Bluetooth Ouija Board

ECE 445 Final Report

Team 59: Luke Staunton, Nikhil Mathews, Oluwatobi Ijose
TA: Jacob Bryan
5/1/18

Abstract

Traditionally, a ouija board uses a planchette to spell out words on the board. Our project uses a magnet module underneath the board and two mechanical rails to move the planchette on the x and y axis to the desired character. We use a phone application to send the desired word to the board via Bluetooth, as well as communicate appropriate control signals. This allows us to mimic the phenomenon of spiritual communication with a completely automated system from the viewer's point of view. The purpose of this report is to outline the design, functionality and verification of our project.

Contents

Table of Contents

1. Introduction

1.1 Objective

We are designing and implementing a system that would allow us to successfully create the impression that there are spiritual forces communicating specific messages to users of a ouija board. By creating an phone application controlled, Bluetooth ouija board, we will be able to perform this illusion by allowing the board to autonomously move the planchette towards the desired characters. A person with the designated app would be able to input a string of choice, and the planchette on the board would maneuver to spell the word, guiding the users hand with the planchette. This makes the illusion much more compelling, especially to non-believers.

1.2 Background

There are many people who genuinely believe that there are actually spirits controlling what is output by a ouija board, and that there are many unseen supernatural elements to our world. Historically, ouija boards have been operated purely by the end user by phenomenon called "ideomotor effect"[1]. With the added benefit of our electromagnet, we can further influence the end users perception. We hope to play on the superstitions of these people, in order to mimic a ouija board that is actually moving on its own. We have seen a couple rudimentary versions of this project online[2], but they aren't very fluid and sometimes do have obvious signs such as clicking noises, or choppy delay between characters, which we plan to eliminate. While there is not an outright need for this solution, it can be more convincing than the original ouija board.

1.3 Subsystem Overview

Our system broke into six major subsystems: phone application, bluetooth module, motor module, microcontroller module, magnet module, and power module. The phone application initiates the illusion and sends the user specified string to the microcontroller. It also notifies the microcontroller that the planchette is on the board, and to initiate search for it. The bluetooth module handles all communication between the system and the phone. The motor module uses feedback control to move the electromagnet to the specific position underneath the board. The magnet module determines if the planchette is coupled or not. All of these modules are then powered via batteries and voltage regulators using the power module. This is visually described in our block diagram in Appendix C, figure C.1.

2 Design

2.1 Physical Design

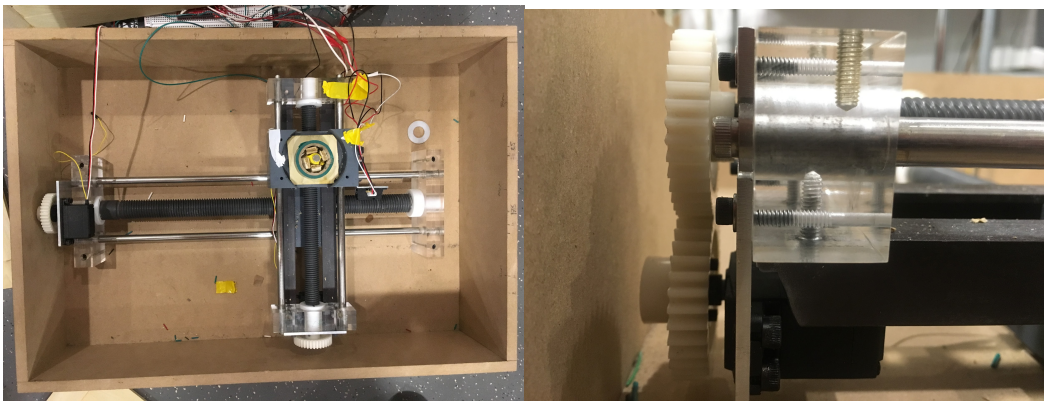


Figure 2.1 (Left): Top down view Figure 2.2 (Right): Servo driving the minor axis

The purpose of the physical design was to create a mechanism that could move the electromagnet across the x-y plane with ease, and offered enough height to contain our PCB and electrical components. The design shop had the rail system already built from a prior group, so we did not have to design it. This did leave us with a physical setup that was untested. We ran a series of tests to confirm that the rails could be moved by the servos we were planning to purchase. We also had a housing built around the rail system as one can see in figure 2.1. With the housing, the overall size of the ouija board system was 21.5 in. x 14 in. x 4.5 in. We used a 1:1 gear ratio to drive the rails with our servos, the justification for this is best explained in section 2.4. Servos were attached above the major axis rail as one can see in figure 2.1 and figure 2.2 shows how the minor axis servo was attached from below.

An alternative physical design would look like figure 2.3, which is the mechanics of an Etch-a-Sketch. This system would have used pulleys attached to servos to actuate the position of the magnet module. We decided against this alternative design because the machine shop already had a manufactured version of the rail system.

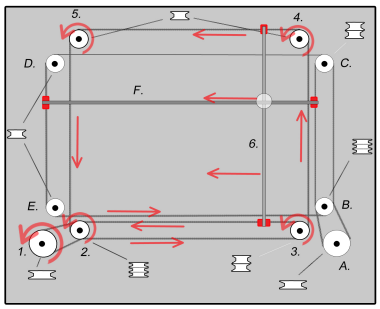


Figure 2.3: Alternative design [3]

2.2 Phone Application

The phone application was intended to interface with the ouija board. We chose the phone application because it allows the user to perform the trick on the ouija board observer from a distance and discretely. The application was developed in Android Studio and used the flow chart described in figure 2.4. The phone application had a single text box input for the word to be displayed and buttons to send the word, initiate snake search on the microcontroller, cancel the first word on the queue, and to request the coupling status of the magnet and planchette. All of these buttons simply relayed the specific command in the specified format on the Bluetooth serial line to the HC-05 for user ease. If a "block" state was reached in figure 2.4, the buttons would be grayed out (removing this functionality). If we did not receive acknowledgement of a command or information sent, there was likely packet loss, so we would timeout and perform another send event and wait for a response again.

