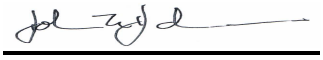
ECE 477 Final Report – Spring 2013
Team #20 – Marble Maze

Left to Right: Mark Sears, John Jachna, Justin Spencer, Jordan Wagner

Team Members:

#1: Mark Sears **Signature:**  **Date:** 4/29/2013

#2: Justin Spencer **Signature:**  **Date:** 4/29/2013

#3: John Jachna **Signature:**  **Date:** 4/29/2013

#4: Jordan Wagner **Signature:**  **Date:** 4/29/2013

CRITERION	SCORE	MPY	PTS
Technical content	0 1 2 3 4 5 6 7 8 9 10	3	
Design documentation	0 1 2 3 4 5 6 7 8 9 10	3	
Technical writing style	0 1 2 3 4 5 6 7 8 9 10	2	
Contributions	0 1 2 3 4 5 6 7 8 9 10	1	
Editing	0 1 2 3 4 5 6 7 8 9 10	1	
Comments:		*TOTAL*	

Table of Contents

Abstract

Our team is implementing a smartphone controlled marble maze. Players guide a marble through a maze by controlling the degree of tilt of the playing surface via a wireless link to a smartphone. The accelerometers in the smartphone capture the players' input and relay it to the game controller. The game controller then adjusts the tilt of the playing surface via stepper motors on each axis to match that of the smartphone. This causes the marble to move along the surface and the time taken to traverse the maze is recorded by sensing the marble as it passes between IR gates at the start and finish.

1.0 Project Overview and Block Diagram

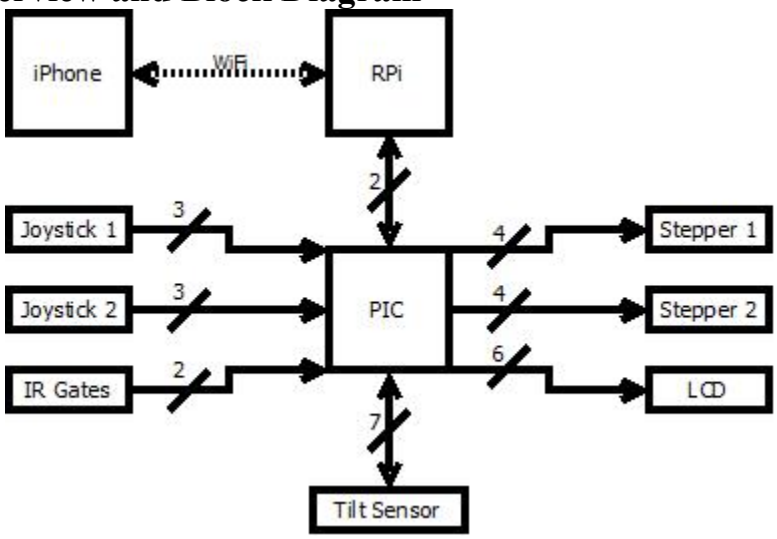


Figure 1.1 - Block diagram of the Marble Maze

The marble maze uses multiple peripherals to take input on the current status. The joysticks and iPhone can be used to control the tilt of the board, and the tilt sensor and IR gates give feedback on what is happening in the game. Results and a basic user interface is displayed on the LCD screen, and sounds and statistics can also be viewed on the iPhone app.

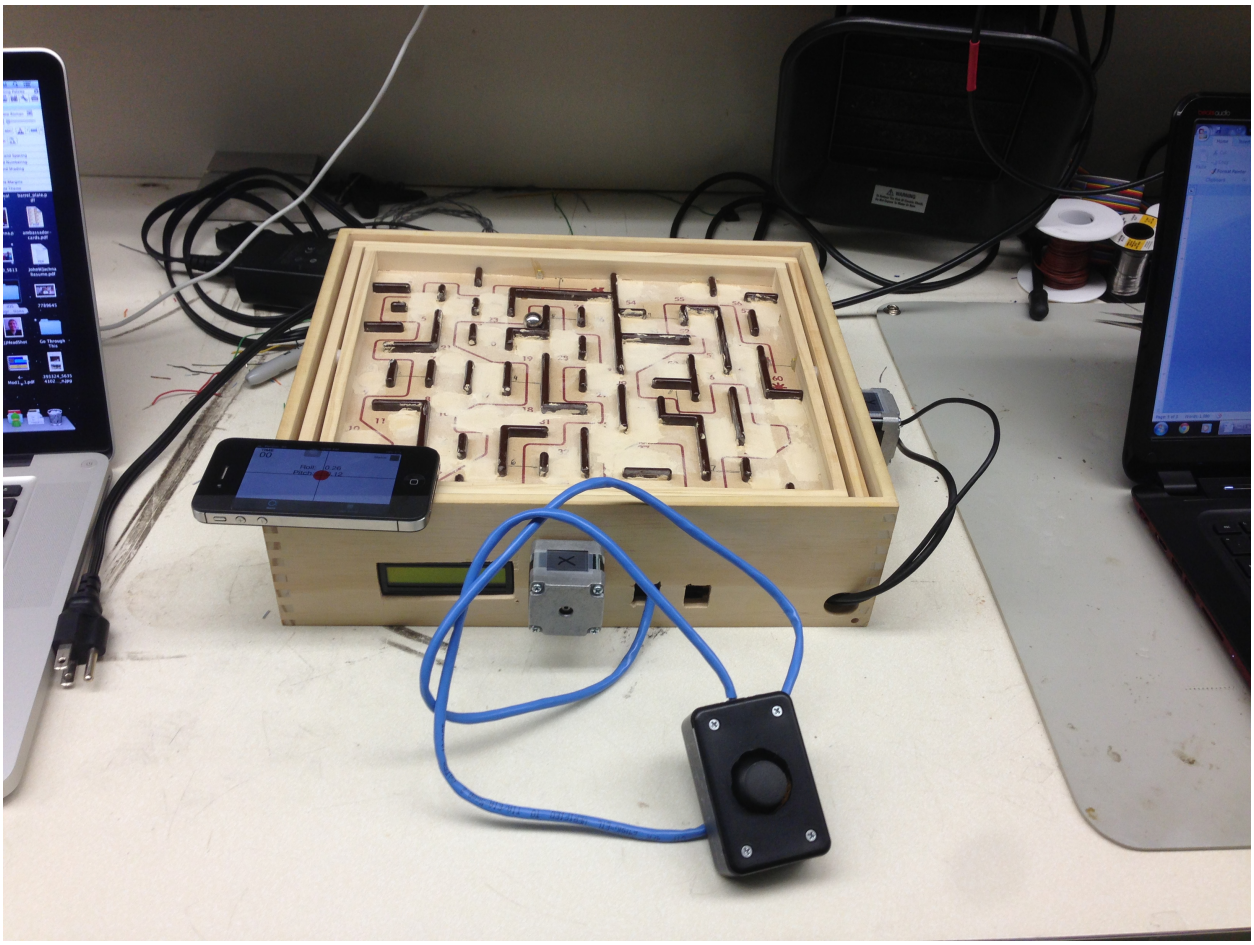


Figure 1.2 - Final Product

2.0 Team Success Criteria and Fulfillment

1. An ability to manipulate the playing surface via stepper motors

Fulfilled.

2. An ability to control the playing surface with accelerometers in an iPhone and resistive joysticks

Partially Preliminary

3. An ability to detect the player's failure or success using IR gates

Preliminary

4. An ability to display game statistics & configuration information on an LCD and play game related tones on a speaker

Preliminary

5. An ability to store separate, specific game data on the board and on the iPhone

Preliminary

3.0 Constraint Analysis and Component Selection

The major design constraints for the Marble Maze come from the wireless communication with the iPhone and the motor control of the board. In order to provide the best control of the board, it is necessary to use motors which are both very accurate and also powerful. We chose stepper motors because of their precision and feedback with step counting. The computation requirements are low, as the motors can only step, at the fastest, once per 10ms. This gives us plenty of time to read analog inputs from the joysticks, display the time on the LCD, and monitor the game state. We did not have any major power constraints because nothing ran off of a battery. The power usage is about 27W, with a 9V wall wart supplying around 3A to our design, when the motors are enabled. Our only major packaging constraint was that everything fit in or be mounted onto our marble maze. Because we have plenty of room on the bottom of the maze, packaging was not an issue.

3.1 Computation Requirements

Our device has relatively low computational requirements. The main task the microcontroller must perform to implement basic functionality is receive data serially from the wireless module, use an algorithm to translate that data and the current position of the board into a direction to turn the motors, and then drive the motors. The microcontroller will be chosen so it has hardware support for the serial communication between the wireless module. For processing how to move the board, the microcontroller will have to read data from the tilt sensor attached to the board, use a stored variable for how many steps off of level the stepper motors are, and decode the rotation of both axis of the phone, as well as compare them to the last known angle. It can then decide how to drive each stepper motor. Some non-critical functions the microcontroller will perform are detecting if a ball has fallen through a hole, possibly by using a sensor and an interrupt, playing different sounds, keeping track of time elapsed for the game, displaying information on LEDs/an LCD, sending data back to the phone through a serial connection to the wireless module, and storing game data. These functions will not happen as frequently and can be performed in between packets received. The only real-time constraint of the device will be to perform motor control before the next packet arrives.

3.2 Interface Requirements

Our device will be interfaced through UART or SPI to the wireless module, requiring two wires to transmit and receive. An SPI port will be used while debugging to transfer data. ADC will be used to read data from sensors, specifically two for a joystick used for debugging and two for a tilt sensor/gyroscope mounted to the board. PWM will be used to control a speaker and numerous LEDs. Two pins will be used for the PWM and an additional for a clock, which will go to a PWM enabled LED driver attached to the LEDs. Four GPIO ports will be used to control the stepper drivers, which will be optically isolated from the two motors.

3.3 On-Chip Peripheral Requirements

For serial communication, the chip needs to support two ports for UART, one for transmitting and one for receiving for the wireless module. A port to transmit over SPI will also be needed for debugging. At least four ADC will be needed, but they do not need a high bit accuracy as it will be for reading from potentiometers in the joystick and gyroscope headings. Two channels of PWM will be needed, one for the speaker and one for LEDs. An input capture timer channel will be needed to generate an interrupt if the ball falls through a hole. The LCD display can use the SPI port previously used for debugging, as long as it is serial enabled. About eight or more GPIO will be needed for ball sensors, LED shift register drivers, and motor drivers.

