Shonna Slay, Shonna Conquer

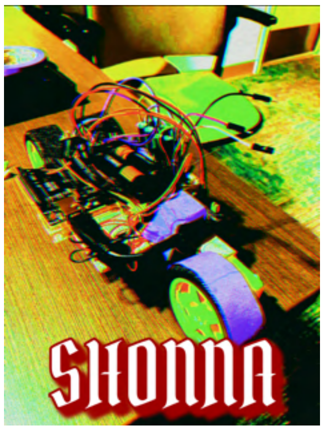
Group 11:

Le Duong

Lilian Le

Naya Wiliams

Regan Van Stavern



Organization	/20
Documentation	/30
Testing	/25
Lessons Learned	/25
TOTAL	**/100**

WOMEN IN
STEM

Table of Contents

5… Objective

5… Introduction

6… Preliminary Car Design (Donna)

- 7… Modifying Dr. Paul D Ronney's Suggested Schematic
- 7… Image of Parts
- 8… Selection of Microcontroller
- 8… Description
- 8… Coding Arduino for 2 Motors
- 9… Forming the Circuit and Wiring
- 10… Powering
- 11… Building Car Frame (No Assembly)
- 11… Testing Results
- 12… Critiques of Preliminary Car Design

12… Shonna 1.0

- 12… Modifying Preliminary Design (Donna)
- 13… Description
- 13… Sketch of Shonna 1.0
- 14… New Arduino and Code
- 15… Removal of Front Motor/Free Front Wheel
- 15… Testing Results
- 16… Critiques of Shonna 1.0

16… Shonna 2.0

- 16… Modifying Shonna 1.0
- 17… Description
- 17… Sketch of Shonna 2.0
- 18… Replacing Front Free Wheel with Second Motor
- 18… Solidifying Connections by Soldering
- 18… Changes to Car Frame
- 18… Testing Results
- 19… Critiques of Shonna 2.0

19… Shonna 3.0

- 19… Modifying Shonna 2.0
- 19… Description
- 20… Sketch of Shonna 3.0
- 20… Grip Tape for Traction
- 21… Addition of Supercapacitors
- 22… Testing Results
- 22… Critiques of Shonna 3.0

23… Shonna 4.0 (Final Design)

- 24… Modifying Shonna 3.0
- 24… Testing Results
- 24… Description
- 25… Schematic for Shonna 4.0
- 26… Sketch of Shonna 4.0
- 27… Adding the Tilt Sensor
- 27… Coding for Tilt Sensor
- 30… Addition of Switch
- 31… Addition of New Battery Pack
- 32… Official Testing
- 32… Reasoning for Final Design

33… Competition Results

35… Final Modifications

- 35… Re-soldering connection to boost converter
- 36… Removal of Switch and Capacitors

36… Conclusions

- 36… Reason for Modifications/Critiques of Shonna 4.0
- 36… What Would Have Been Done Differently
- 38… What We Learned

40… Meeting Agenda

45… References

Objectives

1. Construct an autonomous vehicle that is capable of ascending the ramp, stopping in a position in which the rear wheel is closest to the center of the top platform of the ramp, and able to remain in this position or closer to this position than the opposing vehicle within a 30 second time limit.
2. Utilize a microcontroller, tilt sensor, motors, and relay board to control and operate the vehicle.
3. Build the vehicle so that it weighs less than 1kg and its components are powered by 2 AA batteries. Since the Arduino cannot operate on the 2.95 volts supplied by the 2 AA batteries since it is under 3 volts, a boost converter must be used to provide the arduino with a constant 5 volts.

Introduction

This project has been one long chain of a continuous evolution of different conceptual designs we decided to test. In order to make this report easier to read, we have separated each design into sections by using the car's name. In our first stage, we called the "car" Donna and gave her she/her pronouns. After our first attempt at building the car, we had issues with the Arduino and needed to buy a new one. We saw this as a restart to creating the car, so when we got the new Arduino, we gave the car the name Shonna. Each stage is divided by 1.0, 2.0, and so on until our last and final stage, 4.0. Donna is technically our first stage, but we never got past connecting the motors to the Arduino and battery pack, as the Arduino broke before we could go on. Shonna 4.0 is our final design and what we used in the competition itself.

Preliminary Car Design (Donna)

Modifying Dr. Paul D. Ronney's Schematic

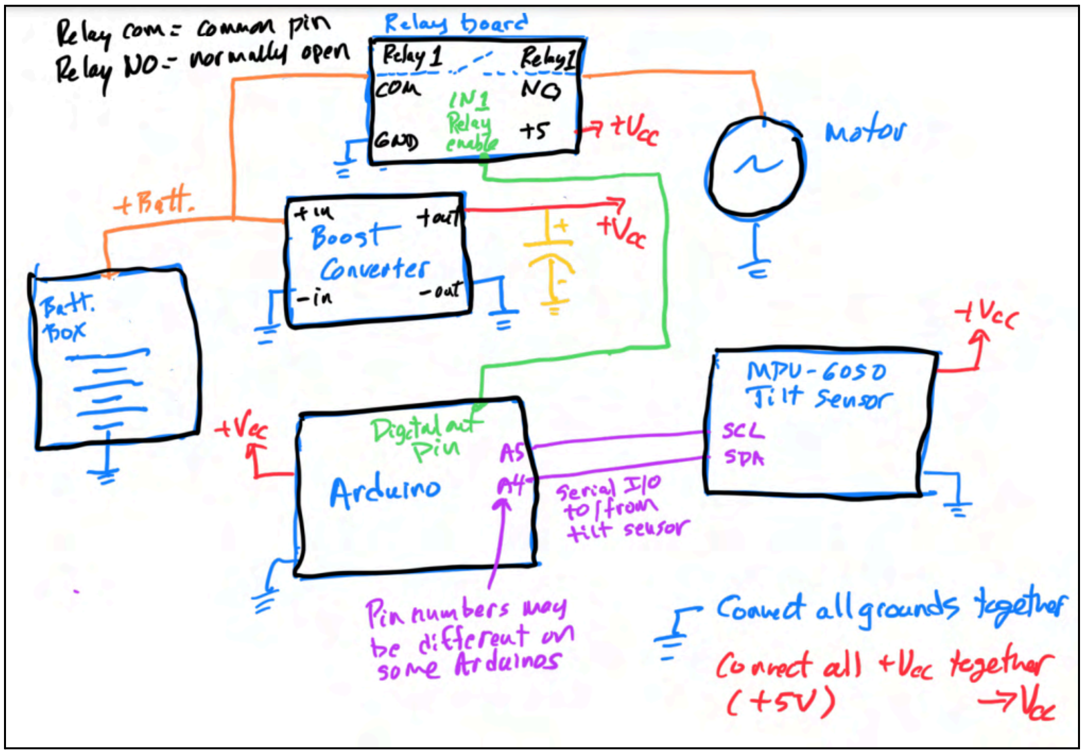


Figure 1a. Dr. Ronney's Schematic for Arduino, Relay Board, Motors, and Tilt Sensor (Source: Dr. Paul D. Ronney)

The only modification we made to Dr. Ronney's suggested schematic was removing the tilt sensor since it was too early in our design testing process to consider it (it would make the prototype more complicated). Our goal for this design was to create a simple microcontroller model to test if the 2 AA batteries would have sufficient current to power the entire car (Arduino, relay board, and motors). Before building the prototype, we tested (in a simple closed circuit) to make sure that the 2 AA batteries had enough current to power the two motors, and they did.

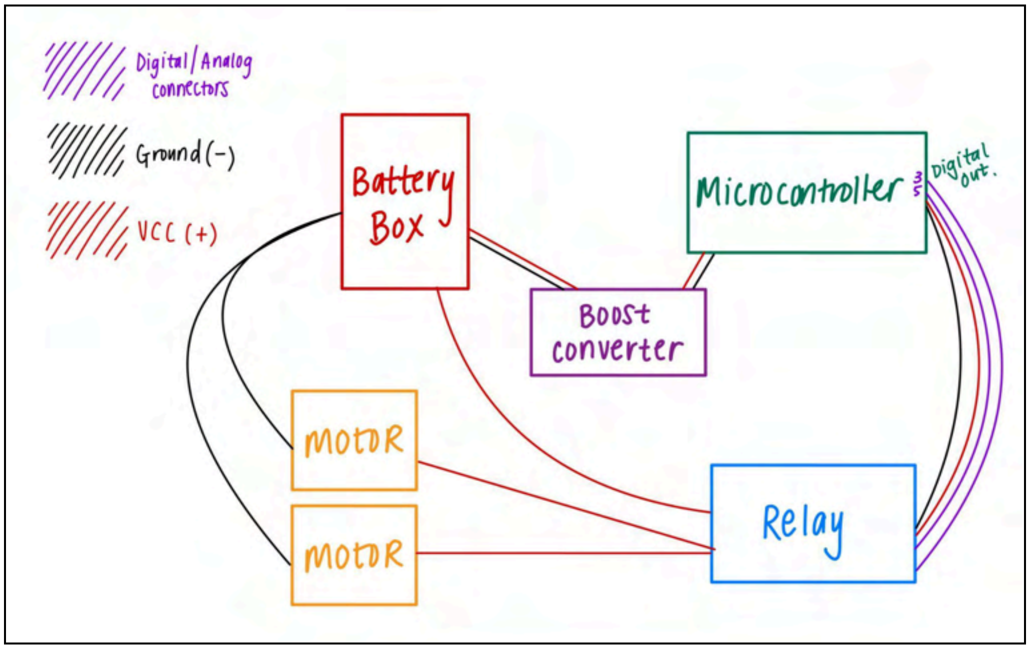


Figure 1b. Dr. Ronney's Modified Schematic.

