Department of Mechanical and Mechatronics Engineering

Engineering Notebook: Digiboard

Group #41

A Report Prepared For:
The University of Waterloo

Prepared By:
Isaiah Erb 20584958
295B Lester St.
Waterloo, Ontario, N2L 0C1

Mar 25 2013

295B Lester St.
Waterloo, Ontario, N2L 0C1

Mar 25, 2019

Prof Sanjeev Bedi
Director, Mechatronics Engineering
Department of Mechanical and Mechatronics Engineering
University of Waterloo
200 University Ave W.
Waterloo Ontario, N2L 3G1

Dear Professor Bedi,

This report entitled Engineering Notebook: Digiboard, was written to record the engineering design and analysis that was performed to create the Digiboard, an electronic console that aims to digitize the board game experience. This project consists of the design challenges that were overcome in the creation of a first prototype for the console. The main challenges that were faced was the development of a new localization and identification method to find passive RFID tags, the creation of a prototype game to showcase the features of the console, and the fabrication of the console itself.

This report was written entirely by me, and has not received any previous academic credit at this or any institution. My role in this project included project manager, software developer and electrical designer. However, I would like to acknowledge the help of my brother Elijah Erb who aided in ideation, engineering inspiration, and electrical testing, mechanical design, and assembly of the project.

Best Regards,

Isaiah Erb
ID: 20584958
4B Mechatronics Engineering

Executive Summary

Board games are an essential part of modern entertainment, and create a way for people to harken back to their roots of simple face-to-face fun in an age of fast paced and draining entertainment. Board games offer a sense of connection, but they often forgo the perks of modernization, and could stand to benefit from the power of computation, animation and communication. In this report, a method of bridging the gap between the board games and a modern digital experience akin to board games is explored.

The primary engineering components of this project delve with the localization and identification of smart game pieces, which are physical and interactable components that can be used in tandem with a touch screen. Additional aspects of the project where engineering knowledge are required are choice of a variety of microcontrollers and computers that are used in the console, the internal power distribution, and the construction and manufacturing of the console itself and its case. Additionally, in order to showcase the features of the console, a game was developed that would use the features of the digiboard's RFID localization and identification boards to improve gameplay. The game that was chosen was a version of Settlers of Catan which features two new game pieces. This was developed in tandem with a Unity plugin that allows for seamless connection between the data provided by the console and the software within Unity.

Finally, the logistical aspects of this project were outlined, delineating the breakdown of roles, budget and scheduling. Recommendations on possible enhancements to the project were explored as well.

Table of Contents

1. Introduction

1.1 Background

Over the past few years, there has been a drastic resurgence of board game popularity. Board game cafes abound, and young people are turning to table top fun more and more often. One possible reason for this is the increasing involvement of technology in our lives, sometimes technology can be a hindrance to personal connection, and board games offer a way to reconnect back to the simple face-to-face fun of old. However, because of the close ties to simple fun that board games have garnered, this market has been slow to modernize. The problem that was identified is that there is a need for modernization within the board game space; there needs to be a way to maintain the personal fun of board games without forsaking the modern perks of technology, touch screens, connectivity, and communication. Digiboard is the solution to this problem, creating a board game console that aggregates all of your board games into one place. Digiboard has an array of RFID sensors beneath a touch screen that detects physical game pieces so that one can still use game pieces, dice, and cards to interact with their game, while also being able to use the screen to relay information in the blink of an eye. No setup required, Digiboard offers the best possible board game experience in a new and innovative way.

1.2 Original Design

The original design that was chosen in the last term was similar to the design that was created as part of the final product. The design that was chosen prominently featured a single square screen similar in size to the board game Monopoly, all enclosed within a case. Internally there would be an RFID sensor array which was composed of MFRC 522 RFID sensor boards which tessellate the entire length of the screen. Additionally there would be an array of hall effect sensors arranged hexagonally so as to have as much overlap as possible. These sensors would detect changes in the magnetic field around them which would signify the presence of a game piece with its embedded magnet in its base. This information would be used to localize the game piece to a specific point on the screen, and identify it as a particular game piece. This information would be collected by a single arduino Mega, which would aggregate the data into a struct and send the data to a raspberry pi. The main computing unit of the console would be a raspberry Pi running the Android operating system which fully features Unreal Engine and Unity. Using these inputs from the Arduino, the raspberry pi would expose the data via its GPIO pins. Finally, the test app that would be developed would be created using Unity and would run as if it were a normal app downloaded from the app store.

2. Final Design

2.1 Identification and Localization

The main differentiating factor between the Digiboard device and a tablet or smart table are the interactions that it makes with smart game pieces: physical objects that can be identified and interacted with on the surface of the board's screen. The process of knowing which piece was placed on the board, and finding exactly where it was placed is a difficult challenge that uses a variety of sensors and technologies to accomplish it efficiently and accurately. In this section the design details of the subsystem that accomplishes this will be further analyzed.

2.1.1 Overview

In order to achieve the desired interaction with game pieces, both localization and identification must occur, and this process must be synchronized so as to avoid misappropriation of an NFC tag's ID to the wrong game piece. First, for the processes of identifying the tags, the method that was chosen was to create an array of 5x6 = 30 RFID sensor boards as shown in **Figure 1**. The sensors are embedded beneath the touch screen and would continuously scan for NFC tags at a very high frequency. This sensor board would relay the ID of the tag that was read and the sensor from which the signal was located to the main controller. This data not only shows the ID of the tag that was read, but also offers a coarse understanding of the location of the tag due to the fact it was found over a specific sensor board.