3.4 Off-Chip Peripheral Requirements

To implement the wireless communication, our design needs a Raspberry Pi with compatible USB wireless dongle. To rotate the board, two stepper motors will be used, as well as two stepper motor drivers. A wall wart will be used to provide power to all devices on the board that require it. A joystick will be used for debugging, and the board will have LEDs, a speaker, and an LCD mounted to it. Underneath the board, there will be a tilt sensor to help level the board and provide feedback, as well as sensors to detect when the ball has fallen through a hole.

3.5 Power Constraints

Our device will be A.C. powered only and the only major source of power dissipation would be from the motors. After designing the voltage regulators and BJTs/MOSFETs for the motors, heat dissipation will be taken into account and a sizeable heat-sink will be used, if needed.

3.6 Packaging Constraints

The main packaging constrain is that our circuit board needs to fit on the underside of the game board. Depending on the design of the game board, a failsafe might be needed to prevent the motors from burning out if they try to tilt the board too far for too long. The game is already designed to be played on a flat surface, so any packaging issues would only arise from mounting additional features onto the game board itself.

3.7 Cost Constraints

Currently, one of the most popular smartphone paired games is the Nerf Lazer Tag . It uses a Nerf gun with an iPhone attached to create a virtual world, generating enemies by adding them to the screen in front of what the camera sees. If two or more players are fighting, the guns can detect when they are shot at using IR transmitters and receivers. The devices plug into the phone, eliminating the need for additional wireless support. A set of two blasters currently retails for around $75. Our device will be competing with this other iPhone enabled toy.

However, because we need additional support for wireless capabilities and are not fabricating our own game board, we expect our device will initially cost more to make, ending up being around $150. The Raspberry Pi and wireless dongle will account for the bulk of the cost, totaling $50. The game board itself will be around $25 and the stepper motors will cost $25 together. The microcontroller and most of the circuit components will be around $25 and the other $25 will come from the speaker, LCD, LEDs, and power supply.

3.8 Component Selection Rationale

When choosing a microcontroller, the on-chip peripheral requirements were the main thing considered, as the computational requirements were low. Two candidates were the PIC18F67J94 [1] and the Freescale MC68HC908GP32 [2]. Both met our minimum requirements for peripherals, but the PIC was cheaper, had more peripherals, more memory, and could operate at a faster clock speed.

For the wireless module, there were three distinct choices. Either to use a standalone wireless device, such as the WizFi210, use a laptop with a USB interface to a wireless transceiver, such as an Xbee, or use a Raspberry Pi with a wireless dongle. The standalone wireless device was the cheapest, but it was the toughest to interface to, as all the configuration had to go through the microcontroller. Using an Xbee and laptop didn't seem like a good solution because it would require the user to own both and iPhone and a computer. Although expensive, using a Raspberry Pi seemed to be the best solution because it is easily configurable, powerful, and small enough to fit on the game board.

Two main choices for motor control were servo motors or stepper motors. Servo motors provide only a limited range of motion, which, if geared properly, could be an advantage because our board only has a limited range of motion as well. However, stepper motors are accurate and easier to track how far they've turned, providing another form of feedback along with the tilt sensor on the board.

4.0 Patent Liability Analysis

The Marble Maze, also known as the Labyrinth toy, has been around for quite a while. The table top tilting game board is older still. The concept, similar to that of pinball or other arcade games, has spawned numerous similar products that involve a marble. Consisting of a tilting platform, and a marble that must navigate through obstacles, there are many similar products and patents to the game, but most if not all of these patents are from the 1970's and are now expired. Infringement of the game itself is not really a major concern, but nonetheless one such patent will be looked at later.

4.1 Results of Patent and Product Search

Below are 3 patents that may cause concern for infringement for our project. Highlighted with each claim are specific phrases for consideration.

Patent 1: Tilt Board Game [3]
Inventor: Sandy F. Kraemer
Filed: July 27, 1972
Abstract: A tilt board game having a frame, a tiltable playing board pivoted at a single point on one side of the frame, and spring supported on the opposite side. Consisting of barriers on the top of the playing surface to guide the ball, and a plurality of holes each with point values spaced around the playing board.
Claims that may cause infringement:
1. A tilt board game for use with a sphere comprising:
a. frame means including a front wall and rear wall,
b. game board means having front and rear edges and a flat planar upper surface including a peripheral wall for retaining a sphere rolling on said surface, said game board means being mounted to said rear wall by a pivotal attachment means centrally positioned at said rear edge,
c.spring means tiltably supporting said game board means in said frame means at said front edge whereby said game board means may be tilted simultaneously about a longitudinal axis and a transverse axis extending through said pivotal attachment means, said spring means being a single compression spring having a generally large diameter centrally positioned near said front edge for biasing said game board means when in operation in a generally horizontal position slightly tilted toward said rear edge,
d. handle means attached to and extending outwardly from said front edge of said game board means through at least one vertical, elongated opening in said frame front wall whereby said game board means may be moved against said spring means for limited tilting around said two axes,

e. a plurality of bores provided in said surface to accept said sphere, said bores being arranged an spaced adjacent said circuitous path to trap and impede the travel of the sphere to test the skill of a player manipulating said handle means.

Patent 2: Motor speed control system [4]
Patent Pending
Inventor: Miu Kwan Cheuk
Filed: March 4, 2011
Abstract: A motor control system which can be connected with Wi-Fi and Bluetooth, comprising of a motor, a power supply, and a cooling fan. There is also a speed control circuit connected with the cooling fan, wherein there is a mode of Wi-Fi and/or Bluetooth in the control circuit, of which it can connect with all kinds of devices which have Wi-Fi.
Claims that may cause infringement:
1. A motor control system which can be connected with Wi-Fi and Bluetooth, comprising of a power supply, a motor, a cooling fan;
the power supply, the motor and the control circuit board are connected with the cooling fan; there is also a mode of Wi-Fi and/or Bluetooth in this control circuit board; the present utility can connect with all kinds of electronic devices which have a mode of Wi-Fi and/or Bluetooth.
2. A motor control system as in claim 1, there is also a button which is connected with the cooling fan for mode operation conversion.

Patent 3: Stepper motor control [5]
Inventor: Robert Pulford Jr.
Filed: June 1, 1999
Abstract: A stepper motor control, including: a driver to drive the stepper rotor; a detector to detect a command step rate signal provided to the driver and to provide an output signal, to provide driving power to the driver, and to increase voltage to the driver in proportion to an increase in magnitude of the output signal. Voltage control can be affected either on the drive board or by controlling the power supply.
Claims that may cause infringement:
1. A stepper motor control, comprising:
a) driver means to drive said stepper motor;
b) detecting means to detect a commanded step rate signal, derived from an on-board or an external source, provided to said driver means, and to immediately convert said commanded step rate signal to an analog output signal proportional thereto;
c) power control means, responsive to said analog output signal, to provide electrical driving power to said driver means, and to increase voltage to said driver means in proportion to an increase in magnitude of said analog output signal.

4.2 Analysis of Patent Liability

For the first patent, there are many similar patents that can be considered very similar to the project's game, however none of them were filed any later than 1980, so all of these have long since expired and should not be a worry to our project now. In the claim, “game board means being mounted to said rear wall by a pivotal attachment means centrally positioned at said rear edge” sounds very similar to the knobs that we have used to tilt our game board. However, specific wording such as "spring means tiltably supporting said game board" is different enough that the claim would not be infringed upon. The game board we are using has no spring for support, and instead is tilted on two axes by the turning of a rod that winds a string. In this case, the function of tilting the playing board and manipulating the marble is very similar, the means by which is it accomplished is very much different.

The 2nd patent claims a motor control with a cooling fan. Not having a cooling fan, but instead utilizing a basic heat sink separates our project from the claim. Also, using a MCU as the brains of the device instead of just having the motor control and fan furthers this. In the claim, “the present utility can connect with *all kinds* of electronic devices which have a mode of Wi-Fi” but our project plans for only an iPhone with a specific app to be able to interface with the Raspberry Pi module, so this again should not pose a problem.

However, the claims made in the stepper motor control patent share many similarities to the method we are using to control our game board. The claim for “driver means to drive said stepper motor” is in our case taken care of by the Step Genie, a chip designed to drive a stepper motor. Since our stepper motor driver IC is a packaged product that we are not liable for, so its use is not a problem for our project. The claim also states that the commanded step rate is “derived from an on-board or external source”. Well, this is pretty all-encompassing, so it is somewhat difficult to avoid this particular part of the claim. Also, “increase the voltage in proportion to an increase in magnitude” does not seem to cause an issue because the stepper drivers will output a set voltage to step the motor, and should not vary in proportion to anything. Instead it acts in a very binary fashion, either on for a single step, or off.

4.3 Action Recommended

For the game board itself, no action is required. As these patents have been expired, the worry of infringement no longer exists. To avoid the 2nd patent, it simply a matter to limit the control to iPhones over the Raspberry Pi connection and not allow for other devices, as it says in the claim. Also, it is worth noting again that this method is patent pending.

For the 3rd patent, to keep from infringing we must ensure that the IC we are using prevents us from being liable. We must also keep from increasing the driver voltage in a proportional manner. However, the means for detecting the step command signal is still in question, and perhaps should be clarified with the stepper driver manufacturer.

5.0 Reliability and Safety Analysis

5.1 Reliability Analysis

In order to determine the reliability of this design, several parts were chosen and analyzed based on several factors. The analysis was done on parts that had a high likelihood of failure, contributed integrally to the entire function of the product, or that could cause physical harm or raise safety concerns to users of the product. Based on these concerns, the parts chosen for analysis were the Fairchild IRF530 power MOSFETs, the Texas Instruments LM2576 Voltage Regulators, and the Microchip PIC18F67J94 microcontroller.

Calculations for the Fairchild IRF530 were performed based on the low frequency transistors specified in section 6.4 of the MIL-HDBK-217F handbook [6] and based on information from its datasheet [7]. The most critical reliability and safety issue in this project revolves around the power MOSFETs that driver the stepper motors. These components have the potential to get extremely hot (170° C maximum junction temperature), and if the part fails in the "stuck closed" position, it could generate extreme amounts of heat (by sourcing 1A of current) that could ignite the wooden game board and endanger the safety of the users. To reduce this risk, a temperature sensor could be installed that would open the circuit if the MOSFETs exceeded the recommended temperature for an extended period of time.

