ECE 4961- Group 2 - MarioKart Bike Final Report

Reed Hester, Chase Griffin, Leah Faulkner
ECE Students, Tennessee Technological University
Cookeville, TN, United States of America
lrhester42@tntech.edu, cagriffin42@tntech.edu, lmfaulkner42@tntech.edu

***Abstract* --- This report introduces group two's project idea and how it was implemented. An introduction of the problem begins the report, followed by the literature review, methodology, results and discussion, timeline and cost, ethics, lessons learned, and conclusions.**

Keywords* --- *Capstone Design, gaming, emulation, exercise, sensors, electromagnetics, engineering

I. INTRODUCTION

A. Background

The Mario Kart exercise bike project is the result of an idea to bring physical activity together with the excitement of the racing video game experience. The objective with this idea is to design and develop a practical way to exercise using the traditional resistive elements of an exercise bike while simultaneously controlling a player in a game of Mario Kart in a simple-to-use, responsive, and enjoyable way. By using the bike as a direct controller, the player will be able to play as fluidly as they could with a standard controller with the added benefit of physical activity. Ideally, the bike will serve as a tool to assist in promoting a healthy lifestyle in an entertaining way.

To achieve this goal, the use of several external control systems combined with the inputs from a standard game controller will be employed. Microcontrollers with code interfaces set up to specifically target the operations for each subsystem of the controls will be utilized as well. These powered subsystems will consist of an adjustable resistance method, a pedaling speed sensing system, an encoded steering system, and an actual standard game controller. The resistance method will vary between two experimental versions of the bike. One method will consist of a magnetic resistance system involving a linear actuator that can incrementally increase or decrease the physical distance of a magnet from the bike wheel, thus creating dynamic resistance based on said distance. The other method of resistance is to use a DC motor connected to a MOSFET that allows the motor to be shorted on command, which creates resistive force when the motor is being spun. The pedaling sensor system will involve a tachometer sensor that measures the speed of the revolving pedals and uses that information to control the acceleration of the player's vehicle. The steering mechanism will govern a rotary encoder which will be used to interpret directional controls. Finally, in order to simplify menu navigation and other game interactions, a standard game controller will be included as well with the traditional button layouts and usage. Each of the sub-microcontrollers in these systems will be routed into a main microcontroller or RaspberryPi device that interprets the received data into usable control information that is forwarded to the device running the game. The game version itself will initially be the Nintendo64 copy running on an emulator program contained on a PC. These design choices have been made out of a combination of the apparent ease of use and design. The use of an emulator software will only be for testing purposes in order to verify the operation of all control systems and will not be used for distribution or copying of the game software or license. An official copy of the game will be employed in the final product. In order to be practical and efficient, the systems on the bike must be simple to understand for the user, easy to access for a varied user base, safe to operate while still being an effective exercise tool, and responsive enough to allow reasonably quick game control.

The measures used to determine the success and achievement of the specifications will vary for each subsystem. Some of them, such as the control accuracy and ease of use, can be qualitatively assessed through regular use. Analytically, MATLAB, LTSpice, and Simulink simulations will be used for the DC motor circuit, the magnet setup, and the pedaling system. Also, control fidelity and response time can be measured using input lag detection software.

B. Motivational Analysis

Imagine yourself blazing down the road as your favorite Mario Kart character, racing and drifting through your favorite tracks, dodging rogue shells and bananas that litter the course, and finally sailing past the finish line just millimeters away from that other player who has been right on your tail since the start. An activity unlike most others, this kind of game inspires a competitive and energizing rush in many players. While most everyone can agree that games such as this are a highly mentally engaging experience, excessive gaming, and a correspondingly sedentary lifestyle, however, can actually be quite dangerous to one's physical health over time. According to the CDC, adult obesity prevalence in the U.S. from 2017-2018 was found to be approximately 42.4% of the population, and has been shown to be steadily increasing since then [1]. While this is not necessarily directly linked to exercise habits, incorporating basic aerobic activity even without diet or routine change otherwise is proven to aid in weight loss and general health improvement [2]. However, many people find going to the gym, or rather exercise in general, to be too much effort, too inconvenient, too difficult, or just plain boring. Whatever the reason, If one could combine that great excitement and enjoyment that, as previously mentioned, often comes with their favorite video games with the extensive benefits of physical activity, it could likely make exercise a much

simpler process that people would feel more readily willing to participate in.

The objective for this project is to develop a system that integrates basic physical exercise with the process of playing a video game, and the end goal is to create a fun and enjoyable way to work out while gaming in order to inspire gamers or other casual groups to fit more physical activity into their day-to-day routines to ultimately improve the general health of society. From these ideas, a logical first plan would be to utilize exercise equipment to control the game itself. Using an exercise-bike style system, the idea being proposed here is to connect the bike to the game in such a way that pedaling and steering motions from the user are tied to the standard control systems of accelerating and steering in Mario Kart. Additionally, feedback from the game itself, such as race speed choice or the act of being struck by an item or obstacle, will be used to affect the resistance against pedaling in varied ways. This is the basic plan for how the whole system will work, and in the following sections, this document will describe in more detail the specifications, constraints, and background behind this system as well as who this team consists of, our skills and resources, and the timeframe in which the development should take place.

C. Problem Formulation

i) Objective

One of the biggest cons of gaming is a lack of physical activity. Most games are played in a seated position with no activity besides small wrist and finger movements. Our objective is to merge exercise with gaming in an enjoyable and practical way, and our approach is through a piece of equipment the majority of people can operate: a stationary bike. To us, the most logical game to pair with such a system is a game involving vehicles. We've chosen Mario Kart because it is a well-known and loved game that all ages can enjoy.

ii) Specifications

The end goal is to build a system that allows the player to immerse themselves in the Mario Kart experience. Therefore, many modifications must be made to meet this goal. Physically, the system must support a wide range of ages and sizes. It must also be stable and comfortable for long gaming sessions. The system must be adjustable in order to provide the rider with varying levels of pedaling difficulty. The handlebars must be able to freely move left or right to provide steering, potentially with some resistance for realism.

Electronically, the system must be able to communicate with either an existing gaming system or a game emulator on a PC. The action of pedalling the bike should tell the game's character to "go" and a rate of pedaling under a certain threshold should tell the character to "stop". Potentially, the resistance of the pedaling could be adjusted electronically for a more seamless experience. Situations like difficulty selection and "crashing" should cause the bike to be harder to pedal. The handlebars should also be equipped with appropriate buttons for actions such as drifting, shooting projectiles/using items, and selecting menu items. If needed, the system should draw power either from batteries or from a standard 120V electrical outlet. Ideally, the system should also apply some sort of regenerative power option if possible. This, combined with low power components, will keep the whole thing as power efficient as achievable.

Safety wise, the system should be calibrated so that there are no sudden changes in resistance that could potentially cause injury. Instead, some ideal gradual rate of change would be the best choice and can be determined through consistent testing. Additionally, there should be some fail-safe in place for the pedal system in case the resistance suddenly gives out, again in the interest of preventing injury. It should be robust enough to handle forceful pedaling while still protecting the user from quick changes in difficulty.