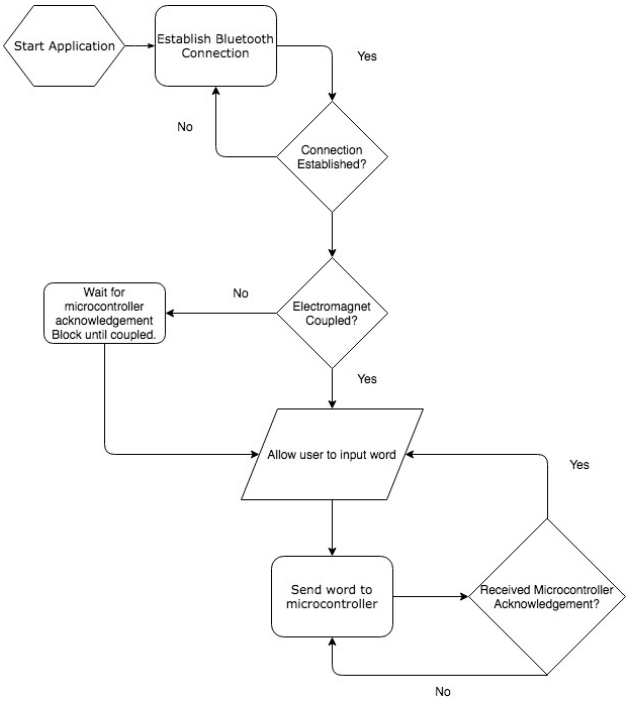


Figure 2.4: Phone application flow-chart

2.3 Bluetooth Communication

For our Bluetooth communication, we used the serial RX and TX lines on the microcontroller connected to a HC-05 chip, connected as shown in appendix B, figure B.2. We only communicate on the serial line to either send control signals such as "snake search," "cancel word," and to determine the coupling status or to send a word to be displayed on the ouija board

to the microcontroller. The base data rate of the HC-05 is 38,400 bits per second, so communication rate wasn't a primary concern given the inherently slow communication rate of human input (approximately 10 bytes/second in our case). However, the main concern was validating that we received data so that there was no packet loss. To solve this, a handshake protocol was established between the microcontroller and phone application. On receiving a word or a command from the phone application, the microcontroller would send back an acknowledgement, that would allow the phone application to send another word or command. This prevented not only loss of data, but also prevented the user from spamming data on the serial line.

Another design addition to our communication was the addition of delimiters for each piece of information. The serial buffer on the microcontroller is flushed slowly relative to the communication rate so there were artifacts of older messages. This lead to message splicing and difficulty interpreting the messages because the microcontroller has no way to know where the beginning and end of message is [4]. To combat this, we used "<" as a starting delimiter and ">" as an ending delimiter. This allowed us to verify if any new data was added to the serial buffer, and if so, where this data was beginning and ending. Lastly, because we only allowed messages to the ouija board to be alphabetical, we used '@' before a command to indicate to the microcontroller that we were issuing a command instead of data to be displayed.

2.4 Motor Module

The purpose of the motor module is to move the magnet module to the position under the desired character on the Ouija board. It does this using two "Parallax 360 Feedback Servo"[5] motors and two "Sharp GP2Y0A41SK0F Analog Distance" [6] infrared sensors. The servos and sensors are then attached to the rails and the IR sensors are attached to the magnet module. The microcontroller module uses this information and a linear feedback loop to calculate the proper speed for the motors. Each axis was treated as an independent control systems. It also uses information from the magnet module to determine if the planchette is under the influence of the electromagnet, we call this being "coupled". If the connection is broken, the motors backtrack the module to find the planchette.

We considered using ultrasonic sensors as opposed to the infrared sensors in the early phases of design. We steered away from this option for two reasons. The first being fear that echoing within the box could throw off the sensors reading. The second being that it required SPI or I2C communication, and we were unsure how well we would be able to support this with our microcontroller code.

We briefly considered stepper motors as opposed to servos, but struggled to find motors with narrow enough steps in the same price range as the feedback servos that we had already researched.

After testing the distance sensors and the motors with the rails, we were able to develop models that mapped the analog distance signal into cm and the desired velocity of the magnet module in cm/s into a pulse width in microseconds for the 20 ms PWM that drives the servo motor. The test process that led to these models can be found in section 3.4 of this document. The motor equations differed for both the axis, and different equations were used for both positive and negative velocities. Equations (2.1), (2.2), (2.3), and (2.4) denote the major axis negative, major axis positive, minor axis negative, and minor axis positive velocity models respectively. Where v is the desired velocity and PWM is the pulse length in microseconds. The output of the distance sensor is mapped by (2.5), where v is the voltage from sensor and x is a position in cm. These mapped values were used with the Arduino Servo library to set the needed PWMs.

$$PWM = int(1460+287*v) \quad (2.1)$$
$$PWM = int(1540+289*v) \quad (2.2)$$
$$PWM = int(1520 +353*v) \quad (2.3)$$
$$PWM = int(1480 + 357*v) \quad (2.4)$$
$$x = 2275/v +1.959 \quad (2.5)$$
$$v[t] = 1/2 * (ref - x[t]) \quad (2.6)$$

We used a simple difference equation seen in (2.6) to calculate output to the motors. Where ref is the reference signal corresponding to the location of the desired character, x[t] is the current position, and v[t] is output velocity to the motor. The coefficient ½ was determined by testing various constants and seeing which had the fastest response, without corrupting accuracy.

This loop was implemented separately for each motor. We went with such a simple difference equation because all experiments with the rails led to a conclusion that they had no momentum, as soon as motor power was turned off they ceased to move. It also allowed our system to be over dampened. Combining that with the built in control system in the servo made such a simple equation possible for accurate results. To have used a more complicated one would have only increased processing time, with no improvement in accuracy.

In order to prevent locking out other methods, we implemented the control loop as a method that gets called with each discrete time step. This method checked to confirm both the major and minor axis measured values were “equal” to their respective axis, and if not set the appropriate output PWM. The method returns true once both axes are “equal” and returns false otherwise. To determine when signals were “equal,” the floating point values of the positions were within a threshold of 0.15 cm difference.

2.5 Microcontroller Module

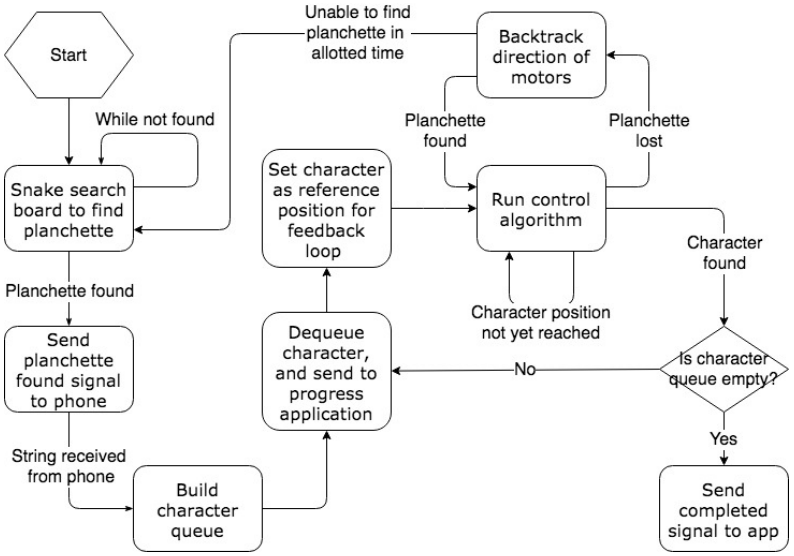


Figure 2.6: Flow chart of microcontroller code

The role of the microcontroller module is to process information received by the various modules and use that to control them. Notably, run a "snake search" to find the planchette, receive and process the needed string from the Bluetooth module, activate the electromagnet and determine if it is coupled, and use the prior two sets of information to set the proper feedback loop parameters to move the magnet module to each of the desired characters. This is more specifically outlined in figure 2.6. The full schematic featured in the Appendix B displays how it controlled each of the modules using various data lines.