Parameter name	**Description**	**Value**	***Comments***
λ_b	Base Failure Rate	0.0045	Power MOSTFET Value
π_T	Temperature Factor	8.3	170° Max Junction Temp.
π_A	Application Factor	8.0	Max Pow. Dissipation 79W
π_Q	Quality Factor	5.5	Non-Military Grade
π_E	Environment Factor	6.0	Ground, Fixed
Entire design:	$\lambda_p = \lambda_b\pi_T\pi_A\pi_Q\pi_E$	9.8604×10^{-6} units/hr	**MTTF:** 11.5694 years

Figure 1: Fairchild IRF530 MTTF Calculations

Calculations for the Texas Instruments LM2576 were based on the guidelines for low frequency diodes in section 6.1 of the MIL-HDBK-217F handbook [8] and based on data from its datasheet [9]. This component is present in both the 3.3V and 5V power supplies. The biggest risk associated with failure of the voltage regulator would be overpowering other components on the printed circuit board and destroying those components due to the voltage overload. This could cause a chain reaction that leads to other failures and irreparability of the game. To try to mitigate the risks of this failure, fuses could be inserted in the circuit to break power to the rest of the board under failure conditions.

Parameter name	**Description**	**Value**	***Comments***
λ_b	Base Failure Rate	0.0020	Voltage Regulator Value
π_T	Temperature Factor	6.7	150° Max Junction Temp.
π_S	Electrical Stress Factor	1.0	Voltage Regulator Value
π_C	Contact Construction	1.0	Metalurgically Bonded
π_Q	Quality Factor	5.5	Non-Military Grade
π_E	Environment Factor	6.0	Ground, Fixed
Entire design:	$\lambda_p = \lambda_b\pi_T\pi_S\pi_C\pi_Q\pi_E$	4.422×10^{-7} units/hr	**MTTF:** 257.9818 years

Figure 2: Texas Instruments LM2576 MTTF Calculations

Calculations for the Microchip PIC18F67J94 microcontroller were performed based on microcircuit section information specified in section 5.1 of the MIL-HDBK-217F handbook [10] and based on information from its datasheet [11]. The biggest reliability issue with this component comes from bugs in the microcontroller code or incorrect inputs from peripherals that cause the microcontroller to output incorrect values. This could be

Parameter name	**Description**	**Value**	***Comments***
C_1	Die Complexity	0.14	8-bit microprocessor
π_T	Temperature Factor	5.6	150° Max Junction Temp.
C_2	Package Failure Rate	0.032	64 pin surface mount
π_E	Environment Factor	2.0	Ground, Fixed
π_Q	Quality Factor	5.5	Non-Military Grade
π_L	Learning Factor	1.0	Production > 2 years
Entire design:	$\lambda_p = (C1\pi_T + C_2\pi_E)\pi_Q\pi_L$	2.5088×10^{-7} units/hr	**MTTF:** 454.7176 years

Figure 3: Microchip PIC18F67J94 MTTF Calculations

5.2 Failure Mode, Effects, and Criticality Analysis (FMECA)

For this design, there are three major functional blocks that could fail: the power MOSFETs, the power supply voltage regulators, and the microcontroller. For each of these blocks, failure modes, effects, and criticality will be analyzed. Before the analysis is discussed, the criticality levels need to be defined:

Criticality Level	Acceptable Failure Rate	Description
High	10^{-9} failures per hour	Personal injury can occur or safety issues arise
Medium	10^{-7} failures per hour	No personal injury occurs. Part failures could destroy other parts or the entire product
Low	10^{-6} failures per hour	No personal injury occurs. Part failures not are inconvenient to user but can be replaced in the field. A failure will not damage other parts

Figure 4: Criticality Definitions

Power MOSFETs: The power MOSFETs have two main failure modes. The first is the condition where the component fails in a stuck open state. In this case, the microcontroller or the motor drivers provide a constant 0 to the MOSFET, causing it to constantly be off. This would cause motors being drive to be unresponsive. It has a low criticality since there is no potential harm to either the user or other circuit components. The second condition is where the component fails in the stuck closed state. In this case, the microcontroller or the motor drivers provide a constant 1 to the MOSFET, causing it to constantly be on. This failure could have dangerous side effects, since sourcing the max current of 1A could generate a large amount of heat that could be capable of igniting the wooden game board. Because of this issue, this failure mode has a high criticality.

Power Supply: The power supply has three main failure modes. The first is a power supply that outputs 0V. This could be caused failure of any of the components that make up the power supply circuit (inductors, capacitors, diodes). A failure of this kind will result in a lack of power to the circuit. Since this is only an inconvenience to the user and doesn't compromise user safety or other parts, it has a low criticality. The second mode is a power supply that outputs more than the expected voltage (either 3.3V or 5V). This could also be caused failure of any of the components that make up the power supply circuit. A failure of this kind will result in the overpowering of components on the printed circuit board, which could in turn destroy those parts or the physical game board. Because of the risk it poses to the reparability of the game, it has a medium criticality. The final mode is a power supply that outputs out of tolerance. This could again be caused failure of any of the components that make up the power supply circuit. A failure of this kind will result in unpredictable output. The output could be over or under the rated power, and there is criticality level medium.

Microcontroller: The microcontroller has two main failure modes. The first is output pins that are constantly outputting 0. In this mode, failure could be caused by software bugs or by failed peripherals that are sending incorrect values. In this mode, peripherals that the microcontroller is talking to could be unresponsive. This failure will only inconvenience the user, so it has a low criticality. The other failure mode is an output pin constantly at 1. Again, this could be caused by software bugs or by failed peripherals that are sending incorrect values. Under most circumstances, this will only cause low criticality errors, but in the case a motor control pin is stuck at 1, it would turn the MOSFET on in the always-closed position, leading to the fire risk described above. Because of this, it garners a high criticality.

6.0 Ethical and Environmental Impact Analysis

6.1 Environmental Impact Analysis

Our project has several stages in its lifecycle, and each stage has associated with it its own impact on the environment. Some of these issues can potentially be eliminated, but many cannot. In these instances we can implement solutions to reduce or mitigate the impact the issues will have.

The product life-cycle includes the following stages:

- Sub-component manufacture
- Sub-component shipping
- Assembly
- Shipping
- Deployment
- Disposal

Our project utilizes several other commercial off-the-shelf products to achieve its functionality. These devices include the Apple iPhone, a Raspberry Pi single-board computer, a USB Wi-Fi adapter, a custom designed printed circuit board (PCB), stepper motors, and a Brio Marble Maze.

The primary environmental issue associated with the sub-component manufacture stage of the device's life-cycle is initial resource usage and the byproducts created during manufacture. Each device requires raw materials to manufacture it. These materials include plastics, metals, and—in the case of the marble maze—wood. All of the electronic sub-components have PCBs inside them. PCBs are manufactured using harsh toxic chemicals [12], such as: ammonium persulfate [13], ferric chloride [14], and lead [15]. By incorporating more functionality into the custom PCB, its component count, and size could be reduced. This would decrease the amount of materials and chemicals used during its production. We could also find substitute components that are made from RoHS [16] compliant manufactures. The Marble Maze is made of wood [17], which isn't hazardous but is a valuable resource. To prevent contributing to deforestation, the wood should come from a sustainable source or recycled material, as our project doesn't require solid-wood construction.

The sub-component shipping stage generates the most excess waste. Each sub-component comes from a different location, is shipped separately using fossil fuel based transportation methods, and uses considerable amounts of packaging materials (i.e. cardboard boxes, packaging peanuts, etc.). The iPhone is manufactured in China [18], the Raspberry Pi in the UK [19], and the Marble Maze in Sweden [20]. Again, by incorporating more functionality into the PCB it's possible that the need for the Raspberry Pi could be completely eliminated, which would remove all of the environmental issues surrounding its use. Also, components could be purchased in bulk to reduce the volume of shipping materials required for packaging.

The assembly stage produces waste in the form of discarded components that aren't required in the final product. For example, the control knobs and part of the control rods are removed from the marble maze to make space for the stepper motors. This waste could be eliminated if the marble maze could be purchased in an unfinished form, including only the parts to be used. If we could find a supplier willing to provide individual maze components we could reduce the much of the waste associated with this stage.

The shipping stage consumes yet more fossil fuels and packaging to deliver it to its final destination. Fuel consumption could be reduced by decreasing the weight of the final product, which would decrease the energy required to relocate it. Only recycled and biodegradable material could be used to package the device for transport, which would mitigate the impact of this stage.

Once the device is deployed to the consumer its environmental impact comes from the electricity used to power the device. Unfortunately, in the U.S. this electricity is most likely generated at a coal-fired power plant [21]. To reduce the impact of this, the device could be made more energy efficient. A master power on/off switch could be added so no energy is wasted while not in use. A timer and sleep mode could also be used to put the device in a low power mode if accidentally left on. Also, through improved design, the power requirements of the device could be lowered by using lower voltage components and smaller more efficient motors. By utilizing high quality components, and over-building the most failure prone elements of the product, we hope to increase its useable lifetime. This would keep units out of the disposal phase for longer periods of time and potentially reduce the total number of units to be created. The most likely areas of failure include the power regulators, the microcontroller, and the power MOSFETs. The reliability of these parts could be improved by selecting components with higher environmental tolerances [22] (i.e. military grade ICs) and by using heat sinks to keep critical components cool.

Once the device enters the disposal phase of its life-cycle recyclability becomes the main concern. Electronics are difficult to recycle by nature [23], so it's important that the device be easy to dissemble into its recyclable and non-recyclable parts. By making this separation easier, we hope to encourage recycling and dissuade users from treating the entire assembly as garbage. If financially feasible, free return shipping could be provided to the consumer to return the device to a recycling facility that could properly dispose of the unit.

It is not possible to completely eliminate the environmental impact of our project, but through the use of smart design and better component and manufacturer selection, it is possible to reduce the overall impact. Many of the solutions outlined above could easily be incorporated into our project, and in the case of large scale production, several more solutions would become applicable as well.