Image of Parts

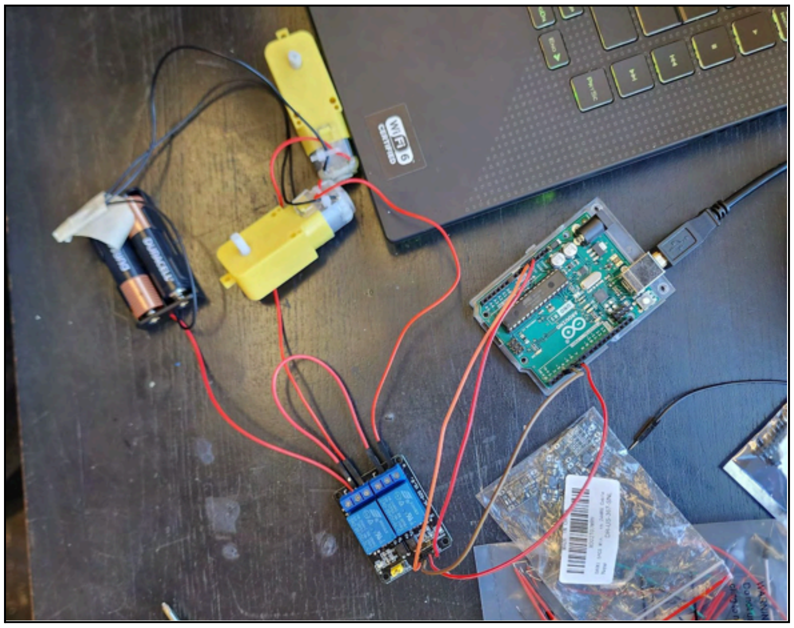


Figure 1c. Donna (w/ Arduino Powered by Laptop): A circuit system between the batteries, motors, Arduino, and relay board. Based on modified schematic.

Selection of Microcontroller

We decided to use the Arduino Uno R3 microcontroller board, and we decided to name it Donna, as it is one of the most popular board choices, and thus there would be many resources on coding, wiring, as well as troubleshooting. However, during our construction process, our microcontroller (both code and bootloader) was corrupted, so we decided to buy a cheaper microcontroller that could handle the same job and also had a working reset button (in case the code became corrupted).

Description

The first "design" of our car included only the Arduino, two double AA batteries, battery holder, a relay, a boost converter, and 2 motors. These components never made it onto the car frame.

Coding Arduino for 2 Motors

```
// Set pin number that controls motor
int motorPin = 3;
int motorPin2 = 5;

void setup () {
// Setup pin so that it will be outputting

pinMode ( motorPin , OUTPUT );
}

void loop () {

digitalWrite ( motorPin ,1) ; // Turn on motor
digitalWrite ( motorPin2 ,1) ;
delay (3000) ; // Wait 3s
digitalWrite ( motorPin ,0) ; // Turn off motor
digitalWrite ( motorPin ,0) ;
delay (3000) ; // Wait 3s

// Go back to beginning of loop ()
}
```

Figure 1d. Code for relay board and motor control (Source: Steven Luna - "Arduino Notes")

The code shown above is our test code to ensure the motors work. The code was modified from Steven Luna's in his "Arduino Notes." We added another digital output for our second motor and named it "motorPin2." In order to test the code, we decided to power the Arduino through the laptop because we knew that the 2 AA batteries had to be able to power the motors alone (based on the closed circuit test. Therefore, that would mean that if the motors did not run, it would most likely mean the error lies within the code and not the other components. Fortunately, the code worked smoothly, and the Arduino could switch on and off the relay board.

Forming the Circuit and Wiring

First, we need to justify why we use certain components, which includes the Arduino board, relay board, boost converter, motors, and battery pack holder. Some straightforward components that we use are motors and Arduino board, which will be what drives the car uphill and programs her to do so, respectively. Next, we used the relay board to switch open and close the circuit between the battery and each motor because the Arduino board cannot output a large enough current/ voltage to power either motor. The boost converter was then used to boost the voltage to a constant 5V from the battery to the Arduino board since it (the board) only operates on +3V, which around the same voltage as the 2 batteries combine. This would mean that without the boost converter, as soon as the battery has been lightly used and dropped to below 3V, the Arduino board would stop operating (Paul D. Ronney). Lastly, we chose to use the battery pack holder as it is an efficient way to hold the batteries together and makes connecting the VCC and GND together a lot easier.

The overall circuit and wiring system of Donna follows the schematic mentioned above. However, the schematic above is a simplified version of the actual circuit. Therefore, we needed further research on how and where the wires should connect the different components. In comparison, the wiring between the batteries, boost converter, and Arduino is straightforward (VCCin and GNDin to the inputs and VCCout and GNDout to the outputs). However, the relay board, battery pack, Arduino board, and motors are more complicated. We learned how to wire these four components during our consultation with Dr. Ronney.

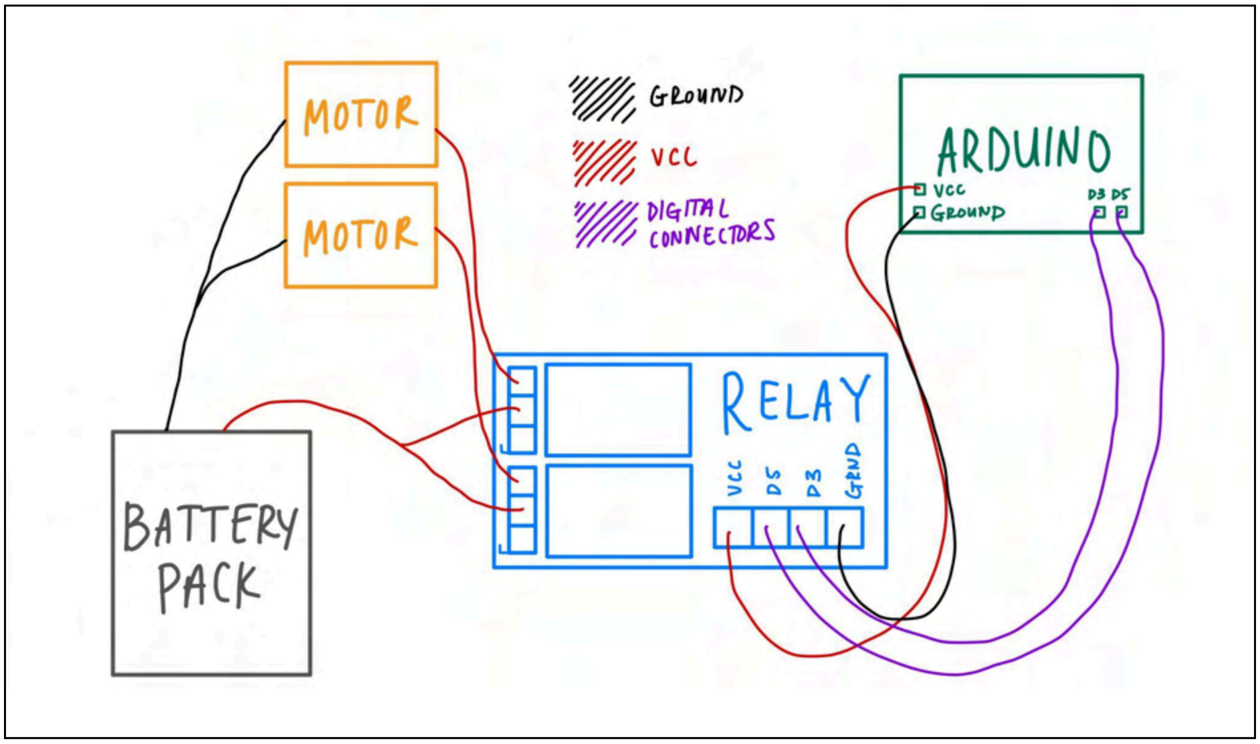


Figure 1e. Sketch of Detailed Schematic of Relay board, Motors, and Battery Pack

In **figure 1e** above, it can be seen that the current that is powering the motor is separate from the current that powers the Arduino board. This is an important concept for us later on when we decided to use the capacitor to solely power the front motor.

Powering

Donna's power system evolved from using the laptop to power the Arduino (refer to figure 1c) to the circuit system seen in the modified schematic above. Unfortunately, we did not take a picture of Donna's actual system layout, which solely depends on the batteries' powers (to power both motors and Arduino).

Building the Car Frame (No Assembly)

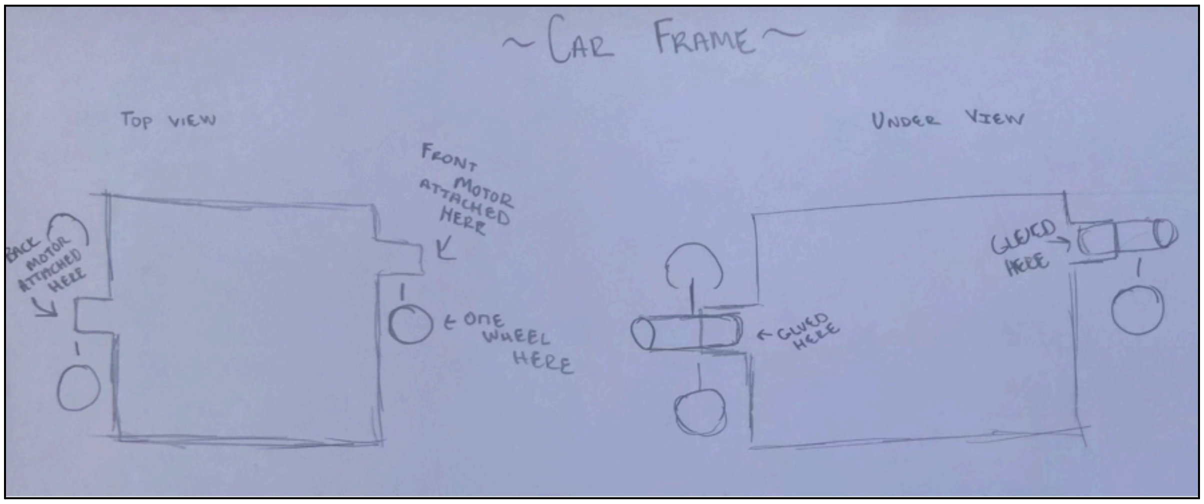


Figure 1f. This is a sketch of the frame we built out of balsa wood for the parts to sit on. In this figure, it is shown how the motors are placed from a perspective above the car and below the car. We used hot glue to attach the motors to the wood.

Since we didn't have a kit, we built our frame for the car out of balsa wood. We did a basic sketch of a square with a small piece of wood sticking out on both the front and back sides of the frame. The piece on the back side was cut out in the middle for the motor with two wheels to be centered on the sides. The piece on the front side was cut out off-centered towards the left side because we only used one wheel for the front motor.

Testing Results

Although this was not an actual design, we got the two motors to run while the Arduino was plugged into the laptop. However, we did not get them to run with the batteries alone, so we concluded that the batteries did not provide enough power for both motors and the Arduino. In hindsight, this could have been due to the poor wire connections since nothing was soldered. The finding that led to the end of Donna was that her Arduino was no longer uploading code. We concluded that the constant switching between 1 and 2 motors for testing corrupted the code and the bootloader on the Arduino.