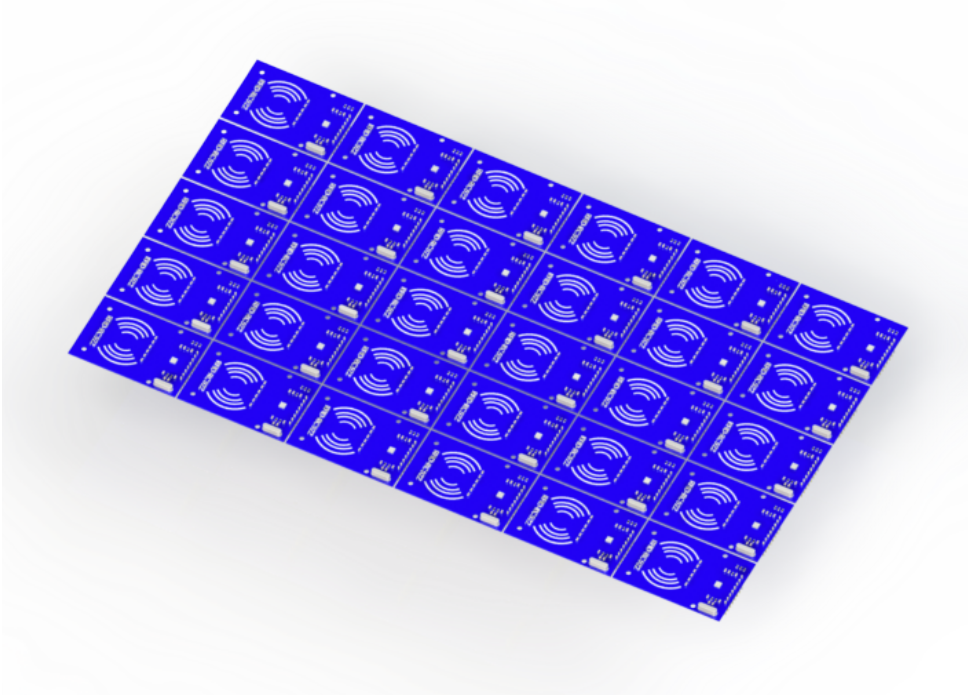

Figure 1: Render of the 5x6 RFID Sensor Array

A much more precise method of getting the location of the tags is required, however. The method that was chosen to localize the game pieces was to utilize the touch screen. The touch panel that was used in this project is of the capacitive variety, which uses the conductivity of the human

body to detect touch inputs. The top layer of the touch screen is coated with a transparent conductor which, upon interaction with a finger or any other conductor, changes in capacitance, which can be interpreted as a touch input. Indirect touch inputs can be simulated by allowing a path for current to flow from your hands to the screen through another conductor, you can use the back of a spoon as a touch input for example. In order to localize game pieces, a small metallic plate was placed at the base of the game piece which was connected to a layer of tin foil wrapped in felt. Touching the tinfoil through the felt allows a path for current to flow from the hand to the tinfoil, then the metal contact and finally the touch screen itself. This allowed for direct simulation of touch inputs, and created a way for pieces to be located extremely precisely. In order to associate the touch input with a NFC tag, and not a physical touch, the software first registers touch inputs to a identified game piece, if the RFID sensor boards also detected a NFC tag within 10 cycles, or approximately 0.16 seconds. If there was a touch input over a particular sensor, and that sensor read a NFC ID within 0.16 seconds, the touch would be registered as a game piece input, not a touch input.

2.1.2 NFC Protocol

The main function of interacting with game pieces is accomplished by both the identification of the game piece and its localization to a point on the surface of the screen. Each smart game piece consists of an NFC tag that could be read by any RFID board that conforms to the NFC protocol. The NFC protocol stands for Near Field Communication, and is named as such since it operates at and operates on the 13.56MHz frequency. Each NFC tag is passively powered, which means that it does not require any external power to function, it receives power directly from the RFID reader which is inquiring a read. This allows NFC tags to be very cheap components often costing less than a dollar. NFC tags are very versatile as well, capable of being read from, and written to. NFC tags range from being able to store 48 bytes of data to upward of 8kB of data, and all of the data is mutable. This allows for unique interactions where the state of a game piece can change when interacting with the RFID reader in different modes.

2.1.3 Modifications

The design for the localization and identification subsystem has changed drastically since the last iteration of the design specified in MTE 481, however only the Localization component had a major overhaul. Initially localization was meant to be performed using hall effect sensors that localize magnets that would be embedded in each game piece. This would require a dense array of hall effect sensors that would measure the analogue values of the magnetic field at their corresponding location. A prototype was created that consisted of three hall effect sensors. Although this prototype yielded results that were promising, and accurate to less than 5mm, the cost of this sensor board would total upward of 200 dollars, which was more than could be handled by the two people working on this project. Additionally, although the accuracy was within the margin of the initial constraint, 5mm accuracy was still quite striking, and other methods such as touch were much more accurate. In the end, this entire sensor board was forgone, trading the utility of the hall effect sensors for simulating touch inputs on the touch screen. This however means that fewer touch inputs from actual hands could be used, which was a tradeoff that was preferred to a lower accuracy and expensive hall effect sensor array.

2.1.4 RFID Sensor Array Design

The exact implementation of the RFID sensor array was guided by a variety of factors: cost, power requirements, data refresh rate, area of effect, communication protocols, RFID interference, and tag modularity among others.

First, the screen size that was being developed for had a 16:9 aspect ratio, and had side lengths of 13.9” and 7.8” respectively. This means that whichever method was chosen for RFID detection, it had to be tileable to this size. Second, the method for detecting the NFC tags must be compatible with most NFC tags and cards which means it must read RFID signals within the 13.56MHz bandwidth, and also support the NFC and MIFARE protocols. Furthermore, it must be simple to integrate into an array, which means it must be able to communicate along a bus via either i^2c or SPI protocols. This is to minimize the amount of communication lines connecting the sensors together. Finally, they have to be relatively large and cheap in order to tessellate the extents of the board without needing excessively many boards and costing too much money.

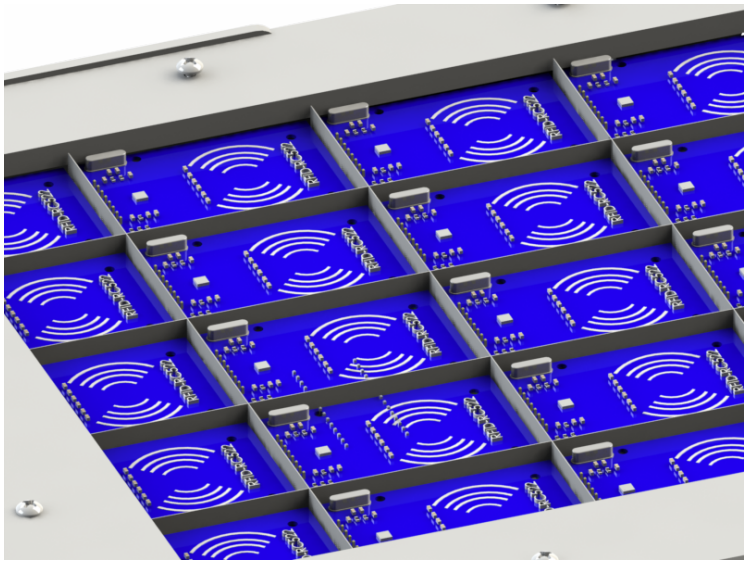

Figure 2: Stainless steel lattice separating sensors to prevent self interference

