Hou Yi: Sun Shooter

Northeastern University
Boston, MA 02115
April 19, 2018

Prepared By:

Fatema Janahi, Selena Nguyen, David Muir, Ciaran Farley

This report includes a summary and reflection upon our solution to Project 3: The Great Museum Caper. We were assigned the problem of creating a fun, safe, and exciting game for children ages approximately 4-9 and given a theme of Chinese mythology. Our solution to this problem was to create a bow and arrow game based on the story of a Chinese mythological archer Hou Yi. As a result, we designed a bow and arrow game in which the player's objective is to shoot down five suns, each worth 100 points. This consisted of a bow mounted on a stand and stringed with surgical tubing, a height-adjustable target and bow stand, and six thin wooden arrows with foamed tips. A box on the bow stand facing the user would display the score on an LCD screen, lighting an LED and sounding a speaker every time a target is knocked over. The results were a success; the children at the Boston Children's Museum enjoyed our game while learning a good deal about its Chinese mythological background, although aiming and user feedback could be improved for an even better experience.

INTRODUCTION TO PROBLEM

We were tasked with making a game for an exhibit for the Boston Children's Museum on April 7th, 2018. The game needed to have physical as well as electrical components and be fun for the kids who go to the museum. It also had to have some basis in Chinese mythology to fit within our section's theme as decided by the Executive Board, and needed to have a 3-D printed component designed in SolidWorks. This component was decided to be a stamp that related to the theme of the project to create a unifying experience among the section's games. The first part of analyzing the problem was to consider the museum's rules. The game had to be safe and not contain any latex. It also had to cost less than $80 to produce and be transportable from Northeastern University to the museum. It, of course, also had to be fun. One objective based off this was that the game should be flashy, and have some sort of audible or visual feedback. The game also needed to be quick to play; if it took more than two minutes children might lose interest. For the functions of the game, our goal was to have the game sense targets when they are knocked down, indicate the player's score, and control the direction and force of shooting arrows.

BACKGROUND INFORMATION

Our game is based on the Chinese mythological archer Hou Yi. According to Chinese folktale, Hou Yi is the god of archery who had descended from the heavens to protect mankind. He used his bow and arrow to shoot down nine of the ten suns that were scorching the Earth. The tale of Hou Yi is the story behind how the sun and moon began to govern day and night.

DESIGN PROCESS

Our initial task was to devise a complex but feasible game that would be fun for children and would implement electronics. Our first idea was a medieval sword fighting game. This would have had two users with foam swords try to hit the other player's armor for points. Force sensitive resistors in both players armors would have recorded hits and then sent this information to a box that would display the score on a LCD screen. This idea was deemed unfit for our client, as many parents and the Museum would have qualms about the swords, even if they were foam.

We quickly moved onto another idea. We brainstormed a target shooting game featuring a bow and arrow. Users would shoot at a wall with targets that has force sensitive resistors, which would then send data to a screen on the user's wrist if hit, increasing the score. Our initial idea called for an adjustable stand for targets and an adjustable height for the bow and arrow. Once we aligned this with the myth of Hou Yi, we aimed to have ten targets. We were luckily able to envision and research what technology this required early on. The force sensitive resistors were deemed to be too expensive at $6 each, so IR sensor and emitter pairs were bought instead. NRF24L01 radio transmitters were decided upon for the wireless communication between the targets and the screen on the arm. Arduino Megas were also bought due to the amount of pins required for the radio transmitters and whatever other components they were paired with.

Figure 1: Initial Drawings of Concepts

The first physical component constructed was the stand for the bow and arrow. Our goal was to make this stand both rotatable and adjustable in height. Brainstorming ways to achieve this was a challenge. We then decided to first construct a four-foot stand out of wood, drilling four angled legs to each side of the post and atop a 2 ft by 2 ft wooden platform as a base.

The next step was building the bow. We first planned to construct the bow out of either flexible wood or plastic, the former being far more affordable. We obtained as flexible wood as we could find, but later decided that connecting three separate, angled wooden parts was much more feasible than attempting to bend sturdy wood into an arc of a bow. We then instead constructed the bow from a wooden plank (Fig. 3). We drilled holes through the top and bottom of the bow, through which we inserted surgical tubing, which served as bowstring. We layered the surgical tubing to reduce elasticity of the bowstring and increase the power of a shot.

We designed a peg system to adjust the stand to set heights. We purchased slim wooden dowels and drilled four identical holes of corresponding radii into the wooden post. We drilled identical holes at the top and bottom of the handle of the bow as well. We cut the wooden dowels into four pieces, fixing them into each hole so that they protrude enough to insert into the bow as well. This way, the bow is attachable and detachable to the pegs at different heights. We used the same sized wooden towels as arrows for the bow. To make the stand rotatable, we decided to add four wheels underneath the base of the stand, which were drilled in and would allow for aiming.

Our next component was the stand for the targets. We constructed this out of wood as well and measured an ideal height corresponding to the tallest height of the bow. For our milestone prototype, we fixed the stand at this height. Under the platform on which the targets would be placed, we attached a hinged wooden box to hold our wiring. We drilled ten holes through the upper platform in which we would place our IR sensors surrounding each target. When some IR sensors broke, we replaced them with photoresistors, which would change the user's score when a target was knocked over and on top of a photoresistor.

Figure 2: Height Adjustable Target Stand with Detachable Bow

Our original plan was to create a wristband consisting the LCD screen that would display the user's score. For our milestone, we decided simply to construct a wooden cuboid that would fit around a child's wrist. This component consisted of a box containing wiring and the LCD screen.

Our next required components were the targets. We used small wooden squares as the targets, which would be placed on the target stand and fall off once hit by the arrows. The remainder of our construction consisted of smaller additions that we made to different main components as we went along. For instance, after testing the bow and arrow, we decided we needed some sort of indentation in the bow to rest and aim the arrow as the user prepares to shoot. We then cut thin wooden sheets into a rectangular slot approximately the size of the arrow and attached this to the bow.

Our next small addition served for aiming the arrow as well. We found that the arrow was still difficult to align and keep in place when drawing the bowstring. For our milestone, we created a small cap out of hot glue to put on the bowstring, using the arrow to create an indent of proper fitting in the hardened glue. This addition to the bow made it much easier to aim and use.

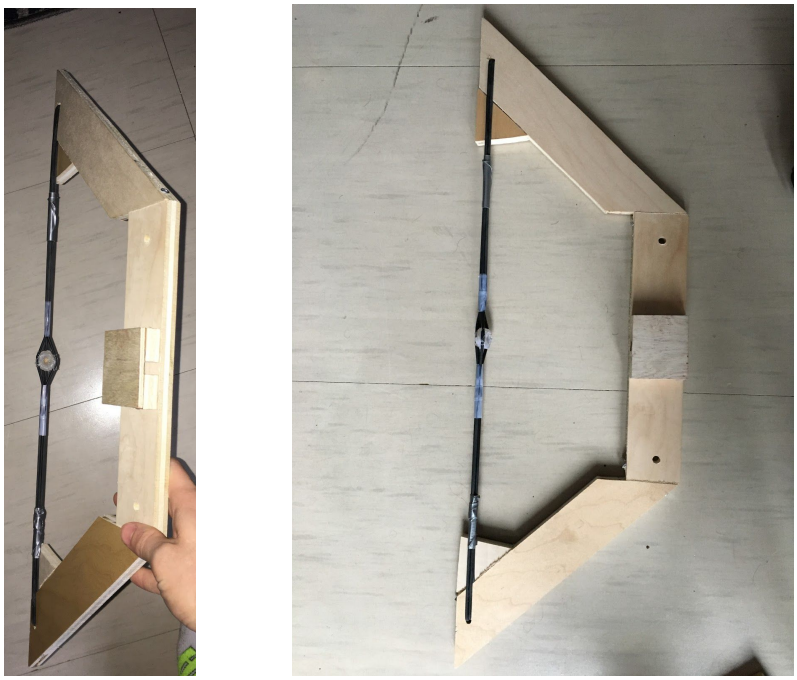


Figure 3: Bow with Initial Aiming Additions

Our work after the milestone prototype consisted of small changes for efficiency and enhancements for visual appeal and feedback. Our main change was in the target stand. We made the stand out of two separable components to make the height adjustable and to match the heights of our bow stand. We also used hinges to attach the targets to the stand so that they would stay in place. To display the user's score, we abandoned the wristband and instead constructed a box attached to the bow stand with the LCD screen (Fig. 4). Lastly, we made adjustments to the bow and bowstring to increase efficiency. For the bow, the wooden component we had previously made to hold the tip of the bow in place was reconstructed. We made an opening on the top so the user could more easily set each arrow into place. For the bowstring, we abandoned the hot glue component because it didn't set the arrow-back into place well enough. Instead, we cut small slits into every arrowback that would snugly fit the surgical tubing. One arrow had a 3-D printed addition with a slit described later. We also duct-taped the surgical tubing so that it was easier to maneuver to fit with the arrow.