The microcontroller we chose for our design was the ATMega328PB [8]. We selected this because it is what comes standard on the Redboard launch pad that our group possessed prior to the beginning of this course, and its compatibility with the Arduino IDE.

In order to ensure that the electromagnet was coupled before setting up the control loop, we implemented a "snake search" function. This moved the magnet module to the bottom right corner of the board and had it snake across the board until it received a coupled signal. This guaranteed connection between the magnet and the planchette before going to the first character. This allowed the initial position of the planchette to be anywhere on the board, and made the illusion more believable.

We later considered an alternative to the "snake search" where the phone would send a "guess character," that the magnet would go to first and then do a spiralling search outward from that point. We chose not to implement this due to time constraints, and failures with the magnet module.

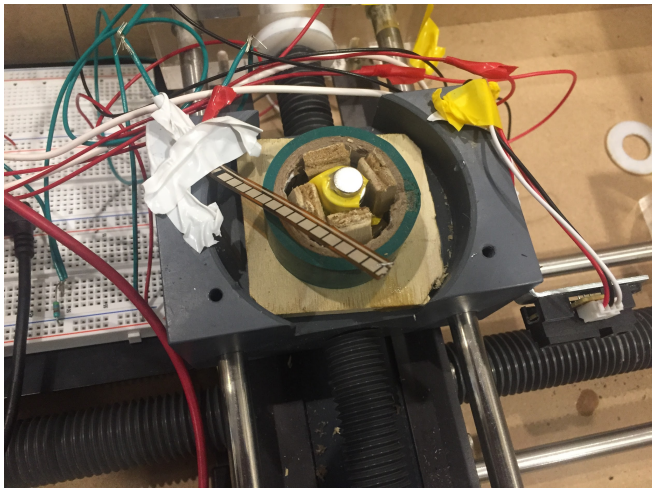
In order to get to the proper reference signals for the control loop, the positions in cm of each character on the ouija board were stored in static arrays in the microcontroller code.

In order to maintain connection with the phone, the microcontroller had to frequently ping the Bluetooth module. This was done with the implementation of a “pseudo-scheduler.” This was done by having a main loop that was constantly iterated through. The loop maintained contact with the Bluetooth module first and then used a series of static variables including a “state” variable to execute the needed method corresponding to its state. For example when the control algorithm was running it would communicate with bluetooth, check if the magnet was coupled, run one iteration of the control difference equation, and then based on if it had reached its destination or not, transition the state variable and repeat the loop.

By having all of the functions corresponding with these states operating in less than 1 ms, we were able to maintain communication and get all of the functionality we needed. We determined this to be the best method as the only alternative would be a threading would have cost massive amounts of time we did not have.

2.6 Magnet Module

The magnet module consists of a housing containing a permanent magnet, and a sensor to detect



coupling. The exact configuration of the magnet module is shown in Figure 2.7.

Figure 2.7: Magnet module

2.6.1 Magnet

The purpose of the magnet within the magnet module is to attract the planchette on the top of the board, and couple with it to allow for effective movement. Initially, this was done with an electromagnet. This electromagnet was constructed with a bolt and thin copper wire wrapped

around it. After wrapping 600 turns of the wire around the magnet, we were able to successfully attract this electromagnet to the planchette on top of the board. However, despite being able to reduce the current through the magnet to 1.8 A, its strength was inconsistent, and we kept running into issues of heat. The magnet continually became too hot to touch, and we felt that unless we designed a cooling system, we would need another solution.

Thus, we decided on the use of a neodymium permanent magnet. This permanent magnet provided more strength, and no current requirements. We did however, explore ways to turn this magnet on and off. Theoretically, if a current is run in the opposite direction of the magnetic field, the fields will cancel, temporarily demagnetizing the magnet. With no way of instrumentally measuring magnetic field (it is possible to model with equations however), tests were simply ran with a similar copper wire around the permanent magnet. Despite applying up to 5A of current to it, we were unable to temporarily demagnetize it. A possible reasoning for this is that the magnet is so strong, that we would need to apply an extremely large amount of current to it to mitigate its magnetic field. This amount of current would ultimately be prohibitive in our project.

2.6.2 Sensors

In order to measure coupling, we needed to figure a parameter to use as a threshold value to discern magnetic coupling. To do so, we began by measuring the change in pressure inside the module when the magnet is levitated, as opposed to on top of the sensor. Our configuration of the module had the pressure sensor at the "floor" of the module, with the magnet configuration on top of it. Upon testing however, it became apparent that there was not a great enough change in pressure to obtain a suitable threshold value.

Pivoting from the pressure sensor, we decided to employ a flex sensor in order to measure this mechanical displacement. The flex sensor was used as a resistor in a simple voltage divider circuit, with a 45 kΩ resistor in series with it, put together on a perf board, as we did not have time to order a new PCB. Our Arduino script outputs the resistance and bend of the sensor, however, the bend of the sensor was sufficient for use as our threshold. First, we placed the sensor again at the floor of the module, and measured the bend of the sensor. We found that while there was a large enough discrepancy in bend values, we also saw that the values were inconsistent, and thus couldn't obtain a repeatable threshold value. Next, we tried placing the sensor at the top of the module, but ran into similar consistency issues.

One viable design alternative for this subsystem is the use of a traditional, flat force sensor. This sensor would be placed on top of the magnet system, and when it is levitated, it would read out the force between the top of the magnet system, and the bottom of the board. The advantage of using such a system would be an increase in consistency, as we won't have to worry about arbitrary bend values, or inconsistent resistance values. Additionally, the flat shape of the sensor

would allow for discrete and convenient placement within the module, and would avoid the issue of the flex sensor being deformed when the module is at the edges of the rail system.

2.7 Power Module

2.7.1 6V battery

We chose to use a 6V Duracell MN908 battery to power our system. The total power usage of our system at peak output is depicted in Table 2.1 below.

Component	Voltage	Current	Power
Flex Sensor	3.3 V	0.303 A	1 W
Motor (x2)	6 V	2.8 mA	.0336 W
IR Sensor	5 V	12 mA	.06 W
Bluetooth Module	5 V	50 mA	.25 W
Microcontroller	5 V	0.2 mA	.001 W
Voltage Regulator (x3)	9 V	100 µA	9E-4 W
			Total = 1.3445 W

Table 2.1: System peak power requirements

2.7.2 Voltage Regulators

Due to the differences in input voltages for the various components of our system, we needed to employ 3 voltage regulators. These regulators will step down the voltage from the battery to 3.3V, 5V, and 6V. They are connected as shown in Appendix B, figure B.3.