D. Constraints

With the specifications and background in mind, there are constraints that must be followed to make the project successful. The system must be housed entirely within or on the stationary bike. No component may be located away from the bike assembly for a more minimalist look. Wires coming out of the bike should be kept to a minimum for both aesthetics and safety. Controls not related to driving or steering should be located on the handlebars for easy access to the user. The pedaling system must be able to withstand the torque produced by an average cyclist. Additionally, the resistance system must be able to handle rapid changes. As far as controls, accessing and modifying game code or memory while running may prove too difficult a process even through the emulator, so external control systems or alterations to already existing controllers may be the only feasible options. If necessary, the user may need to make the initial base resistance choice separately from race speed choice due to this issue. Since this bike would most likely be operated in a gym or other public setting, it would be powered ideally by a standard wall outlet, meaning the system needs to comply with the power rating and safety standards accordingly. Below is a list of known safety standards concerning the project.

a) Electromagnetics

- 299 - IEEE Standard Method for Measuring the Effectiveness of Electromagnetic Shielding Enclosures
- 299.1 - IEEE Standard Method for Measuring the Shielding Effectiveness of Enclosures and Boxes Having all Dimensions between 0.1 m and 2 m
- 1309 - IEEE Standard for Calibration of Electromagnetic Field Sensors and Probes, Excluding Antennas, from 9 kHz to 40 GHz
- C63.2 - American National Standard for Electromagnetic Noise and Field Strength Instrumentation, 10 Hz to 40 GHz Specifications

- C63.14 - American National Standard Dictionary of Electromagnetic Compatibility (EMC) including Electromagnetic Environmental Effects (E3)
- C95.1 - IEEE Standard for Safety Levels with Respect to Human Exposure to Electric, Magnetic, and Electromagnetic Fields, 0 Hz to 300 GHz

b) Electronics
- NESC HBK - National Electrical Safety Code (NESC) Handbook

c) Batteries
- 946-2020 - IEEE Recommended Practice for the Design of DC Power Systems for Stationary Applications

E. Salient Outcomes

The most important outcomes of the final version of this project include the functional control scheme, the basic resistance system, and the basic health system. These systems represent the core intentions of the project itself. The control system works perfectly well to play a race in-game. Steering, acceleration, drifting, and item throwing are all present and work great, allowing for an entirely plausible simulation of a controller. The resistance system is subtle, but effective at forcing the user to pedal a bit harder in certain stages. This in addition to the base resistance introduced by connecting the motor to the flywheel driven by the bike tire creates a solid workout that isn't too difficult to play through multiple races with. Additionally, it has been noted that most people hardly notice the exercise during gameplay due to being distracted by the enjoyment of the game, only mentioning the burn afterwards. This was one of the major goals of the project in general. Lastly, the MET health system is simple to operate and informative, making the process of calorie burn estimation easy for the user to gauge if desired.

F. Report Structure

The structure of this report contains a review of the literature surrounding the topic of exercise bikes and their extended use in the gaming setting, the method of which the group has approached the problem outlined in the introduction in both the design and build choices, the qualitative and quantitative experimental results of our final version of the project after each subsystem was implemented, the timeline of progress and final cost of the actually construction and assembly of the design, all ethics considered during the project and all lessons learned after the fact, conclusions including possible future work, and lastly and appendix containing personal notes of contribution and skills acquired during the project all in that order. Each section is labeled accordingly. Any more information regarding specific details of the project can be found on the official Github repository.

II. LITERATURE REVIEW

Before addressing the MarioKart bike, it is necessary to explore what kinds of stationary bikes are on the market. Generally, there exists three types: upright, recumbent, and indoor cycles. Upright bikes hold the user in a vertical position, similar to a bicycle. They typically have smaller footprints because the pedals are below the seat. Next is recumbent bikes, which keep the user in a fully-seated position. The pedals are directly in front of the seat, allowing the user to relax the rest of their body and focus on pedaling their legs. Finally there's indoor cycles, which closely resemble road bicycles. The handlebars are directly in front of the seat, meaning the user's back is nearly parallel to the floor. While this position may seem less comfortable, most users report that this is the most realistic riding position that stays comfortable through long sessions [3]. All of these bikes have adjustable seats and can support around 250-300 pounds, making any of them suitable for our requirements

The above bikes are all constructed in the same basic fashion. The user sits on the seat and turns the foot pedals, just like a normal bicycle. The pedals are attached to a chain or belt, which drives a large flywheel. The flywheel is where the resistance of the pedaling is typically adjusted. There are several ways to add resistance to the system, but the two most common ways are through friction or magnetics. Friction bikes have a contact pad attached to a rod that the user can adjust. Turning a knob will move the rod up or down, causing the pad to be pushed into or pulled away from the flywheel. The benefit to these is that the knob includes an emergency stop feature where the user can push the rod all the way in and the contact pad forces the flywheel to stop spinning. Magnetic bikes work very similarly, with one key difference: there is no contact with the flywheel. Instead of a contact pad, a series of magnets is attached to the adjustable rod. The closer the magnets get to the flywheel, the higher the resistance becomes. This is due to a phenomenon called eddy currents. When a non-magnetic metal comes in contact with a magnetic field, eddy currents are produced in the metal. These currents flow perpendicular to the magnetic field, creating a friction-like resistance between the magnet and the metal [4]. What's useful about this is that there is no actual contact or friction involved. This saves the materials from wear and greatly prolongs the lifespan of a magnetic bike. Magnetic resistance is highly adjustable and some bikes even come with an electromagnet that is electronically controlled, allowing the resistance to be adjusted by changing the strength of the magnet rather than its position. Either form of magnet resistance would be suitable for our purposes. One con of magnetic bikes is they are generally more expensive than other variants.

An uncommon way to add resistance to one of these stationary bikes is to attach the flywheel to a permanent magnet DC motor. A DC motor has two wires, positive and negative. By shorting/connecting these two wires, a force called back torque is created. Back torque is caused by the internal voltage created when the wires are shorted. When

back torque is present, the shaft becomes harder to spin [5]. To control when the two wires are shorted, some kind of switch is required. A switch that is easily controlled by a computer is called a transistor. Transistors are essentially voltage controlled switches. When a signal is sent to one, the switch is flipped. One of the most common kinds of transistors used in small-scale electronics is called a MOSFET (Metal Oxide Semiconductor Field Effect Transistor) [6]. MOSFETs have 3 terminals: the source, the drain, and the gate. Current travels from the source, through the MOSFET, and out of the drain. When a certain voltage is applied to the gate, the MOSFET allows the current to travel through. A MOSFET acts as an open when there is no gate voltage, and as a short when there is voltage at the gate. By varying the voltage at the gate, the level of resistance between the source and drain can be changed. Higher voltage at the gate means lower resistance. With some sort of microcontroller, the gate voltage could potentially be controlled by the game.

Another common feature - or lack thereof - of stationary bikes is immobile handlebars. Some bikes come with an additional arm workout in the form of two handles that move with the motion of the flywheel, like an elliptical stepper machine. However, this kind of motion is not what this project requires. We need bicycle-like steering. With any stationary bike selection, modification would be required to allow for free steering. Most stationary bikes come with bent or vertical handlebars, which fortunately would provide our users with a more realistic riding experience.

To get the bike to communicate with the game, it needs switches, buttons, and sensors. Most of the hardware required comes conveniently packaged in video game controllers. Buttons to scroll through menus and select options, use items, and drive the Kart are readily available in these controllers. Additionally, these buttons are designed to be pressed often and be responsive since they are for gaming. To control the game, either an electromechanical or a computerized approach will be taken. In either case, a sensor to measure the speed or frequency of the rotating pedals will be needed. A tachometer is a sensor designed to measure the RPM (rotations per minute) of rotating objects. For this project, an optical tachometer would be suitable. These operate by shining a light, like an LED, against the rotating object. The light reflects off the surface back into the sensor, triggering a signal. The rate at which the signal is triggered is used to determine the RPM of the object [7].