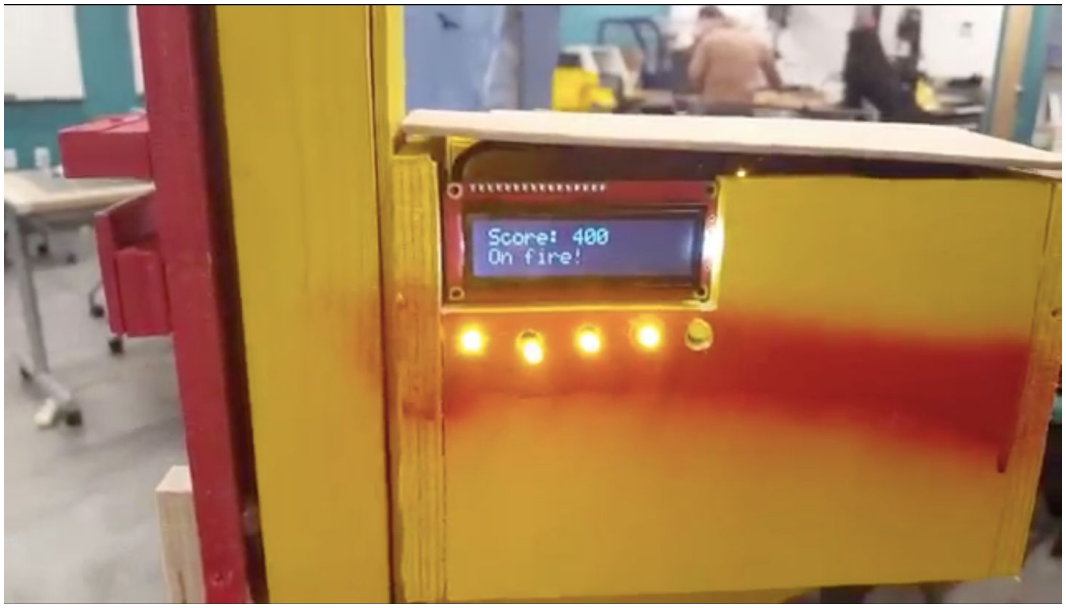


Figure 4: Box with LCD Screen

To enhance the visual appeal of our project, we painted and decorated a bed sheet to drape over the stand. We also used larger targets out of thinner wood that looked like suns to represent the suns shot down by Hou Yi. We spray-painted these suns, bow stand, and arrows, and added score values to the targets. And finally, to make our game lighter and more ergonomic, the base of the bow stand was cut down. Our final product was fully painted, easily portable, adjustable for both components, and easier for children to handle.

DESIGN PROCESS: ELECTRONICS

Once the radio transmitters arrived, a test was set up on regular RedBoards. The aim of this test was to simply turn a light on via a button connected to a separate RedBoard (Figure 5). The wiring for this was based off the schematic seen in Figure 4. Once this test worked, the radio transmitters were connected to the Megas.

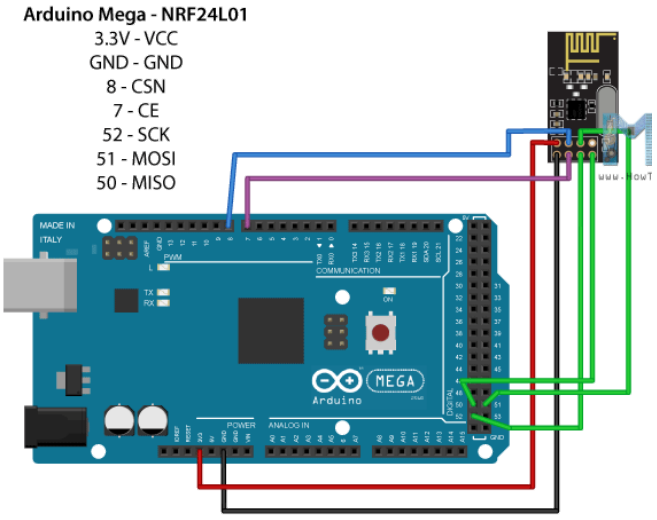


Figure 4: NRF24L01 Wiring Guide[1]

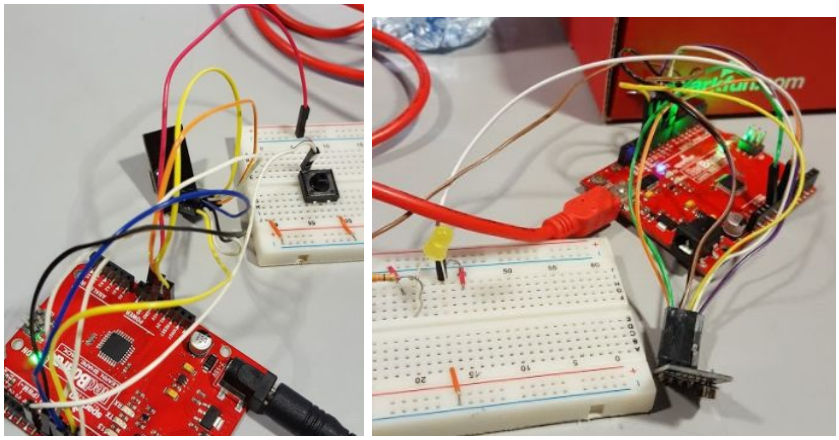


Figure 5: Testing Transmitters on RedBoards

The next step taken was wiring the IR sensors and emitters (Fig. 7). The ends of the sensors and emitters were soldered to jumper wires that had their black connection ports taken off. The ground and power ends were labeled with yellow and black electric tape respectively. Connecting these to a breadboard proved tedious, albeit successful. The serial monitor was used to verify that they were working as expected. A LCD screen was then wired to other Mega(Fig. 6) . The wiring for the screen required many connections in specific places, leaving little room for error (Fig. 7). This led to a prolonged troubleshooting session when the LCD screen didn't work. A different LCD screen was substituted to narrow down the problem. The screen continued to fail to print any text. We soon discovered the back of the screen was painfully hot, leading us to look for wiring issues. We found two wires had been switched, fixed this error, and used a new screen again. Finally, it worked.

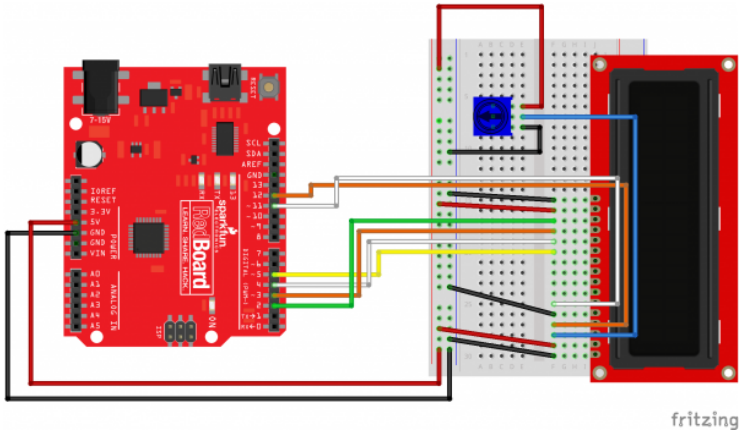


Figure 6: LCD Screen Wiring Guide[2]

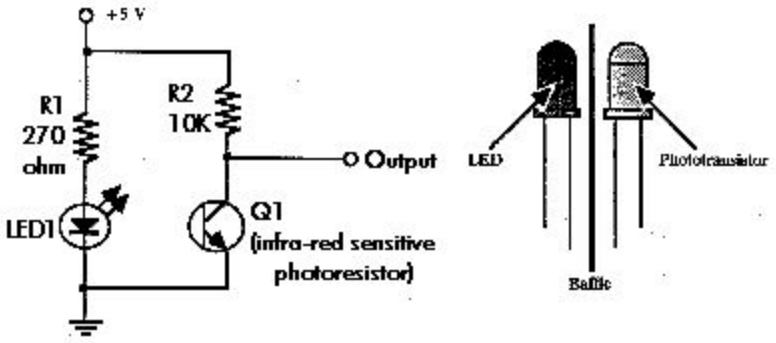


Figure 7: IR Schematic

The final step was to set up wireless communication between the Megas where the action of a target being knocked over changes the screen's display. This was done through trial and error of several methods of sending different kinds of data. Sending characters was settled on as the best way. The electronics were now ready to be put into the structures built.