6.2 Ethical Challenges

The primary ethical challenge facing our device is product safety [24]. Potential hazards include electric shock, chocking on the marble, and getting a body part pinched by the playing surface. Some of these risks can be reduced substantially, but none can be completely removed. It's important to keep in mind that there will always be a potential for personal injury, and the goal is to make that potential as small as possible. Other ethical issues we must address include providing the consumer with a reliable and long-lasting product in a way which doesn't increase the environmental impact of the device.

The risk of electrical shock can be reduced by ensuring all electric components are physically isolated from the user by placing them inside the housing of the device and using appropriately sized and insulated wiring. The device should also be sealed, such that the user has no access to the electronic components. All external metal components should be grounded, and a fuse or circuit breaker to limit the device current should be incorporated.

The choking hazard could be eliminated by sealing the marble in the playing surface with a transparent plastic overlay. This would allow gameplay to proceed as normal, while removing access to the item small enough to choke on. The overlay should be constructed from a shatterproof material that doesn't introduce any additional hazards. Alternatively, a minimum age limit could be suggested to prevent the most at risk users, children under four [25], from coming in contact with the hazard. The downfall of this approach however is that there is no way to enforce this limit.

The risk of getting pinched by the playing surface of the game could be reduced by adding a force sensor to the moving mechanisms. This sensor would detect how much torque was being applied to the game surface and could turn the motors off if something was preventing it's free movement. An emergency kill switch could also be placed prominently on the external packaging of the device that would stop all motion of the playing surface. A torque-limiting clutch could also be used to connect the motors to the playing surface, although this method may be prohibitively expensive.

The next most effective way to increase the safety of our project is by including warning labels and a user manual. All potential hazards can be individually labeled to specifically inform the user of the danger present, and more detailed information, along with instructions for safe use, can be included in a user manual.

Another ethical challenge we face is balancing the solutions implemented to reduce the environmental impact of the device with the ability to deliver a quality durable product. The solutions to these two issues are sometimes in direct opposition to one another, so it's critical to analyze which solutions provide the most benefit. For example, it's likely that higher quality components might weigh more and thus increase the associated transportation costs, but by using them the usable life of the product is extended considerably. Another area of conflict resides in the users' ability to disassemble the device for disposal, while still providing adequate physical protection from the electronics while the device is in operation.

7.0 Packaging Design Considerations

7.1 Commercial Product Packaging

Product #1

Figure 7.1: Baseball

A commercial product similar to our project is the Front Porch Classics Baseball [26] desktop game. Baseball is a tabletop game that simulates a version of the sport baseball using a marble and flipper to represent a ball and bat, and these are built into a wooden frame shaped like a baseball field. The flipper is used to hit the marble across the playing surface and into one of several holes, which simulates batting and scoring. The playing surface has holes in it, and the interior is sloped to return the marble to the player.

Baseball is constructed out of unfinished wood in a baseball diamond shape. Utilizing wood as the primary construction material is low cost, the game costs approximately 23 dollars, and makes the product's frame sturdy. The game's durability is increased by using wood, because of its capability to absorb many small impacts without breaking, whereas plastic could shatter if not specifically designed to be impact resistant. Wood has low tooling requirements and is easy to modify. The downside of using wood is increased weight. Baseball weighs approximately 2.8 pounds. The same product utilizing plastic could be made thinner and lighter. Using plastic would also allow the package to be more streamlined. Wood construction is limited in the way it can be shaped and connected. Plastic could be molded into a unified game board without any hard edges. The overall size of Baseball is approximately 14” x 12.5” x 3.5”, except for the difference in shape, this size is comparable to that of Marble Maze.

The base packaging of both Marble Maze and Baseball share many similarities. They're constructed of unfinished wood, and roughly the same size and weight. Due to the nature of the Marble Maze project and it's similarity to Baseball, no packaging features present in Baseball will be incorporated into the final design of Marble Maze.

Product #2

Figure 7.2: Pinball

Another similar commercial product is Table Top Pinball. This game comes in several versions similar in most respects, differing only in the theme. The version of Pinball analyzed for this report was Spider-Man 4 [27]. Pinball is a miniature version of a full-sized arcade pinball amusement machine.

Pinball is constructed almost entirely of plastic. The exterior is colored to match its theme, and shaped to eliminate hard edges. The formed plastic creates a very sleek look for the game. Pinball has a hinged “scoreboard,” where the player's stats are displayed electronically. It also has clear plastic over the playing surface to prevent the loss of the ball. Pinball costs approximately 20 dollars and weighs 2.6 pounds. Its dimensions are 16.3” x 9.8” x 8”. These specifications aren't significantly different from those of Marble Maze, even though the game play is totally different.

One of the features of Pinball which could be adopted by Marble Maze is its use of a clear plastic topper to enclose the playing space and prevent the loss of pieces. To implement this feature however, a mechanism to reset a marble at the starting position of the maze would be required, and currently we haven’t found any such suitable mechanism. Another feature utilized

by Pinball is the incorporation of an external LCD screen to display the user's score. Marble Maze could use an externally mounted LCD screen to display the elapsed playing time and other metrics.

7.2 Project Packaging Specifications

The packaging for the Marble Maze project is going to be the same packaging as the donor Marble Maze with a few enhancements. It will be a wooden box, approximately 1 foot square, with external ports for power and joysticks, and an LCD. The LCD will be placed on one of the sides of the box to be determined after selecting a motor mount position. The controlling electronics will be mounted out of sight, inside the box, and the ports will be flush mounted in the side. The goal is to utilize the donor packaging as much as possible, with the additional features added in the most unobtrusive way possible, such that the modification isn't immediately noticeable.

7.3 PCB Footprint Layout

The components utilized in this project will be surface mount devices in the smallest form factor available. The microcontroller (PIC18F67J94) comes in in a 64-pin TQFP [3] which is the smallest form factor available for this device. The stepper motor controllers are available in 24-pin SOIC packages [28]. The project will utilize an LED driver chip, TI TLC5916IPWRG4, available as a 16-pin TSSOP [5]. A shift register will be used for I/O expansion, and is available from ON Semiconductor in a 16-pin SOIC [29]. Utilizing these components along with the previously designed on board power supply, the initial dimensions of the PCB are approximately 70 mm x 50 mm. This equates to a PCB area of 35 square centimeters.

8.0 Schematic Design Considerations

8.1 Theory of Operation

The Raspberry Pi was designed to be powered from a USB connection, so a supply voltage of 5V was essential. Luckily, the GPIO pins all use 3.3V [30], so no level conversion is needed to interface with the microcontroller, which runs at 3.3V. To achieve a wireless connection with the iPhone, the Raspberry Pi will use a wireless dongle that plugs straight into its USB port.

After many different design choices were considered, the triple axis accelerometer [31] showed to be the best choice for solving the problem of initial board orientation. When first powered up, the microcontroller needs to level the game board before the game starts. Additionally, to prevent the motors from spinning the knobs beyond the maximum angle for the game board, a feedback mechanism was needed. With a triple axis accelerometer, the flat orientation can be found at the start of the game and the rotation of the board can always be known. This, in addition to step counting with the motors, could provide a very effective feedback mechanism for board control.

The stepper motor and corresponding driver circuits, StepGenies, were chosen to try and match either the 3.3V or 5V supplies that were already being used. The goal was to eliminate the need for a third voltage regulator on the board. When the drive mode is fixed, each driver requires three inputs: step, direction, and enable [32].

Unfortunately, when our stepper motor drivers arrived, they showed no signs of life. We quickly ordered new stepper motor drivers, the L297[33]. These provided built in drivers, which also eliminated our need for hexfets. Because the addition of the motors on the axels and the circuit components on the inside of the marble maze hindered the motors in different amounts per direction, the triple axis accelerometer proved ineffective for providing feedback. Instead, we stuck to counting steps taken in either direction.

For the final operation, the board uses the ADC to read the joystick directions and then calls the motor stepping functions if a game is currently running. It decides which way, if any, the motors should step and then runs through one full cycle for each motor. It also checks the input from the Raspberry Pi indicating the game status. The Raspberry Pi monitors the IR gates to determine when a game starts or stops. If a game starts, the PIC uses a timer interrupt to keep track of hundredths of a second and displays the time, in tenths of a second, on an LCD. When the game ends, the microcontroller reads the high scores from flash and determines if the game time generated a high score. It displays an appropriate message, stores the high scores in flash if they have changed sends the scores to the Raspberry Pi, and then waits for the next game to start. When interfacing to our LCD, we chose to design a serial controller using an Arduino. Because the Arduino runs at 5V, it worked perfectly with the 5V LCDs available from the parts room. A logic level converter is used when transmitting UART data from the PIC to the Arduino, and the Arduino simply converts each character received into a character on the LCD, with the exception of 25,

which clears the LCD. The Raspberry Pi monitors the IR gates because they were not functioning correctly with the microcontroller. When it detects that a game has started, it signals the PIC and the iPhone, which both start timing. Once the game ends, the PIC sends the high scores to the Raspberry Pi, which passes them along to the iPhone over a wireless connection, which the RPi generates. The iPhone can then keep a list of all high scores for the user to view.

8.2 Hardware Design Narrative

The PIC transmits data to the Raspberry Pi over a UART connection. Because the Raspberry Pi is monitoring the two IR gates, there is a GPIO line from the RPi to the PIC indicating if a game is currently running or not, on pin RC5. This pin was chosen because it did not interfere with any of our debugging setups. The PIC supports PPS Lite, or peripheral pin selection, for the UART line [34]. It can reconfigure the UART connection to be on any RP pin. We chose RP39 because it was closest to where the RPi header was on our PCB. The connection runs at 1200 baud. It was initially set to 9600 baud, but lowered because the complimentary RX portion had to be done in software, on port RG0. Joysticks are connected to the PIC with two analog input pins and one digital input pin each. Joystick 1 used RE0 for the digital input and AN17 and AN18 for the analog inputs. Joystick 2 used RF4 for the digital input and AN10 and AN11 for the analog inputs. These were selected because they were all in a group on our PCB headers. For the UART connection to the serial LCD controller, RP1 was selected because it was easily accessible on our pin headers. This connection runs at 9600 baud.

Because the same family of voltage regulators [35] were used for both the 3.3V and 5V supply, the capacitors are the same for both. The unregulated voltage capacitors are 100uF and the regulated voltage capacitors are 1000uF. The microcontroller uses its internal crystal and runs at 8MHz. The PIC uses .1uF capacitors for decoupling, as recommended by the datasheet. Each motor draws around 1A of current during operation, and the Raspberry Pi draws about .7A. Because of this, we chose a power supply rated at 9V and 4A to give plenty of overhead with current.