With all of these factors in mind, the sensor board that was procured was the MFRC-522 board[1], a widely available and ubiquitous choice for RFID sensing. This board detects both MIFARE and NFC RFID tags and can therefore detect fob keys and RFID enabled key cards as well as normal passive tags. This board communicates via SPI so it can easily be connected through a bus in order to read and write values to it. The final dimensions of the board are also 2.36” x 1.57”, which divided into the main screen size yields 5.89 and 4.96 sensors respectively. This means that the entire screen can be tessellated relatively well with 6x5 sensors leaving a small gap in between each sensor. The current draw of each sensor was approximately 10mA in idle, and 30mA when reading or writing to an NFC tag. With 30 sensors, this means the sensor array would draw a maximum of 0.900A of current, which is quite significant at its worst case. Finally, shown above in **Figure 2**, one problem that was found during testing was that each sensor would interfere with each other when placed close enough. This would cause compromising read errors which were unacceptable. However it was discovered that simply placing thin tin foil between the sensors provided a barrier for the emitted

signal, and would allow for complete isolation of the signal. In the final design, this was brought to fruition by using a lattice of stainless steel plates 10 thousandths of an inch thick in between the lengths of each sensor.

2.1.5 Data collection

In order to read the data that was being transmitted on each sensor, a controller must communicate via SPI and poll the sensors within the sensor board to gather the data from each RFID sensor. In testing, an Arduino UNO was used to verify communication between the sensors, via SPI, and it was selected as the main controller. The SPI protocol uses four signal lines to transfer data, the SS line which is the chip select pin, the MISO and MOSI lines, which stand for Master Input Slave Output and Master Output Slave Input, respectively, and finally the CLOCK line which synchronizes the signals. Why was the nano selected, Chip select limits for SPI connection with sensor, why three nanos instead of another amount, communication between all of them via i^2C and SPI. While MISO, MOSI, and CLK are shared busses, the SS line is a chip select, and a unique line must run from the controller to each slave within the SPI bus. This means that a total of 30 SS lines, and 3 bus lines must be sent from the controller. Instead of distributing all these lines to a single microcontroller, 3 Arduino Nanos were used to split the load in three. This means that each arduino nano has its own MISO, MOSI, and CLK signals and 10 chip selects to its trisection of the board.

The sensors operate on an update frequency of 10HZ, and this data is collected on each arduino is packaged up into a better configuration for i2C transfer. On the main controller for the project, there is a secondary arduino that can be read from directly, so there is a secondary i2C bus that will transfer each third of the sensor data to the main controller every cycle. This data is packaged up into a series of bytes which correspond to a struct containing the number of tags detected per sensor, the sensor location, and the data dump and ID from each RFID tag that was sensed. This is all the data that is required to completely localize and identify the tag for the software component. Eventually this data is read into the software using a method that will be described in section 2.4.3

2.1.6 Board Design

The final components that are necessary for the completion of the sensor array are the lines that connect each sensor's pins to their respective SPI bus, and their connection to power and ground. This is not a trivial task as each board requires 7 lines to be connected, which totals to 210 lines. If using solder or wires, this would be a major point of failure since there would almost certainly be a less than desirable connection out of 210 lines connected manually. Instead, a board will be created that will connect all the required lines from the arduino to each of the sensor board pins. Each board has a power, ground, MISO, MOSI, Clock, Interrupt, Reset, and Chip Select pin, however the Interrupt pin was left disconnected. The schematic for the board is shown in **Appendix A**. The schematic that was created essentially connects together each sensor board's MISO, MOSI, RESET, CLOCK, 3.3V, and GND connections, and connects each sensor's Chip select, or SDA to a particular digital input pin on the arduino nano. This creates the bus that was desired. Additionally, the I2C pins for the arduino are connected to an external header that will be used to connect each of the arduino Nanos to a single I2C bus. All the unused pins were also attached to an external header

in case their functionality was needed at a later time. Finally, there were mounting holes placed on the board for structural support of the board and boards that will be mounted on top of this board.

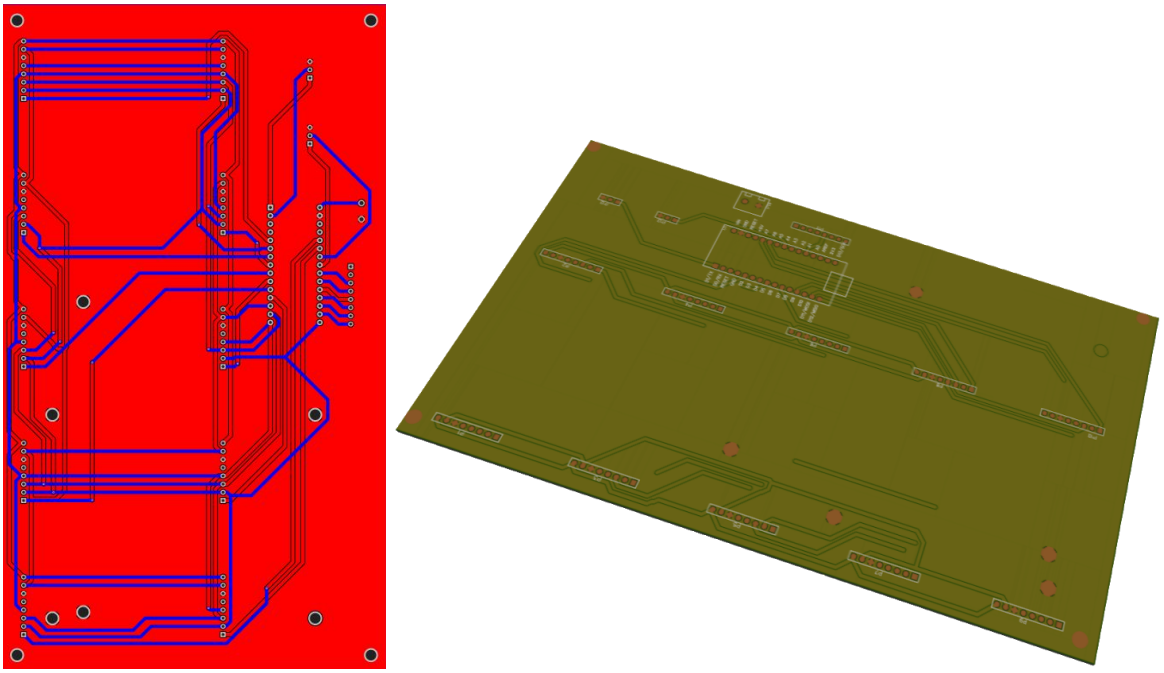

Figure 3: (Left) Front and back traces for the PCB, (right) Final render of the Sensor Array PCB