3 Design Verification

3.1 Physical Design

When we first received the rail system, we needed to verify that the static friction would be lower than the stalling torques of market available servos. To test the static friction, We wrapped a string around the gear and attached a cup of weights to the end of it. We slowly added more weight until the weight overcame the static friction and put the module in motion. After this we counted the weights in the cup, and stopped the motion of the rail to ensure that this was the minimum amount of weight needed to overcome the static friction. Once we'd repeated it a few times and was confident on that it was the proper weight we measured the radius of the gear, and using calculated the torque. On the shorter axis, this torque was 3.405 oz-in and was 16.75 oz-in

for the longer rail. Because the stalling torque of the servos was reported to be 34.7 oz-in in the datasheet [5], we knew that the mechanical system would work.

3.2 Phone Application

In order to test the phone application, we first had to test simply the send and receive functionality with another phone. We developed a log that would output debug messages for each button. We used two phones, one with a test application with Bluetooth messaging enabled to verify that our development app was working as described. First we checked that we could in fact connect to the external test device. Then, we sent a message in the specified format <“word”> to the test application, checked that it was received on the test application, and also followed the specified flow chart specified in figure 2.4. If a message is sent, we can’t send another message until we receive acknowledgement, so to see that behavior, we waited a couple seconds on the test application to send acknowledgement back to the development device. In this manner, we were able to verify the state diagram.

Next, we connected the development app to the HC-05 module connected to our microcontroller. We tested that we could send and receive information in the described format and receive a response from the microcontroller. Since the protocol for sending and receiving messages were similar on the microcontroller and our test app, we had to simply time the responses to ensure that there was not too much delay. We did find a splicing bug if too much data was fed to the serial line in quick succession, but we eliminated this by forcing the phone application to only read data from the line at 300 ms intervals, which allowed enough time for the microcontroller to populate the serial buffer with a full message.

An internal log on the phone application was also instrumental in understanding the microcontroller responses and the data we were sending to it.

3.3 Bluetooth Communication

To avoid packet loss, we implemented a handshake protocol to limit this. However, in practice there were no instances of packet loss unless we abruptly left the range of the Bluetooth module. We tested sending words that were 10 characters long (the maximum we specified was 10 bytes/second), and they were delivered successfully from a range of 5 meters. This range was enough to be outside the view of the person using the ouija board. This was critical because otherwise the ouija board observer could see the phone application use and this could cause suspicion.

We were also able to successfully receive and queue up characters in the appropriate order on the microcontroller. Based on the speed of input from a human user, it was very unlikely that the words would be sent in the wrong order. However, by printing the queue contents as words were

received, we were able to visually see that the words were received in the order that they were sent.

3.4 Motor Module

The verification of the motor module was centered around the precision of the final location of the magnet module. All of our requirements can be found in Appendix A. There were also test done to calibrate the mapping functions.

To map velocity to PWM we attached the servos directly to the driving gear with tape. Then powered the servos at 6V using a DC power source in the senior design lab. We then programed the function generator to a square wave with a 20ms period and plugged that into the input of one of the servos. We taped a ruler to the lab bench and measured the initial position of the module. As the function generator output was turned on, we started a stopwatch. We then arbitrarily stopped the function generator as we stopped the timer. We then measured how far the module had moved. We did this for eight different duty cycles for both rails and used the data to create figures 3.1 and 3.2. This data was used to derive equations (2.1), (2.2), (2.3), and (2.4).

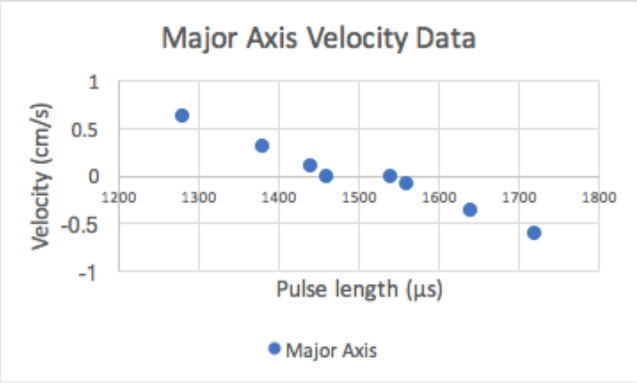


Figure 3.1: Major axis velocity vs. PWM width

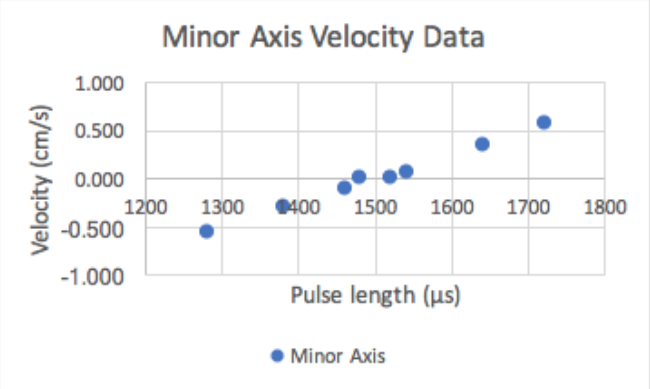


Figure 3.2: Minor axis velocity vs. PWM width

The easiest part to verify was that there was no overshoot. This was done by removing the lid to the board and sending arbitrary characters from the phone app, then watching the magnet module move. It was visually confirmed that it never exceeded the desired position after testing over 20 characters, therefore showing it was over dampened.

The precision of the distance sensors measurement was also one of our requirements. To verify this, we removed the lid and positioned the module at various positions along the major axis. We programmed the ATMega to pipe out a value f(data), which was 1000/V, where V is the voltage read from the sensor. We then used this data to build a best fit affine linear model to measure distance. This data can be seen in figure 3.3.

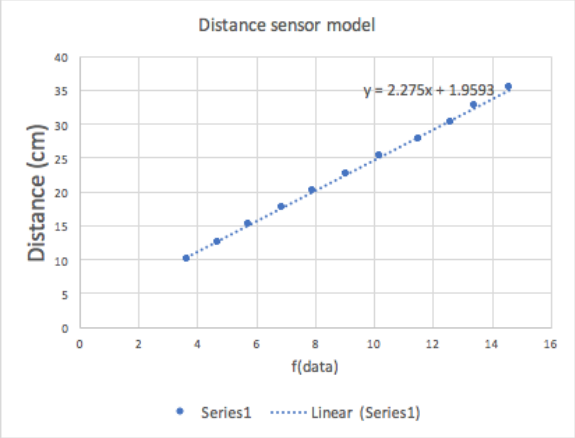


Figure 3.3: Distance data and linear model

Since our model was data driven, the mean square error from the data set depicted in figure 3.3 was 0.046 cm*cm. We also never had error greater than .75 cm, therefor meeting our requirement. Unfortunately we did not reach this level of accuracy with the lid attached.

In order to verify our last requirement we once again entered in random characters from the phone app and then measured to confirm that it was within a 1.25 cm radius of the reference position stored in the static array within the microcontroller module.