A small housing was built under the top plank of the target stand, and holes were drilled on the top plank for the IR sensors and emitters. The breadboard and Mega were placed underneath in the housing (Figure 8) and the sensors and emitters were connected to male female jumper wires to feed them through the holes. The LCD screen and its Mega were placed in the box described earlier, completing the project for the milestone.

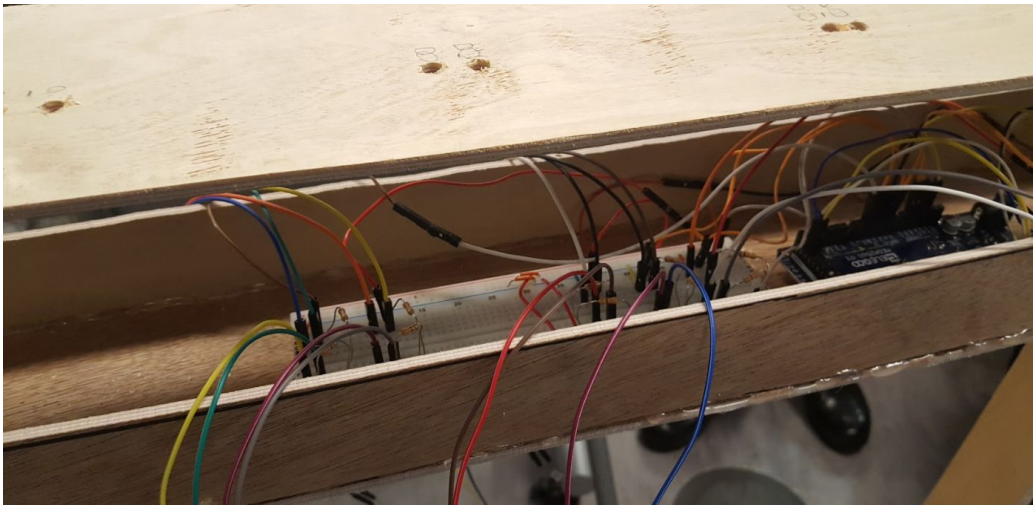


Figure 8: Wiring for Targets Under Stand

Furthermore, the final milestone design we settled on can be summarized by the flowchart below in figure 12. The first Arduino Mega is wired to the infrared sensors, thus it receives readings from them, and to the radio transmitter. The radio transmitter can then transmit data to the other transmitter, which acts as a receiver. Since it is wired to the second arduino mega, which is connected to the screen, data is able to be outputted to the screen.

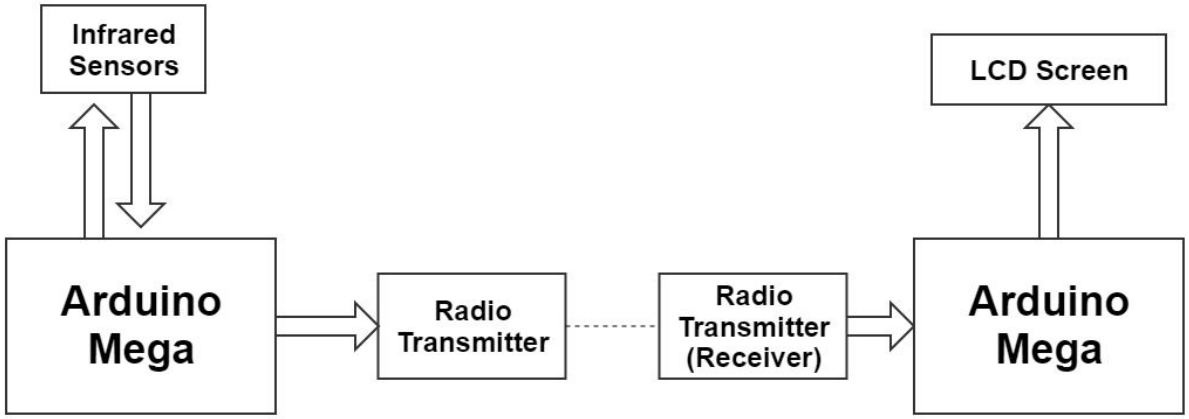


Figure 14: Design Flow Chart

After the milestone, we prioritized making sure the target sensing system worked, and adding more user feedback via LEDs and a speaker.

FINAL DESIGN

Our final design utilizes three primary physical components: the bow stand, the bow, and the target stand. The bow stand is a sturdy, but fairly lightweight design. The base of the stand is a wooden cross with arms a foot long and 3 inches wide. A ¼ inch hole was drilled at the end of each arm for wheels that were inserted to aid mobility and aiming. Each arm also has a triangular support made from 2 by 4s. These were screwed in to support a 4 foot tall post that stands in the middle of the cross. The post had 4 ¼ inch holes drilled in it, 8 inches apart. Four ¼ inch pegs made from a wooden dowel were glued into these holes to act as the height adjustment mechanism for the bow. Lastly, a 4 inch by 6 inch by 3 inch electronics box was built from plywood and wood screws and was glued to the upper part of the bow stand post. This box housed the electronics that controlled the LCD screen so the user could see their score. The entire stand was then painted yellow, and is shown in its final state in Figure 10.

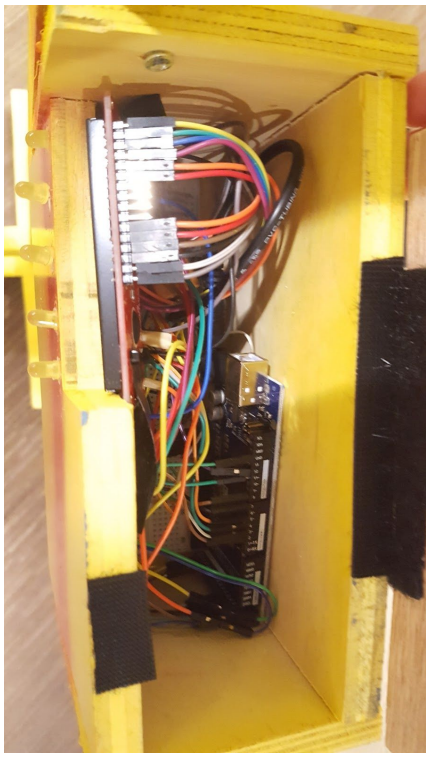
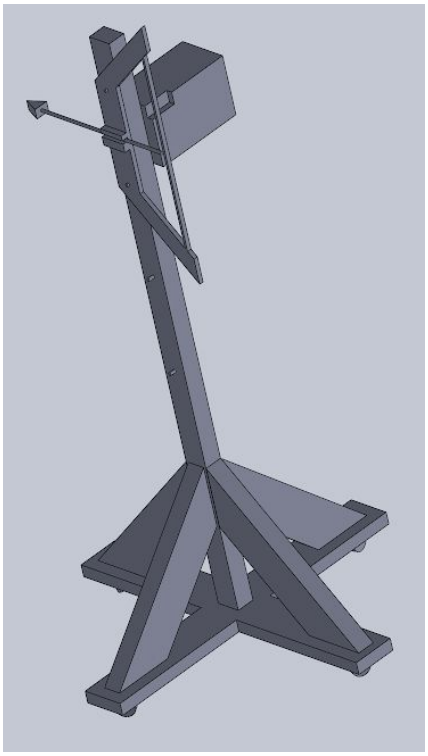




Figure 10: Bow Stand (left) - Electronics Box (Middle) - Solidworks Model (right)

The bow itself is composed of three 8 inch pieces of a skinny wooden plank. These pieces were screwed together to form a bow shape. Two ¼ inch holes were drilled on the top and bottom of the bow. 6 feet of surgical tubing loops through these holes to provide the elastic shooting force to the ¼ inch wooden dowel arrows. Two more ¼ inch holes were drilled on the top and bottom of the bow handle, 8 inches apart. These holes attach the stand and bow together, while also providing a mechanism to adjust between three different height choices. The arrows were foot long ¼ inch wooden dowels with sturdy triangular foam tips. The backend of the arrows had small slits cut into them. In order to make using the bow as easy as possible, the surgical tubing was taped in a way that the arrows could be easily hooked onto one strand of the tubing. Small wooden blocks were also glued to the front of the bow to guide the arrow as it is shot. These small additions to the arrow and bow allowed younger kids who weren't as nimble to shoot the bow with less difficulty. The final design of the bow was painted red and is displayed in figure 11.