The next step was to convert this schematic into a PCB. There were a variety of factors that were considered when designing the PCB. The first aspect of the design that had to be completed was routing which is the process of applying copper traces to connect all the lines. This was completed using the Autoroute feature of the design tool that was used. However there are a variety of other design constraints that must be followed as well. One calculation that is extremely important for the board is the trace width. For this calculation, the maximum current across any individual trance must be calculated since the trace width is inversely proportional to the resistance across the trace. As stated earlier the sensors draw a maximum of 10mA each, so on a single board with one third of the sensors, they will draw a maximum of 300mA. Next, according to prior testing with the Arduino nano under nominal conditions draws 50mA, and in high stress conditions with a large computation load draws 65mA which yields a total of 365mA of current running through the main power and ground lines at the worst case. In order to calculate the trace width, a calculator[2] was used with an input of 215mA of current and a copper weight of the standard 1 oz. ft^2. This yielded a minimum trace width of 0.195mm. A large safety factor of 5 was placed on this, and a trace width of 1mm was used in the final design to complete avoid any faults. Next, the auto router was constrained to draw traces in only 45 degree angles since there are limits on the drilling capabilities of most pcb manufacturers. Finally, a copper ground plane was placed on the top layer so as to decrease the internal resistances between individual ground contacts, and to distribute heat in a more effective manner. **Figure 3** showcases both sides of the board as well as a final rendering.

2.2 Computation and Operating System

2.2.1 Overview

The next subsystem that will be discussed is the computation system. This consists of the computer that will control the logic and graphics of any game that is developed, and aggregate the data from the sensors. The main design decisions that were made for this component was the selection of the computer that would be used to run the board games, and the operating system that would be used. In the end, the LattePanda development board was selected, along with the Windows 10 operating system in order to support the Unity and Unreal Engine communities.

2.2.2 Modifications

Originally, the computer that was selected to run this project was the Raspberry Pi 3B, and the operating system that was selected was the Android Things OS. Both of these selections were not included in the final design. Although Android offers the capability of playing Unity and Unreal Engine games in the same vein as Windows 10, its hardware integration support is much more limiting, and often has to be programmed in manually instead of having naturally supported drivers. Furthermore, after testing the performance of the Raspberry Pi 3B trying to run Android, it was clear that there were severe limitations on the quality that was to be expected from games if limited to the Raspberry PI. Frame rates of test applications dropped well below the unperceivable range, and became slightly unbearable. This prompted the decision to change the OS to Windows 10 to continue to support Unreal Engine and Unity games.

2.2.3 LattePanda and Windows 10

Since this project was primarily a prototype meant to showcase the potential of this a board game console, the most important factor when deciding the operating system of choice is one which creates the most ideal environment for hardware integration and software development. Windows 10 has the advantage here as there are countless drivers for any touch screen monitors, mice, and many other peripherals. After this decision, the LattePanda development board was the obvious choice to run windows 10 in a small form factor. The LattePanda has 4GB of RAM, 64GB of internal storage, has 2 USB 2.0, 1 USB 3.0, HDMI output, a fully fledged internal Arduino Uno, and runs off a 5V supply or microUSB. It is an extremely versatile computer, and only draws 200mA at maximum load. Upon running the LattePanda board, it was clear it was very capable of running any low-intensity games at a decent frame rate, however, since there is no dedicated graphics card, the fps of many games do tend to drop on this system.

2.3 Power Design

2.3.1 Overview

Another important aspect of the design is the power requirement of the system and its individual components. In order to power the entire system, each component must have a supply with its rated voltage, and the supply must be able to source the total current that will be drawn by the system without any decrease in voltage. Calculating the total current drawn by individual components, and the maximum rated voltage will give an idea about what power supply will be used.

2.3.2 Voltage and Current Requirements

Table 1: Voltage and Current for System Components

	Max Current Draw [A]	Rated Voltage [V]
LattePanda	0.5	5
Touch Display Board	1.7	16
RFID Sensor Board x30	0.9	3.3
Arduino Nano x3	0.195	9
TOTAL	3.295	

Table 1 below shows a summary of the electrical characteristics for all of the components within the system, outlining both the maximum current draw and the rated voltage. The total amount of current drawn by the system in worst case is 3.295A, so the power supply that will be used must be able to source at least that much current without dropping voltage. The maximum voltage that must be supplied is 16V, which means that, unless the design implements step up converters, the supply must have at least 16V as an output. With these design specifications taken into consideration, the supply that was chosen was a AC to DC converter which rectifies and regulates the input and supplies a maximum of 4A of current at 16.7V. This coupled with a method of stepping down voltage will allow power to be supplied to the system.

Buck converters, or step down converters are DC to DC converters that step down output voltage while stepping up the current that can be supplied. The 16.7 input voltage can be converted to the required rated voltage of each of the subsystems of the Digiboard using three buck converters, one to step the input down to 9V, 5V, and 3.3V respectively. The 16V display board can receive the 16.7V directly since it is regulated, and is within the tolerance range specified in the datasheet of the display board. The buck converter that was used was the LM2596 variety, and takes an input from 3V to 40V and regulates down to any voltage at least 1.5V below the input. An important caveat is that its maximum output current is 3A, but this is within the constraint of the maximum current needed which is 0.9A for the RFID sensor board. A potentiometer is located on each buck converter which allows the user to change its regulated voltage level.

2.4 Gaming Platform

2.4.1 Overview

The creation of the hardware is only one aspect of the Digiboard console. There is also the board games that accompany it. In order to see the full potential of a digital board game platform, an example game was created to showcase the capabilities of the system. An entirely new experience based on the board game *Settlers of Catan* was developed in order to show how modern board games can be greatly enhanced through the use of animation, a touch screen, and guided play while maintaining personal interaction through the use of game pieces. In addition to the fully featured Catan game that was created, game pieces tailored for the new Catan game were fabricated to complement the board game experience, each with its own RFID tag to be localized and identified. This game was created to showcase the innovative capabilities of the board game console, but in order to integrate with the hardware of the system, a method for transmitting information from the sensors of the system to Unity and Unreal Engine were developed. This data is exposed to the engines through the use of custom plugins which were developed for each platform so as to create a simple to use method of integrating with the Digiboard platform. This would allow other developers who wished to create a new board game for the platform to integrate their game seamlessly with the system.