3.5 Microcontroller Module

The microcontroller was very easy to validate as very few of the requirements were quantitative. A simple string from the phone application was able to confirm that the entirety of the string “please pass us” could be loaded into the character queue, and cause the magnet module to make 12 distinct stops. This proved the verification of the queue requirement. During our demo we showed the success of the snake search, confirming that requirement as well.

The only numerical requirement was the timing requirement of the control loop. This was verified using timestamped printouts, and even with the delay that comes with printing to serial we were significantly below are requirement of 160ms max delay.

3.6 Magnet Module

3.6.1 Magnet

To verify our actual magnet, we applied a current to the coil electromagnet. And measured the current output vs applied voltage. Additionally, we did qualitative analysis on whether or not the magnetic strength could sustain the planchette.

Table 3.1: Magnet Module Verification

# of Turns	Voltage Applied [V]	Current [A]	Coupled [Y/N]	Suitable Strength [Y/N]
100	2.3	3.6	N	N
200	2.3	3.6	N	N
400	2.6	3.6	N	N
600	2.3	3.3	N	N
1000	2.6	2.8	Y	N
1200	2.3	2.1	Y	Y
1300	2.3	1.8	Y	Y

As noted in the design section, the heat generated when applying current through the electromagnet prevented us from using this system. Additionally, the use of the permanent magnet mitigated the need for the BJT circuit as well.

3.6.2 Sensors

To verify the pressure sensor functionality, we tested it's values when the magnet was levitated above it, versus when it was sitting on it in the module. The testing procedure was as follows:

1. Make necessary connections to power and program pressure sensor
2. Place sensor on the "floor" of the housing
3. Place ouija board top on housing
4. Run the motors, while simultaneously reading out pressure values
5. Remove plachette to simulate decoupling, and read out pressure values

Table 3.2 details the process of our progression through four tests, we noted the uniformity of our values, and concluded early on that this was not a suitable solution. The slight variations in out values was inconsequential.

Table 3.2: Pressure Readings

Test #	Pressure [psi] - Coupled	Pressure [psi] - Not Coupled
1	12.99	12.99
2	12.98	12.99
3	13.01	13.00
4	12.99	12.98

To verify the flex sensor, we created our module configuration, and read out values for the bend of the sensor. First, we got bend values for the configuration with the sensor at the bottom of the module, and then for when it was at the top of the module. The testing procedure is the same as when we were testing the pressure sensor.

Table 3.3: Flex Sensor Readings

Test #	Bend, Bottom, Coupled	Bend, Bottom, Not Coupled	Bend, Top, Coupled	Bend, Bottom, Not Coupled
1	34	23	29	20
2	31	25	29	23
3	33	27	27	24
4	30	38	26	23

As one can see, it appears that the coupling and decoupling bend values approach each other as more tests were done. This is because our sampling rate for coupling did not account for the amount of time the sensor needed to mechanically recover after being deformed.

3.7 Power Module

3.7.1 6V Battery

The MN908 battery is rated at 13 Ah, thus should be able to power our system for 9.7 hours, per the calculation below.

$$\frac{13\ [Ah]}{1.3445\ [A]} = 9.7\ hours$$

3.7.2 Voltage Regulator Verification

To verify the voltage regulators, we simply used the multimeter in our provided kit to measure the voltage at the outputs. The results can be seen in Table 3.4

Table 3.4: Voltage Regulator Verification

Regulator	Output
3.3V	3.28 V
5 V	4.84 V
6 V	5.92 V

4 Cost

All costs associated with parts used in our project, either for design or testing, are listed in Table 4.1. All labor costs are listed by team member are listed in Table 4.2 and are calculated using the formula:

$$\text{Total Individual Cost} = \text{salary} \times \text{hourly rate} \times \text{actual hours spent} \times 2.5$$

Finally, the total cost associated with this project is listed in Table 4.3.

4.1 Itemized Parts List

Figure 4.1: Parts Cost

Part	Unit Cost	Units Needed	Sub Total Cost
Atmel ATMega328P	$2.50	1	$2.50
HC-05 Arduino Wireless Bluetooth Receiver	$7.69	1	$7.69
Sharp GP2Y0A41SK0F Analog Distance Sensor	$9.55	2	$19.10
HoneywellSSCMNNN030PA2A3 Pressure Sensor	$31.88	1	$31.88
Parallax 900-00360 DC Servo Motors	$24.99	2	$49.98
Texas Instruments LP38692MP Voltage Regulator	$1.58	3	$4.74
FLORA SEWABLE 3-PIN JST WIRING A (wiring for distance sensors)	$1.95	4	$7.80
LED DURIS S 5 GREEN 540NM 2SMD	$0.73	2	$1.46
IC USB SERIAL FULL UART 20SSOP (serial processor)	$2.12	2	$4.24
Resistors of values 2k, 1k, 10k, 10, 39, 30, 16.2, 27 ohms	$0.04	80	$3.20
Capacitors of value .1uF, 47 pF, 1uF, 10uF	$0.20	15	$3.00
SWITCH PUSHBUTTON SPST 1A 30V (Push button)	$0.74	3	$2.20
OSC XO 16.000MHZ HCMOS TTL SMD (16MHz Oscillator)	$1.31	3	$3.93
MOSFET N-CH 60V 0.31A SOT323 (Mosfet)	$1.31	3	$3.93
Total Cost			$144.76

4.2 Labor Cost

Table 4.2: Labor Costs

Team Member	Hourly Rate ($)	Total Hours	Total Individual Cost ($)
Nikhil Mathews	50.00/hr	132	16,500
Luke Staunton	50.00/hr	132	16,500
Tobi Ijose	50.00/hr	132	16,500
Total Team Cost ($)			49,500

4.3 Grand Total Cost

Table 4.3: Grand Total Costs

Parts ($)	Labor ($)	Grand Total ($)
144.76	49,500	49,644.76

5 Schedule

Table 5.1 lists weekly progress by each team member during the course of this semester.