9.0 PCB Layout Considerations

The PCB for the Marble Maze has four major components that need to be laid out. The PIC18F67J94 MCU will be placed at the center, with a nearby oscillator. On one side of the board, the stepper motor drivers are positioned such that the motors control lines will be connect to a header. Sequestered in a corner are the voltage regulators, which need to be separate because of rapidly switching signals. The raspberry pi and tilt sensor are placed around the edge of the board, and extra MCU pins are routed to headers.

9.1 PCB Layout Design Considerations - Overall

To minimize EMI, the voltage regulator circuit should be isolated as much as possible from the rest of the board. Similarly, the signals to the raspberry pi should be kept separate. Also important with the raspberry pi is to connect it to the same ground as the rest of the board, since otherwise the signals might not be interpreted correctly. The MCU and crystal oscillator are centrally located to facilitate routing to the other devices, and easier access to all the pins. The tilt sensor is placed next to the power regulators, and has convenient access to all the pins it needs on the microcontroller. Finally, the 2 stepper motor drivers, and the hexfets that go with them, are located on the right side of the board, with large power and ground traces running to them.

Ground routing is done by making a U shape around the edge of the board, with as big a trace as will it. Ground is kept entirely on the bottom of the board to keep it as separate from the power lines as possible. This ground is fed to the Raspberry Pi, to ensure it runs on the same ground voltage as the microcontroller. Vdd, which is a regulated 3.3V, is kept on top of the board as much as possible, and runs straight through the middle of the other components. Both power and ground traces are set to 40 mils width as much as possible, going down to 10 mils only to fit the small MCU pads. Vcc, a regulated 5V used for the stepper motor drivers, will be the second power line on the board.

All traces are run at 45 degree angles (chamfer/bevel), unless there is a junction of two or more traces, in which case a right angle is formed for optimal performance. This is done to decrease transmission reflections [36] and to avoid acute angle “Acid traps.” At this stage in the design, the programmer and another header still need to be moved to a more convenient place to be routed easily to the microcontroller.

9.2 PCB Layout Design Considerations - Microcontroller

The PIC18F67J94 microcontroller [37] used for our project is placed at the center of the board. The microcontroller sits in the middle, oriented such that its pins are located on the side

where they are needed. Surrounding the microcontroller are four power supply bypass capacitors, one for each Vdd and ground pin. This is accomplished by immediately spreading the other pin traces away from the MCU, to allow space for the capacitors without interfering with another trace. It was also considered to put the capacitors on the bottom of the board, but we found there was enough room to keep them on the top layer. However, the oscillator and its supporting resistor and capacitor are placed on the bottom layer, right below the microcontroller. This is done in order to keep it as close to the microcontroller as possible, although moving it to the top of the board might be an option if there is enough room close by.

In order to accommodate the small pins of the microcontroller, a 10 mil trace is used for each pin. After the bypass capacitor, the traces are increased to the 40 mils of the power and ground lines on the rest of the board. The traces run directly to the component they are connected to, or to a header if they are unused pins. Currently, there are quite a few unused pins on the microcontroller, but we may have need for those pins later in the semester.

9.3 PCB Layout Design Considerations - Power Supply

The power supply circuitry, including a barrel jack to connect to a wall wart, is located in the lower left of the PCB board. Large aluminum electrolytic capacitors were selected as the bulk capacitors [38], a 100uF Cin and a 1000uF Cout. These capacitors are located near the edge of the board, close to the barrel jack. All power supply components isolated from the other major components on the board, to help prevent the fast switching of the regulator from causing unwanted noise in other signals. An inductor and a diode are also needed as support circuitry to the LM2576 voltage regulator [40]. The circuitry is duplicated for a 5 volt regulator chip and a 3.3V regulator, as both voltages are required by other components.

Thick traces were used for all power components, and these traces extend around the board, but without forming a complete loop. When we have moved farther along in the project, we will be able to add in a ground plane to the PCB design, which should help ensure better function of the ground and power lines. The ground line, which is kept at a 100 mil trace where possible, runs up and around the top of the board, ultimately feeding to the hexfets and motor drivers on the far side of the board. The Vdd line moves up the middle of the board, for ease of access to all the components that require it. The Vdd line can be seen on the bottom copper in appendix A. The Vcc line is run along the bottom of the board directly to the two motor drivers, and this can also be seen in appendix A, on the top copper.

10.0 Software Design Considerations

10.1 Software Design Considerations

There are several considerations that that have guided the design of this software. As far as memory is concerned, the PIC 18F67J94 has 128kB of flash and 3,862 bytes of RAM [41]. Apart from the program code, the only memory item that needs to be stored in nonvolatile flash is the high score data. All other variables are session-specific and can be placed on the stack or the heap as needed.

There are also several external interfaces for the design. First, the Raspberry Pi is connected to the microcontroller via UART. The two joysticks and the triple axis accelerometer tilt sensor will be interfaced via analog to digital convertor ports on the microcontroller. An LCD will be connected via SPI. Finally, the motor drivers and IR gates will be connected via general-purpose digital input and output pins on the microcontroller. Figure 1 below explicitly details the port connections on the microcontroller.

10.2 Software Design Narrative

As stated above, the microcontroller application will be constructed in a command-driven program style with several modules performing the chip functions. The first module is the Initialization module. This module will initialize ports and set session variables and reset any flags to their inactive state. This module will run once on power up independent of interrupts. The next module is the Level Board module. This module is responsible for calibrating the tilt of the board to a level position so that all subsequent board adjustments are made relative to this zeroed starting position. This module will be run immediately following Initialization and after a player completes a game. The remaining modules are all associated with peripherals that will generate interrupts to trigger different game events. The most important module is the Motor Control module. This module will listen for data coming in from the Raspberry Pi over UART (or via the joysticks on the ADC ports if not using an iPhone), interpret the data, and send the appropriate command to the motor drivers to step the motors to the position requested. This module will also rely on feedback from the triple axis accelerometer to determine if it has reached the targeted position.

The iPhone app is being written in Objective-C, and is utilizing several of Apple's built in frameworks to access accelerometer data and to send requests of HTTP [42]. Currently the iPhone app reads position data, displays it on screen with a GUI, and can send position data by HTTP request and receive responses. A code listing for the iPhone is available at github [43] and will be updated as completed.

11.0 Version 2 Changes

- Pick better motor drivers from the start. From a more reliable and well-tested vendor.
- Get beefier power for the motors. They work, but the power could easily be improved. Not even drawing max amperage, the regulator gets rather hot.
- Use unipolar stepper motors instead of bipolar. Unipolar seem easier to control.
- Choose a microcontroller that does not use peripheral pin selection or has much better documentation on it.
- Order parts earlier and begin interfacing immediately. Try to have a working prototype using the development board in case there are setbacks with the PCB.
- Route all pins from the microcontroller to headers, then to any components on the PCB.
- Make additional header pins for power and ground other than the ones needed for the microcontroller pins.
- Use the amount of microcontrollers in stock at popular electronic supply outlets as another component criterion.
- Order a development board with headers that are easier to access and that has as many header connections as it does pins on the microcontroller.

12.0 Summary and Conclusions

We designed and constructed a smart phone controlled mechanized marble maze. This product is meant as a fun pastime for children, but is certainly entertaining to adults as well. The product is basically a heavily modified store bought game, with electronics and high resolution stepper motors added on. The tilt of the maze is dictated by the orientation of an iPhone, utilizing the accelerometer in the phone and translating that motion into motion on the board. There are also joysticks to play the game, in the absence of an iPhone. The board runs off a 9V power supply, and keeps track of player statistics and high scores.

As for new knowledge that we gained over the semester, working with Eagle to create a real, working PCB from scratch was fairly new territory. All the tools, techniques and procedures used in the design and construction of the PCB gave us a new skillset. Another major part was learning to use a new microcontroller. From learning what pins to connect to program it, to how to initialize and write to pins, it was a process of reading documentation and finding what we needed to know. Also, a lot of learning went on to effectively control the stepper motors without killing the rest of the circuit. Learned about wireless routing capabilities of Linux, web servers and services, and interfacing several applications together in order to connect the iPhone to the RPi and the MCU. Overall, it was a good experience that we gained a lot from.

References

[1] Microchip. (2012). *PIC18F97J94 Family*. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/30575A.pdf

[2] (2008, January). *MC68HC908GP32* [Online]. Available: http://cache.freescale.com/files/microcontrollers/doc/data_sheet/MC68HC908GP32.pdf?fpsp=1&WT_TYPE=Data%20Sheets&WT_VENDOR=FREESCALE&WT_FILE_FORMAT=pdf&WT_ASSET=Documentation

[3] [Kraemer, Sandy F. 1974. Tilt Board Game. United States http://www.freepatentsonline.com/3787055.pdf

[4] Cheuk, Miu Kwan (Hong Kong, HK). 2012. MOTOR SPEED CONTROL SYSTEM. United States KING GOLDEN LIMITED (HONG KONG, HK http://www.freepatentsonline.com/20120223662.pdf

[5] Pulford Jr., Robert (Wolcott, CT) 2000. Stepper motor control United States. Tri-Tech, Inc. (Waterbury, CT) http://www.freepatentsonline.com/6150789.pdf

[6] Military Handbook: Reliability Prediction of Electronic Equipment [Online] https://engineering.purdue.edu/ece477/Homework/CommonRefs/Mil-Hdbk-217F.pdf

[7] Fairchild Semiconductor IRF530 Datasheet [Online] http://www.datasheetcatalog.org/datasheet/fairchild/IRF530.pdf

[8] Microchip PIC18F97J94 Datasheet [Online] http://ww1.microchip.com/downloads/en/DeviceDoc/30575A.pdf

[9] Texas Instruments LM2576 Datasheet [Online] http://www.ti.com/lit/ds/symlink/lm2576.pdf

[10] Wikipedia. (2013, April 12). Printed Circuit Board. [Online]. Available: http://en.wikipedia.org/wiki/Printed_circuit_board#Chemical_etching

[11] Wikipedia. (2013, April 12). Ammonium Persalfate. [Online]. Available: http://en.wikipedia.org/wiki/Ammonium_persulfate

[12] Wikipedia. (2013, April 12). Ferric Chloride. [Online]. Available: http://en.wikipedia.org/wiki/Ferric_chloride