2.4.2 Unity

The platforms that were decided to be supported were Unity and Unreal Engine since these Engines are the two most popular in the gaming community, they have the most support, and are for which extensible plugins are extremely easy to write. The prototype Catan game that was developed however was created with the Unity platform. Unity is a scripting gaming engine that allows users to create games extremely quickly with a very easy to use interface and C# as the main scripting language. Games are developed as a compilation of scripts that create logic to move, spawn and mutate assets such as images, 3d models or sounds, and Unity provides tools to make these interactions extremely easy. Additionally, there are many plugins and assets that are created by Unity or the Unity community that ease the process and provide resources make development even easier.

2.4.3 Digiboard Plugin

The Digiboard plugins that were developed bundle the logic that creates an access point for the Arduino data that is available on the LattePanda's Arduino Uno, if it is connected. This is facilitated through a C++ library that the developers of the LattePanda created called Firmata which exposes the pin data of the arduino to the host computer. This data is then converted into a struct data type which can be transferred and used within the game. This C++ code that accesses the data is compiled into a DLL or Dynamically Linked Library which can be called directly from the plugin's Unity C# script. This C# script keeps a record of all the sensor data as well as a record of the touch data recorded by Unity's input manager. The timing and location of each touch and RFID identification is compared to discern which touches are player's fingers and which are associated with a game piece. If the script has determined that a touch was over a particular sensor that has

detected an NFC tag within a 200ms margin of time, then it is ignored as a Digiboard touch input, and is published instead as a game piece location. This data is compiled and published at a frequency of 10Hz. It is published as a global object which can be accessed directly in the user's code if they choose to implement the plugin into their code.

2.4.4 Catan Game

The game Settlers of Catan is a board game about competing for resources in a crowded area and creating a sprawling civilization before anyone else. It is a 2 to 4 player game that allows different players to build settlements and cities around resource hexes. Each resource hex has a number which, upon rolling that number, players with adjacent cities or settlements collect the associated resource. Players can also build roads to branch away from their initial settlements and find other resources they need. **Figure 4** shows a screenshot of the game which conveys the different settlements, roads and resources.

Figure 4: Zoomed Screenshot of the Catan game created for Digiboard

One central aspect of traditional Catan is that once a 7 is rolled, which is the most common roll out of two dice, the Thief is activated, and the player is able to choose a hex on top of which the Thief will remain, blocking all resources collected on the hex he stands on. In this version of Catan, two game pieces were created , the Thief, and the Terraformer. The Thief does the same as the traditional Thief in Catan, but the Terraformer does something that would be extremely impractical if it were a real board game, it changes the resources of all adjacent hexes to a random resource, but since this is a computer game, it can be performed instantly. The Thief and Terraformers are not simple 3d objects in the game, however, the player has to physically move the pieces in question onto the board game to play his turn. Figure 5 shows the state of the board game once either the Terraformer or the Thief has been placed, and this is all completed using the plugin that fetches data from the Digiboard's RFID sensor array.

Figure 5: The rock hex has been selected after a player places a Terraformer or Thief game piece

These interesting game mechanics would not be possible without the speed of a computer, and player interaction with game pieces and the digiboard are meant to allow players to get the best of both the board game and video game worlds with this experience.

2.5 Mechanical Design

2.5.1 Case Design

The mechanical design component of this project consists primarily of the case. The case is the main structural component of the design, and provides support ands space for each of the sensor and control boards inside the console. In order to ensure that the packaging of the components was possible, and to know where to put mounting holes on the boards before being sent to fabrication, a CAD model of the entire Digiboard was created to inform the design. This CAD model was comprehensive and included every model that was to be included in the final product. The final CAD model of the case and all its components is shown in **Figure 6** below.

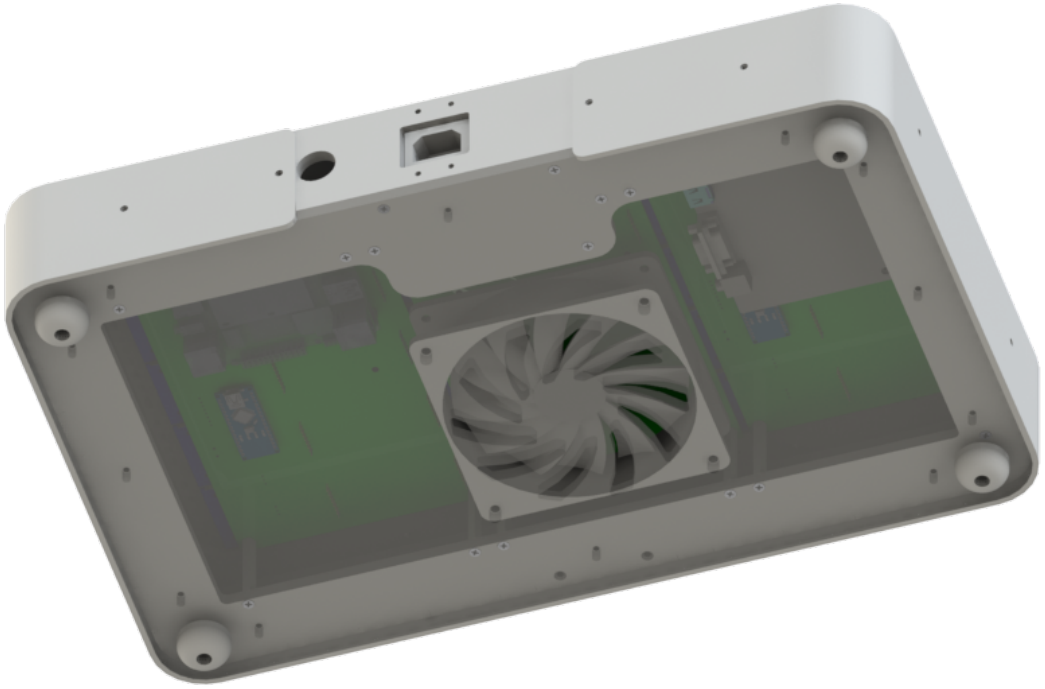

Figure 6: Rendering of the case of the Digiboard

The design of the case began with the screen, the LCD screen that was used as the basis of the touch screen was not symmetrical as would be expected, it had a pronounced bezel on the bottom side which also supported the variety of inputs and outputs to the panel. This meant that the screen would be placed offset relative to the spacing of the board in order to make it appear symmetric. One of the main design goals was to place the RFID sensors as close as possible to the screen so as to ensure that there was a high margin for piece detection. Furthermore, the sensors must be placed such that their working area spans the entirety of the screen. Since each of the sensors have a deadzone at the bottom where the circuitry and connective pins are located, these areas do not sense RFID tags. When placing them relative to the screen, they were placed in order to minimize deadzone and to maximize the working sensing area.