Critiques of Preliminary Car Design

For our preliminary design, several issues arose that prevented the car from functioning correctly. Our first issue with the preliminary design was an insufficient current to power everything in the circuit. When we assembled the circuit and formed the wiring according to our modified schematic, we noticed that the two motors were inconsistent on 2 AA batteries.

A significant issue with Donna was that her Arduino stopped working. After several tests, we believed our code became corrupted after the Arduino stopped responding. Due to too much inconsistency with the input current that enters the battery (since we kept changing the circuitry to power from 1 to 2 motors), we ruined our Arduino and the code saved on it. We then tried resetting the Arduino board by pressing the reset button, but it did nothing. We suspected the reset button was malfunctioning, and we attempted to upload a new code to the Arduino from our computer, but that also did not work. Our new code was not uploading correctly, as it took longer than usual, and a red error message was displayed. Further research led us to believe that we may have corrupted the bootloader of the Arduino board, which would require us to use another Arduino to burn a new bootloader (Arduino.com).

Overall, there were many issues with Donna, but the most prevalent were the lack of current flowing through the circuit and the corrupted code and bootloader. In order to resolve these issues, we planned to solidify the wire connections by soldering and buying a new Arduino board.

Shonna 1.0

Modifying Preliminary Design (Donna)

Since we needed to buy a whole new Arduino because Donna's code and boot loader was corrupted, we named our second design, Shonna 1.0. Shonna 1.0 differed from Donna because we uploaded a new code for 1 motor to the new Arduino board and replaced the front motor with a free wheel.

Description

The first real design of our car included the new microcontroller, two double AA batteries, a battery holder, a relay, a boost converter, one motor, and three wheels. For this test, all of the components of the car were on it but secured by tape in order for us to determine whether or not the motors were capable of powering the total weight of the components of the car without putting them on permanently. The third wheel was a free wheel, attached to the front of the car using cardboard and tape..

Sketch of Shonna 1.0

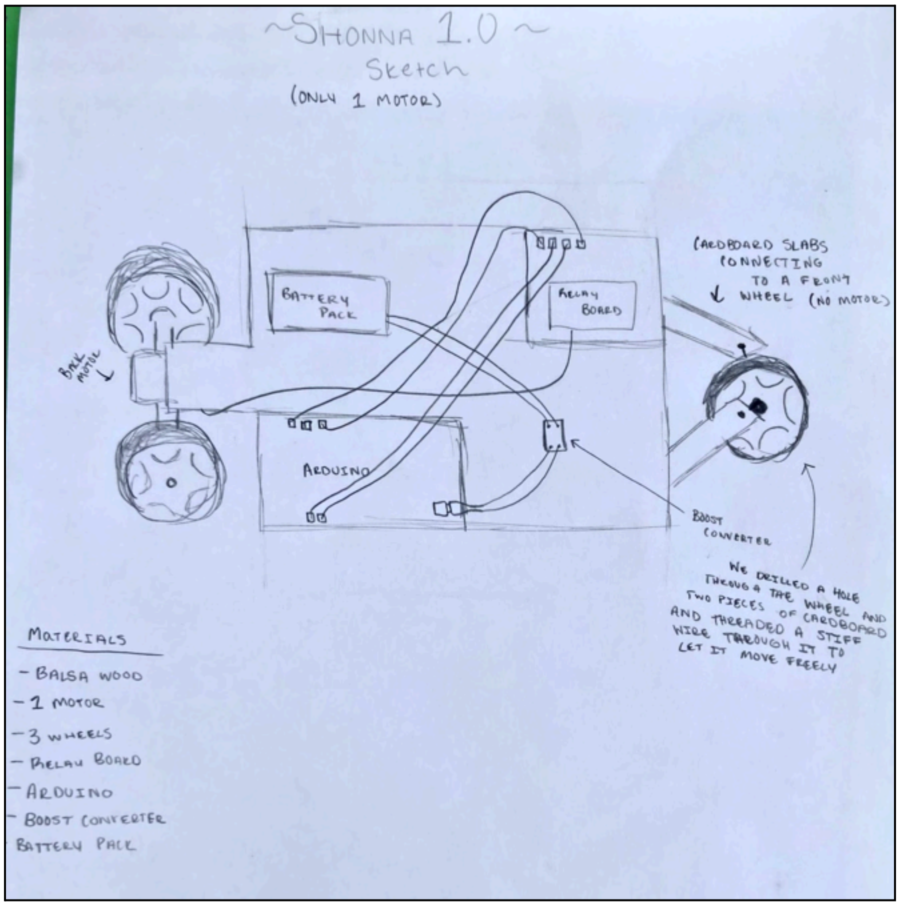
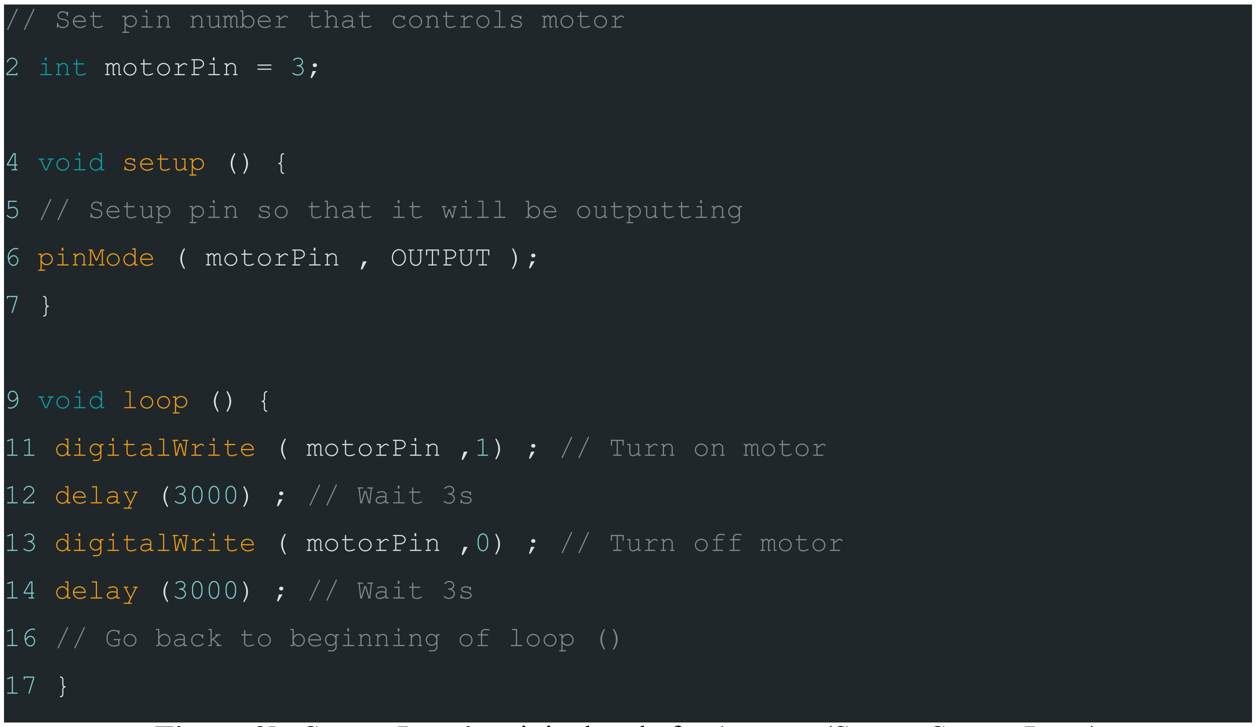


Figure 2a. This is a sketch of the car with a new Arduino after our last Arduino broke.

We also decided that we should only use one motor and have a freely moving wheel in the front. We connected the front wheel using two small cardboard cut-outs and drilled a hole through the wheel. We threaded a stiff wire through the cardboard pieces and the wheel to keep it in place.

We later changed back to two motors as this sketch did not have enough power to push the car up the ramp when we tested it. Also, the cardboard was too flimsy to keep the car moving straight.

New Arduino and Code



```
// Set pin number that controls motor
int motorPin = 3;

void setup () {
// Setup pin so that it will be outputting
pinMode ( motorPin , OUTPUT );
}

void loop () {
digitalWrite ( motorPin ,1) ; // Turn on motor
delay (3000) ; // Wait 3s
digitalWrite ( motorPin ,0) ; // Turn off motor
delay (3000) ; // Wait 3s
// Go back to beginning of loop ()
}
```

Figure 2b. Steven Luna's original code for 1 motor (Source Steven Luna)

As mentioned before, we needed to buy a new Arduino board on Amazon because our previous Arduino board's boot loader had become corrupted. Once we received the new Arduino, we uploaded a new code for only one motor. The new code for one motor was successfully uploaded to the Arduino board, and it could control the one rear motor.

Removal of Front Motor/Free Front Wheel

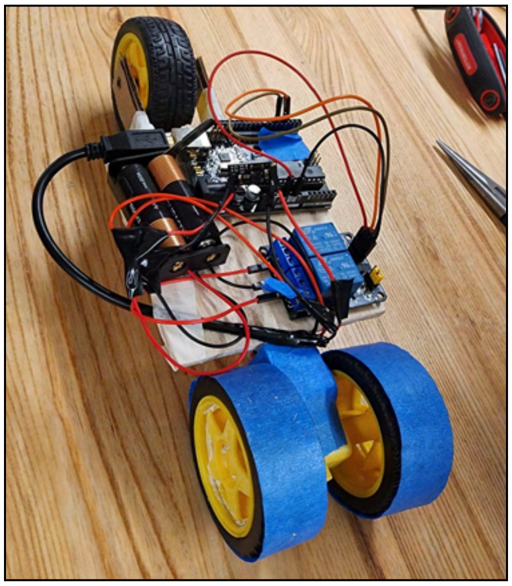


Figure 2c. Shonna 1.0: Removal of Front Motor/Free Front Wheel

We had decided to modify our design to have only one motor because our two AA batteries could not power both motors. Using only one motor would have also made our car lighter, requiring less power to ascend the ramp. As seen in the sketch of Shonna 1.0, the operating motor was located at the rear of the vehicle. To attach the front wheel, we drilled a hole into one of the wheels and inserted a beam through the hole, which would then be connected to the wooden base of the car by two cardboard slabs.