Table 5.1: Updated Project Schedule

Week	Luke	Nikhil	Tobi
2/26	Completed slide deck and presentation. Made list of high level software requirements for integration. Met with machine shop about refining physical prototype. Continued measuring parameters for the feedback control model	Developed an integration plan with Tobi and Luke. Listed high level software requirements between all modules/drivers.	Use wire for extensive testing of needed current and force.
3/5	Started PWM driver Determined concrete changes that need to be made to physical prototype. Started modeling layout of our special components for PCB layout.	Finished programming basic iOS application. Added a Bluetooth manager to manage peripheral connections.	Write drivers for I2C communication protocol between the microcontroller and the pressure/IR sensor
3/12	Finished PWM driver for motors. Integrated motors and distance sensors to rails and measured the effects of their inputs and outputs. Developed a simplified control model. Finalized housing design requirements with the machine shop.	Wrote microcontroller code to receive text on bluetooth serial line and reply. Used a test application to communicate with microcontroller.	Use physical measurements to determine clearance between electromagnet housing and components
3/19	**Spring Break**	**Spring Break**	**Spring Break**
3/26	Finalized PCB layout and order it. More parts purchased. Physical rails and enclosure finalized.	Setback because iOS applications did not work with HC-05 bluetooth module. Expanded microcontroller code to output acknowledgement signals on text received from test app.	Test batteries in conjunction with voltage regulators to ensure proper functionality
4/2	Wrote pseudo-scheduler for micro control loop and snake search functions	Added delimiters in code to prevent data splicing on microcontroller end. Added support for all control signals (snake search, cancel word and verify coupling).	Begin process of soldering PCB. Continue battery safety and power testing.
4/9	Set back due to lack of functioning electromagnet system	Added a beta Android application with a basic log. Added queue support on microcontroller.	Aid in other aspects of the project, namely connectivity between sensors, microcontroller, and application
4/16	Debugged full system integrated system. Experimented with alternatives to pressure sensor.	Polished the log and debugged splicing issues with phone app. Soldered USB serial PCB and parts of AtMega PCB.	Prepare appropriate method of presentation for mock demo.
4/23	Prepared mock presentation.	Prepared for demo. Helped with alternative approaches (flex sensor) for pressure module.	Continue refining the product demonstration.
4/30	Worked on final report sections relating to motor modules, microcontroller, and PCB layout	Worked on final presentation slides, rehearsed the delivery and worked on final paper.	Divide up sections for report. Aid in both final report and presentation.

6 Conclusion

6.1 Accomplishments

Our team successfully developed a proof of concept for this bluetooth controlled ouija board. We were able to show all of the mechanical functionality, successfully created an app to interface with our system, and wrotre successful scripts to interface and control the various sensors and modules in our system. Our control algorithm was instrumental in ensuring a proper positional control of the magnet module, and we managed to overcome obstacles due to wear and tear on our motors.

6.2 Uncertainties

While our control algorithm and app interface were important in ensuring that the planchette moves to the right location, an integral part of the illusion is magnetic coupling. Due to lack of

proper sensor vetting, we did not choose the best method of implementing this integral functionality. In section 2.6.2, we discuss a different sensor that could potentially solve this issue. Additionally, when putting the lid on the actual body of the box, we experienced interference with our IR sensors. This caused inaccurate readings, that could have been mitigated if detected earlier.

6.3 Future Work

Coupling Recognition: For a fully functioning product, our first change would be to implement a method of consistently measuring the displacement between the decoupled and coupled state of the magnet module underneath the board. Since our original design didn't have a discernable difference between the two states, it was difficult to achieve a threshold value between the two. In our revised design, we used a flex sensor, but the inconsistency in readings because of distortions in the alignment of the sensor were problematic. An alternative approach would be to 3D print a wider sheet that behaves similarly to the flex sensor but would not be hindered by stability issues and therefore, less inconsistent readings.

Sound Reduction: Our prototype board generated a fair amount of noise with the DC motors and worm gears. Given that the original ouija board doesn't generate any noise, this could easily discredit our device and break the illusion. A few potential approaches to mitigating the noise are to soundproof the box, lubricate the worm gears and motors or to use less noisy motors entirely.

Randomized Movements: Another addition that could enhance the illusion is randomly moving the planchette after completing a word. Currently we have the control loop stop the motors for a few seconds at the last (x, y) position on the board after completing an input word. However, adding random movements between words would make the movement seem much less mechanical and more believable.

Spiral Search: An alternative search mechanism that we considered was a spiral search from a specified origin point, rather than a snake search from the origin (0,0). By looking at the board, the user could roughly determine where the planchette is sitting on the board and convey that character to the microcontroller. The microcontroller would direct the magnet module to that character's mapping on the board and then spirally search about that center point with a minimally increasing radius. This spiral would likely capture the planchette position and couple much quicker than the relatively slower snake search.

6.4 Ethical Considerations

The primary ethical concern associated with this project is the potential for it to ruin one's reputation. Our project can be described simply as the illusion that a supernatural being is

attempting to communicate with the user. Although it may sound ludicrous to some, others are easily convinced that such spirits may be trying to tell them things. This proposes the risk that the project we designed could be used to maliciously deceive people in such a way that harms their reputation. We've reviewed the IEEE guidelines for ethics[10] and although we cannot fully prevent someone from misusing our project to deceive or hurt someone's reputation, we have
considered this and have made it apparent that this product is purely for entertainment purposes.

References

[1] A. Romano, "How Ouija boards work. (Hint: It's not ghosts.)", *Vox*, Sept. 29 2017. [Online]. Available: https://www.vox.com/2016/10/29/13301590/how-ouija-boards-work-debunked-ideomotor-effect. [Accessed: 19- Feb- 2018]

[2] "Animated 'Haunted' Ouija Board," *instructables.com*, Nov. 7 2007. [Online]. Available: http://www.instructables.com/id/Animated-Haunted-Ouija-Board. [Accessed: 21- March- 2018]

[3] "Etch A Sketch," *wikipedia.com.* [Online]. Available: https://en.wikipedia.org/wiki/Etch_A_Sketch. [Accessed: 4- April- 2018]

[4] "Writing data to bluetooth peripheral fails," stackoverflow.com, Aug. 14 2016. [Online].Available:https://stackoverflow.com/questions/38945114/writing-data-to-bluetooth-peripheral-fails. [Accessed: 23- March- 2018]

[5] Parallax, "Parallax Feedback 360° High-Speed Servo", 900-00360 datasheet, Sept. 2017.

[6] Sharp, "Distance Measuring Sensor Unit", GP2Y0A41SK0F datasheet.

[7] GM Electronic, "HC-05 Bluetooth Module User's Manual", HC-05 datasheet.

[8] Atmel, "Atmel-42735B-ATmega328/P_Datasheet_Complete", ATmega328/P datasheet, Nov. 2016.

[9] Texas Instruments, "LP3869x-ADJ 1-A Low Dropout CMOS Linear Regulator With Adjustable Output", LP3869 datasheet, Dec. 2004 [Revised February 2016].

[10] "IEEE IEEE Code of Ethics", *Ieee.org*, 2018. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 19- Feb- 2018]

[11] "MN 908 Alkaline Lantern", *duracell.com.* [Online]. Available: https://www.duracell.com/en-ca/product/mn-908-alkaline-lantern/ [Accessed: 2-May-2018]

Appendix A: Requirements and Verifications

Table A.1 System Requirements and Verifications

Requirement	Verification	Verification Status (Y/N)
Phone Application		
● Phone application follows the state diagram in figure 2.4 with no error.	1. Try all state combinations and look at the phone screen to see that the correct next state behavior is achieved by application.	Y
● Receive and display information about the state of the planchette (coupled or not) according to the microprocessor with no error, and a delay of less than five seconds.	1. Apply pressure to the pressure sensor and look at the phone screen to see that the that coupled acknowledgement is received from microcontroller. Visibly confirm coupling LED is on. 2. Remove pressure and start a stopwatch when LED shuts off. Verify visually on phone screen that decoupled acknowledgement is received before five second mark.	Y
● Indicate to microcontroller that the planchette is on the board and it can begin to snake search to look for it.	1. Send “snake search” signal to microcontroller from phone app. 2. Observe visually that the snake search functionality begins on the electromagnet module.	Y
Pressure Sensor		
● Microcontroller detects absolute pressure within the threshold of [121.57, 121.57+mag force] within a resolution of +/- 0.25%, as per sensor documentation.	1. Attach pressure sensor to microcontroller. 2. Apply various amounts of pressure and read output. 3. Program behavior for pressures above and below threshold. 4. Used for coupling detection.	N/A

