1. General Information

Student Name: Casey Merola, Keegan Smith, Jeffrey Nguyen

Student ID number:

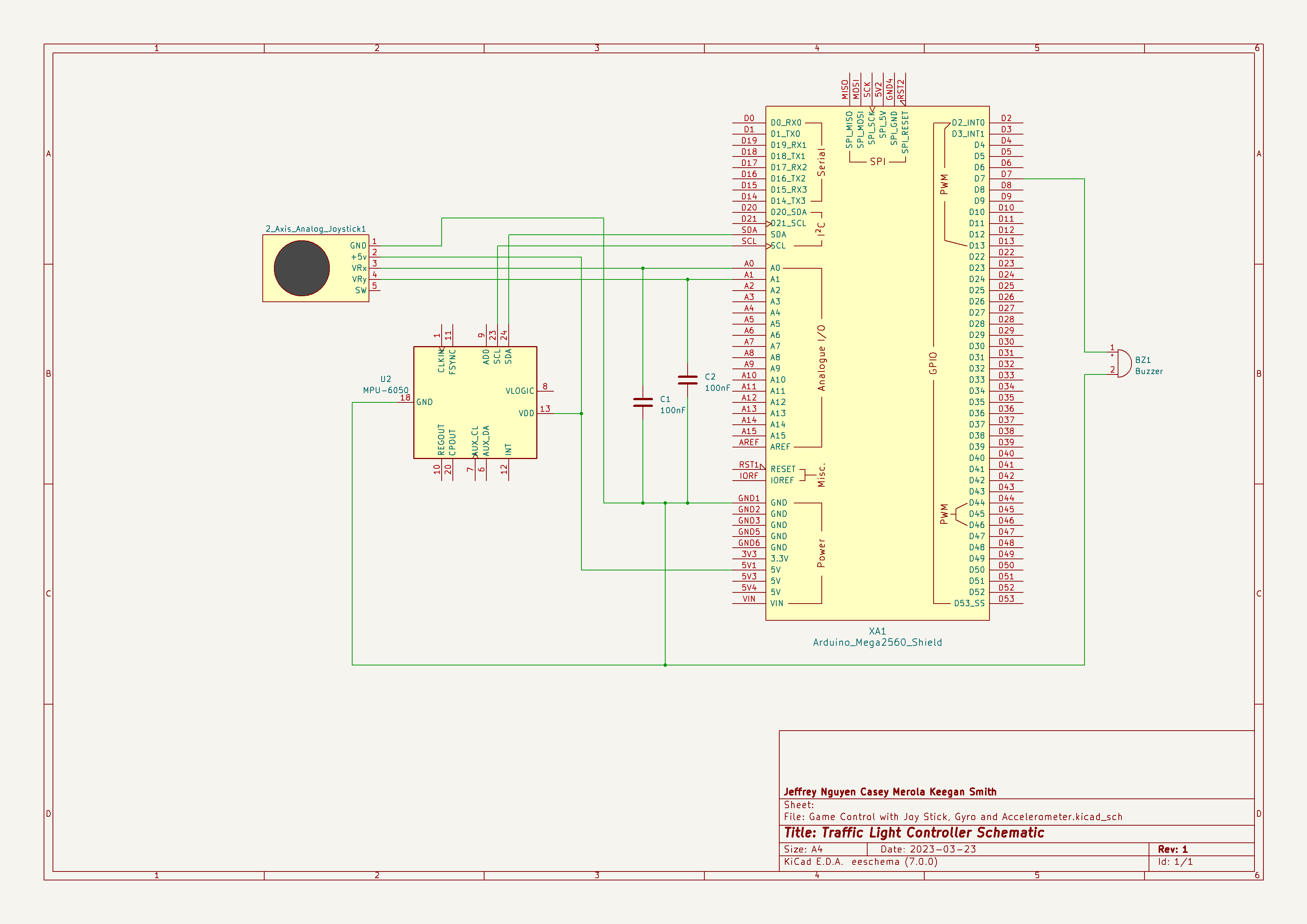
Team Name/Number: The Capybaras

Team member names: Casey Merola, Keegan Smith, Jeffrey Nguyen

Date of completion: 03/26/2023

Demonstration method: Zoom

1. Design:
   1. Hardware Design



**Figure 1. Schematic**

This project was completed by connecting the gyroscope to 5V and ground, with the SDA and SCL connected to their corresponding pins on the Arduino (pins 20 and 21 respectively). This enables the gyroscope to communicate serially with the Arduino using I2C protocol. The joystick is connected to 5V and ground, with the two analog outputs from the joystick connected to the first two analog input pins from the Arduino (A0 and A1). The buzzer is connected to pin 7, but any pin would work for this, as it is only a digital output.