Testing Results

This design successfully went forward on a flat surface but veered leftwards. However, when we tested it on the ramp, it did not go up any distance. We concluded two significant issues with this design: our wheels needed more torque since they were so large, and another was that one motor would not provide enough power to go up the ramp. A minor issue was that we needed to find a better placement of the free wheel.
(Link to video of design test for Shonna 1.0)

Critique of Shonna 1.0

For Shonna 1.0, the issue was that the front wheel was unstable, partially because we had only attached it with tape and cardboard, and the car tended to steer to the left. Ideally, we desired our car to have a straight path to avoid collisions during the competition.

Although the video only shows our test for Shonna 1.0 on a flat surface, we determined that one motor was insufficient to provide enough torque to carry her up the ramp. At this point, we realized that the one-motor design did not improve the amount of current output that was supplied to the motors.

Another issue was that we had many problems with our wire connections. For Donna and Shonna 1.0, we decided to form temporary connections by simply twisting the wires and applying electrical tape because we wanted the flexibility of modifying our design before soldering. We eventually realized that these weak connections were an issue because our method of twisting the wires was ineffective and would constantly become undone. This resulted in our motors not working consistently due to poor wire connections.

In our attempt to figure out the problem, we toggled the switch on the battery pack, where we realized that the two double A batteries could power both motors. We then concluded that our problem was within the wire connections and not the motors. Therefore, we decided to go back to our original Donna design with two motors but solidify the connections so that the current from the batteries could successfully flow to the motors.

Shonna 2.0

Modifying Shonna 1.0

The modifications we made to Shonna 1.0 resulted in the production of our next model: Shonna 2.0. Once we realized that the battery packs were able to power both motors, we replaced the free-moving front wheel with a new wheel attached to a motor. We also solidified our wire connections by soldering to ensure that the circuit was not loose.

Description

We wanted to make Shonna 2.0 work with the wheels we already had, so to increase the torque for the second test, we wrapped masking tape around the wheels, ensuring it was tight enough so that the masking tape would not slip on the wheels. We also re-examined our wire connections from our first design and got two motors running on the batteries for our second design by solidifying those connections. At this point, we re-uploaded the code for two motors onto the new microcontroller. Since the third wheel was now connected to a motor, we used a piece of wood to hot glue the motor and wheel piece to the car. Other than that, the car's structure was mainly identical to that of the previous test (held together by tape).

Sketch of Shonna 2.0

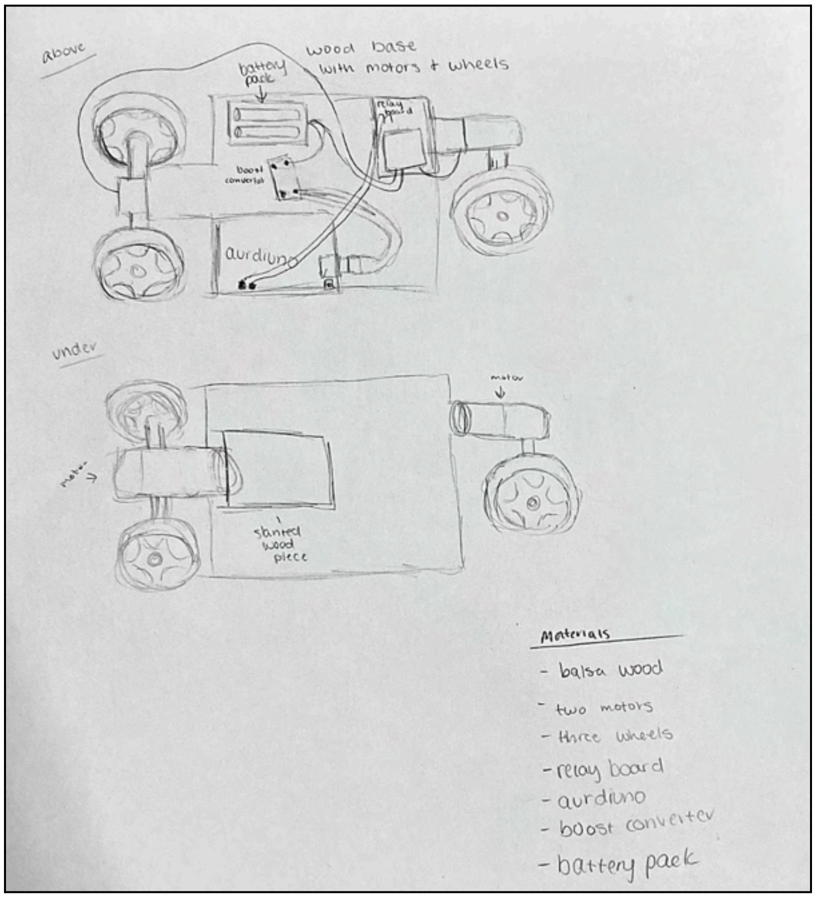


Figure 3a. This is the sketch of Shonna 2.0 with 2 motors. The figure shows placement of the motors on the frame of the car that we envisioned.

Replacing Front Free Wheel with Second Motor

For Shonna 2.0, we reverted to our original design for Donna, but this time with better wire connections. Similar to Donna's design, Shonna 2.0 had one motor in the front that was attached to a singular wheel and one in the rear with two wheels attached to its axle. We detached the taped front free wheel and re-attached a new wheel and motor unit. We added the front motor to the circuit by soldering the Ground cable to the battery pack and attaching the VCC cable to the relay board.

Solidifying Connections by Soldering

As mentioned in the critiques of Shonna 1.0, one of our major problems was with the connections between the wires. In order to solve this issue, we started soldering most of our wire connections to improve the current flow in the circuit. We soldered the input Ground and VCC pins from the battery pack to the boost converter and soldered the output ground and VCC pins from the Arduino board to the boost converter. In addition, we soldered the Ground connection of the motors to the battery pack.

Changes to Car Frame

The changes made to the car frame for Shonna 2.0 consistently remove the front free wheel and replace it with a second motor. To physically attach the wheel to the vehicle's body, we sanded the top of the motor before hot-gluing it to a small, narrow piece of wood. We then attached the narrow piece of wood to the base of Shonna 2.0 to form the connecting beam to the front motor. The narrow piece of wood needed to be off-centered so that the front wheel aligned with the vehicle's center.

Testing Results

Again, this design successfully went in a straight line on a flat surface; however, we had a terrible quality battery case, making the connection very inconsistent– since the two motors needed much power and the battery case was affecting the output from the batteries. The motors would only start some of the times that we turned the battery switch on. We knew this because when the motors did not turn on, we felt a quick jolt from the motors trying to start.

Critiques of Shonna 2.0

The issue with Shonna 2.0 was that our battery packs were very unstable. The battery packs we bought on Amazon could have been better quality because the switch had to be placed precisely for the battery output to power the circuit. This became a critical issue because it would take multiple tries to switch the battery pack on. After asking around different groups and the lab TAs, we tried to find other battery packs with switches (what we needed). Therefore, we decided to buy new battery packs on Amazon that had better quality on/off switches, but there was an issue with the delivery of the battery packs.

We only realized after the King of the Hill competition that the faulty battery pack was the cause of our less-than-ideal output powering the Arduino board and the motors. Since the Arduino board is supposed to power the relay board, the decrease in output voltage from the battery pack severely affected the performance of the motors.

Shonna 3.0

Modifying Shonna 2.0

Our next design, Shonna 3.0, was created through a series of additions made to Shonna 2.0: the addition of the supercapacitors and the grip tape. Before we went with the capacitors on Shonna 3.0, we were trying to figure out an easier way to run both motors simultaneously. An older student in BHE described the gearbox idea and how it could help give our car more torque with less speed. Since we were more concerned about getting Shonna up to the top of the ramp and less about the speed, we considered making a gearbox. We already have gearboxes within the motors, so we could try to code for the motors to move. Therefore, we ended up proceeding with the supercapacitor option on Shonna 3.0.

Description

Since we were unsure if the new (better) battery case from the one we had on Shonna 2.0 would arrive on time for us to replace the old one due to problems with the delivery, we decided to find another way to supply power to our two motors. We constructed a capacitor system to power our front motor, allowing the double AA batteries to power only the rear motor and the microcontroller (we know from Shonna 1.0 that this model works for the old battery pack we had).

Sketch of Shonna 3.0

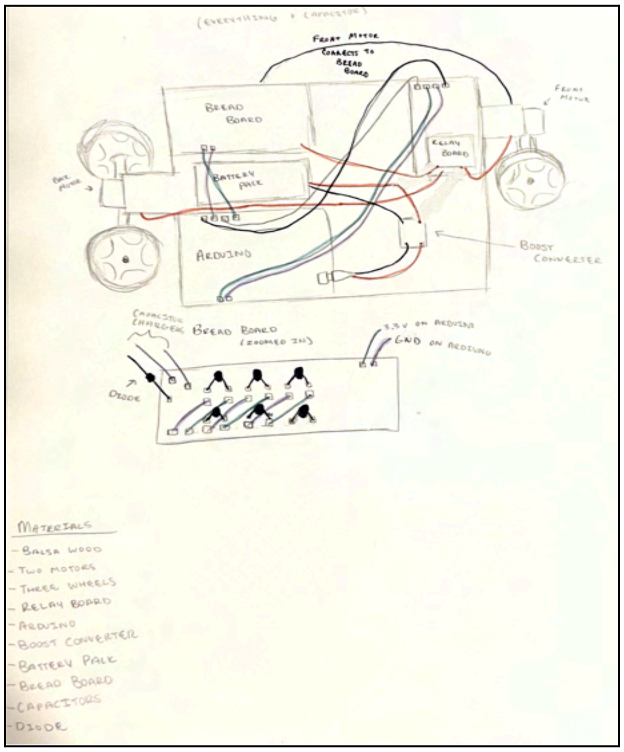


Figure 4a. This is a sketch of our second design for the car. We added in capacitors at this point and had it connected to the front motor.

Grip Tape for Traction

We had been using tape around the wheels for traction, but grip tape would work much better because it is more reliable since the stickiness of the tape goes away after a while. The grip tape worked much better, and we stuck with tape for the front wheel since it would help hold itself at the top of the ramp better because it was sticky.