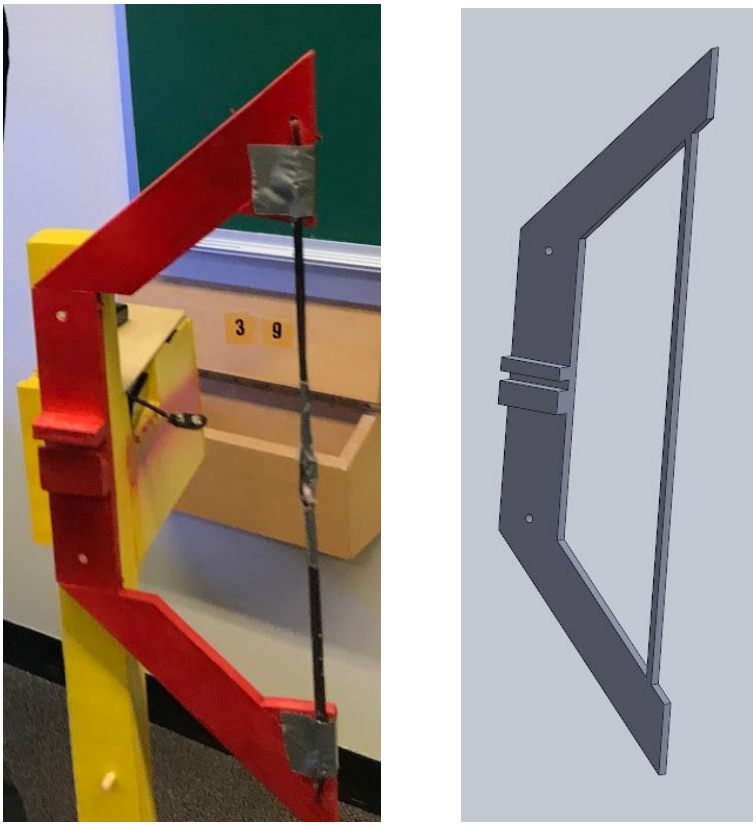


Figure 11: Bow (left) - Solidworks Model (right)

The target stand is made of two 2 foot by 6 inch wooden planks with 4 foot tall wooden posts screwed in on either side. These wooden post were cut in half and offset slightly to allow height adjustment of the stand. The lower level had the 2 parts of the post completely overlapping while the upper level had only ½ overlapping. The make adjusting the height easier, small wooden girders were added to the out part of the post to keep the two halves in line with each other. The upper plank has ten ⅛ inch holes drilled in it for sets of an emitter-sensor pair/photoresistor. In between each emitter and sensor/above the photoresistor the targets were attached with a hinge. The targets themselves consisted of small wooden blocks that the hinges screwed into with yellow painted suns above. Beneath this upper plank was an electronics carriage. An ⅛ sheet of wood was cut and glued in a box that was subsequently glued under the targets. The front of the box was attached with hinge to allow access to the electronics in case of an issue. Lastly, a bedsheet was draped around the entire target stand. Not only did this add visual appeal because it was painted sky blue with clouds and birds, it protected the electronics and stopped stray arrows. The target stand is shown below in figure 12, and the entire final product is shown in figure 13.

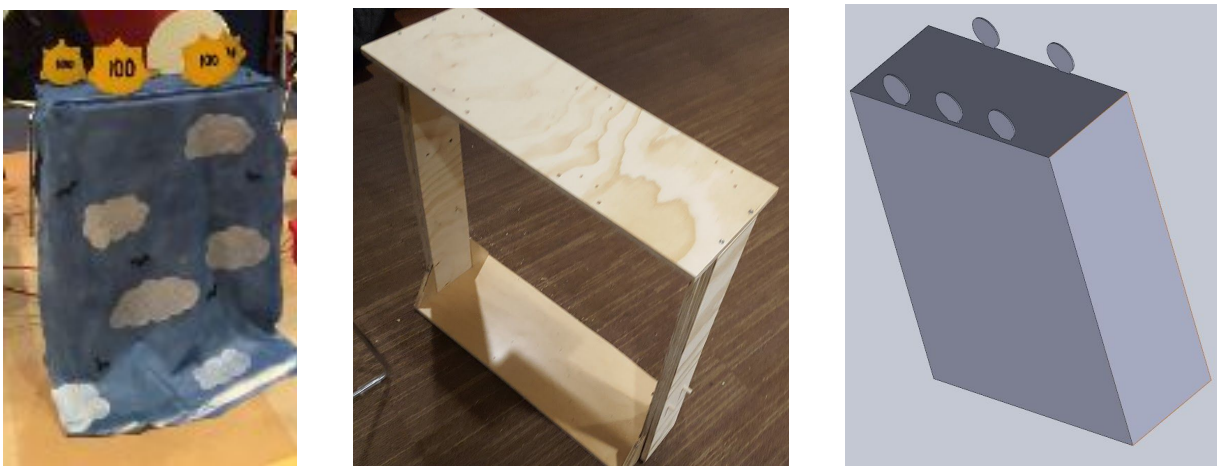


Figure 12: Final Target Stand (left) - Inside of Target Stand (middle) - Solidworks Model (right)

Figure 13: Final Product of the Entire Game (left) - Solidworks Model (right)

We also 3-D printed two parts: a bow and arrow silhouette for the stamp (Fig. 14) and the back of an arrow (Fig. 15). The back of the arrow part had a slit for easier placement on the bow string. The bow and arrow silhouette was glued on a wooden block to use as a stamp.

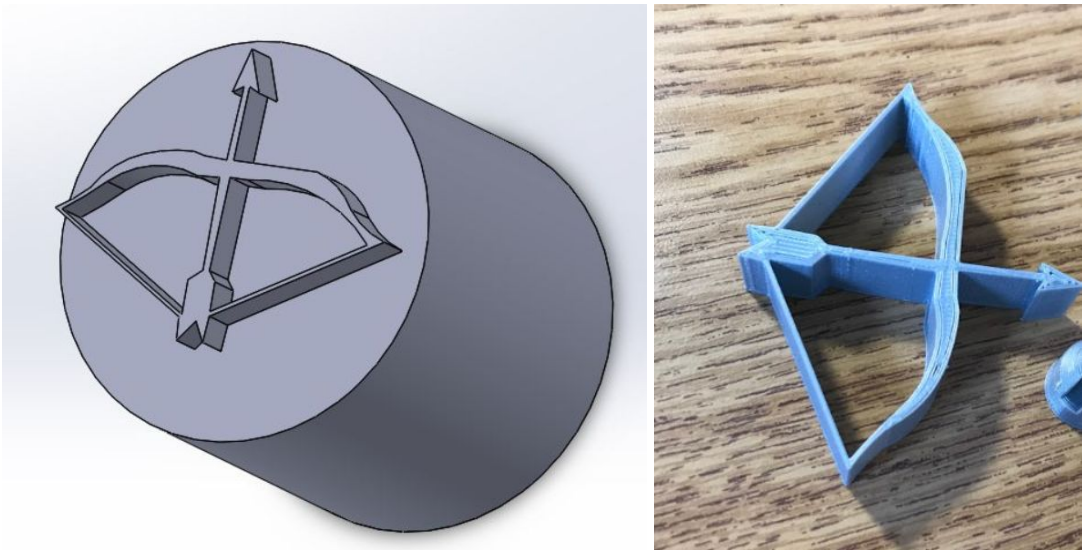


Figure 14: SolidWorks Model of Stamp Design and Actual Part

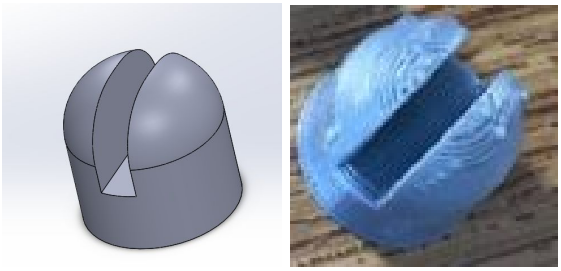


Figure 15: SolidWorks Model of Back of Arrow and Actual Part

FINAL DESIGN: ELECTRONICS

For the final demonstration, it was decided that a 16 by 2 screen would be used, as its size was suitable to attach to the bow stand without hindering its' movement. In addition, we decided to forgo using the RedBoards at our disposal in favor of third party replicas of the Arduino Mega. This circuit board has 54 digital pins and 16 analog, providing more pins to be used for the infrared sensors, radio transmitters, and LCD screen. While not all 54 digital pins are needed, a regular RedBoard's 14 were insufficient. A radio transmitter being used with either a screen or five IR sensor/emitter pairs requires more. The simplicity of only having two circuit boards, although more expensive, was chosen over I2C communication, which establishes wired communication between two RedBoards or any other kind of board to essentially pool their resources together.