[13] Wikipedia. (2013, April 12). Lead. [Online]. Available: http://en.wikipedia.org/wiki/Lead

[14] Wikipedia. (2013, April 12). Restriction of Hazardous Substances Directive. [Online]. Available: http://en.wikipedia.org/wiki/RoHS

[15] Brio. (2013, April 12). Labyrinth Product Information. [Online]. Available: http://www.brio.net/ToPlay/3_years/Games/34000_Labyrinth.aspx

[16] Costello, Sam. (2013, April 12). Where Are Apple iPods Manufactured?. [Online]. Available: http://ipod.about.com/od/glossary/f/where-is-ipod-manufactured.htm

[17] Upton, Liz. (2013, April 12). Made in the UK. [Online]. Available: http://www.raspberrypi.org/archives/tag/made-in-the-uk

[18] Brio. (2013, April 12). About us. [Online]. Available: http://www.brio.net/About_BRIO/The_Company_BRIO/About_us.aspx

[19] U.S. Energy Information Administration. (2013, April 12). Electricity in the US. [Online]. Available: http://www.eia.gov/energyexplained/index.cfm?page=electricity_in_the_united_states

[20] IEEE. (2013, April 12). IEEE Code of Ethics. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html

[21] BabyCenter. (2013, April 12). Choking Hazards for Children. [Online]. Available: http://www.babycenter.com/0_choking-hazards-for-children_1004.bc

[22] University Games Corporation (2013, Feb 7) Front Porch Classics Baseball [Online]. Available: http://www.universitygames.com/ugitem.asp?itemno=FP10055&brand=FP

[23] Amazon (2013, Feb 7) Spider-Man 4 Table Top Pinball [Online]. Available: http://www.amazon.com/Spider-Man-4-Table-Top-Pinball/dp/B00859UTQI/ref=sr_1_6?s=toys-and-games&ie=UTF8&qid=1360307396&sr=1-6&keywords=pinball

[24] Microchip (2013, Feb 8) PIC18F97J94 Data Sheet [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/30575A.pdf

[25] Mouser (2013, Feb 8) Microchip Technology MTS2916A-HGC1 [Online]. Available: http://www.mouser.com/ProductDetail/Microchip-Technology/MTS2916A-HGC1/?qs=sGAEpiMZZMvYc73DDAmzVlIqDWvAO99k62nbBc9m93Y%3d

[26] Mouser (2013, Feb 8) Texas Instruments TLC5916IPWRG4 [Online]. Available: http://www.mouser.com/Search/ProductDetail.aspx?qs=vPP9GyyTAo0Mny%2fTOeRlLQ%3d%3d

[27] Mouser (2013, Feb 8) On Semiconductor MC74HC4094ADR2G [Online]. Available: http://www.mouser.com/ProductDetail/ON-Semiconductor/MC74HC4094ADR2G/?qs=sGAEpiMZZMtsbn1GaJysl%2f0XjUzVqixQgeZo7sILZVc%3d

[28] Embedded Linux Wiki. (2012, Dec. 6). *RPi Hardware*. [Online]. Available: http://elinux.org/RPi_Hardware

[29] Freescale Semiconductor Technical Data. (2008, Nov.). *+-1.5g, +-6g Three Axis Low-g Micromachined Accelerometer.* [Online]. Available: http://www.sparkfun.com/datasheets/Components/General/MMA7361L.pdf

[30] Step Genie. [Online]. http://www.stepgenie.com/StepGenieSpec.pdf

[31] L293 QUADRUPLE HALF-H DRIVERS [Online]. Available: http://www.datasheetcatalog.org/datasheet/texasinstruments/l293d.pdf

[32] *SGS-Thomson Microelectronics.* (1996, Aug.). *L297 L297D*. [Online]. Available: http://pdf1.alldatasheet.com/datasheet-pdf/view/22436/STMICROELECTRONICS/L297.html

[33] Texas Instruments. (2004, May). *LM2576/LM2576HV Series SIMPLE SWITCHER 3A Step-Down Voltage Regulator*. [Online]. Available: http://www.ti.com/lit/ds/symlink/lm2576hv.pdf

[35] Motorola (1995) System Design and Layout Techniques for Noise Reduction in MCU-based Systems [Online]. Available: https://engineering.purdue.edu/ece477/Homework/CommonRefs/AN1259.pdf

[**36**] Microchip (2013, Feb 8) PIC18F97J94 Data Sheet [Online]. Available:

http://ww1.microchip.com/downloads/en/DeviceDoc/30575A.pdf

[**37**] On Semiconductor (2004) LM2576 Switching Regulator [Online]. Available: http://pdf1.alldatasheet.net/datasheet-pdf/view/11659/ONSEMI/LM2576T-5.html

[**38**] Lennard.net. How to create CAM files from Eagle for PCB manufacture[Online]. http://www.lennard.net.nz/electronics/How%20to%20Create%20CAM%20files%20from%20Eagle%20for%20PCB%20Manufacture.pdf

[**39**] Schindler & Schill Gmbh (2013) Online Gerber-Viewer [Online]. Available: http://www.gerberviewer.com/default.aspx

[**40**] PIC18F97J94 FAMILY [Online] http://ww1.microchip.com/downloads/en/DeviceDoc/30575A.pdf

[**41**] Resources for Apple Developers [Online] https://developer.apple.com/resources/

[**43**] API Reference – Bottle Dev Documentation [Online] http://bottlepy.org/docs/dev/api.html

[**43**] MarbleMaze Application GitHub Repository. https://github.com/jjachna/MM-iPhone

Appendix A: Individual Contributions

A.1 Contributions of Mark Sears:

My biggest contribution to the project was the PCB board design and routing. Of all the members, I spent the most time working with Eagle by far. As such, I have a much better understanding of how to use Eagle than I started with at the beginning of the semester. Other than the PCB, I spent time working to get the power supply, motors, LCD and other components connected properly.

For the first half of the semester most of my time was taken routing the PCB layout. Making sure everything was connected to what it needed to be, isolating components, and keeping all turns to 45 degrees. While our board is comparatively small, it still had quite a large number of vias, and some routes that needed to run a long distance. Notably, the header for the MCU programmer needed to be connected to very specific pins, including the MCLR pin which was on the far side of the chip and required an especially long trace. Since our MCU had a lot of extra pins, routing these to headers without interfering with the rest of the board was another task of mine. I shuffled all the components and headers around until I was able to lay them out and connect them in a rather small total area for the board. While laying all these things out, I slowly became accustomed to Eagles interface, features, and little tricks to make things easier. If I had to lay out another board, it would easily take half the time.

After the PCB was fabricated, I shifted my attention to work on the power supply and motors. The power supply proved a bit troublesome, and I spent a lot of time debugging where the problem was and how to fix it, usually by replacing a part or fixing a connection. As for the motors, a major difficulty arose when the first stepper driver chip we had did not appear to work as intended, so I laid out an appropriate circuit and wired up a different driver and soldered it to a separate board to connect directly to the bi-polar stepper motors.

As team leader, I often asked other teammates on how they were progressing, and what they needed from other members in order to continue. I set up times to work, and helped ensure that homeworks and project work occurred in a timely manner. Keeping the group in a positive attitude was a must when things weren't working, and stress levels were high.

A.2 Contributions of Justin Spencer:

My contribution to this project was multifaceted, but was primarily focused on the physical construction and component integration of the project with a supporting role in the other facets of the design.

During the design phase I helped create the project's block diagram and establish its desired functionality. I purchased the initial marble maze chassis, which was later returned for a better model, and took the measurements of the device. Using the measurements I constructed a CAD representation of the packaging design. I also created and maintained a CAD representation of the project block diagram.

As the software for the project was being developed I provided several suggestions that were incorporated into the final design. This included using the Raspberry Pi as a web server to reduce the data transmission delay. I also helped debug some software issues that came up during development. This included using the Raspberry Pi init boot parameter to bypass the login credential requirements of the OS when the credentials became mysteriously invalid. This saved several hours of work which would have been necessary to reinstall the Raspberry Pi operating system. Also, an issue with the microcontroller developed where it appeared that receiving single characters didn't work. I suggested that the issue was a failure to send a terminating character to the microcontroller. Implementing this change solved the transmission issue which allowed further development and debugging to proceed.

Once we acquired the PCB, I populated and soldered all of components on the PCB. I also de-soldered and replaced components as we broke them, including the microcontroller which was successfully replaced several times, and altered the design (e.g. switched motor driver ICs). I set up, performed, and monitored the initial and subsequent component "burn ins". As we encountered problems with components, I performed bench tests including checking power output waveforms, voltage levels, and locating short circuits. I then replaced any required components. After, identifying a short on the PCB, I cut the trace in several places and "fly-wired" between PCB vias to correct it.

I designed, prototyped, and built most of the dongles used in our project. This included the user joysticks, the joystick ports, their crossover cables, the IR gates, and the stepper motor drivers. The dongles were all constructed using their respective components, prototyping boards, and lose wires attached back to the PCB via headers. The dongles were prototyped on a breadboard, laid out on prototyping-board, bench tested, and the connected to the PCB with headers.

I performed the majority of the physical systems integration and construction. I did all the physical modifications to and integration of components on the marble maze chassis. This included cutting mounting holes for the joysticks, stepper motors, and LCD screen. I physically mounted all components to the board, and I selected and implemented the method used to connect the stepper motors to the maze surface. This included cutting the maze "drive shafts",

mounting the motors in line with their axis, and directly coupling them together with hardware. I mounted all the components to the chassis and routed their wires back to the PCB headers. I altered the original chassis to provide space for attaching the PCB and dongles inside the marble maze chassis. I filled the holes on the surface of the maze and later removed excess filler.

During the integration phase, I spent considerable amounts of time connecting and debugging all the components. Some of the issues that I addressed were incorrect voltage levels being delivered to IC's, incorrect output pin assignments, and poor solder joints. One of the tricky bugs I solved was identifying the pin headers as faulty connections to the PCB. I traced the problem, using a digital multimeter and logic analyzer, and found that the pin headers didn't work well with the ribbon cables initially used on the dongles. This was a difficult bug because it was intermittent and not visibly apparent. I fixed this problem by substituting different (i.e. thicker) wiring for the ribbon cables, reattaching all the pin headers, and then checking each connection individually for continuity.