* 1. Software Design

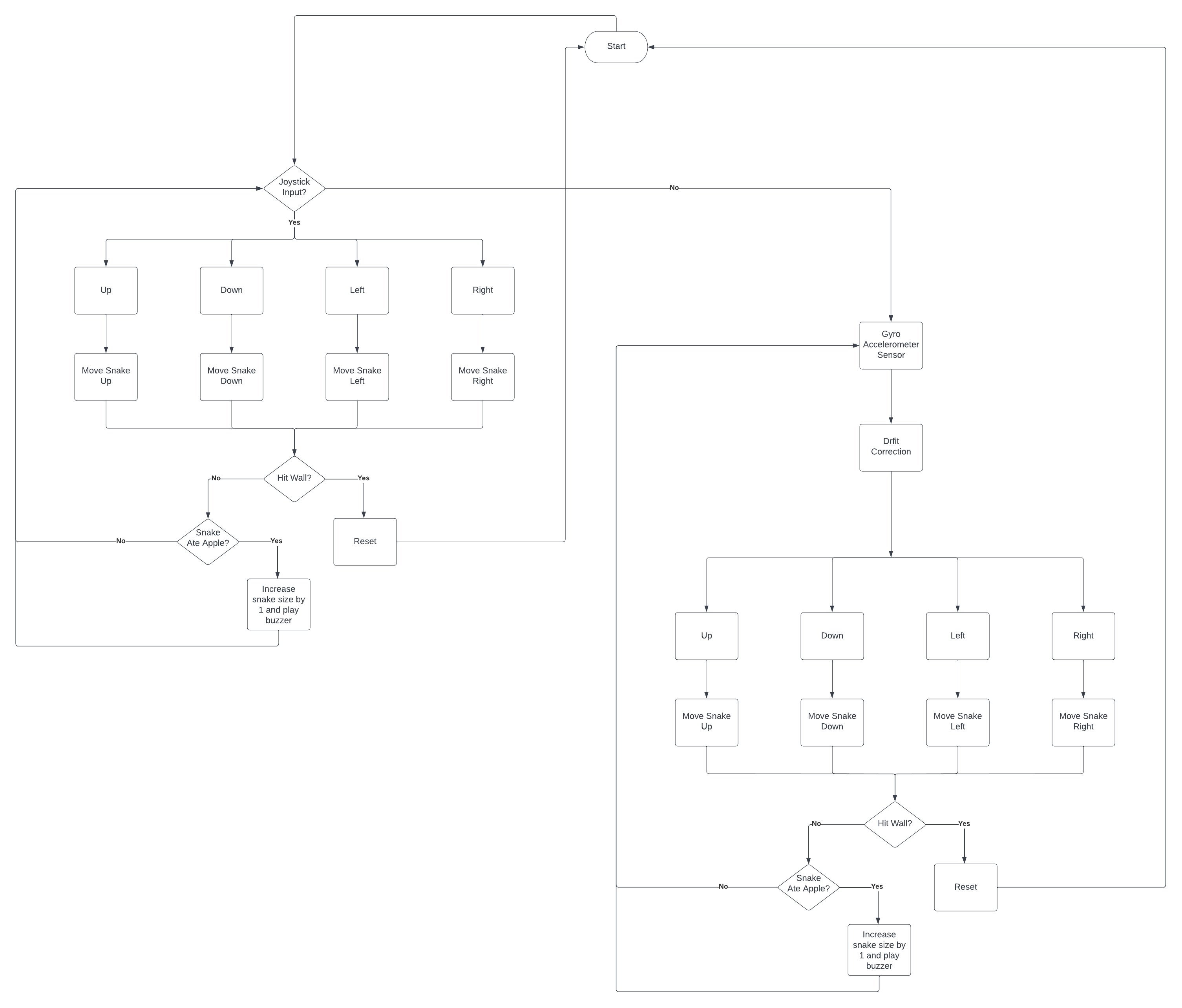


Figure 2. Flow Diagram

The project works by accessing the Wire.h file, allowing for communication between the gyroscope and the Arduino using the I2C protocol. The values for pitch and roll are taken from the gyroscope, to which a weighted average is taken in order to reduce the level of drift seen in the raw measurements. This is done by finding the slope, or difference, between the new measurement and the old measurement. After that, the new slope is determined by multiplying the slope, from the new minus the old measurement, by a constant, c1. Then, the slope calculated from adding the previous slope value multiplied by its own constant, c2, to the previously calculated slope. These two constants always add to one, and for the first 100 measurements, they are both equal to 0.5 to create a solid base on the readings. After that, c1 is reduced such that the old slope value has more sway in the readings, thus lowering the total slope of the drift.

The joystick is implemented by reading the analog voltage that it outputs, which is converted to a digital value from 0 to 1023. The value is centered by subtracting 512 from the measured voltages, which is then used to determine if the joystick is within the 90-degree range of each direction. This makes sure that the direction is measured accurately and will not have a large dead zone.

The two movement controls are implemented by first checking if the joystick is pressed in a direction, which overrides the gyroscope. This is done because if both were read at the same time there may be issues with moving the gyro contrary to the joystick and it would not be easy to operate. If the joystick is not being moved, the gyroscope is checked, and if it is over the threshold in any direction, the direction is sent to the snake game. Each direction is given a number, 1 for up, 2 for down, 3 for right, and 4 for left, which is sent serially to the python game.

The python game is edited with the correct communication port which is different for every device, and the serial port is read and evaluated if it is 1, 2, 3, or 4. If the game reads a direction, the corresponding function is called, either go\_up, go\_down, go\_right, or go\_left, which moves the snake in the game. When the apple is eaten, the python writes 'x’ to the serial port. The Arduino checks for an 'x', and if it reads it, the buzzer is set off until a timer interrupt turns the buzzer off.

GitHub Link: <https://github.com/CaseyMerola/lab2.git>

* 1. Results

The program works as expected, and the joystick and the gyroscope can both be used to control the snake, which beeps a buzzer when an apple is eaten. The joystick is programmed such that the stick does not have to be firmly in a single direction to trigger the movement, which makes the directional travel much easier. Any input within the general area of a direction is considered that direction, which helps make the stick easier to control.