• Maintain supply current at $30\mu A \pm 5\%$.	1. Ensure current to pressure sensor is within tolerance range with use of an ammeter.	N/A
• Maintain supply voltage range between 2.7 V - 3.3 V.	1. Use oscilloscope to measure open circuit voltage of sensor and make sure it is within voltage range. 2. Using the same method as above, make new measurements once pressure sensor is pressurized and ensure voltage stays within the tolerance.	N/A
Electromagnet		
• Maintain a current of 44.9 mA +/- 5% through electromagnet.	1. Wrap coil around predetermined number of zinc washers. 2. Use oscilloscope to provide a current through wire. 3. Through intuition, determine needed current to successfully attract planchette through the board. 4. Update current requirement value based on experimentation.	N/A
DC Motor with Encoder		
• Rail stops within 1.25 cm radius of the desired location.	1. Set a desired location location from the microcontroller. 2. Measure distance of rail from desired physical location with a ruler along the major axis and the minor axis. Using pythagorean theorem calculate radius of error.	Y
• Rail moves to desired position with no overshoot.	1. Remove top of ouija board. 2. Set a desired location from the microcontroller with x and y location greater than that of the initial positions of the rails. 3. Using a pencil record on the interior how far it traveled past the final x and y coordinates. If it moves past at all then it fails the test.	Y

	4. Repeat steps 2 and 3 with a desired location with a smaller x and y location than the current rail position.	
● Relay physical position to microcontroller within a margin of 0.75 cm.	1. Attach oscilloscope lead at the positional data output. 2. Move the rail to an arbitrary position. 3. Read scope data and confirm measurement is accurate.	Y
Bluetooth Module		
● Can properly communicate a single word (10 bytes/second) with the phone application from within 5 meters.	1. Send signal from the phone app and utilize a measuring tape to check the maximum distance that the BT module can receive signals from.	Y
● Can receive entire words in the correct order from application without dropping packets over 80% of uses (BLE has no mechanism to recover lost packets).	1. Send multiple words via phone app. 2. Verify visually on phone screen that bluetooth module sends back acknowledgement signal for each word. 3. Ensure they're received in the correct order on the microcontroller queue.	Y
Microcontroller Module		
● Can build a queue of ten characters that need to be visited by the planchette based on input from the phone application.	1. Send a string of ten characters to the microcontroller. 2. Confirm that the rails move to the positions corresponding to the characters by measuring the locations they stop at. Also confirm that the order these positions are reached is consistent with the input.	Y
● Output to motor changes within 160 ms after a change in the distance	1. Attach an oscilloscope lead to the output to the motor, and a second oscilloscope lead to the input from	Y

signal.	the distance sensor. Set the oscilloscope to roll mode. 2. Place an obstruction in from of the distance sensor to spoof the appearance of the position being off. 3. Confirm that the duty cycle being sent to the motor changes within 160 ms from the change in the distance sensor signal.	
● Performs snake search function upon request of the phone application.	1. Have the phone application send the signal to perform the sweep function while pressurizing pressure sensor. 2. Confirm that the electromagnet module is search across the x and y directions. 3. Remove pressure from the pressure sensor and confirm that this stops the sweeping.	Y
Batteries		
● Provide up to 5 hours of run time.	1. Connect batteries to system 2. Continually all parts of system 3. Ensure the batteries are functional for up to hours.	Y
Voltage Regulators		
● Supply 5 V +/- 5% to bluetooth module, ATmega328P, and IR sensors.	1. Perform LTSpice simulation on regulator schematics with different parameters to ensure that voltage stays within range.	Y
● Supply 6 V +/- 5% to each DC motor.	1. Complete same procedure as above.	Y
● Supply 3.3 V +/- 5% to pressure sensor.	1. Complete same procedure as above.	Y
● Output of 0V +/- .05 V. when enable is low.	1. Use regulator on PCB. 2. Use voltmeter to measure output.	Y