A.3 Contributions of John Jachna:

I did my best to contribute my unique skill set and talents to this project, as well as supporting my teammates in accomplishing their tasks. First, I design, programmed, and tested the iPhone application used to send position information to the game board and to display high score data saved on the microcontroller. This allowed for a novel, interesting, intuitive way for users to interact with the game and have fun. Though I have some experience with iPhone app development, there were a few frameworks needed in this application that I needed to research in order to properly implement the functionality required. This included using the built in gyroscope, creating URL requests, and adding table view and the custom game controller view to a tab bar application.

In addition to the iPhone app, I completely set up Raspberry Pi for our project applications. The first function the Raspberry Pi needed to perform was wireless networking. I was able to first set up the Raspberry Pi to connect to an existing wireless network. Though this appeared to be sufficient for the project, it was later decided that the game would be more portable if the Raspberry Pi could act as a wireless router. To do this, I needed to find the Linux packages that enabled a DNS/DHCP server, an IP table, and wireless network broadcasting, figure out how install them, and set them up. In attempting to set up these features, I discovered that the chipset used in the wireless dongle that we purchased was incompatible with the standard release of HostAPD, a package that enables network broadcasting. I had to troubleshoot this issue and modify the standard release so that it worked with our chipset.

After this, I worked to enable UART serial communication on the Raspberry Pi. The first time I installed the Raspberry Pi OS something was not installed correctly, so I needed to diagnose that issue and ended up reinstalling the operating system so that it could work. When this was solved, I researched how to send and receive UART on the Raspberry Pi.

The final component I was primarily responsible for was the web server. I researched different options for deploying a web server and concluded that a python web server running on the micro-framework “Bottle” was most efficient for our project. In this web server, I designed and implemented various web services so the iPhone could send and receive data to the rest of the project. Since I worked on two of the three communicating systems, I put significant effort into designing the data packets and web service requests to be used. I also created the high-level software design on which the microcontroller program was based.

I worked very closely with Jordan to help him debug the microcontroller and the peripherals. I also assisted in helping pick out parts to be used on our board, as well as providing another pair of eyes to check the PCB and other circuits.

A.4 Contributions of Jordan Wagner:

As I am about to enter into a career in embedded software engineering, my major role was writing all the software that ran on our PIC microcontroller and interfacing with most of our parts. Aside from that, I helped in designing our schematic, using the datasheets from the parts we had selected to know which pins connected where, what decoupling capacitors were needed, and which pins had to be routed to headers.

At the beginning of the semester, I did a lot of work selecting the microcontroller based on our constraints and picked out a lot of the parts that we would need to get our design to work. After we decided on which parts we would use, I helped design the schematic with the parts and ensure all the pinouts were correct. I also spent a lot of time familiarizing myself with the PIC's 700 page datasheet and looking through all the possible peripherals we could use and deciding on which would be best to do certain functions, such as drive the motors, communicate with the Raspberry Pi, and monitor digital inputs from buttons/gates.

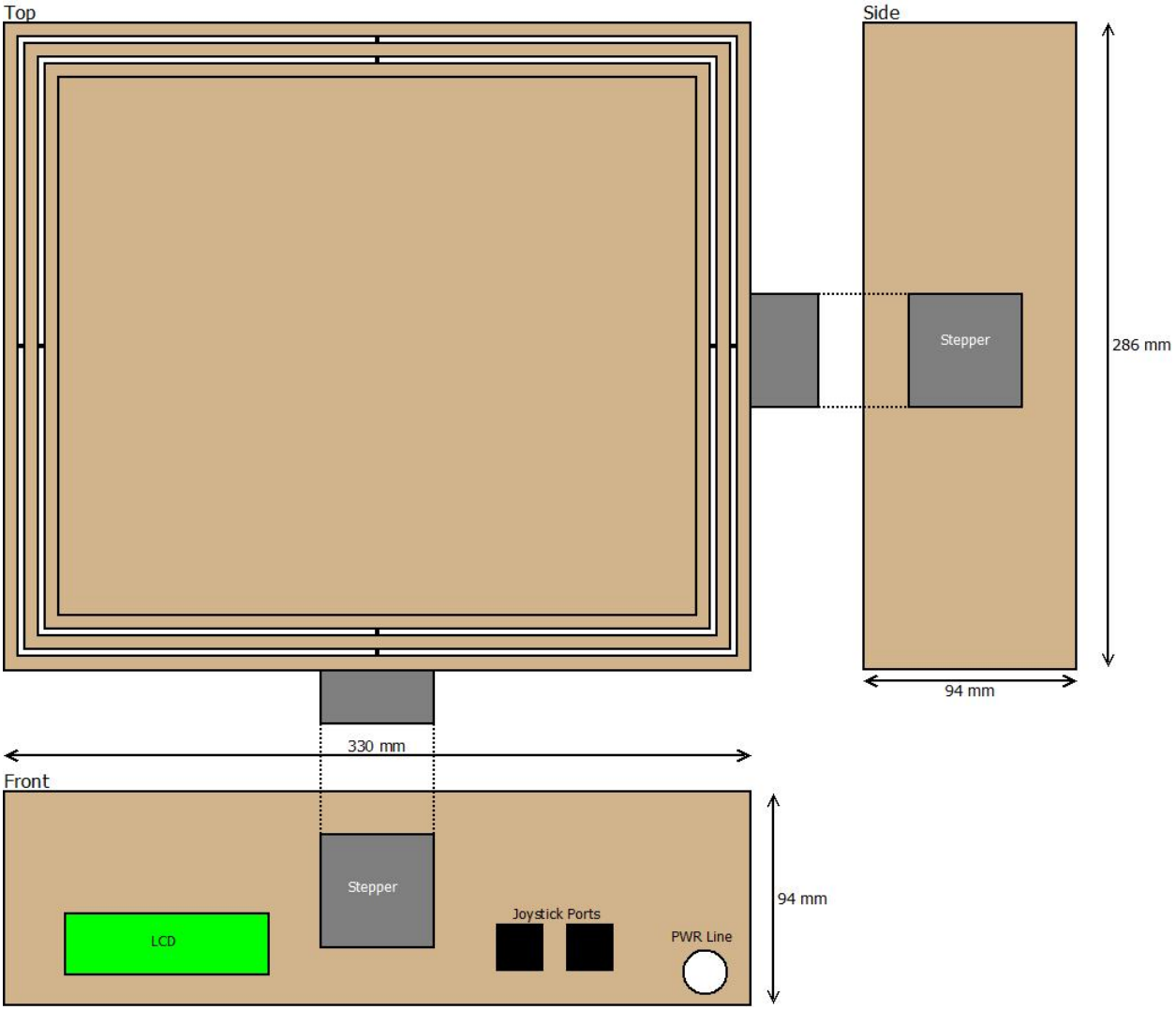
Once our development board arrived, I began learning in depth how to control different features of it. Once I had the correct software to connect and program it, I was able to load on the first heartbeat program. After that, it was just a matter of reading the data sheet and learning new peripherals. I began writing functions that could be used in later programs for things such as delays, timers, ADC, and UART. I spent a large amount of time learning new peripherals and then writing a program that could test them out with one of our components, as they slowly arrived. I also spent time writing functions for interfacing with parts such as shift registers or an LCD that did not require complex peripherals but still needed a lot of debugging. My main downfall was not having a way to test UART receive, which ended up not working and used up a lot of time. After I got an LCD functioning by using an Arduino as a serial controller, it was very easy to interface with other components, such as our accelerometer. Instead of simply toggling LEDs, I could now print informative messages and debug efficiently. I was able to write functions that could be used with our PCB for the accelerometer, joysticks, timers for motor control and keeping track of play time within a game, sending and (I assumed) receiving over UART, an interrupt service routine, and writing to and reading from flash, all before our PCB was functional.

Once the PCB was assembled, I spent a lot of time loading heartbeat programs onto it and testing that it worked correctly. Because I was the only one who knew how to program our microcontroller, I had to work alongside Justin and Mark a lot when testing that the motors worked and the PCB was functioning again when it would burn out. When the PCB wasn't working, I'd work with John on getting the PIC and Raspberry Pi to communicate and inventing the protocol we'd use for transmitting tilt angles, high scores, and game times.

Finally, after our PCB was assembled and the motors were mounted and working correctly, I began the process of assembling all my functions and code snippets from the development board and compiling the main program for our marble maze. I quickly realized that a complicated algorithm for varying speeds based on angle readings was not the best approach for the first version of our software and instead began writing code to get our game functioning enough that we could have our PSSCs checked off. I decided that, because UART Rx was still not working and I, nor anyone on the forums I posted to, knew why, that the best course of action would be to get the game working with joysticks and write a UART receiver function in software when I didn't have access to the PCB. After staying up all night coding a couple times, I was able to work out the bugs in the software from the new code I had added and from all the different functions running together. I was able to successfully play a few games of Marble Maze with a joystick and have everything function as expected, as well run a working echo program using a software UART receiver. Unfortunately, when our final design was being package, something went wrong and it ceased to function at all. Although the PICKit 3 could still detect and program our microcontroller, I could not get a heartbeat program to run and, seeing as we were out of spare microcontrollers, stopped the further development of our software.

Overall, I helped contribute to the design of our game, the choice of most of our parts, the final packaging, and some of the hardware debugging. I also wrote all the PIC's software and interfaced the PIC with most of the parts in our final design.