Regarding the running of the game itself, it may prove difficult to use an actual console given how securely and privately Nintendo handles its proprietary systems. Therefore, for the purposes of this project, an emulator will be used in order to allow us to run the game client on a PC, which will give us greater options regarding interactions between the control system and the game data. An emulator is generally a piece of software that allows one computer to simulate the processes and conditions of another virtually identically. This allows the host machine to access programs and features that are typically only available to the system being emulated [8]. An emulator will allow more versatility to the type of controls available for usage in this project, as well as increased performance from the game itself as compared to the original console. Most emulation software designed for games provide the option to create a custom map of controls based on inputs received from the control system. This makes it quicker and easier to test and replace control options than that of a dedicated console. For the purposes of this project, the N64 version of Mario Kart will likely be used due to its relative simplicity and ease of emulation.

III. STANDARDS CONSIDERED

a) Electromagnetics

- 299 - IEEE Standard Method for Measuring the Effectiveness of Electromagnetic Shielding Enclosures
- 299.1 - IEEE Standard Method for Measuring the Shielding Effectiveness of Enclosures and Boxes Having all Dimensions between 0.1 m and 2 m
- 1309 - IEEE Standard for Calibration of Electromagnetic Field Sensors and Probes, Excluding Antennas, from 9 kHz to 40 GHz
- C63.2 - American National Standard for Electromagnetic Noise and Field Strength Instrumentation, 10 Hz to 40 GHz Specifications
- C63.14 - American National Standard Dictionary of Electromagnetic Compatibility (EMC) including Electromagnetic Environmental Effects (E3)
- C95.1 - IEEE Standard for Safety Levels with Respect to Human Exposure to Electric, Magnetic, and Electromagnetic Fields, 0 Hz to 300 GHz

b) Electronics

- NESC HBK - National Electrical Safety Code (NESC) Handbook
- IEC 60529 - Electrical enclosure protection

c) Batteries

- 946-2020 - IEEE Recommended Practice for the Design of DC Power Systems for Stationary Applications

IV. METHODOLOGY

i. INTRODUCTION

The MarioKart Exercise bike system is a device with the ultimate goal of offering an engaging and practical way to combine the benefits of physical activity with the fun and excitement of playing a video game. In order to achieve such a goal, a design that can communicate the user's motion into utilizable control data is required. It must also offer a similarly fluid experience in controlling the game as one would find with a typical game controller. To be viable for the average consumer, the design must be modular enough to be applicable to a standard bicycle design and non-invasive enough to the original frame to be easily

assemblable. Above all, it should also maintain the safety and comfort of the user during operation as well.

The system has been broken down within this document into several functional subsystems, each with a specific purpose relevant to the goals previously outlined. The diagram below represents the conceptual structure of the project as a whole.

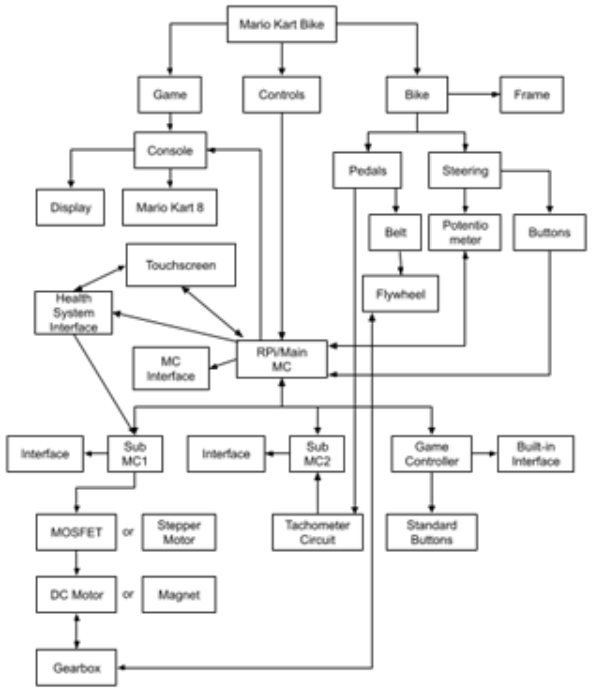

Fig. 1. Block Diagram of Mario Kart Bike System

As shown in the block diagram, the system will ideally have two versions: a DC Motor resistance system and a traditional magnetic resistance system. The rest of the subsystems on these bike designs will be exactly the same, with the only experimental variable being the method of resistance generation. A conceptual diagram with the location of each general connection between subsystems is shown below.

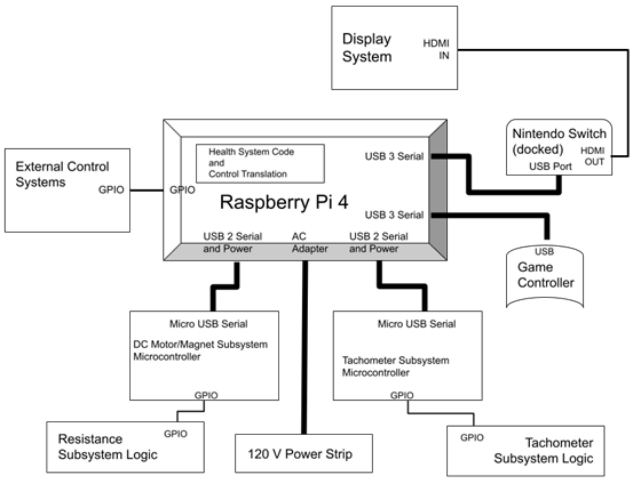

Fig. 2. Conceptual System Diagram

ii. GAME SYSTEM

A. PC Subsystem

i) Running the Game

In order to run the game software itself, the project requires either a video game console or some form of emulation (i.e. running the game through emulation software on a personal computer). The decision was made to use the Nintendo Switch console to run the game, due to potential copyright issues pertaining to emulation. Additionally, the Nintendo Switch provides access to the latest version of MarioKart, which will allow users to enjoy the most up-to-date experience of the franchise.

ii) Displaying the Game

In terms of video and audio output, the Nintendo Switch console that is running the game will be connected to two PC Monitors via an HDMI splitter. This will allow two players to view the same game session on two different bikes with separate monitors. Therefore, the players will be able to play together in real time as they normally would in-game. In addition, there will be a touchscreen display that serves as both a monitor for the health system and a selection menu for certain game and bike functions. The monitors and HDMI cables have already been freely obtained. The only component on the bill of materials for this system will be the splitter.