Addition of Supercapacitors with Breadboard

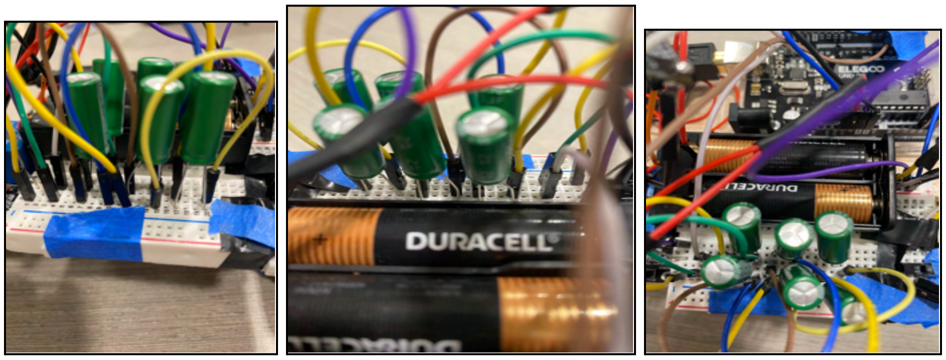


Figure 4b: Front, back, and aerial view of supercapacitor configuration

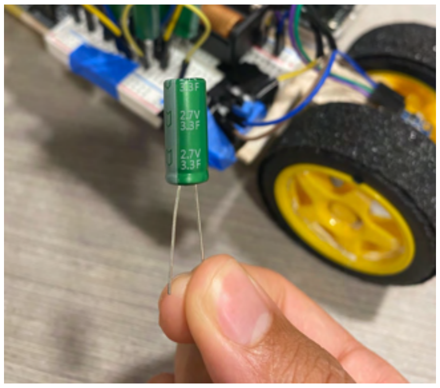


Figure 4e: Supercapacitor used

Shonna 3.0 has the addition of a set of 6 supercapacitors in a parallel configuration, all of which are connected to the front motor and are precharged before each match. Each has 2.7 V and 3.3 F. This parallel configuration would add more current (Farret) to the capacitor system. The calculation for the total farret is 3.3 F * 6 = 19.8 F, and the voltage remains the same at around 2.7 V since it is in parallel and not series (Allaboutcircuits.com). We decided to use a parallel configuration instead of a series because while a series would increase the voltage significantly, meaning the front wheel would get more torque as well as speed, the current would stay the same at 3.3 F. 3.3 F (from testing) was used up too fast to power Shonna 3.0 up the ramp and throughout the 30 seconds of the match. We also decided to keep the breadboard (with the supercapacitors attached) on the final version, as creating a more direct connection would be time-consuming and difficult. Additionally, the breadboard comes in use, and we later decided to add a mechanical tilt switch to Shonna 4.0.

Testing Results

From our test, the 19.8 F current calculated above carried sufficient current to keep the front wheel spinning fast for around 30 seconds, and the torque was enough to drive Shonna 3.0 up the hill. Although the breadboard did not add much weight to Shonna 3.0, it did make her bulkier and caused some problems.

Our testing of Shonna 3.0 with tape instead of grip tape shows us the inefficiency of replacing the tape on the rear wheels after every match. Adding the grip cuts this step, allowing us to focus on other vehicle parts between matches. At the same time, it is still grippy enough to help Shonna 3.0 climb the hill.

This design successfully went up the hill; however, we still needed a tilt sensor, so it did not stop at the top of the ramp. While we could not record Shonna 3.0 going up the hill, the same result but with the tilt sensor attached can be seen in the official testing video below.

Critiques of Shonna 3.0

One problem with Shonna 3.0 was that she required an external power source to charge the capacitors. Although the contest parameters do not say anything against using separately charged supercapacitors, using the 9-volt battery to charge the capacitors between each use meant that we used more than just 2 AA batteries to power our car.

Also, the capacitors and the breadboard added extra weight to our car. This extra weight means more power would be needed to make the car travel up the ramp. The capacitors also lose charge quickly and would have to be recharged between each round.

Another issue arose when Amazon sent us an empty package. When we ordered our new battery pack, we waited around two days for the package, but it arrived with nothing. Therefore, we were forced to order a new battery pack which delayed our progress by four days. Our current battery packs were still a severe issue because they were unreliable, and we were worried that the new packs would not come in time. Therefore, we started to plan for an alternative solution involving a mechanical switch to efficiently power the tilt sensor, which will turn on the Arduino board.

<u>Shonna 4.0 (FINAL DESIGN)</u>

Modifying Shonna 3.0

Shonna 4.0 is the final design we used for the competition and has several modifications from Shonna 3.0. Shonna 4.0 consists of the tilt sensor, a mechanical switch used to turn the Arduino on and off, and a new battery pack.

Testing Results

The car went up the hill and stopped at the top. The only change that we wanted to make at this point was to add a delay to the tilt sensor so that the car stopped a little closer to the center of the platform, however, the microcontroller stopped uploading code. Because of this, we went into the competition without a delay on the tilt sensor.

Description

We added a tilt sensor to Shonna 3.0 and coded it to stop immediately when the car is horizontal. We also added a mechanical switch so that we could turn the tilt sensor on and off. We were also able to replace our battery pack to a better quality one, making the connections more solid. Since this was our final design, we put everything together using hot glue in order to make everything secure.

Schematic for Shonna 4.0

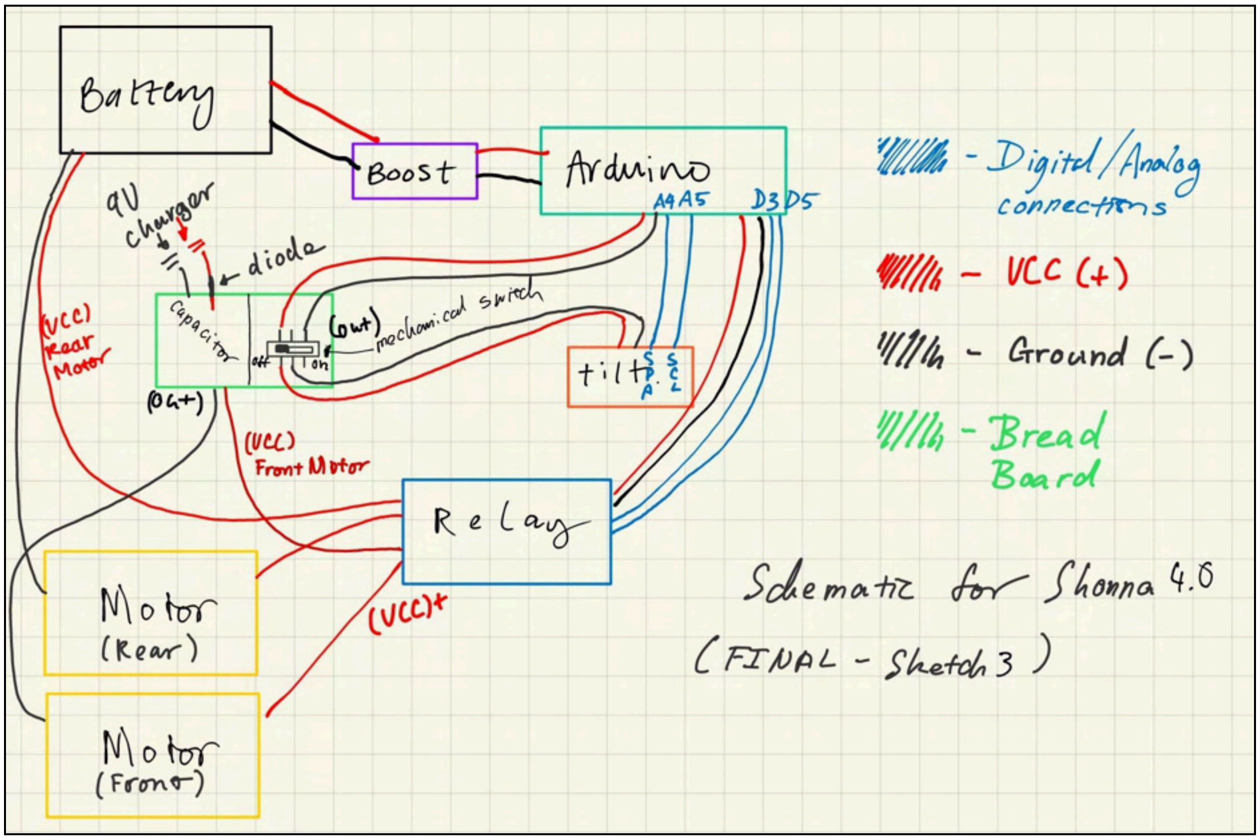


Figure 5a. Schematic for Shonna 4.0

Sketch of Shonna 4.0

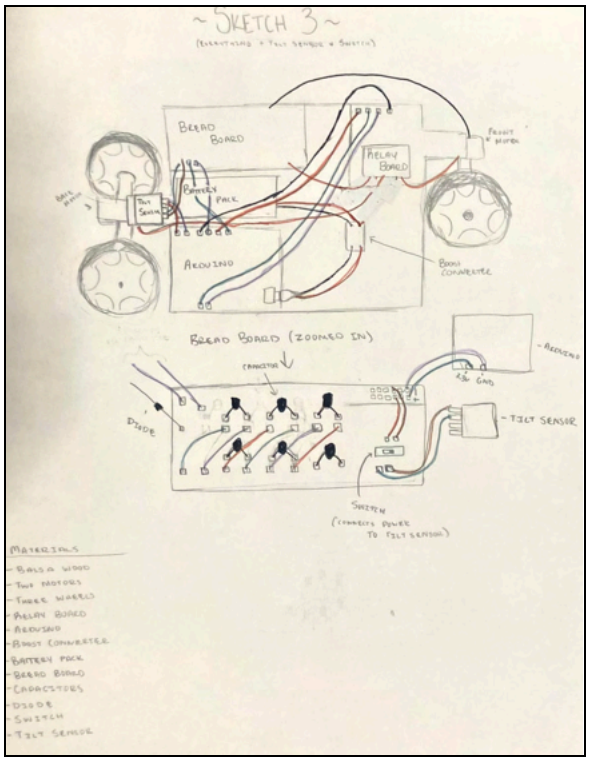


Figure 5b. This is our final sketch of Shonna. In this sketch, we added in the tilt sensor and switch. The tilt sensor is used to make the motors move when angled up and stop the motors when leveled. The switch was used to turn the relay board on/off for an easily accessible way to start the car on the ramp.