The sensors also needed to be positioned such that the 10 thousandth of an inch metal lattice can be placed between each sensor. This was modeled in CAD to ensure that the lattice would not touch the screen and cause it to damage the display. A layer of quarter inch pliable foam was

added on top of the lattice to separate it from the display. The sensors themselves are soldered to the connective board below, and are further supported by adhesive foam in order to protect the solder joints and prevent any damage in the event of a drop or applied force transferred through the touch screen. Additional foam was placed above the sensors in contact with the top layer of foam to provide cushioning support on both sides of the sensors. In order to achieve perfect spacing for the RFID sensor boards relative to the screen, standoffs were used which mounted the board to the case.

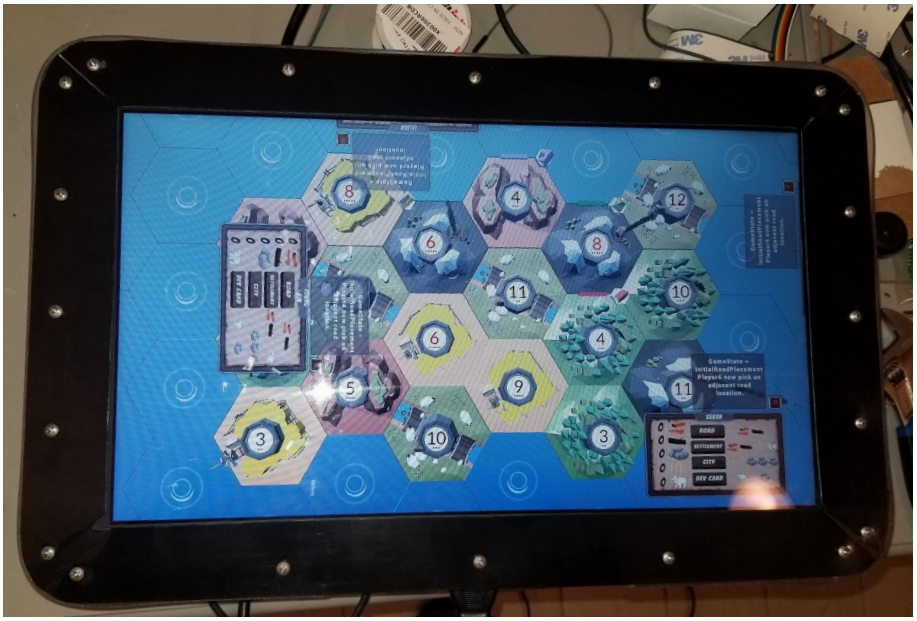

Figure 7: Front of Digiboard console

For the case itself, the material was chosen as per the specifications of the UW laser cuttin6g services, so 1/4” clear acrylic was chosen. This was the strongest material that was able to be cut by the laser cutter, and provided very good support for the case. The case was designed such that the mounts for the integral components such as the screen and sensors matched up, then the second priority was making it symmetrical relative to the screen. The base plate of the case was created to be removable for easy access to the internal electronics during prototyping. The base also has cutouts for the 12V fan which provided cooling for the entire system. On the bottom of the case were four rubber bumpers meant to raise the chassis up by a few millimeters. This provided an escape path for airflow from the fan below. A hole at the back side of the case provided an intaking location for air, as well as a secondary access point for cables used for debugging and power.