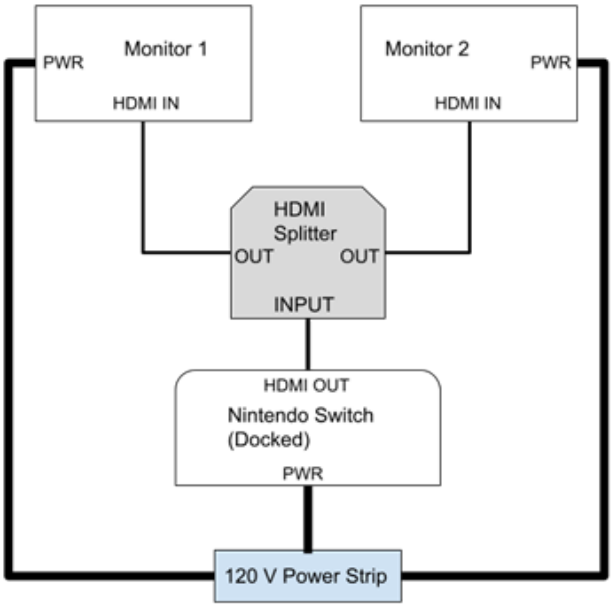

Fig. 3. Display Schematic

iii. CONTROLS SYSTEM

The purpose of the controls system is to provide a means for the player to use their bicycle to play MarioKart. The system must interpret various mechanical inputs, like pedaling and steering, while also providing feedback to the player in the form of dynamic pedaling resistance. It must also contain all of the necessary control options to cover each in-game action like a standard game controller would. The system must also be modular for ease of replacing components.

A. Main Microcontroller

The main microcontroller in this design needs to cover both the control of data to and from the subsystems and the translation of signals into usable control data. Additionally, certain peripheral components for other game control purposes will be connected directly to the main microcontroller and interpreted according to their associated function (e.g. initial resistance selection). These desired functions necessitate ample GPIO and serial data connections, as well as the ability to power said subsystems and components. In order to keep the design as compact and modular as possible, each of these constraints should be met by a single microcontroller system controlling a series of sub-microcontrollers with less complex and more specific duties. Each GPIO pin can supply up to 16 mA of current, with a maximum possible draw of 30 mA. Each of the sensors we will be using require less than this amount for proper operation. The 5V power rails are able to supply power directly from the connected power source. The main microcontroller selected to fulfill the general needs of the design and serve as the overarching manager for each subsystem controller is the Raspberry Pi 4 Model B: 4 GB memory version.

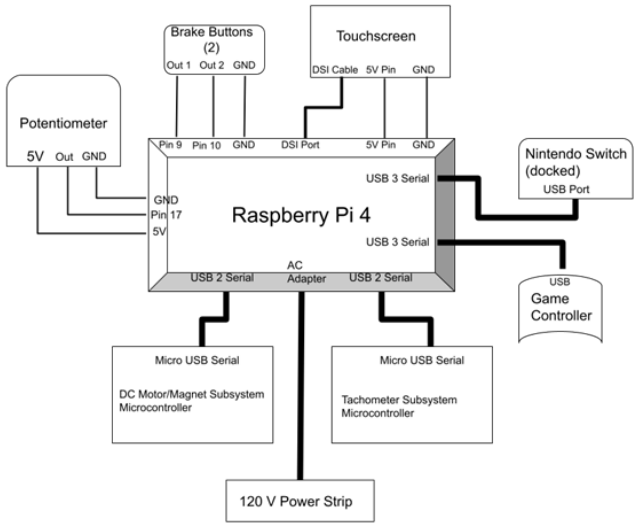

Fig 4. Raspberry Pi I/O Diagram

This version of Raspberry Pi (RPi) possesses two USB-2 and two USB-3 ports which will allow it to connect to both sub-microcontrollers through two separate data lines using mini-USB to USB Type A cables. This way, sensor information obtained from each subsystem and current game state information can be transmitted through serial data, and each sub-microcontroller can be powered from those same connections. Additionally, the RPi contains a 40 pin GPIO header with two 3.3 V and two 5.0 V power pins, 4 dedicated ground pins, and several custom function pins that will be used to manage both the external buttons and the touchscreen device which all require direct connection to the RPi. It also contains an AC adapter which allows it to be powered from a standard wall outlet.

Control translation will take place directly on the RPi as well. Because it will be receiving player activity data from each subsystem, the RPi's primary purpose is to decipher the incoming signals and turn them into functional control information to be sent to the Nintendo Switch console and used for gameplay. There are several open-source, RPi-compatible libraries in existence that aid in the conversion of external data into control data (such as "NSGadget_Pi" which uses an RPi and other external electronic devices to impersonate a Nintendo Switch controller), and the RPi community has a large resource base of other programs with very similar functions that will help simplify the control operations as well.

B. Resistance Subsystem

i) Sub-microcontroller

The sub-microcontroller selected for both the DC motor and the magnetic resistance system is the Arduino Nano. This microcontroller contains two true analog outputs which are capable of outputting a voltage ranging from 0 to 5 V. Additionally, it contains 8 dedicated analog inputs that will be used to obtain feedback voltage and current values from the DC motor circuit. The fidelity of the digital-to-analog converter can be set to either 8 or 10 bits within the software. At the maximum fidelity of 10 bits, the user can adjust between 2^10 or 1024 partitions of voltage. Each partition modifies the voltage by approximately 4.883 mV. Though the analog input from the hall sensor will only reach approximately 0.7 V, an amplifier will be used to step this range up to 4 V, which leaves 819 partitions worth of fidelity. This should give the board an adequate degree of sensitivity for controlling the DC motor resistance system precisely enough to simulate a dynamic range of resistance settings that is both realistic and challenging to the user.

ii) DC Motor Resistance

The DC motor will serve as the main form of resistance for one of the bikes. Before choosing a motor to fit in this system, it's important to know what kind of load it will be receiving. Based on research, it has been found that the average person pedals a bike at 50-90 RPM with an applied torque of 30 ft-lbs. Since DC motors are built for the opposite (high speed and low torque), a gearbox will be needed to transfer the input from the pedals into what the motor can handle.

Using AutomationDirect's website, the team has found a gearbox and motor combo that will meet the required specifications.. The gearbox is a 20:1 speed reducer, but if it is run from the output shaft, it becomes a 1:20 speed increaser. The output shaft can handle 345 in-lbs (28.8 ft-lbs) at 88 RPM. Although 28.8 ft-lbs is slightly less than 30 ft-lbs, this gearbox will work because it's rated for that torque at the higher end of a cyclist's RPM. As a cyclist increases their RPM, their torque will decrease.

The chosen motor is a 0.33 HP permanent magnet direct current (PMDC) motor that is rated to run at 1800 RPM. Since a 1:20 ratio was chosen for the gearbox, if the rider of the bike pedals at 90 RPM (an average person's maximum effort) then they will match the rated RPM of the motor. The motor runs at 180 V with a peak current of 1.75 A, meaning the maximum power is around 315 W. The motor is sold in a 90 V variant as well, but the 180 V model was chosen because it has a lower peak current with the same power output. Average to healthy cyclists can produce anywhere from 100-300 W when pedaling, so this motor is rated to handle whoever drives it.

The reason a PMDC motor was chosen is because they have a constant current (at high RPM) and a variable voltage when used as a generator. The voltage has a linear relationship with the input RPM, while the current remains virtually constant once a certain RPM is reached. The speed increase from the gearbox ensures that the current will be in that constant range. The motor and gearbox are represented visually in the DC Motor Control section below.

iii) DC Motor Control

The purpose of the circuitry around the DC motor is first of all, to control the resistance across the positive and negative terminals, and second of all, to measure the voltage and current being generated by the motor during operation. In short, the resistance across the terminals will be controlled by an n-channel MOSFET, the voltage will be measured by an op amp configured as a diff amp, and the current will be measured by a hall sensor.

The point of measuring the voltage and current at all times is so that the performance of the MOSFET can be monitored. As the MOSFET is under operation, it will heat up. As the MOSFET heats up, it will become less conductive. To mend that, it will need a higher gate voltage. By monitoring the voltage and current being generated by the motor, the magnitude of that adjustment can be determined.