Adding the Tilt Sensor

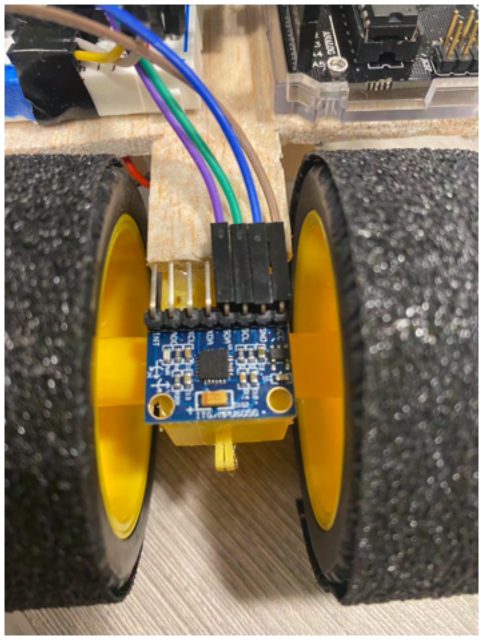


Figure 5e. Image of tilt sensor (located at rear wheels) (MPU 6050 Accelerometer/Gyroscope)

After adding the breadboard and capacitors for the front motor, we needed to connect the tilt sensor and code it so that we could get Shonna 4.0 to only move when angled upwards and stop moving when leveled. We connected the tilt sensor directly to the Arduino board (refer to **Figure 5a** for the specific inputs and outputs of the wires). After connecting the tilt sensor fully, making sure the connecting pins don't touch each other, we got the sensor to turn on. But we needed to code for it, and find a way to make the on/off button more accessible during the competition since we have been turning it on from the switch on the battery pack.

Coding for Tilt Sensor

```
#include <MPU6050.h>

// I2Cdev and MPU6050 must be installed as libraries , or else the . cpp
/.h files
// for both classes must be in the include path of your project
# include " MPU6050 .h"
# include " Wire .h"
```

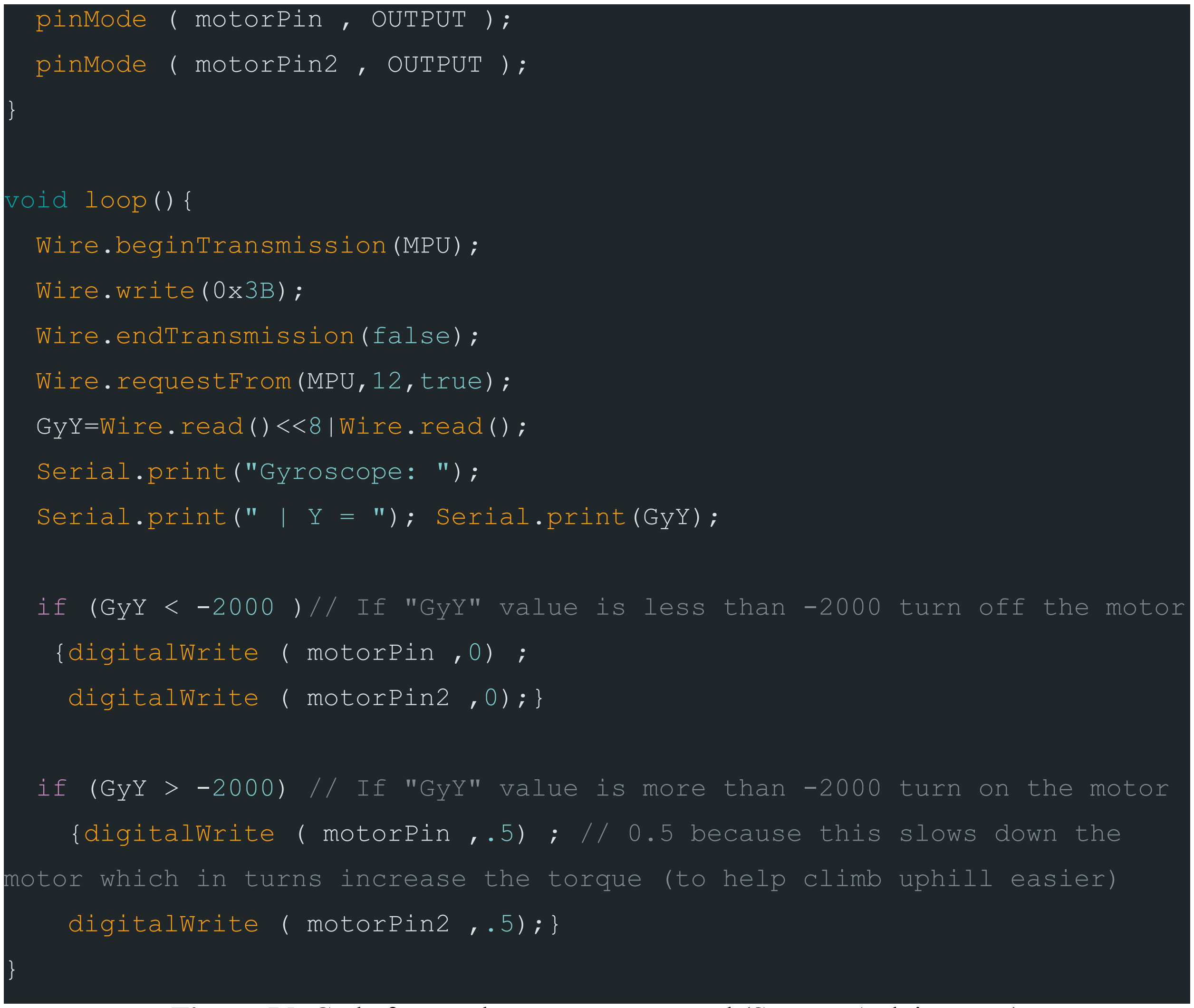
```
// Create object to read accelerometer data
MPU6050 accelgyro ;

// Create variables to hold acceleration values
int16_t ax , ay , az ;
int16_t gx , gy , gz ;

void setup () {
// Initialize communication with accelerometer .
// The Wire library requires special communication via
// pins A4 ( connected to SDA ) and A5 ( connected to SCL).
Wire . begin () ;

// initialize serial communication
Serial . begin (38400) ;

// initialize accelerometer
accelgyro . initialize () ;

// verify connection
Serial . println (" Testing device connections ...");

if( accelgyro . testConnection () ) {
Serial . println (" MPU6050 connection successful ");
} else {
Serial . println (" MPU6050 connection failed ");
}

}

void loop () {
// read raw accel / gyro measurements from device
accelgyro . getMotion6 (& ax , & ay , &az , &gx , &gy , & gz );

// You can also use these lines to ...
```

```
 // accelgyro . getAcceleration (&ax , &ay , &az); // Get acceleration
data
 // accelgyro . getRotation (&gx , &gy , &gz); // Get gyroscope ( rotation
) data

 // display tab - separated accel / gyro x/y/z values

 Serial . print ("a/g:\t");
 Serial . print ( ax ); Serial . print ("\t"); // Print acceleration in x
 Serial . print ( ay ); Serial . print ("\t"); // Print acceleration in y
 Serial . print ( az ); Serial . print ("\t"); // Print acceleration in z
 Serial . print ( gx ); Serial . print ("\t"); // Print rotation in x
 Serial . print ( gy ); Serial . print ("\t"); // Print rotation in y
 Serial . println ( gz ); // Print rotation in z

 }
```

Figure 5c. Code for accelerometer - was not used (Source: Steven Luna)

Above is the code from Steven Luna's "Arduino Notes," which we did not end up using as we could not download the "I2Cdev" library. We then decided to use outside source to code for our tilt sensor.

```
#include<Wire.h>
const int MPU=0x68;
int16_t GyY,Tmp;
int motorPin = 3;
int motorPin2 = 5;

void setup(){
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
  Serial.begin(9600);
```



```
  pinMode ( motorPin , OUTPUT );
  pinMode ( motorPin2 , OUTPUT );
}

void loop(){
  Wire.beginTransmission(MPU);
  Wire.write(0x3B);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU,12,true);
  GyY=Wire.read()<<8|Wire.read();
  Serial.print("Gyroscope: ");
  Serial.print(" | Y = "); Serial.print(GyY);

  if (GyY < -2000 )// If "GyY" value is less than -2000 turn off the motor
   {digitalWrite ( motorPin ,0) ;
    digitalWrite ( motorPin2 ,0);}

  if (GyY > -2000) // If "GyY" value is more than -2000 turn on the motor
    {digitalWrite ( motorPin ,.5) ; // 0.5 because this slows down the
motor which in turns increase the torque (to help climb uphill easier)
    digitalWrite ( motorPin2 ,.5);}
}
```

Figure 5d. Code for accelerometer - was used (Source: Arduino.com)

The code above was our code that was uploaded to our Arduino board. The code doesn't require downloading any libraries, which made it more straightforward. The code has been modified by combining the original code with the two-motor code in **Figure 1d.** During that process, we also removed 5 other variables (AcX, AcY, AcZ, GyX, and GyZ). In the tilt sensor, two main things are being measured: Acceleration in x, y, z planes (signify by "Ac") and angular change in x, y, z planes (signify by "Gy"). Since Shonna 4.0 is only going up the hill (and potentially being pushed back down in the same direction), only one of the angle change variable is needed. We ended up with GyY (angle change in the Y plane) as it was the value that changes when we tested the readings through the serial monitor.

Then, we adjusted the threshold to tell the Arduino at which point the car is leveled. We decided the threshold through testing multiple times the estimated GyY value for when the car is leveled.