The gyroscope control is a little more difficult to control, since it is programmed to detect a movement that may be difficult with the device connected to a breadboard. The measurement of the difference between data points makes the gyroscope fairly accurate, and it can move the snake in the desired direction most of the time. The only issues with the movement are likely due to the physical limitations of the connected breadboard.

1. Problems Encountered and Solved:

Drift from the gyroscope and the accelerometer were one of the most prominent issues. The rate of drift was so large that it became difficult to detect changes for a reasonable amount of time. This was overcome by taking a weighted average to mathematically reduce the drift significantly.

There were not many issues with the joystick interface, but when only one axis was considered for the considered, the joystick had to move very straight, and often would not register if it was a few degrees off. This was solved by considering both directions of the joystick to calculate regions for each direction rather than single values, which made the joystick easier to use.

Interfacing with the snake program had difficulties with the lack of debugging available. The only way to really debug the program was to print values to the console, and since it was reading the serial line to function, the console became slower and cluttered. This was solved by thinking through the program without brute forcing the serial communication.

1. Personal Contribution to the Lab (Technical Details):

Casey Merola

* Interfaced with joystick
* Programmed serial communication with python game
* Contributed to report

Keegan Smith

* Interfaced with Gyroscope/accelerometer
* Reduced drift from gyroscope/accelerometer
* Contributed to lab report

Jeffrey Nguyen

* Created the Schematic
* Created Joystick Symbol
* Created flow diagram
* Contributed to report

1. Lessons Learnt:

This lab taught the basic principles of serial communication, and showed that different programs, different languages, and different devices can be interfaced with, making communication relatively easy. The lab also taught about the basic principles of the I2C protocol, which was used to interface with the gyroscope. It demonstrated how easy this communication is at a high level, and in practice the protocol can be implemented with a few functions and an imported library. The power of serial communication was also learnt in this lab, since the gyroscope only needed to be connected with 4 wires, with only 2 data wires, and is able to transmit many values of position in many directions. The lab also taught how to correct for drift, which is a difficult issue for accelerometers when measuring position, as position is solved by integrating the acceleration twice, increasing the error significantly.