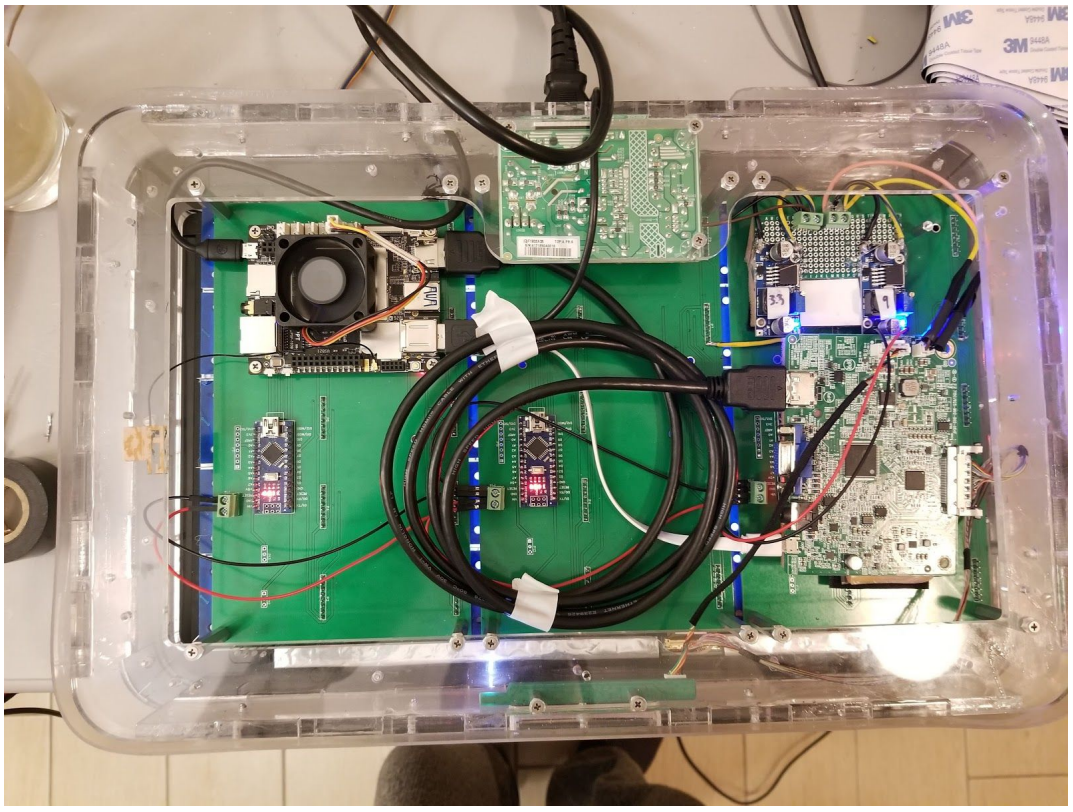

Figure 8: Final assembly of the Digiboard console

2.5.2 Manufacturing and Construction

The manufacturing process of the case was relatively straightforward. The bulk of the design was simply laser cut 1/4” acrylic which eased the manufacturing process immensely. The first assembly step was to solder all of the relevant electrical components onto the sensor board. For the entire design, a single thread size was selected so that the assembly process would be much more painless, and only one tool needed to be used. Next, once the six sides of the board’s case were all laser cut, they were assembled together. They each have interweaving jigsaw-like appendages which were wedged together and fused with acrylic epoxy in order to create the outer shell of the case. After a few hours of curing, the circuit boards could then be mounted to the static plate of acrylic on the bottom side of the case, not the removable plate. Next the screen could be mounted directly to the top plate of the acrylic case. Next the foam was applied to the back of the LCD screen to protect it once the lattice was in place. Finally the wires connecting all the electrical components could be connected, and finishing touches could be made. 4 bezels of plastic were applied to the top of the case to hide the imperfections of the top plate and give it a more polished look. A 1/16” polycarbonate trim was applied to the outer edge to prevent any fingers from entering the case. The fan and rubber bumpers were mounted, and then finally the base plate was screwed in.

2.7 Testing and Performance

2.7.1 Commissioning Schema

Commissioning is the process of creating the testing procedure that will be used to validate the design. There are a variety of sub components that need to work independently and as a group once each is integrated to the entire system. Creating a method to test individual components and the entire system is necessary to validate the project.

In order to ensure that the entire design would function, each subsystem needed to be tested individually. Furthermore, since many of the components that were to be procured or manufactured came in sets of more than one such as the thirty sensors or four arduinos, it was decided that smaller scaled versions of each subsystem would be tested to ensure its functionality before buying and creating the real model to scale. This would save on costs, and would ensure the functionality of the system at hand. For the electrical subsystems, this meant that a prototype with breadboards would be made before buying any components in bulk. Researching expensive components and comparing against what other people in the community have been able to do is another good way of validating components that cannot be tested at a smaller scale such as the LattePanda.

2.7.2 Localization and Identification Validation

For the localization and identification subsystem, ensuring a scale model of this system worked correctly was the top priority. In order to validate this at scale, 5 sensors were purchased and tested with an Arduino Nano, breadboards, and jumper cables. This scaled system was setup so that it would replicate exactly what the larger system would look like with each of the pins connected along a bus and separate chip selects. The arduino was programmed so that it could read the data and output serial data to the console. RFID tags were brought close to it while it was running to verify if it could read. It was during this phase that it was noted that there would be significant interference and noise from adjacent sensors, as no data could be read if two sensors were too close together. To try to alleviate this, the approach of wedging tin foil between adjacent sensors was tried, and was surprisingly very successful. The tinfoil isolated the RFID signal within its zone, and almost got rid of interference. This was later brought to fruition with the metal lattice. Once this system was validated, five more sensors were purchased to verify that the ten sensor load that would be seen on the real board would function. The only worry was that the SPI bus would be overloaded with ten sensors, however upon testing, the system worked as intended.

2.7.3 Computational hardware and OS Validation

Another aspect of the project that needed to be validated was the computational hardware and operating system. One of the biggest unknowns of this project initially was which operating system would be able to support the intensive graphics of video games but also be relatively easy to prototype within. Initially the priority was to use something that had the best support for video games and provide the best graphical support, which erred on the side of the Android, but upon running Android on the Pi, it was clear that there would be a lot of hiccups with that operating

system on that hardware. Testing a graphically intensive game was a good test to see if the our prototype game that would be developed would even run on the system. Because of the ill results of this test, it was decided that the better operating system would be Windows 10 on the LattePanda. This was further corroborated after communicating with other users of the LattePanda. Many users were able to play a variety of games on the LattePanda without much issue or drops in frame rate.

2.7.4 Electrical Validation

The final validation that was required was electrical validation. This was the process of ensuring that the electronics of each subsystem was functioning properly before everything was integrated into a single unit. Since there were effectively four different voltages levels all being sourced from the same voltage supply, each tier of voltage was tested separately using a power supply first. Once each of these subsystems were validated to be working, then they would be connected to the buck converter to receive power directly from the board power supply, however they would still be connected one at a time, and their corresponding current draw and voltage drop would be measured. This ensured that the power supply was supplying voltage properly to each sub component. After each system was verified to be working as specified, they were all connected in parallel.