NRF24L01 radio transmitters were chosen for the wireless communication desired (Figure 16).

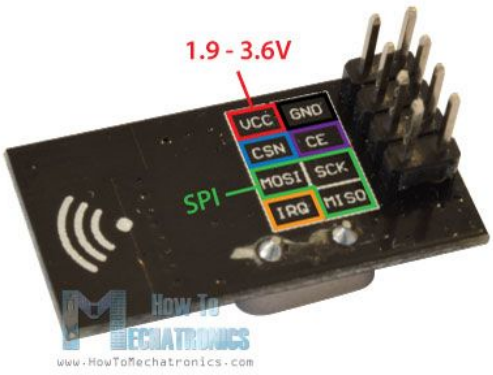


Figure 16:NRF24L01 Transmitters [3]

Furthermore, following the milestone, the core of the coding and wiring stayed the same. The method of wireless communication via radio transmitters carrying the score was not changed since this worked well and was liked by users during the milestone display. However, several people told us that more feedback that a target had been hit would make the game more fun. We added five yellow LED lights to the bow stand, which were coded to light up one by one as targets were hit. A piezo speaker was also added that made a brief sound after a target was successfully knocked over. We experimented with a time limit for players to knock over the targets, but found the process too difficult to code and decided it wouldn't add much to the game.

The most significant change was replacing the IR sensors with photoresistors. After working well at the milestone stage, their readings became erratic and ineffective. We replaced all but one of them with standard Arduino photoresistors (Figure 17).

Figure 17: Photoresistor

Holes were drilled behind the targets where possible to take advantage of stark contrast in light when the targets were knocked down. For the back targets, holes were drilled next to the targets. After much frustration from reaching through wires to press reset on the Mega, a reset button was added to the target stand.

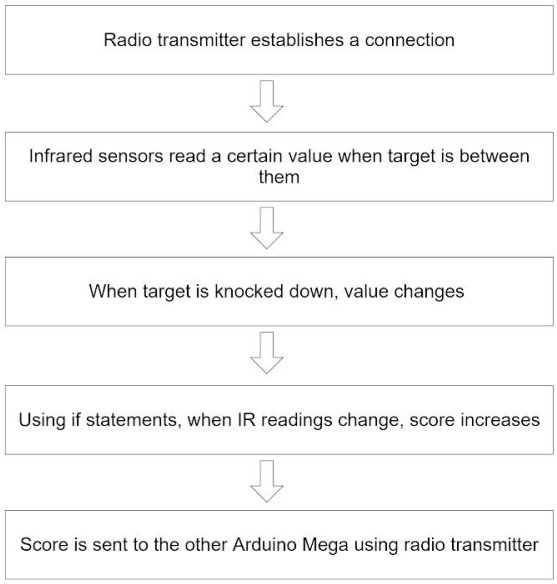
CODING PROCESS

The code mainly relied on the use of the nRF24L01 Library, which is a library that allows the use of functions that send and receive data between two Arduino Megas through the radio transmitters. Accompanying libraries were also used, as they contained functions required for the set up of the communication.

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
```

Because two arduino megas are communicating, two sets of codes were required: the target code and the bow stand code. Even though it was decided that the Mega would be placed on the bow stand instead of the player's arm, this did not impact the code much.

Target Code:



Bow Stand Code:

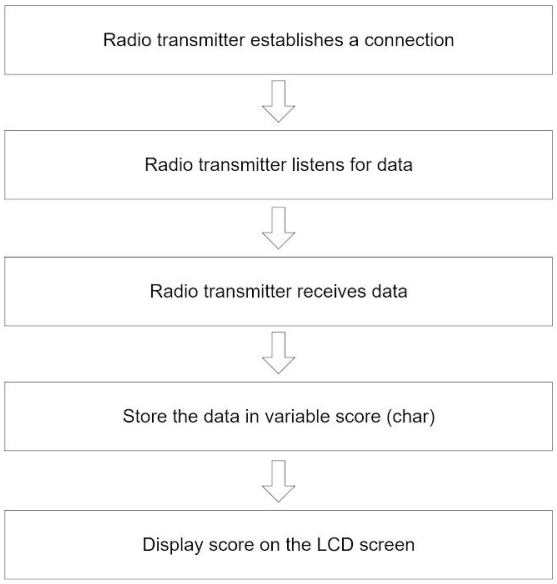


Figure 18: Milestone Code Flow Chart

First of all, the transmitter pins are declared in both the codes.

```
RF24 radio(7, 8); // CE, CSN
```

There is an array to hold the radio address that the two radio transmitters will be communicating through. This same address is later used in the receiver code (arm code) to receive data.

```
const byte address[6] = "00001";
```

Furthermore, the way that the infrared sensors work is that there is a beam of infrared radiation coming from the emitter and is read by the detector. This gives the detector a very low reading. However, if there is something in between the sensors, a target in this case, then the reading becomes very high. The score takes advantage of this through a series of if statements.

In order to ensure that the program stops taking more readings after the target is knocked down, variables a through e were created and set to 0. As long as their values are 0, the program

would continue to check the sensor. However, if the state of that sensor is less than a certain reading, which means that the target has been knocked down, a is incremented by 1, as well as the score. Therefore, that sensor will stop reading because a is no longer 0.

```
IR1state=analogRead(IR1);
    if (a==0)
    {
    if (IR1state < 300) {
      score++;
      a++;
    }
    }
```

Furthermore, after a series of trials, it was decided that the easiest way to send the scores to the receiver would be by sending the score as a character. Therefore, the next part of the program converts the score to an unsigned char named data, which was declared earlier.

```
if (score==1)
{data='1';
}
```

After that, through the use the radio.write function provided by the nRF24L01 Library, the characters are sent to the receiving radio transmitter.

```
radio.write( &data, sizeof(unsigned char));
```

When the emitters and detectors were moved onto the wooden stand that we built, the distances between the two were larger than we expected. This reduced the significance of the difference between the readings when a target is present and when it is not.

We plan to fix this issue by manipulating and calibrating the sensors for the milestone demonstration. In the long term, we plan to make the targets' bases smaller in order for the infrared sensors to be able to be placed closer to each other.

For the arm code, we relied on the use of the same libraries, including an extra one for the LCD screen, which is the LiquidCrystal.h library. We also set up the arduino to receive information from the same radio address of the transmitter using the same code that was put in the transmitter code:

```
const byte address[6] = "00001";
```

For the setup part of this code, we had to use several commands to tell the transmitter to begin listening to the information being sent through a certain address, which was done through the following:

```
radio.begin();
radio.openReadingPipe(0, address);
radio.setPALevel(RF24_PA_MIN);
radio.startListening();
```

In the main loop, we made it so that if information was available through the radio transmitter, that information was placed into a new variable data. Since we knew it would be receiving characters, we made the variable a char variable.

```
if ( radio.available()) {
   // Go and read the data and put it into that
variable
    while (radio.available()) {
              radio.read( &data, sizeof(char));
   }
```

The only other code included in the main loop is a function called screen. This function was created separately, to keep things organized. In this function, there is a series of if statements that checks which character is currently being received, in other words what the score currently is, and outputs the score and a message. After several tests, it was determined that the data being received was in terms of ASCII values, and thus it was decided that the easiest way was to keep them that way.