The MOSFET will serve as the main control of the connection between the positive and negative leads of the motor. Transistors in general are typically known as switching devices, where moving from a low state to a high state is the top priority. But for this use case, the MOSFET will be operated in its linear or ohmic region. Between the transition from low to high, there is a range of gate voltages that keep the MOSFET in a partially open/closed state. The goal here is to control the MOSFET with a microcontroller's analog output pin. That means the range of gate voltages for the ohmic region should be less than 5 V. The motor will be generating up to 180 V and 1.75 A during peak operation, so the MOSFET should be rated for such high parameters. The chosen MOSFET is rated for well over this use case, and the ohmic region is between a gate voltage of 3-5 V. The specific package being used is the TO 247.

The protection diode included in the circuit is meant to be a safe-guard. Under normal operation, the current will be coming out of the positive terminal of the motor. However, if the rider were to pedal the bike backwards, the polarity of the motor would be flipped. That means the current would flow in the other direction. To prevent damage to other components in the circuit (mainly the microcontroller), a diode needs to be added. If the current comes out of the positive terminal like it should be, then the diode will see no current. If the current comes out of the other terminal of the motor, then the diode will be fully conductive and keep all of the current in a loop with the motor. The chosen diode is rated for 400 V and 2 A with a forward voltage of 1.1 V. The MOSFET and diode make up the bare minimum requirements for controlling the DC motor. They are represented in Figure 5.

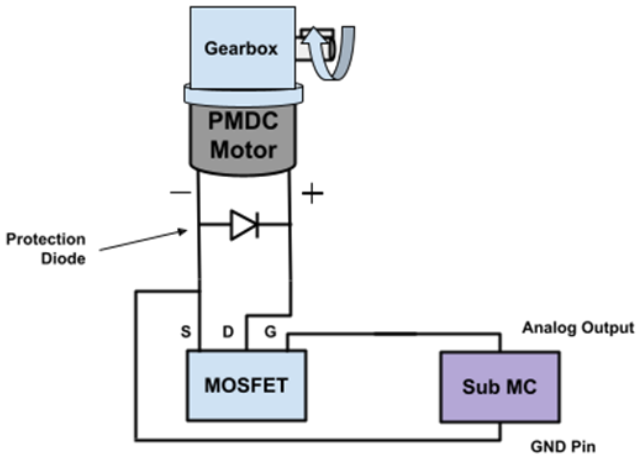

Fig. 5. MOSFET and Diode Circuit

The sub-microcontroller being used to control the MOSFET and collect the measurements of voltage and current is the Arduino Nano. This board has two true analog 5V output pins and several analog to digital input pins. The analog-to-digital (a2d) conversion happens on the board itself, and it has a configurable resolution. In the code, the programmer can set the fidelity to 8 or 10 bits. The team has chosen to use a 10 bit fidelity. 10 bits means 1,024 individual partitions. The pins can read signals up to 5V, which means there are 5/1,024 = 4.883 mV per partition. That's also 204.8 partitions per one volt. This will be a useful consideration because if the measurement devices can output a signal close to 5V, they can take advantage of the full resolution.

To measure voltage, an op amp will be connected across the positive and negative terminals of the motor. However, the 180 V being generated is too dangerous for most op amps and for the microcontroller. Therefore, the op amp needs to have a resistor network surrounding it that both protects its own inputs and configures the op amp into a differential amp. A network of four resistors can accomplish both tasks. By connecting the op amp in a negative feedback loop, it can be configured to have a gain less than one. That will be very important in stepping down the motor's 180 V to 5 V or less so that the microcontroller can understand the measurement and be kept safe. Aiming for 5 V or less also ensures that the op amp will be taking advantage of the full resolution of 1024 partitions. It is also important that the inverting and non-inverting terminals of the op amp don't see too much voltage or current. With some circuit analysis, these specs can be met. By setting R1=R2 and R3=R4, the process becomes even simpler. Through nodal analysis, the following equations were generated:

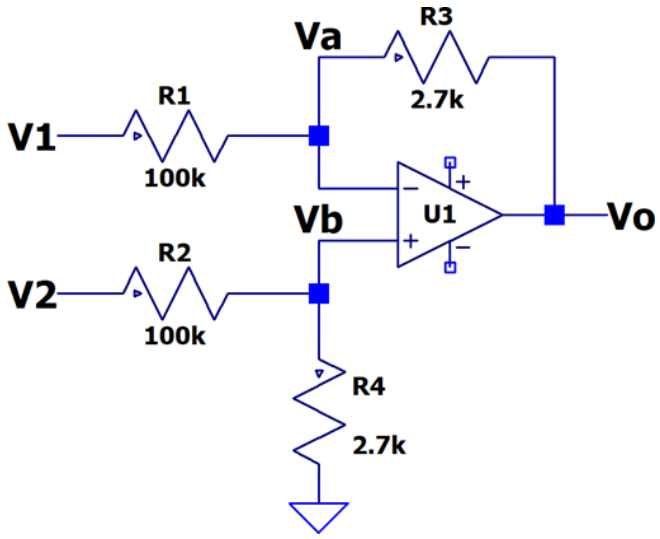

Fig. 6. Op Amp Voltage Measurement Circuit

$$V_a = \frac{V_o}{\frac{R_3}{R_1}+1} \quad (1)$$

$$V_a = \frac{V_2}{\frac{R_2}{R_4}+1} \quad (2)$$

$$V_o = \frac{R_3}{R_1}\left(V_2 - V_1\right) \quad (3)$$

$$I_1 = \frac{V_1 - V_a}{R_1} \quad (4)$$

$$I_2 = \frac{V_2 - V_b}{R_2} \quad (5)$$

The measurement will be taken by connecting the positive terminal of the motor to V2 and the negative to V1. The most voltage V2 will see is 180 V, and the least V1 will see is 0 V. Vo needs to be 5 V or less, meaning the ratio of R3 and R1 can be solved for. It comes out to 0.0278. Additionally, in order to keep the current low, all of the resistors need to be in at least the kiloOhm range. I've chosen to make R3 and R4 2.7 kOhms and R1 and R2 100 kOhms. With those chosen, that means:

$$V_a = 4.869\,V \quad (6)$$

$$V_b = 4.732\,V \quad (7)$$

$$I_1 = 48.69\,\mu A \quad (8)$$

$$I_2 = 1.753\,mA \quad (9)$$

With those specs in mind, an op amp can be chosen. The team has picked the LM 741 op amp, which can handle up to 30 V at each terminal.

To measure current, the team has chosen to use a hall effect sensor. It will have to be able to measure up to 1.75 A and potentially handle high voltage. The chosen chip is meant to have the full load of current injected into it. It is built with a layer of insulation between the input pins and the rest of the internal hardware. The hall effect takes place

through this layer of insulation. Only the magnetic field will travel through the insulation. Then, the chip will output a voltage proportional to the current it is receiving. The chip will output 400 mV per amp it reads. This will leave the output voltage well under 5 V for the microcontroller. It is an 8 pin chip and pins 1-4 act as the IN and OUTs for the current to travel through. The purpose of including 2 IN pins and 2 OUT pins is so that the current can be split and a smaller trace can be used on the PCB the chip is mounted to. In particular, the A4 model should be used because it can read a minimum of 125 mA and a maximum of 12 A.