Addition of Mechanical Switch

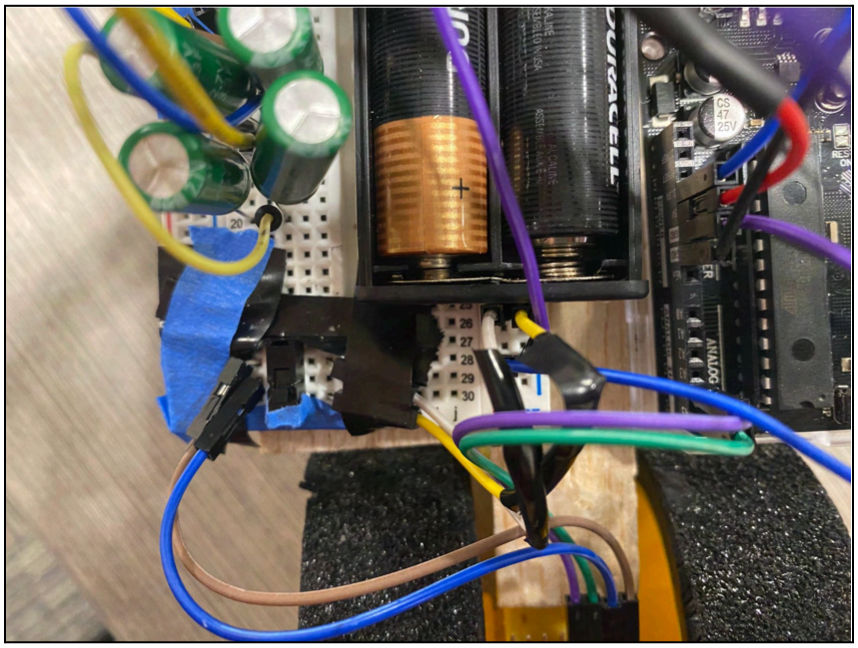


Figure 5f: Image of Mechanical Switch (on the left side connected to the blue, tan, yellow, and white wires)

The mechanical switch was a decent fix to the issue of turning on the car more accessible and without having to turn the switch on the battery pack. Another issue we had with turning on the entire car using the battery pack's switch was that when we turned it on from the battery pack, the coding, at the start, would be inconsistent and the sensor sometimes become corrupted and needs to be reset. To fix this, we would keep the battery and relay board on and attach a switch to the breadboard so we can turn on the tilt sensor from this. When the relay board is turned on and the switch is off for the sensor, nothing would happen. When we switch the sensor back on, the board requires a reset for everything to start running. So we use the reset button to turn on the Arduino and starts the code, rather than turning everything on at once through the battery pack.

Addition of New battery Pack

After the issues we had with the old battery pack (one with a knife switch), we decided to get new packs with a standard switch. This battery pack was more efficient than the knife switch since the connection was consistent and more secure. The knife switch would damage the connection between the batteries and the positive and negative wires. With the new battery pack, we could ensure that the battery would never lose connection and would not cause any issues supplying energy to the rest of the car since the switch is secure.

Official Testing

In our official test of Shonna 4.0, the only issue was that she stopped too early. In order to combat this, we moved the tilt sensor back more towards the rear of the car so that she would stop a little later, thus ending closer to the center of the ramp. (Link to video of Shonna 4.0 official test) (Link to 2nd video of Shonna 4.0 official test)

Reasoning for Final Design

Our final design (for the competition) was Shonna 4.0, and we stuck with her, instead of changing back to Shonna 2.0 but with a better battery pack and hence more solid connections, because the capacitors gives the car an extra boost of power, which allows the battery to only focus on powering the rear motor. While this extra boost of current is perhaps not needed for going up the ramp, it could have helped going against cars that are built to keep pushing the opponent down the hill. We tested her enough to know that she would go up the ramp and was capable of stopping at the top of the ramp and go back up if she gets pushed back down the hill. We worked until the last day on this final design, so we had to decide fast on what to go with as we had many different ideas to perfect our design. The capacitors worked well as long as we had them charged up every time we wanted to test Shonna 4.0 on the ramp. The mechanical switch helped start the car easier since we did not want to go underneath the battery pack to reach the switch during the competition. While the capacitors were complicated and needed to charge to work, we had a pretty simple charging setup (9V battery charging capacitor through diode and GND of capacitors - refer to **Figure 5a**) and the system overall was able to meet the goal of making it up the ramp and stopping at the top. Since our motors have good traction and current power, we assumed that if the opponent tried to push our car down during the competition, we would be able to hold on to the ramp.

Competition Results

Round 1:

On the day of the competition, Tuesday, October 18th, Shonna 4.0 competed in 5 rounds against groups 3, 4, 6, 16, and 9 in that order. Regarding strategy, we decided to replace the AA batteries and charge the capacitors using a 9-volt battery before each round to ensure that the motors had the most voltage possible. We charged the capacitors to around 3.8 volts and checked the voltage with a multimeter. We also replaced the masking tape on the front wheel between each round so that the wheel had enough traction and decreased her chances of slipping. For the competition, we instructed the TAs to start the car by pressing the ON button on the Arduino.

In the first round against car #3, our car successfully ascended the hill and stopped in an ideal position. Although the car did not stop where the back wheel was at the center of the ramp, Shonna 4.0 stopped where the back wheel rested at the ledge of the flat section of the hill, whereas the opposing group's car barely reached the top of the ramp.
(Link to video of Round 1: Shonna 4.0 vs Group 3)

Round 2:

For round 2, Shonna 4.0 competed against group 4. Before the start of the round, we replaced the AA batteries, re-charged the capacitors, and replaced the tape on the car's front wheel to maximize traction. During this round, Shonna 4.0 reached the top of the hill after the opposing car, but she won the round because her rear wheel was closer to the center of the ramp. Compared to the first round, the tilt sensor performed better because the rear wheels rested on the flat section of the ramp rather than at the ledge. At this point, Shonna had a record of 2-0 for the competition. In the video, our car is on the left side of the ramp. (Link to video of Round 2: Shonna 4.0 vs Group 4)

Round 3:

In round 3, our car competed against group 6. As mentioned before, we replaced the AA batteries, re-charged the capacitors with a 9-volt battery, and replaced the tape on the front wheels. However, prior to this round, we decided to replace the grip tape on the back wheels with regular masking tape to increase traction. Since the masking tape had adhesives, we believed they would stick better to the ramp to prevent the car from slipping.

In this round, Shonna 4.0 successfully ascended the ramp, but unfortunately, she could not stop at the top. Shonna 4.0 had lost for this round because the car traveled down the other side of the hill, which was further from the center than group 6. We had deduced that the issue in this round was due to the tilt sensor not functioning correctly. The tilt sensor could not accurately read when the car was level (we suspect this was because the code was corrupted- see page _ for explanation) on the top of the ramp, and therefore the motors kept running. In the video, Shonna 4.0 is the car on the left. (Link to video of Round 3: Shonna 4.0 vs Group 6)

Round 4:
Prior to the start of the 4th round, we realized that the tilt sensor became inconsistent within the first 30 seconds when we initially turned on the battery pack. In other words, the tilt sensor would not signal to turn on the motors when the car was at a slant and would also not signal to turn off the motors when the car was level. To solve this issue, we decided to replace the batteries and then test Shonna 3-5 times before placing her on the ramp for the competition. Similar to previous rounds, we re-charged the capacitors and replaced the tape.

In round 4, Shonna 4.0 competed against group 16. In terms of strategy, we planned to avoid a collision with the opposing vehicle by placing Shonna 4.0 on the right side of the ramp. For this round, our car did not function properly because it failed to move when the ON button was pressed. This was the first instance in which this issue arose (we believe that Shonna 4.0's failure to move was due to overcharging the capacitors- see page 35 for further explanation). Fortunately, Shonna had won this round because the opposing car had traveled past the top of the hill and down the ramp past Shonna 4.0. Although Shonna 4.0 was unable to move, she won round 4 against group 16 because the opposing vehicle had traveled further from the center of the ramp. (Link to video of Round 4: Shonna 4.0 vs. Group 16)

Round 5:

Prior to the final round, we realized that Shonna 4.0 did not move because the front wheel did not have enough torque. There was an issue with the amount current powering the front motor because the wheel was not running with the same amount of power. We attempted to solve this issue by constantly re-charging the capacitors to its full extent and replacing the batteries.

In round 5, our car competed against group 9. Shonna 4.0 had lost this round because she still did not move whereas the opposing vehicle had reached the top of the hill. Similar to the previous round, our car did not move because of the issue with the current supplied to the front motor. At the end of the competition, Shonna 4.0 had a record of 3-2 and received 4th place in the final rankings. In the video, our car is on the right side of the ramp. (Link to video of Round 5: Shonna 4.0 vs. Group 9)

Final Modifications

Re-soldering connection to boost converter

After the competition, we attempted to fix Shonna 4.0 because she experienced several malfunctions within the last three rounds. Some disruption in the circuit prevented the tilt sensor from working correctly and powering the Arduino. Another issue was that the front wheel severely decreased in torque and could not carry Shonna 4.0 up the hill.

First, we detached some of Shonna 4.0's components and noticed that the tilt sensor would work when we bent the output VCC cable a certain way on the boost converter. Therefore, we figured that one of our soldered connections to the boost converter was loose, and we re-soldered the pin to the boost converter.

After re-soldering the loose connections, we realized the tilt sensor still needed to be fixed because the motors would continue to run even when the car was level. Next, we experimented with the connections and eventually removed the switch, capacitors, and breadboard.

Removal of Switch and Capacitors

After we removed the switch, capacitors, and breadboard, we re-wired the ground cable of the front motor to the negative terminal of the battery pack and the VCC from the relay board to the positive terminal of the battery. Here, the tilt sensor was successful in powering the Arduino when the car was slanted and turning it off when the car was level.

Conclusions

Reasons for Final Modifications

In our final modifications, we omitted the supercapacitors and the mechanical switch. In doing so, we connected the ground and VCC of the front motor directly to the battery. We decided to make these final modifications because we wanted the tilt sensor to be consistent when it signaled the Arduino to turn on and off and provide the front wheel with sufficient torque. Shonna 4.0 failed in rounds 3 and 5 because of the tilt sensor malfunctioning and the loss of torque in the front wheel. Due to the final modifications made at Biegler after the competition, we resolved these issues. In the end, we did not need the supercapacitors, the breadboard, or the mechanical switch since the two double A batteries were sufficient to power both motors.

What Would We Have Done Differently?

If we were to do this project again, we would have done more testing to determine the issues with our car. We made a mistake by concluding that our two double AA batteries would need more power to power our two motors. However, the issue was the connection of our wires, not the need for more power. Because we did not conduct further testing, we added a capacitor to our car to get more power. This led to another instance where we failed to test our components adequately– we should have tested the capacitors more extensively. After we got the capacitors to work for the first few times from charging them with a 9-volt battery, we concluded that this was an effective way to operate our capacitors and stopped our testing there. In the end, since we consistently over-charged our capacitors with the 9- volt, we ruined our capacitors and they conveniently stopped working during the competition.

We could have improved our capacitor system by having the double AA batteries charge the capacitors directly and having a mechanical switch for our tilt sensor since we want our tilt sensor to be off while the capacitor charges. This change would make charging the capacitors much more efficient and straightforward compared to how we charged our capacitors in between rounds. This would also provide the max amount of power for our motors.