```
if (data==48)
{ lcd.clear();
    lcd.print ("Score: 0");
    lcd.setCursor(0, 1);
    lcd.print ("Fire!");
}
```

Even though it was initially difficult to send information through the radio transmitters, changing the data type from strings to characters solved the problem.

FINAL CODE

Even though the majority of the code was complete by the milestone demonstration, it was decided that more feedback was needed to make the game more enjoyable. These feedback mechanisms were coded into the bow stand code in order to have the feedback be viewable and audible by the player. In addition, after four of the infrared sensors were replaced by photoresistors, the target stand code also required changes. However, these changes were minimal and only consisted of changing the reading values in the if statements. This is because the photoresistors were wired into the same analog pins as the infrared sensors.

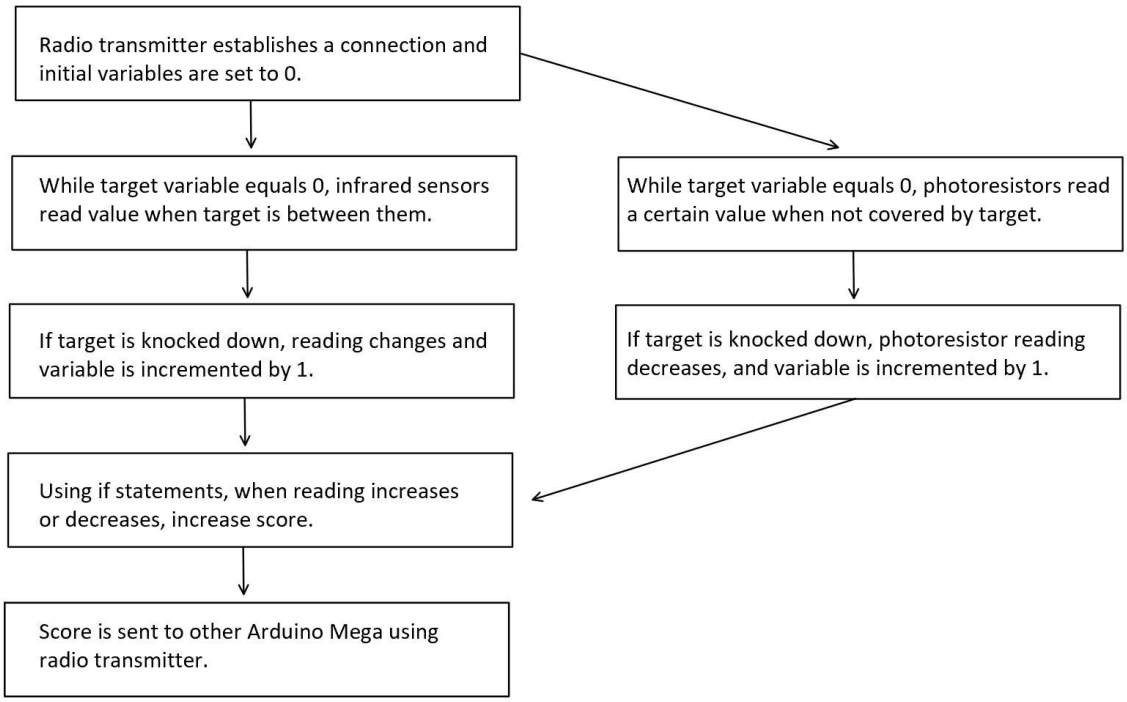


Figure 19: Final Target Code

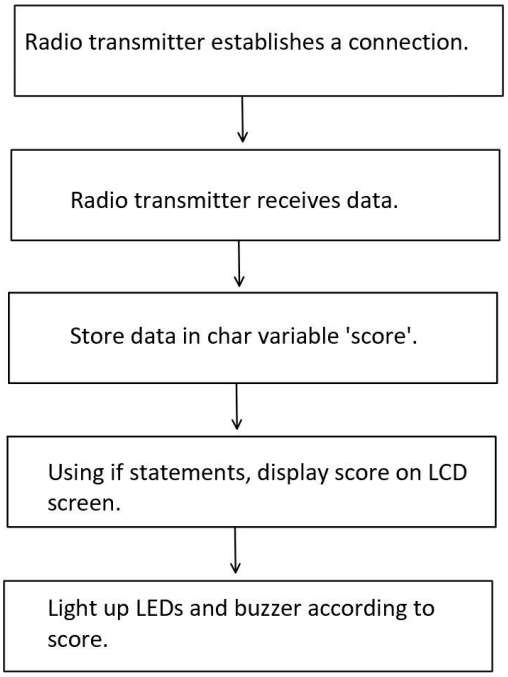


Figure 20: Final Bow Stand Code

First of all, the feedback mechanisms that were added to the game were five lights that correspond to the targets being hit, and a piezo speaker to indicate when a target was hit. The lights were turned on using the `digitalWrite` function within the if statements that corresponded to the values being received by the neurotransmitters. After testing this, it was discovered that lights did not turn off on their own when the game was reset, thus requiring this to be coded in the if statements as well.

```
digitalWrite(led1, HIGH);
digitalWrite(led2, LOW);
digitalWrite(led3, LOW);
digitalWrite(led4, LOW);
digitalWrite(led5, LOW);
```

Furthermore, the speaker was also coded within the if statements. However, the speaker would sometimes sound when a target was knocked down and not stop for the duration of the game. This was solved by creating variables from b to f and setting them equal to zero, and placing a while loop with each of them within the if statements. Therefore, when the if statement runs, the buzzer sounds and immediately stops when the value is incremented by 1.

