin	2	0.19	0.38	1551.04
2 - 9V Batteries/Charger	1	32.39	32.39	323900
Connector Socket (MSP) 20 pin	1	0.45	0.45	2094.16
			TOTAL	**Price per Unit**
			81.17	**55.2593**

Section B: Final Physical Setup

Final Physical Setup