Since the chip outputs 400 mV per amp it measures, it will output a peak of 700 mV because the peak current of the motor is 1.75 A. To ensure that the resolution of the measurement is usable, the output of the hall sensor needs to be amplified. Since the resolution of the arduino's a2d conversion is 204.8 partitions per volt, if the signal is left unaltered, only 143.4 partitions will be available. The advertised peak current is 1.75 A, but just to be safe, it should be assumed that it could get higher. That means the chip's 0.7 V max should be amplified to around 4 V for safety. To accomplish this, another LM741 amplifier circuit will be implemented. The op amp will be configured as a non-inverting amplifier, which looks like this:

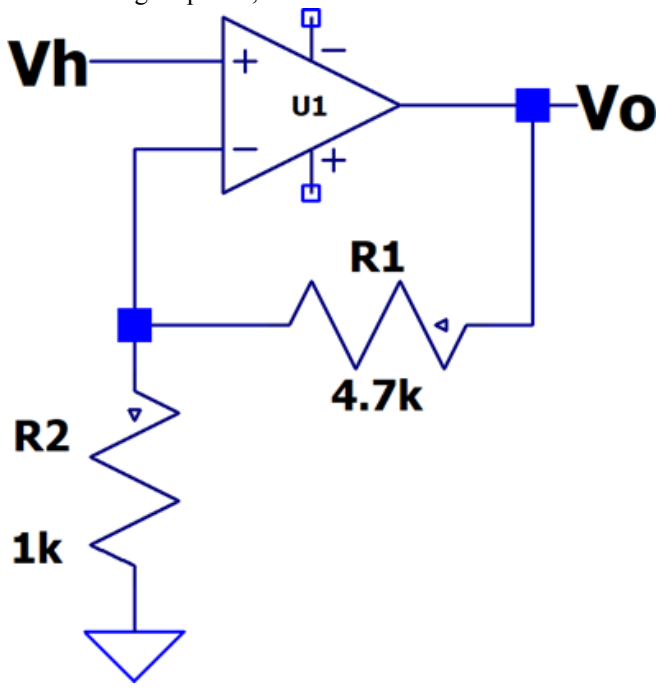

Fig. 7. Hall Sensor Preamp Circuit

$$V_O = V_h\left(\frac{R_1}{R_2} + 1\right) \quad (10)$$

$$\frac{R_1}{R_2} = \frac{V_O}{V_h} - 1 \quad (11)$$

$$I_O = \frac{V_O - V_h}{R_1} \quad (12)$$

Vh is the signal from the hall sensor, and Vo will be what is sent to the microcontroller. Vh has a typical maximum of 0.7 V, and to be safe, the op amp should output 4 V. That means the ratio of R1 and R2 should equal 4.714. If the maximum output is around 4 V, that means the resolution is 819.2 partitions. That's 80% of the maximum resolution.

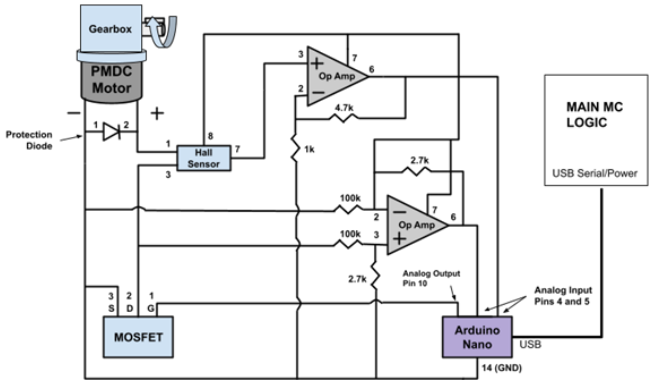

Fig. 8. Complete DC Motor Resistance Control Circuit

With all the components combined, the final circuit is shown above.

iv) PCB Design

The PCB must serve as a solid structure to mount the components of the circuit to and act as a conducting path for the appropriate connections. Sections of trace that are directly connected to the motor must be at least 40 mil in order to handle the max current of 1.75 Amps. There must be solder holes in places where a wire must be connected to the board. No connectors are necessary because the connections are meant to be permanent. The diagrams below show the layout of the PCB and highlight the trace connections.

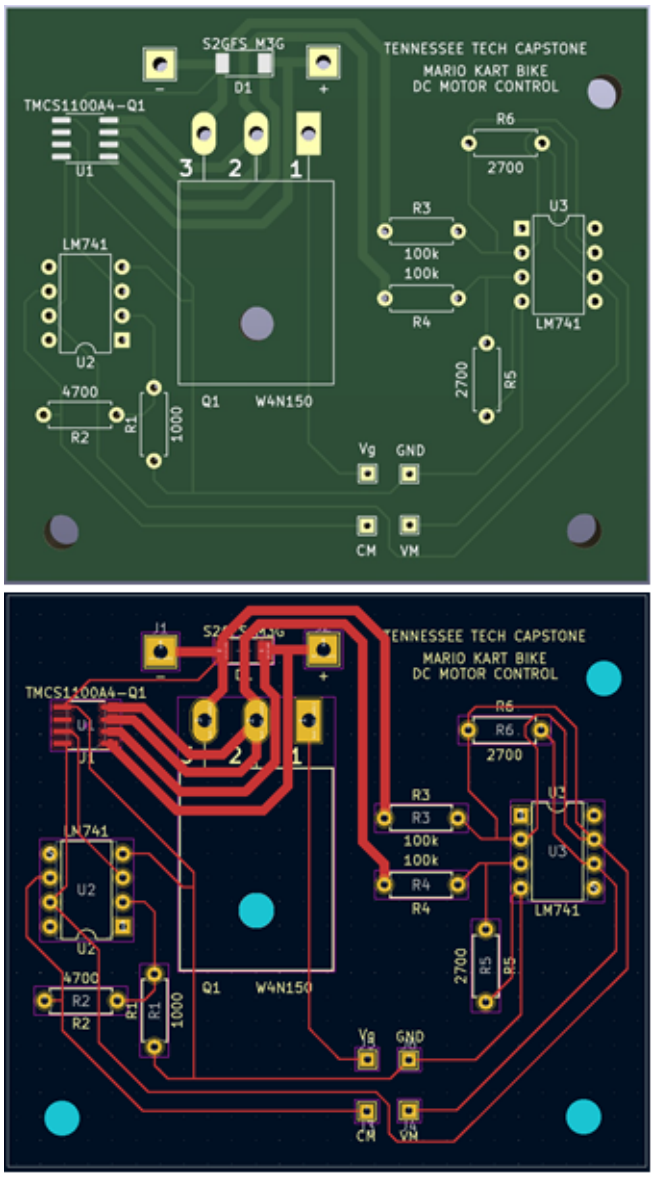

Fig. 9. PCB Design for the DC Motor Circuit

v) Magnetic Resistance

The magnetic system will serve as the main form of resistance for a second bike setup. The magnet must be electronically adjustable such that the distance between it and a flywheel can be modulated, and it must be able to provide resistance similar to what would be seen on various terrain inclines outdoors.

The most efficient method determined for use in the current bike design is to utilize a magnetic resistance component that is shipped along with the rear wheel frame, which will be shown in more detail later in the report. As for the magnetic component, it is essentially a magnet and flywheel working in tandem to provide resistance. It operates using eddy currents, the same principle used in designing auto-belay systems for climbers and repellers. There is a cable attached to the magnet that, when tensioned, moves the magnet closer to the flywheel. The closer these two get to each other, the greater the resistance.

vi) Magnet Control

The controller for the magnet must be able to electronically adjust the position of the magnet based on player preference and in-game factors. The current plan is to use a stepper motor to induce tension on the cable rather than the manual adjustment it ships with.