An alternative to the capacitor system as a whole is running our two motors on our double AA batteries alone, as seen in **Figure 6A**. This would have been our least complicated option; however, we ruled it out because we had already started developing a capacitor system when our new battery packs arrived. We should have tested the second design with our new battery pack. If we had done this before the competition, we would have seen that our car consistently works and goes up the ramp when powered by the double AA batteries alone, and we would have probably gone with this design as our final one.

Another thing that we would have done differently starts with a cheaper microcontroller (rather than the Arduino brand). Replacing our Arduino brand microcontroller solved many of our problems, as the reset button worked and adequately uploaded code. However, we decided to replace our Arduino pretty late in the game. We were trying to make the Arduino microcontroller work instead of replacing it, and it caused much stress toward the deadline.

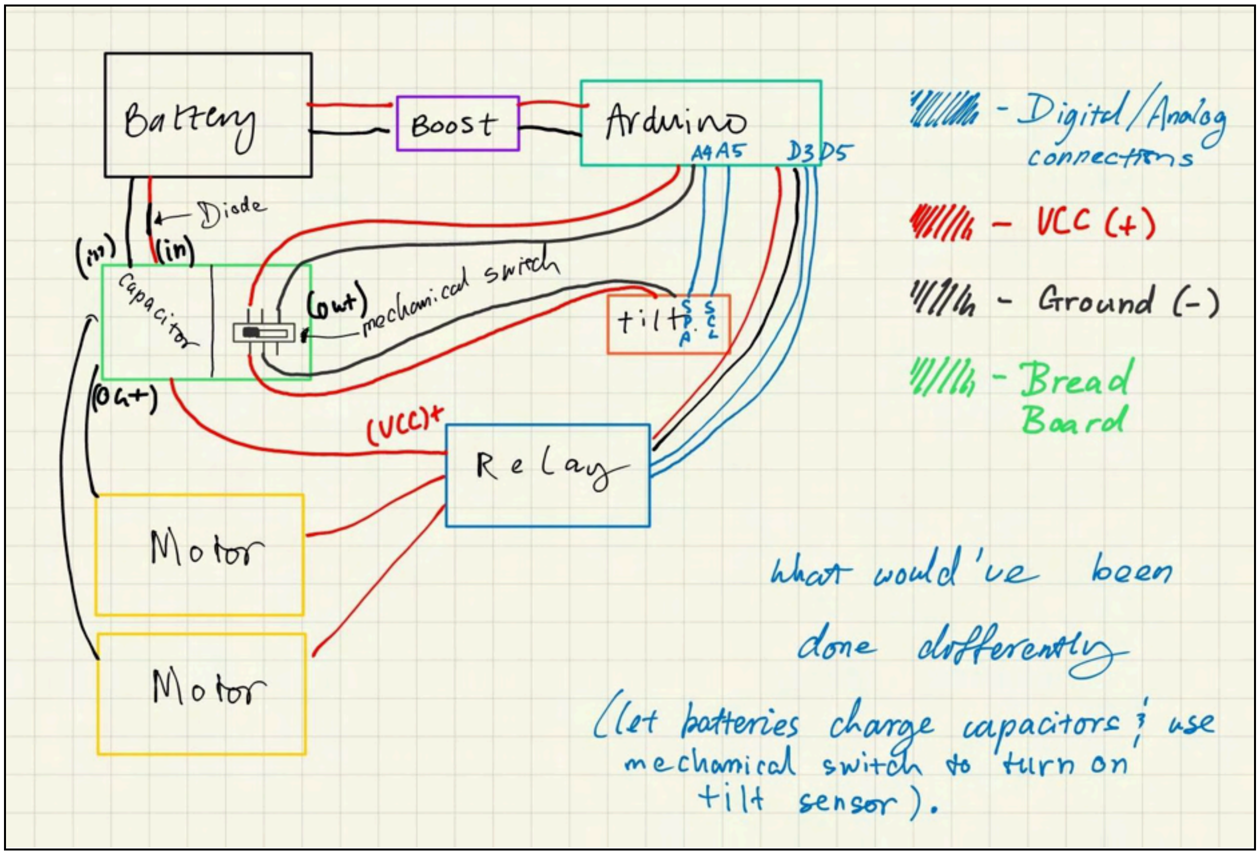

Battery
Boost
Arduino
A4 A5
D3 D5
Diode
(in)
(in)
Capacitor
(out)
mechanical switch
tilt
SDA
SCL
(out)
(VCC)+
Relay
Motor
Motor
- Digital/Analog connections
- VCC (+)
- Ground (-)
- Bread Board
what would've been done differently
(let batteries charge capacitors; use mechanical switch to turn on tilt sensor).

Figure 6A. What would have been done differently if we were to continue to use the capacitors (if we let batteries charge the capacitors when turning on the car, instead of precharging the capacitors).

What We Learned

The biggest lesson that we learned is to approach problems more thoughtfully and carefully. When we encountered an issue with our circuit, we would panic and move wires around until the problem was solved, which was unwise (and unscientific) of us because we ended up changing things that didn't need to be changed. Along the same vein, we should have considered smaller issues, like wire connections, that we thought had a miniscule impact, but ended up being the source of a lot of issues.

Since most of us came into this project with no background knowledge and because we chose to not start out with a kit, this project taught us a lot about circuitry and the engineering process in general. We learned to deal with having non-ideal materials and figured out how to make it work.

We overcame a lot of obstacles along the way and kept each other going when things went wrong. Although we didn’t do as well as we hoped we would in the competition, it is hard not to be proud of Shonna.

Meeting Agendas

September 16th:

Objective:

- Finish reviewing King of The Hill project guidelines.
- Purchase certain materials needed to construct the car
    1. Arduino board
    2. Tilt sensors
    3. Motors
    4. Batteries
    5. Battery packs
    6. Boost converter
    7. Relay board
    8. USB B cable
- Research the uses of each component.

Summary:

- We compiled a composition with an inventory of commodities to purchase for the motorcar
- We made a unanimous decision to not buy a pre-made kit because it permitted us more space for inventiveness and had the potential to function better than cars built with the kit.

September 27th:

Objective:

- Figure out how to code for Arduino.
- Put together a working circuit (get the motors to run on the double AA batteries).

Summary:

- Motors were running in a closed circuit.
- Got started on Arduino programming (downloaded Arduino IDE).

September 30th:

Objective:

- Understand and put together schematic for Arduino board, batteries, motors, and relay board.
- Create test code for the motor.
- Test motor with relay board and battery.

Summary:

- Learned how to use the multimeter.
- Wrote code for relay board and motor.
- We got the motors to work with the test code through the relay board.

Set backs:

- We needed to figure out how to power the Arduino using the batteries.

October 4th:

- Objective:
    - Create a cable to connect the battery to the Arduino board.
- Summary:
    - We got 2 motors to work with the battery while the Arduino board is powered by the computer.

October 10th:

- Objective:
    - Figure out why the Arduino's code isn't working.
    - Start building the frame of the car
- Summary:
    - The bootloader of our Arduino board corrupted, so we ordered another off-brand Arduino. This new Arduino had a better reset button and handled inconsistencies a lot better.
    - We used the bandsaw to cut out a square piece of balsa wood to be the base (the platform that we could put our circuit on) of our car.

October 11th:

- Objective:
    - Soldering first time/soldering most of the wires and creating a holder for the front wheel to go on.
- Summary:
    - We soldered most of the wires including the ones connected to the boost converter.
    - We created a holder for the front wheel out of cardboard slabs and a stiff wire. This did not work as well as it was too flimsy and would drive to the left.

October 12th:

- Objective:
    - Solder all of the wires and test the strength of the connections
- Summary:
    - Lilian and Naya met to solder the rest of the wires.
    - Decided to bring back second motor/remove the motorless wheel in the front
    - We got the motors to run on the batteries at the same time.

October 13th:

- Objective:
    - Figure out the capacitors and breadboard connection to the front motor.
    - Attach front motor somehow
- Summary:
    - Started working on capacitors and trying to figure out how to charge them and connect to the front motor.
    - Reattached the front motor to the car and kept the two wheels in the back and one wheel in the front placement.

October 14th:

- Objective:
    - Get the tilt sensor to work.
    - Connect the front motor to the tilt sensor somehow.

- Summary:
    - We got the tilt sensor to work but not yet connected to the front wheel/motor.
    - Wired the breadboard and connected it to the front wheel motor.

October 16th:

- Objective:
    - Code for a delay on the tilt sensor, figure out how we are going to charge our capacitors.
- Summary:
    - We figured out how to code for a delay on the tilt sensor and connected it to the front motor through the breadboard
    - Figured out that we can charge the breadboard by supplying it with energy through a 9 volt battery.
    - Got the tilt sensor connected to the breadboard and connected to the front wheel motor.

October 17th:

- Objective:
    - Do the final touch ups of the car
    - Test the car
    - Final placements of the parts onto the base of the car.
    - Add some sort of switch/button for easy access of starting the car.

- Summary:
    - Regan cut out and sanded a little piece of wood to put under the car to keep it from getting stuck on the ramp
    - We attached everything to the base of the car and rewired everything because we had an issue
    - Tested the car a couple times on a makeshift ramp and got it to go up and stop when leveled on the top of the ramp using the tilt sensor.

October 18th (post competition):

- Objective:
    - Figure out why the front motor stopped working
- Summary:
    - We tried to re-solder the connections to the boost converter as we thought it might’ve gotten loose.
    - That didn’t help the front motor move and we came to the conclusion that the capacitors were overcharged and therefore disconnected the front motor to the tilt sensor.
    - We took the capacitors out as a whole and connected the front motor directly to the battery pack which fixed the issue and it worked.

References

How to do a loopback test. (2022, June). Arduino.com. https://support.arduino.cc/hc/en-us/articles/360020366520-How-to-do-a-loopback-test

How to use the accelerometer- gyroscope GY-521. (n.d.). Arduino Project Hub. Retrieved October 31, 2022, from https://create.arduino.cc/projecthub/Nicholas_N/how-to-use-the-accelerometer-gyroscope-gy-521-6dfc19

Luna, Steven. AME 101 *Arduino Programming Notes*. University of Southern California. Fall 2022.

Ronney, Dr. Paul D., AME 101 *King of the Hill*. University of Southern California. Fall 2022.

Series and Parallel Capacitors. (n.d.). All About Circuits. https://www.allaboutcircuits.com /textbook/direct-current/chpt-13/series-and-parallel-capacitors/