3. Logistics

3.1 Schedule

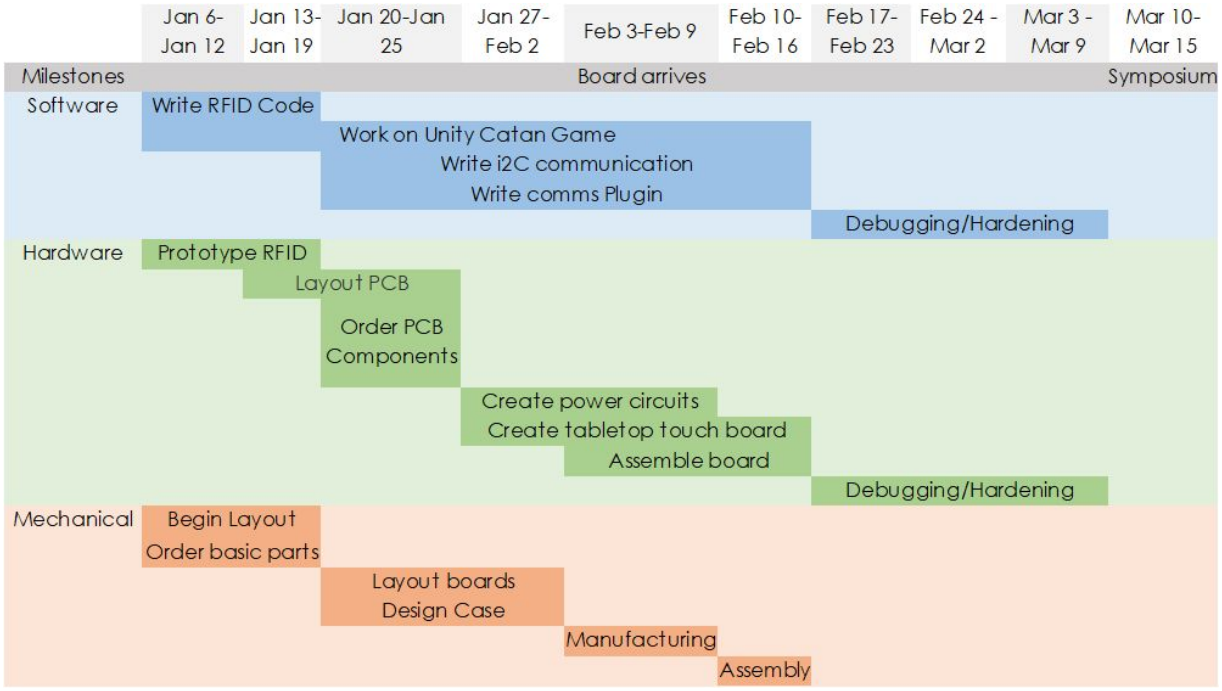

Figure 9: Initial schedule for the term

As shown in **Figure 9**, the schedule for the project was broken down into three parts, software, hardware and mechanical. This schedule was very liberal since it allowed for a three week period near the end of the term before symposium for debugging and hardening of the final design. This however was not enough time, and there were numerous delays. The major choke points and waiting periods occurred after ordering the PCB and after ordering the RFID sensors, both coming from China. The shipping of the boards was delayed by two weeks causing a detrimental delay in the schedule by two weeks. The sensors were also delayed by just over one week, which caused an earlier delay before more involved testing could begin. This meant that there was not much time for debugging or hardening before symposium.

3.2 Budget

Table 2: Bill of Materials for the Digiboard Project

ITEM	PART #	DESCRIPTION	QTY.	Price	Total
1	Screen	LCD Display, touch screen, interface board	1	273	273
2	RFID-RC522	Sensor boards	30	2.5	75
3	NanoV3 v4	Arduino Nano	3	5.99	17.97
4	91780A527	Female Threaded Hex Standoff, Aluminum, 3/16" Hex, 5/16" Long, 4-40 Thread	7	0.42	2.94
5	91780A531	Female Threaded Hex Standoff, Aluminum, 3/16" Hex, 7/16" Long, 4-40 Thread	4	0.57	2.28
6	LattePanda	Windows 10 Enabled development board	1	280	280
7	91780A630	Female Threaded Hex Standoff, Aluminum, 1/4" Hex, 1-3/8" Long, 4-40 Thread	12	0.64	7.68
8	91771A110	18-8 Stainless Steel Phillips Flat Head Screw, Passivated, 4-40 Thread Size, 1/2" Long	33	0.0461	1.52
9	97975A205	Rounded Head Thread-Forming Screws for Brittle Plastic, 410 Stainless Steel, Number 6 Size, 3/8" Long	21	0.0562	1.18
10	91099A155	18-8 Stainless Steel Phillips Flat Undercut Head Screws, Passivated, 4-40 Thread Size, 1/4" Long	3	0.036	0.11
11	91099A165	18-8 Stainless Steel Phillips Flat Undercut Head Screws, Passivated, 4-40 Thread Size, 3/8" Long	3	0.0459	0.14
12	Cooling fan	Corsair 12V desktop computer fan	1	0	0
13	Bumpers	Rubber Bumper with Unthreaded Hole, SBR, 1" OD, 1/2" High, 7/32" Mounting Surface Height	4	0.552	2.21
Total					$664.02

Table 2 outlines the bill of materials for the project. The initial calculated budget that was split amongst two people was $700, and the final budget totalled just under that result. This should have been over the initial budget that was specified in term 1, however with the changes to the localization and identification system, the cost was drastically reduced.

4. Conclusion

In conclusion the initial problem which was to create a console that provides a happy medium between the worlds of board games and video games was completed. The Digiboard console complete with an RFID sensor array which localizes interactable game pieces functions as intended, and meets most of the criteria and constraints that were initially set out upon. Although there were some modifications to the initial design, localization and identification were rescoped in order to save on costs and create a more accurate model. The computation system was also changed from a raspberry pi running Android to a LattePanda running Windows 10. This change offered more computational and graphical power while still supporting the Unity and Unreal Engine platforms. In order to showcase the system, a version of The Settlers of Catan game was created which leveraged the sensor board to terraform or block resources on the map. This proved to be a fun way to interact with the board and utilize the touch screen and video game aspects of the console that would be impossible to replicate with a physical board game. Finally, the mechanical components such as the case were fabricated to house all the components to create the final design.

5. Recommendations

This project was meant to be a prototype which shows the vast potential of a board game console. There are so many interactions that are possible with a unique system like Digiboard. However, since this is just a prototype there are a variety of things that can be improved upon. One recommendation for future work would be to look into utilizing the RFID signal itself to localize game pieces instead of having two separate mechanisms for localization and identification. This is currently possible with technology such as RFIND, however it is very expensive. Another aspect that can be improved upon is to work on attempting to implement this system with an Android controller instead. There are many cheaper Android boards, however hardware integration is more challenging with them, this would allow for much better graphics processing, and would offer the possibility of using an Android skin as the custom operating system for the board instead of running only a Unity game. Another possible enhancement would be to use a 16V battery as a voltage source instead of using a rectifier. Every internal component save for the LCD interface board uses a buck converter to regulate voltage anyway, so implementing a battery wouldn't be too difficult. Another possible enhancement would be to encapsulate the entire RFID sensor board into a single board instead of soldering individual sensors to the board. This would greatly save on costs, and would allow the board to not have any dead zones unlike the state of the board right now.

6. Teamwork Effort

This project was done with two group members Isaiah Erb, and Elijah Erb. Isaiah Erb specializes in software design, project management, hardware design, and embedded programming. Elijah Erb specializes in mechanical design, manufacturing, procurement, hardware design, and graphic design.

The group member Isaiah Erb programmed the Catan game, and created the plugins that allowed other users to integrate with the hardware. He programmed the Arduinos to collect data from the sensor boards via the SPI protocol, and talk to the main arduino via I2C. He prototyped the initial localization method for localizing magnets, and created a tabletop prototype of the RFID sensor system with Arduinos. He worked in tandem with Elijah to design the electric circuit board that facilitated the connection of all the sensors.

The group member Elijah Erb laid out the entirety of the CAD for the system. He designed, manufactured, and assembled the case and all its components. He helped design and layout the PCB, and conducted most of the soldering of the board. He created most of the graphical designs that were showcased during the symposium, and also created the felt plushies that differentiated the two game pieces that were shown during the symposium.

References

[1] MFRC522 Datasheet. Contactless Reader IC. NXP,
https://www.elecrow.com/download/MFRC522%20Datasheet.pdf

[2] Trace Width Calculator.
http://circuitcalculator.com/wordpress/2006/01/31/pcb-trace-width-calculator/

Appendix A:

Sensor Board Schematic