```
while (b == 0)
{

  tone(buzzer, 500, 1500);
  delay(100);
  b = 1;
}
```

ANALYSIS

All the materials cost under $80, meeting our financial constraint. The game was played a total of 68 times, with 10 children playing twice. Data was recorded for 48 of these playthroughs and input into MATLAB for analysis. The raw data can be found in Appendix E. Our average user was 6.9 years old, hit 1.1 targets, and spent 71.18 seconds playing our game. Various trends we observed can be found in graphs obtained from the data (Fig. 21, Fig. 22).

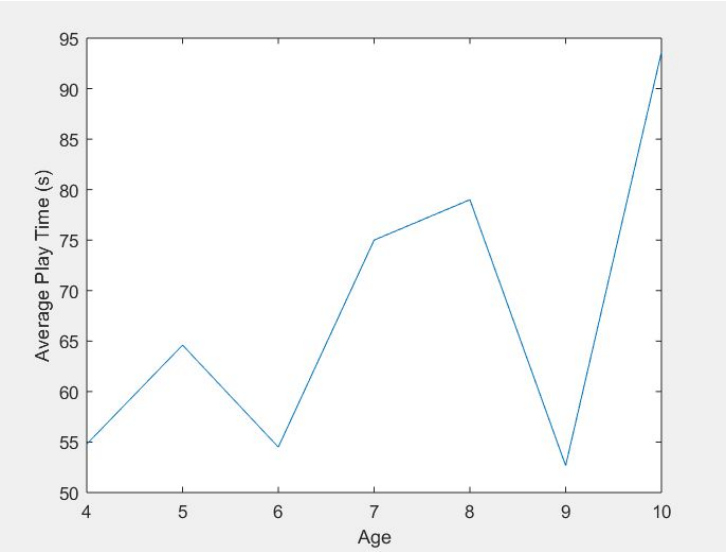


Figure 21: Playing Time vs. Age

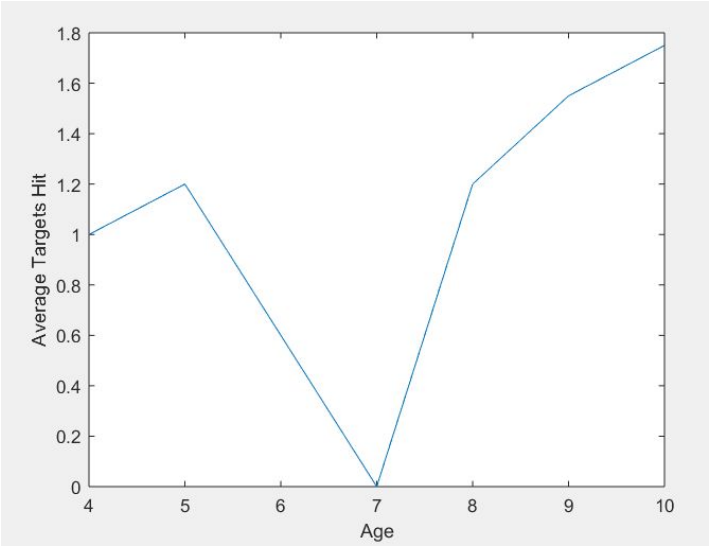


Figure 22: Average Targets Hit vs. Age

One would might expect that younger children would always take longer to play the game. However, we found that at the museum younger children were generally quicker to fire off their arrows, regardless of aim or force. Older children spent more time moving the stand around and aiming, which generally increased their time played. None of the players whose data was recorded went over two minutes. This satisfied our goal of short playing time to avoid forcing many children to wait.

The quintessential determinant of success for this project, beyond how well hardware or coding worked, was whether or not the game was fun for the children, which in turn has several components. All children who played were able to fire arrows with minimal assistance after being shown once how to use the bow. They found the shooting component alone fun; the arrows flew very quickly after being released. They also were elated after knocking down targets. From

the simple benchmark of how much users enjoyed the game, the project can be considered a success. There are many more components of the project worthy of analysis that had varying amounts of success.

Hitting the targets was not as easy as we had hoped. Although, on average, a child was able to hit at least one target, hitting one target isn't as satisfying as knocking down multiple ones each play through. A few factors contributed to this. The most important was the size of the targets. They were only approximately ½ square feet in area each, so the game required some skill to hit them. Sometimes the targets didn't fall down even when hit by the arrow; we still counted these as hits. In addition, the target stand was a bit higher than the level at which the kids were firing the bow from, which added more difficulty. Making larger targets and lowering the height of the target stand would have likely increased the amount of targets hit, making the game more fun.

While the physical aspect of the game functioned close to expectations, feedback via electronics left much room for improvement. We found that most children didn't notice the changing score on the screen, partially due to the delay from the radio transmitter. In addition, they didn't take much notice of the LEDs we added below the screen or the sounds from the piezo speaker. If these were augmented so that children always recognized feedback, then their experience would be much improved. Lowering the screen to a height closer to eye level for children would obviously make it more noticable. The screen was on the other side of the stand as the bow, which didn't make it less noticable for the college-age people we showed it to during the design process, but made it harder to see for the actual group we were designing for. Switching it to the same side was the bow would make it much more prevalent. As for the LEDs, adding "super bright" LEDs(Fig. 23) would have made the game more flashy and appealing.

Figure 23: Super Bright LED[4]

We would also replace the piezo speaker in the future with a bluetooth speaker for louder sounds, as the museum was simply too busy for a piezo speaker to make a difference.

With a projectile firing game, safety was imperative. No one was hurt or in danger the whole duration of the exhibit. We set up our project against a wall so that no one would walk

behind the targets. What we didn't anticipate was people walking in front of the targets. Many parents and children would attempt to take a shortcut right through the space between the bow and target stand. We mitigated this danger by having one group member by the bow stand to stop a child from firing while people were close. We also put one group member on each side of the "firing range" to stop people from going through in the first place. Caution tape, or some other physical barrier, would have made this task much easier.

CONCLUSION

We were assigned to create a fun game for an exhibit at the Boston Children's Museum on April 7th, 2018 for under $80 using Arduino components. Our game was chosen to be based off Chinese mythology. We chose a bow and arrow target shooting game because the age group is very excited by flashy physical components and bored by complicated gameplay. This was then adapted to the story of Hou Yi, an archer who shot down extra suns in his legend.

The electronic basis of our game was accomplished using IR sensors and emitters, a LCD screen, and radio transmitters. If the arrow knocks over a target on the stand, an IR sensor or photoresistor will send a message via a radio transmitter to a LCD screen in a box near the user, which will display the score. We decided that implementing electronic components to provide visual and audible feedback of the user's actions and scores would enhance the experience. This was accomplished using LEDs that light up according to number of targets hit, as well as a speaker that sounds each time the user scores.

Our final game allows children seven arrows to score as many points possible, out of five targets worth 100 points each. The stand for the targets and the bow and arrow were built out of wood, screws, and hot glue. The bow and arrow uses layered surgical tubing as bowstring, wooden arrows with foam tips for safety, and arrow-backs with small slits that fit the bowstring to allow the user to pull back and guide the arrow. The bow is connected to the stands by connecting any two of the multiple dowels on the stand through the holes on the bow. This serves as a method of adjusting the height for different users. The target stand is adjustable accordingly. Wheels beneath the bottom of the bow stand were also attached to increase the user's ability to aim. Our targets designed to look like suns and are connected to the stand via hinges. A decorated bed sheet drapes over the target stand to increase visual appeal. All physical components of the game were also built to be as light and portable as possible.

Although it has simple game mechanics, it serves to fit the desires and abilities of our target users. The children ages 4-10 at the museum were completely satisfied with our game. Our milestone prototype was a functional prototype working at its most basic level, but many improvements were made for our final product. Throughout our design process, we have completed our project as to fully satisfy both the users' and client's needs. Future work would include making the targets easier to hit and more lights and sounds for when a target is knocked down.

APPENDIX A: WORKS CITED

"Arduino Wireless Communication - NRF24L01 Tutorial." *HowToMechatronics*, 28 Feb. 2018, howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/.

"Arduino Wireless Communication - NRF24L01 Tutorial." *HowToMechatronics*, 28 Feb. 2018, howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/.

"Breadboard - Self-Adhesive (White)." *Learn at SparkFun Electronics*, learn.sparkfun.com/tutorials/sik-experiment-guide-for-arduino---v32/experiment-15-using-an-lcd.

'Super Bright LED - White 10mm." *COM-11118 - SparkFun Electronics*, www.sparkfun.com/products/11118.

APPENDIX B: CODE

B1 - Target Code

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
RF24 radio(7, 8); // CE, CSN
const byte address[6] = "00001";
//pins for IR sensors
const int IR1 = A1;
const int IR2 = A2;
const int IR3 = A3;
const int IR4 = A4;
const int IR5 = A5;

int score = 0;

//booleans of reading to send to receiver
int IR1state = 0;
int IR2state = 0;
int IR3state = 0;
int IR4state = 0;
int IR5state = 0;

int testIR1state = 0;
int testIR2state = 0;
int testIR3state = 0;
int testIR4state = 0;
int testIR5state = 0;

int a = 0;
int b = 0;
int c = 0;
int d = 0;
int e = 0;

unsigned char data;

void setup() {
  radio.begin();
  radio.openWritingPipe(address);
  radio.setPALevel(RF24_PA_MIN);
  radio.stopListening();
  pinMode(IR1, INPUT);
  pinMode(IR2, INPUT);
  pinMode(IR3, INPUT);
  pinMode(IR4, INPUT);
  pinMode(IR5, INPUT);
```

```
  Serial.begin(9600);
}
void loop() {

  //Testing readings
  //IR1state=analogRead(IR1);
  // Serial.println("IR1: ");
  //Serial.println(IR1state);


  // IR2state=analogRead(IR2);
  //Serial.println("IR2: ");
  //Serial.println(IR2state);


  //IR3state=analogRead(IR3);
  //Serial.println("IR3: ");
  //Serial.println(IR3state);

  //  IR4state=analogRead(IR4);
  //   Serial.println("IR4: ");
  //  Serial.println(IR4state);

  // IR5state=analogRead(IR5);
  // Serial.println("IR5: ");
  // Serial.println(IR5state);

  delay(2000);

  data = '0';
  if (a == 0)
  {
    if (IR1state < 400) {
      score++;
      a++;
    }
  }

  if (b == 0)
  {
    IR2state = analogRead(IR2);
    if (IR2state < 200) {
      score++;
      b++;
    }
  }

  if (c == 0)
```

```
  { IR3state = analogRead(IR3);
    if (IR3state < 700) {
      score++;
      c++;
    }
  }
  if (d == 0)
  { IR4state = analogRead(IR4);
    if (IR4state > 600) {
      score++;
      d++;
    }
  }
  if (e == 0)
  {
    IR5state = analogRead(IR5);
    if (IR5state < 400) {
      score++;
      e++;
    }
  }

  if (score == 0)
  { data = '0';
  }
  if (score == 1)
  { data = '1';
  }
  if (score == 2)
  { data = '2';
  }
  if (score == 3)
  { data = '3';
  }
  if (score == 4)
  { data = '4';
  }
  if (score == 5)
  { data = '5';
  }

  delay(1000);
  radio.write( &data, sizeof(unsigned char));
}
```