C. Tachometer Subsystem

i) Sub-microcontroller

The sub-microcontroller selected for the tachometer subsystem is also the Arduino Nano. This microcontroller is required to convert the signals from the tachometer component into RPM in order for that information to accurately serve as the acceleration control in-game. It also possesses the minimal GPIO needed to operate the tachometer through the power, ground, and data pins found on the board. Additionally, the acceleration signal will be pulse width modulated upon leaving the Nano, not by using the PWM pins, but rather through a calculation within the program to allow uniform communication through the USB serial data lines, in increasing frequency based proportionally on increasing RPM. This will allow for a smoother acceleration control experience that more closely resembles the real-life process of gradually gaining speed when accelerating on a bicycle.

ii) Tachometer Board

For the tachometer component, this design necessitates the ability to optically measure the rotational speed of either the pedaling from the bike user or the rotation of some other mechanism on the bike. It also needs to return some sort of signal indicating the measured RPM or otherwise a signal that can be interpreted as an RPM value through some calculation made outside of or on the selected device..

The component that will serve as the optical tachometer designated for use in this subsystem is the KY-032 infrared obstacle avoidance sensor. This component contains an IR LED and an IR-sensitive photodiode that is capable of detecting reflected light from the LED off of some sort of reflective surface. The output signal returned by the device can follow both TTL and CMOS logic, and it sends a low level if an obstacle is being detected or a high level if no obstacle is being detected. The device contains two potentiometers, one that controls the flashing frequency of the LED through the NE555 timing chip, and the other that controls the brightness of the LED which allows for indirect control of the distance of detection by the photodiode. The board contains 4 pins, one for ground, one for power, one for the logical output of the system, and one for enabling the

device. The enable pin is unused as long as the jumper on the board is kept in place, therefore it is left open. The control circuit for this subsystem is shown below.

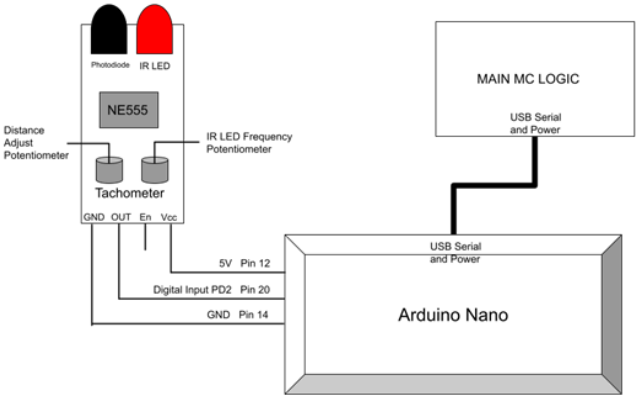

Fig. 10. Tachometer Control Circuit

The device's original intended function by the designer is to detect the IR light that the LED emits being reflected off of an object and subsequently signaling that an object has been detected in front of the device with a low logic signal. However, the true purpose of this device in this design is for use as an optical tachometer to measure the RPM of a spinning gear. This can be achieved by connecting it to the Arduino Nano and feeding it the detection data. Although this device is typically used to simply detect when an object is in front of it, the Arduino can be programmed to calculate RPM from the measurement. As previously stated, the component returns a logical low (~0 V) when the photodiode picks up reflected light from an object in front of it, and for a spinning object such as the pedals of the bike or the flywheel, it detects this signal multiple times with each pass of the object. The RPM can be determined from this by a simple calculation:

RPM = ((# of events * time interval of measurement) / (# of objects in rotation))

"# of events" is how many times the component has signaled a detected reflection, "time interval of measurement" is how many times a minute the RPM has been sampled, and "# of objects in rotation" represents the number of objects that pass in front of the detector that are part of the same spinning apparatus (e.g. there 2 pedals on the bike, but only one spinning apparatus).

This component can be directly powered and controlled by the Arduino Nano with it's digital GPIO pins and 5 Volt power pins, and it can perform all of the necessary calculations within the microprocessor on-board. Additionally, in order to achieve a more realistic acceleration control experience, the Nano can be used for pulse width modulation of the outgoing control signal in a way that causes gradual increase in speed rather than an all or nothing trigger.

The photodiode on the sensor has certain mechanisms to prevent signals that aren't intended for use from being picked up. This requires certain parameters to induce a desired logic signal that are outlined in the description below. The IR-sensitive photodiode used on this component is the HS0038B. It possesses a bandpass filter, an integrator stage, and automatic gain control to filter out noise or disturbances from the desired IR signal detection. In order to distinguish a true data signal from said noise/disturbances, the signal received must be of or near the carrier frequency of 38 KHz, which is possible to achieve from the flashing LED using the tuning potentiometer and an oscilloscope to monitor the exact frequency as it is changed. Additionally, the reflected signal must be received in bursts of 10 – 70 cycles for optimal performance with a non-reflective gap time in between of at least 14 cycles. The datasheet for the device does not give an exact rise/fall time for the logic signal found at the output pin, however, the times for each component of the device are available. The circuit schematic of the device is shown in Figure 11 [9].

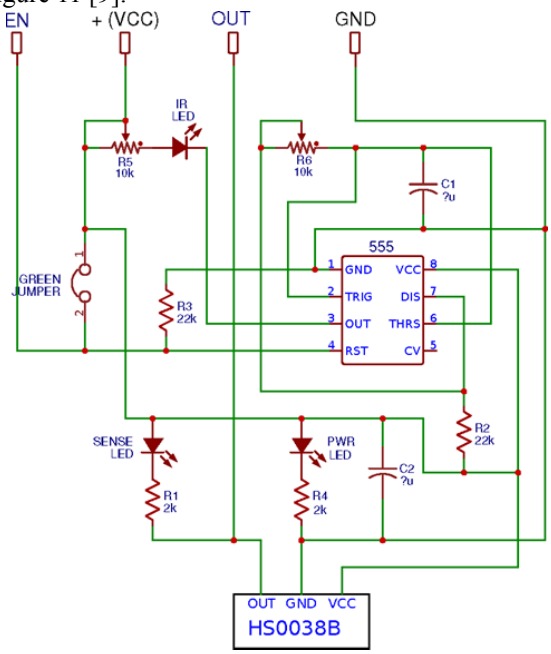

Fig. 11. Circuit Schematic of the Components on the KY-032 [9]

The only devices involved in the output logic signal are the HS0038B IR-Photodiode and the NE555 timing chip. The photodiode has a maximum rise/fall time of 100 nanoseconds, and the NE555 has a maximum rise/fall time of 300 nanoseconds, resulting in a total logic delay of 400 nanoseconds in a worst case scenario. At 38 KHz frequency, the LED will be flashing once every 26.316 microseconds. For 10 cycles to be achieved and including the initial fall time as the logic signal is set to low, the light must be reflected for at least 263.56 microseconds. The minimum gap time of 14 cycles will take up at least 368.824

microseconds of time, including the initial rise time as the logic signal is returned to high. Including these along with the dynamic response of the photodiode itself, this results in a minimum marginal total of 632.384 microseconds of operational time to achieve a sound logic signal. The object planned for use to measure RPM will be the gear of the bike itself with the holes and surface shrouded by a dark covering (black paper, tape, etc) and a white line on the covering to act as the reflective portion. The maximum estimated RPM for the gear when the bike is being used is approximately 90 RPM. This means there will be 1.5 rotations every second, or 0.6666 seconds per rotation. That equated to 666.66 milliseconds of time to work with in total for each rotation of the gear. Since only 632.384 microseconds (or 0.094%) of the rotation time is necessary for a functional logic signal, it will be simple to place a reflective strip of adequate width to both meet the minimum operational time needed and fit well onto the gear. Exact dimensions of the strip will be a minimum of 0.094% of the circumference of the gear in order to comply with this design. Figure 12 is a model of the design plan.