Appendix B: Packaging



Appendix C: Schematic

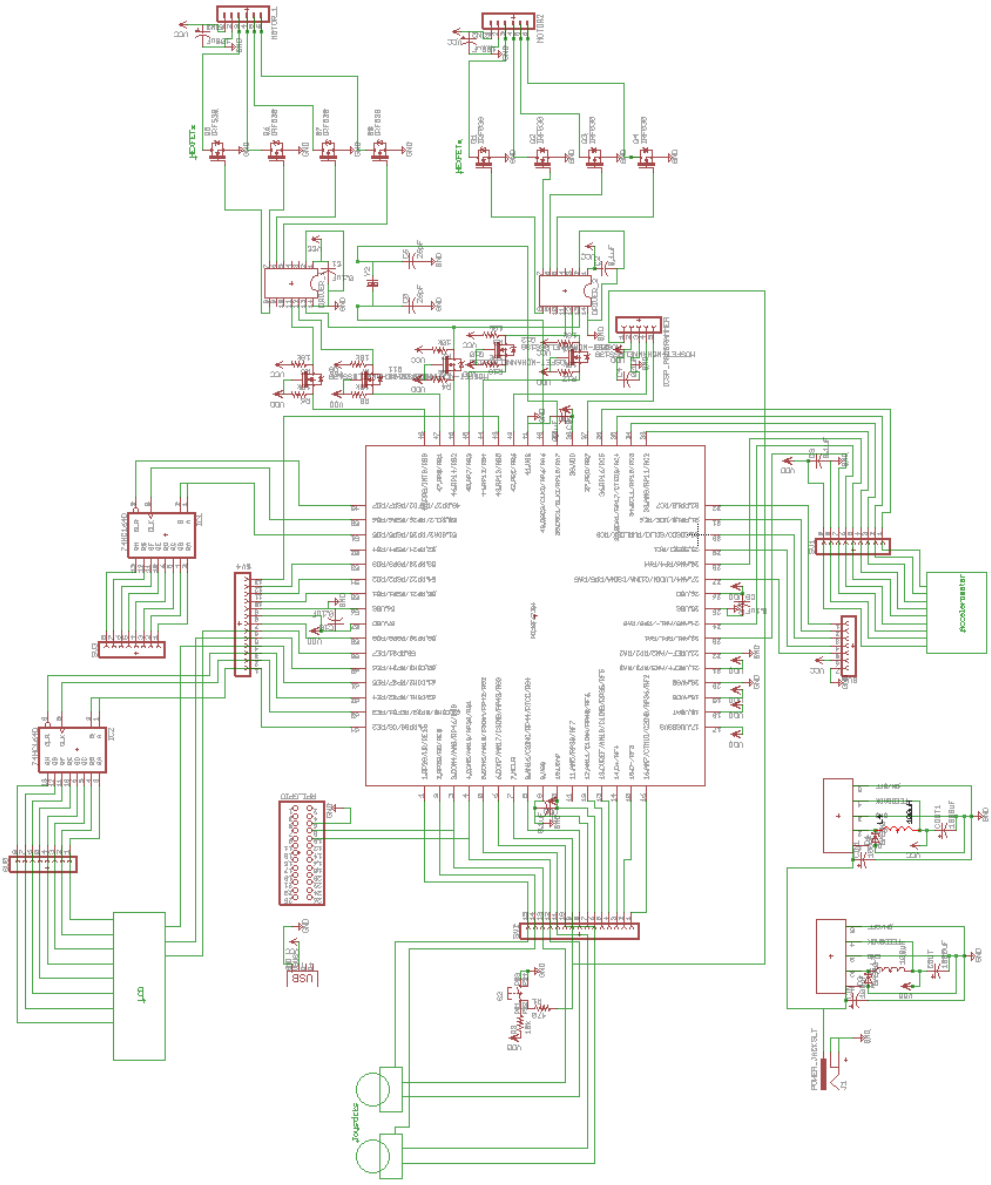


Figure C-1

Final schematic shows all MCU pins an their connections with power, motors, and peripherals. All unused pins are fed to headers for surrounding the MCU.

Appendix D: PCB Layout Top and Bottom Copper

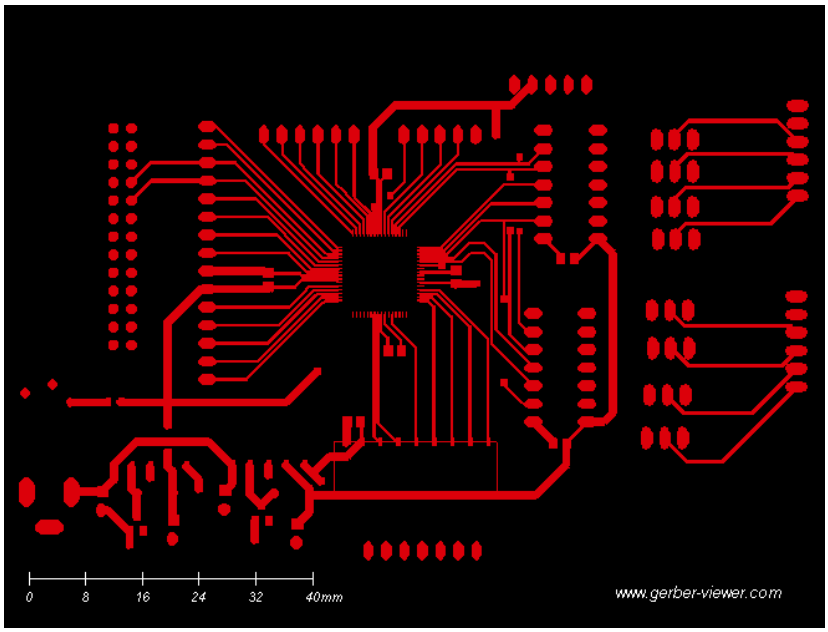


Figure D-1

Top Copper. Shows the routing to the microcontroller, headers, and other lines

Bottom Copper

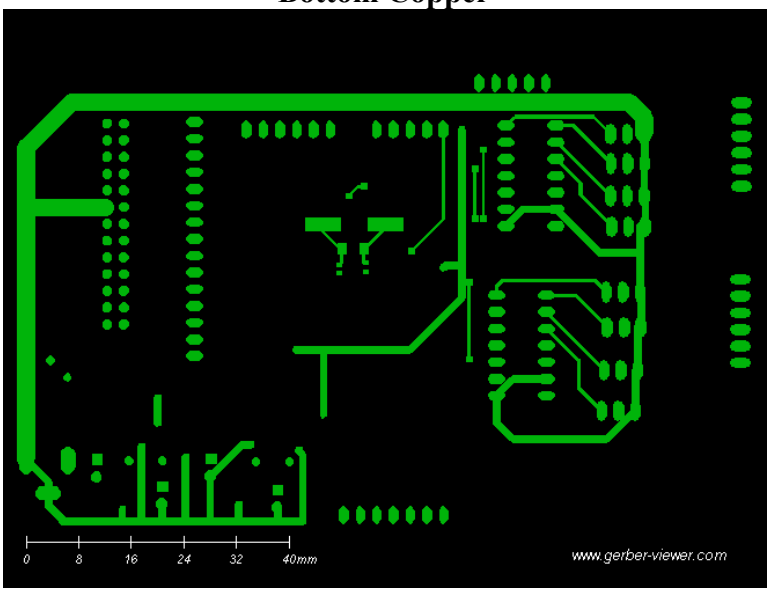


Figure D-2

Bottom Copper. Shows a ground line running around the edge of the board. Both coppers images were generated with a gerber file online viewer [39].

Appendix E: Parts List Spreadsheet

Vendor	*Manufacturer*	*Part No.*	*Description*	*Unit Cost*	*Qty*	*Total Cost*
Digikey	Fairchild Semiconductor	BSS138	Logic Level Converter	$.25	1	$.25
Digikey	Texas Instruments	LM2576T-3.3	3.3V Regulator	$2.83	1	$2.83
Digikey	Texas Instruments	LM2576T-5.0	5V Regulator	$2.83	1	$2.83
Digikey	Texas Instruments	SN74HC164DR	Shift Register	$.49	2	$.98
Newark	Raspberry Pi	MODB-512M	Raspberry Pi	$35.00	1	$35.00
Adafruit	Adafruit	1140	Pi Shell	$7.90	1	$7.90
Adafruit	OURLiNK	814	WiFi Module	$11.95	1	$11.95
Sparkfun	4UCON	PRT-09011	USB Female A Connector	$1.25	1	$1.25
Digikey	Microchip	PIC18F67J94	Microcontroller	$3.34	1	$3.34
Jameco	Reliapro	39BYG001-R	Stepper Motor	$21.95	2	$43.90
Mouser	Condor/SL Power	CENB1040 A0903F01	Plug-In AC Adapter	$23.36	1	$23.36
Amazon	Brio	B0001WGISK	Marble Maze Game	$39.99	1	$39.99
Sparkfun	Sparkfun	COM-09032	Joysticks	$3.95	2	$7.90
Digikey	TE Connectivity	5555165-1	RJ11 Jack	$.93	4	$3.72
Sparkfun	Sparkfun	SEN-00241	IR emitter and detector	$1.95	2	$3.90
Digikey	Micro Commercial	MMBD301-TP	Schottkey Diode	$.70	2	$1.40
Microtivity	Texas Instruments	L297	Stepper motor driver	$4.95	2	$9.90
Sparkfun	Arduino	DEV-11113	Arduino Pro Mini 5V	$9.95	1	$9.95
Sparkfun	Xiamen Amotec Display Co.	LCD-09052	LCD display	$14.95	1	$14.95

Total:	**$225.27**

Appendix F: FMECA Worksheet

Failure No.	Failure Mode	Possible Causes	Failure Effects	Method of Detection	Criticality	Remarks
A1	Output stuck open	Error from microcontroller or stepper drivers; part failure	Motors unresponsive	Observation	Low	Inconvenient, not dangerous
A2	Output stuck closed	Error from microcontroller or stepper drivers; part failure	Power MOSFETs source 1A current constantly; could cause combustion	Observation	High	Dangerous to user

Figure F-1: FMECA Table for Power MOSFETs

Failure No.	Failure Mode	Possible Causes	Failure Effects	Method of Detection	Criticality	Remarks
B1	Output = 0V	Failure of power supply circuit components (capacitors, inductors, diodes)	Lack of power to circuit	Observe	Low	Inconvenient to user
B2	Actual output > expected output (either 3.3V or 5V)	Failure of power supply circuit components (capacitors, inductors, diodes)	Power MOSFETs source 1A current constantly; could cause combustion	Observation	High	Could destroy components on board
B3	Output out of tolerance	Failure of power supply circuit components (capacitors, inductors, diodes)	Unpredictable output	Observation	Medium	Could overpower components or under-power components and cause them to function incorrectly

Figure F-2: FMECA Table for Voltage Regulators

Failure No.	Failure Mode	Possible Causes	Failure Effects	Method of Detection	Criticality	Remarks
C1	Output continuously 0	Software; peripheral failures	No output to peripherals	Observation	Low	Inconvenient to user
C2	Output continuously 1	Software; peripheral failures	Constantly on output to peripherals	Observation	High	If motor control always on, could lead to combustion

Figure F-3: FMECA Table for Microcontroller