Appendix B: Schematics

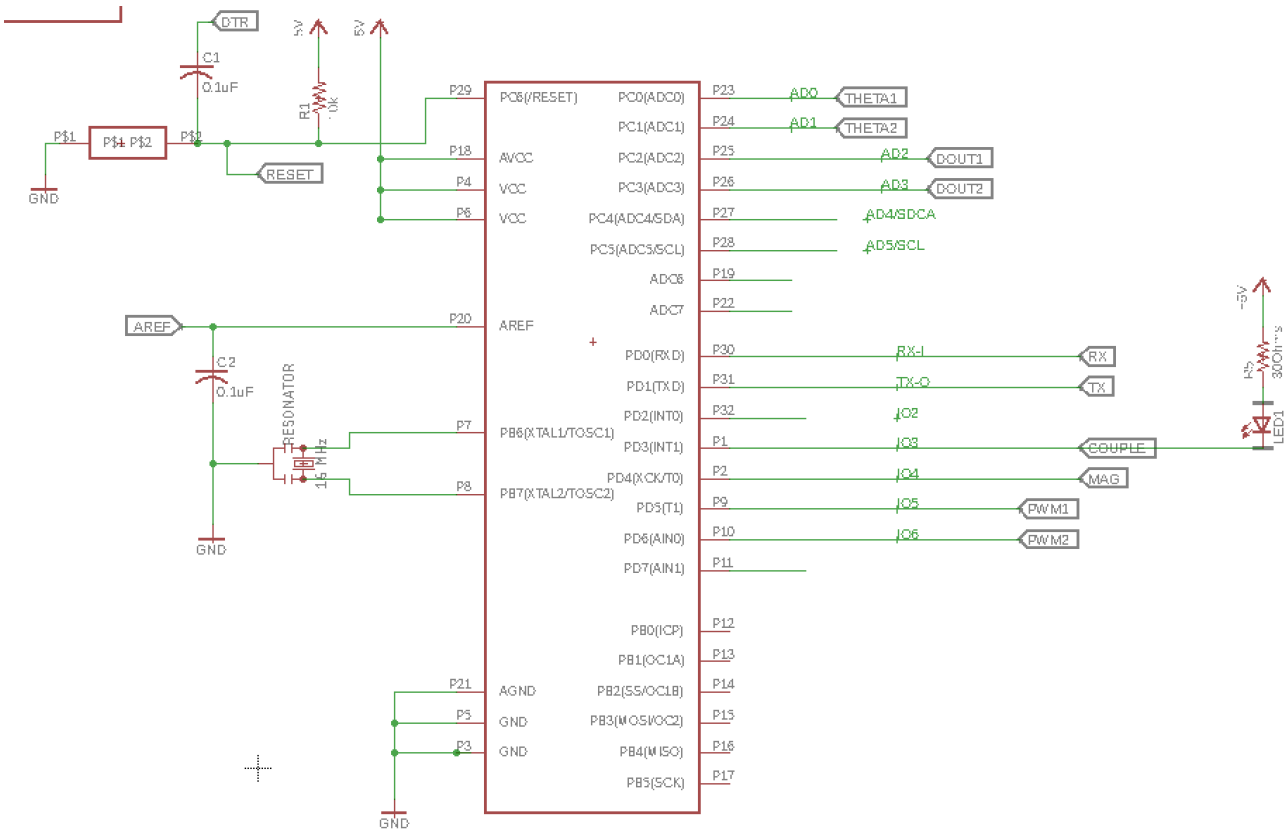


Figure B.1: Microcontroller[8] Schematic

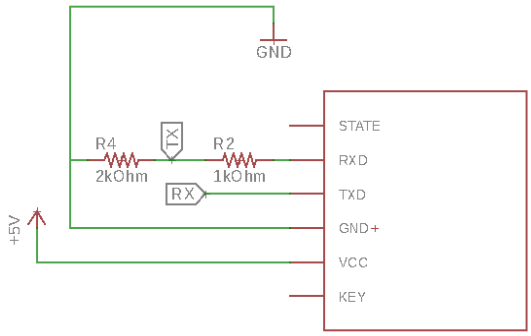


Figure B.2: Schematic of Bluetooth[7] module circuit

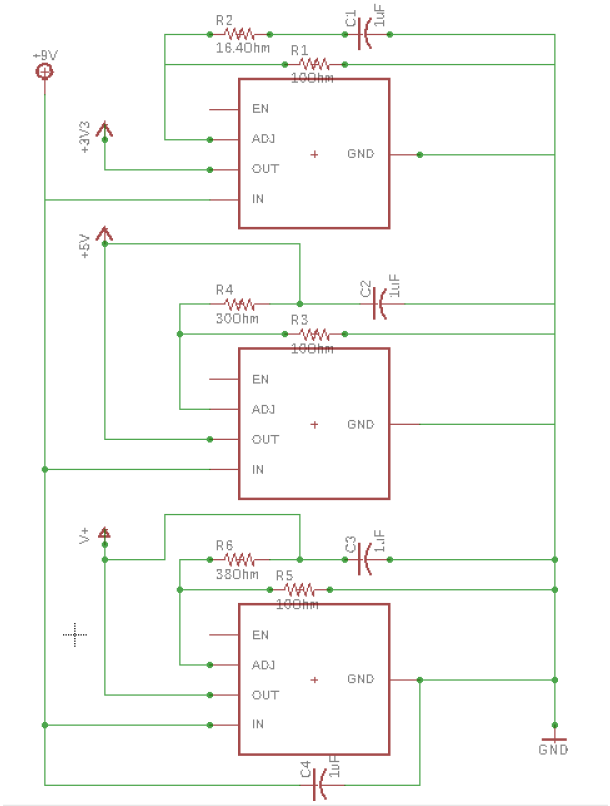


Figure B.3: Voltage regulator[9] schematic and related routes

Appendix C: Block Diagrams

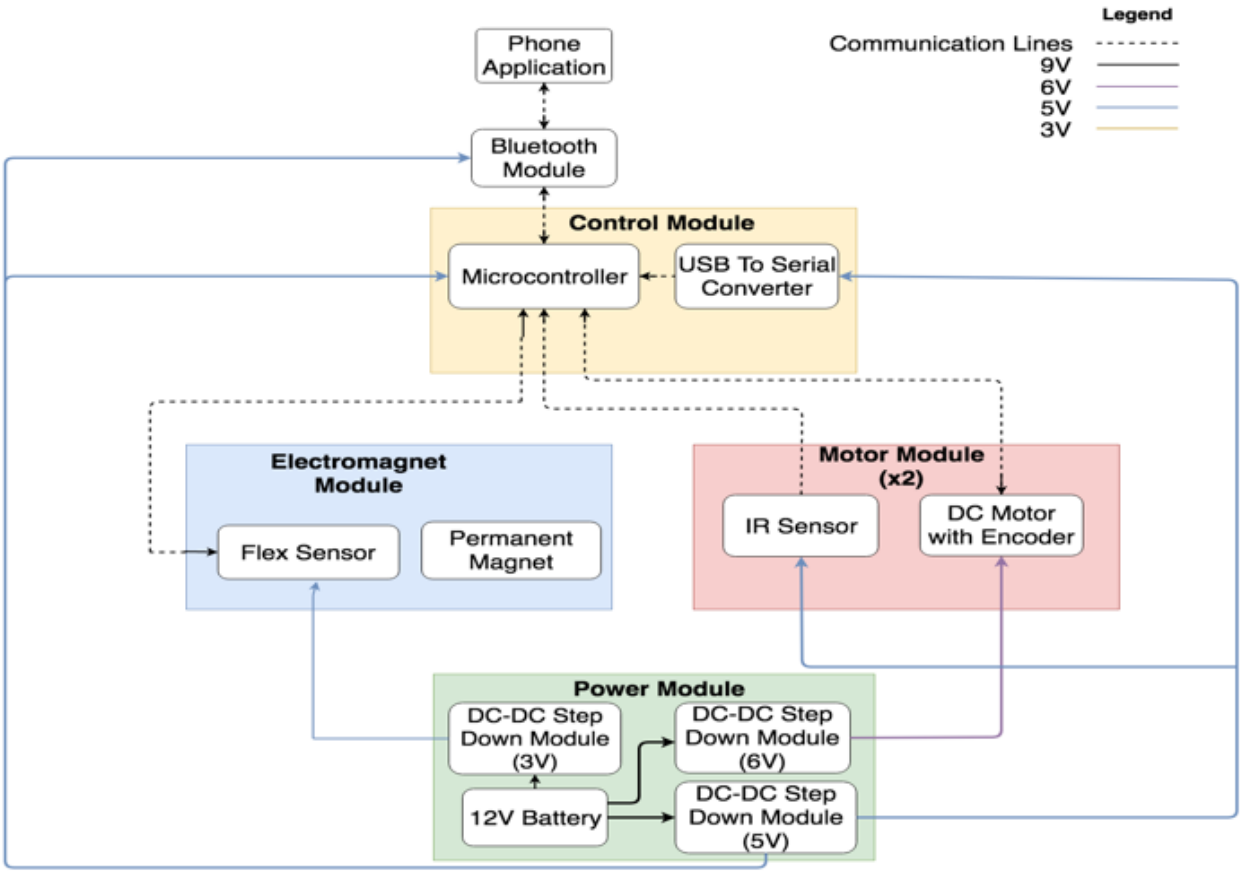


Figure C.1: High Level Block Diagram