B2 - Arm Code:

```
#include <LiquidCrystal.h>
#include <SPI.h>
```

```
#include <nRF24L01.h>
#include <RF24.h>
RF24 radio(7, 8); // CE, CSN
const byte address[6] = "00001";

//integer for the score
int score = 0;
unsigned char data;
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

const int buzzer = 22;

int led1 = 27;
int led2 = 29;
int led3 = 31;
int led4 = 33;
int led5 = 35;


int b = 0;
int c = 0;
int d = 0;
int e = 0;
int f = 0;

//function
void screen();

void setup() {

  pinMode(buzzer, OUTPUT);
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);
  pinMode(led4, OUTPUT);
  pinMode(led5, OUTPUT);

  lcd.begin(16, 2);
  lcd.clear();
  Serial.begin(9600);
  radio.begin();
  radio.openReadingPipe(0, address);
  radio.setPALevel(RF24_PA_MIN);
  radio.startListening();
}

void loop() {
  if ( radio.available()) {
```

```
    // Go and read the data and put it into that variable
    while (radio.available()) {
      radio.read( &data, sizeof(char));
      Serial.print(data);
      delay(1000);
      screen();
    }
  }
}

void screen()
{
  if (data == 48)
  {
    digitalWrite(led1, LOW);
    digitalWrite(led2, LOW);
    digitalWrite(led3, LOW);
    digitalWrite(led4, LOW);
    digitalWrite(led5, LOW);

    lcd.clear();
    lcd.print ("Score: 0");
    lcd.setCursor(0, 1);
    lcd.print ("Game has started!");
  }

  if (data == 49)
  { digitalWrite(led1, HIGH);
    digitalWrite(led2, LOW);
    digitalWrite(led3, LOW);
    digitalWrite(led4, LOW);
    digitalWrite(led5, LOW);
    lcd.clear();
    lcd.print ("Score: 100");
    lcd.setCursor(0, 1);
    lcd.print ("Nice work!");

    while (b == 0)
    {
      tone(buzzer, 500, 1500);
      delay(100);
      b = 1;
    }
  }

  if (data == 50)
  { digitalWrite(led1, HIGH);
    digitalWrite(led2, HIGH);
```

```
  digitalWrite(led3, LOW);
  digitalWrite(led4, LOW);
  digitalWrite(led5, LOW);

  lcd.clear();
  lcd.print ("Score: 200");
  lcd.setCursor(0, 1);
  lcd.print ("Keep going!");

  while (c == 0)
  {
    tone(buzzer, 500, 1500);
    delay(100);
    c = 1;
  }
}

if (data == 51)
{
  digitalWrite(led1, HIGH);
  digitalWrite(led2, HIGH);
  digitalWrite(led3, HIGH);
  digitalWrite(led4, LOW);
  digitalWrite(led5, LOW);

  lcd.clear();
  lcd.print ("Score: 300");
  lcd.setCursor(0, 1);
  lcd.print ("Wow!");

  while (d == 0)
  {
    tone(buzzer, 500, 1500);
    delay(100);
    d = 1;
  }
}

if (data == 52)
{
  digitalWrite(led1, HIGH);
  digitalWrite(led2, HIGH);
  digitalWrite(led3, HIGH);
  digitalWrite(led4, HIGH);
  digitalWrite(led5, LOW);
  lcd.clear();
  lcd.print ("Score: 400");
  lcd.setCursor(0, 1);
```

```
    lcd.print ("On fire!");

    while (e == 0)
    {
      tone(buzzer, 500, 1500);
      delay(100);
      e = 1;
    }
  }

  if (data == 53)
  { digitalWrite(led1, HIGH);
    digitalWrite(led2, HIGH);
    digitalWrite(led3, HIGH);
    digitalWrite(led4, HIGH);
    digitalWrite(led5, HIGH);

    lcd.clear();
    lcd.print ("Score: 500");
    lcd.setCursor(0, 1);
    lcd.print ("YOU GET A STAMP!");

    while (f == 0)
    {
      tone(buzzer, 500, 1500);
      delay(100);
      f = 1;
    }
  }
}
```

B3 - Transmitter Test

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
RF24 radio(8, 10); // CE, CSN
const byte address[6] = "00001";
const int button=7;
boolean buttonstate=0;
void setup() {
  radio.begin();
  radio.openWritingPipe(address);
  radio.setPALevel(RF24_PA_MIN);
  radio.stopListening();
  pinMode(button, INPUT_PULLUP);
  Serial.begin(9600);
}
```

```
void loop() {

  buttonstate=digitalRead(button);
  radio.write(&buttonstate, sizeof(buttonstate));
  delay(1000);
}
```

B4 - Receiver Test

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
RF24 radio(8, 10); // CE, CSN
const byte address[6] = "00001";
boolean buttonstate=0;
const int led=7;
void setup() {
  Serial.begin(9600);
  radio.begin();
  radio.openReadingPipe(0, address);
  radio.setPALevel(RF24_PA_MIN);
  radio.startListening();
}
void loop() {


  if (radio.available()) {

    radio.read(&buttonstate, sizeof(buttonstate));
  if (buttonstate == LOW) {
    digitalWrite(led, HIGH);
    Serial.print("gogirl");
  }
  else {
    digitalWrite(led, LOW);
  }
  }
}
```

B5: MATLAB

```
load('targets.txt')
load('age.txt')
load('timetoplay.txt')
avgage=mean(age)
avgtargets=mean(targets)
avgtime=mean(timetoplay)
```

```
ages=[4:10];
avgtime=[54.8 64.6 54.5 75 79 52.67 93.6];
avgtargets=[1 1.2 .6 0 1.2 1.55 1.75];


figure(1)
plot(ages, avgtime)
xlabel("Age")
ylabel("Average Play Time (s)")
figure(2)
plot(ages, avgtargets)
xlabel("Age")
ylabel("Average Targets Hit")
```

APPENDIX C: TIMESHEET

	Ciaran	David	Selena	Fatema	Purpose
1/21	3-3:45	3-3:45	3-3:45		Order parts, finalize details
1/27	10-11:15			10:30-11:15	project proposal
1/29	7-7:30		7-7:30	7-7:30	project proposal
1/29		8-11:30			cardboard prototype
2/2	3-4:45			3-4:45	soldering and coding
2/4		12-1	12-1		buying wood
2/5	7:40-7:45	7:40-7:45	7:40-7:45		updates
2/10	12:00 - 2:30			12:00 - 2:30	wiring
2/17	12:00 - 2:30			12:00 - 2:30	coding
2/19	4:45-5	4:30-6	4:30-6		building
2/23	2-5	2-5	2-5	2-5	building
2/25	5-5:30	5-5:30	5-5:30	5-5:30	working on infographic/report
3/2	3-4	3-4		3-4	report
3/16	2-4	2-4		2-4	buying materials
3/23	3-5	3-6	3-6	3-6	assembling wood and electronics
3/30	3-5	3-6	3-6	3-6	assembling wood and electronics/ making arrows
4/5			2:40-5	2:40-6	final touches
4/5	6-9		6-9	6-9	final touches/decorating
Totals	23:05	19:20	18:40	26:50	

APPENDIX D: EXPENSE SHEET

Item	Quantity	Price	Purchased from	Date of purchase
IR sensor	5	$15.99	SparkFun.com	1/21
Radio transmitter	2	$11	AliExpress.com	1/21
Patent application	1	$3	Professor Hertz	1/23
Arduino Megas	2	$30	Amazon.com	1/28
Screws	1 Pack	$4	Economy True Value	2/20
Dowels	2	$3	Economy True Value	3/18
Spray Paint	3	$12	Economy True Value	3/23
Total:				78.99

APPENDIX E: RAW DATA FROM MUSEUM

Age	Targets Hit	Time Played
8	1	58
9	2	68
5	1	73
4	1	69
5	2	54
6	0	63
10	0	68
8	2	64
8	2	56
8	1	80
5	1	85
10	2	79
8	0	95
6	1	82
9	1	83
4	2	75
8	3	93
5	2	85
8	2	87
5	2	47
6	0	45

10	0	51
8	0	92
8	1	96
7	0	81
9	1	98
10	1	91
10	0	64
9	2	65
5	1	61
7	0	69
10	2	93
9	2	88
9	2	75
4	1	94
5	0	68
5	0	65
6	2	68
8	1	57
5	2	45
4	0	80
6	5	69
4	1	49
4	0	45

5	0	63
9	0	62
10	1	62
5	2	57