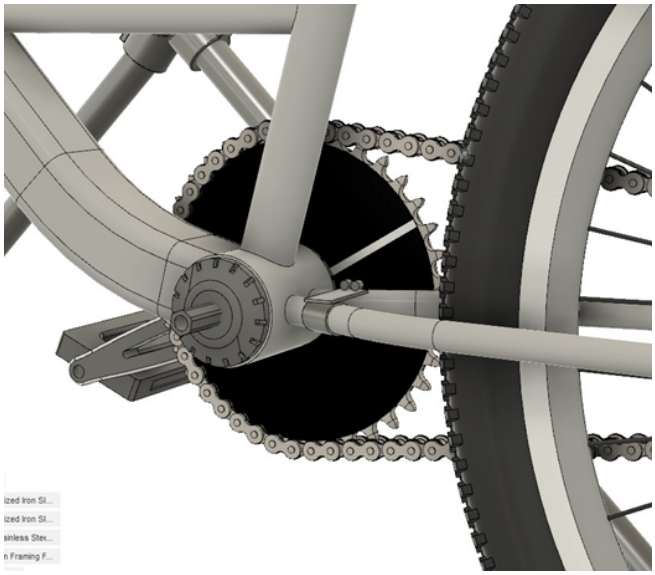

Fig. 12. Prototype CAD Model of the Tachometer Mounting Setup

The device is represented by the small gray plate and two cylinders in front of the pedal mount and is facing the gear. The gear has been shrouded with a black covering, and a white line, which will serve as the reflective patch for achieving the correct cycle count, has been placed on it. As the user pedals and the gear rotates, the white stripe will periodically pass in front of the device, causing the IR light to be reflected. It will be reflected for at least 10 – 70 cycles of flashing depending on the user's speed. This reflection will trigger the logic level of the device to low, and then a counter variable within the Arduino program controlling it will increase. Once the white stripe is passed completely and at least 14 more cycles of flashing have passed, the light will no longer be reflected, and the signal will return to a logical high. This process will repeat until a certain amount of time has passed, and then the rpm calculation will be run with the result being sent to the RPi. This will continue again in its own cycle throughout the use of the bike.

D. Steering and External Controls

i) Directional Control

The control system design necessitates some way to convert the steering motion from the user's physical movements on the bike into directional movement and control data that can be used to correspondingly control the character in-game. The component that will serve this purpose must have a reasonable similarity to the steering function of a standard game controller. It must also be able to rotate with the bike at least 90 degrees in both directions for left and right turning. In order to obtain a precise and dynamic measurement of the steering direction, this design will include a potentiometer component that captures degrees of rotational motion from the user and returns that information to the Raspberry Pi to be interpreted as control data. The component selected for this system is a STEMMA 10k Ohm Rotary Potentiometer.

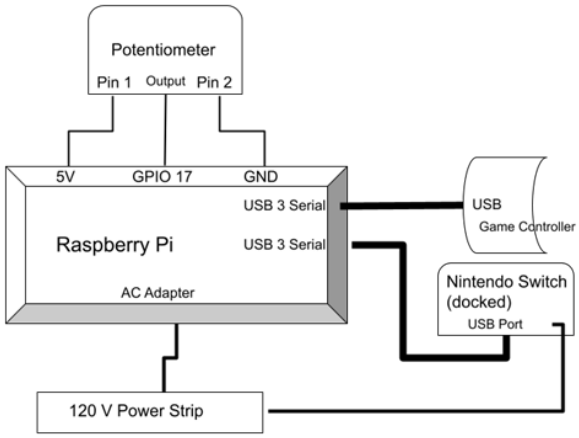

Fig. 13. Steering Component Schematic

This component functions like a standard potentiometer (i.e. variable resistance) where the turning of the dial increases or decreases the amount of current passing through it. The dial has a total rotational range of 270 degrees and will be mounted so that the midpoint at 135 degrees aligns with the center of the bike. Although the user would typically be turning no more than 90 degrees in each direction, the extra room will prevent accidental overturning from breaking the component. This potentiometer component is already contained on a board with connecting wires for each pin. It contains 3 total pins, one for power,

one for ground, and the last one for the output signal. The power and ground signals can be directly connected to a 5V and ground pin on the RPi, and the output will go to an analog input GPIO pin. The analog input on the RPi has 10 bits of fidelity, which provides 1024 partitions, similar to the Arduino Nano. Since the user will only be turning 180 degrees total for gameplay (or ⅔ of the total arc), there are only 682 partitions that can be utilized. So, each partition represents 0.26 degrees of turning. This is comparable to the turning fidelity of the accelerometer on a Nintendo Switch controller. The partitions will be split in half, with 341 designated for each direction. The further to each side the user turns, the more partitions are counted by the receiving analog input pin on the RPi. This signal can be translated as a correspondingly increasing or decreasing turn signal in-game. In terms of the physical properties of the translation of bike motion into control data, this design is outlined in detail in the steering portion of the bike system below.

ii) Handlebar Buttons

The MarioKart game has several in-game control options other than the basic steering and acceleration operations. Throwing items at other players and drifting are part of the player's arsenal, meaning that this design needs some external controls to provide users with the same options as a standard controller. The design will include the attachment of clamping aluminum alloy bike brake levers. The levers will be detachable such that the system remains modular and can be applied to a variety of bike designs. The chosen brake levers are actually meant for E-Bike brakes, so they have a built-in switch. Using a realistic braking system simplifies the drifting and item throwing actions in-game and makes them easy to access while controlling the rest of the bike system. The type of switch used in those E-Bike levers is a "normally open" switch (connected/closed when the button is pressed). More specifically, the buttons will be pressed when the user applies adequate pressure to the levers, causing the circuit to connect. The resulting signal will be sent to the Raspberry Pi as control data for the corresponding button function through digital GPIO pins as shown in the diagram.

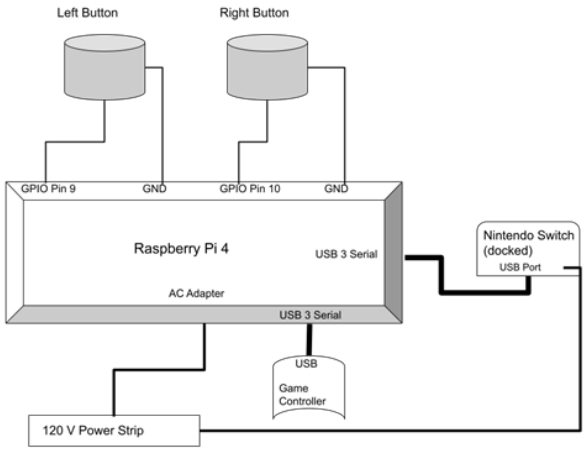

Fig. 14. Handlebar Buttons Schematic Diagram

The brake levers themselves look like any other standard bike brake lever, except they have two wires coming out of the back. The placement of these brake levers is shown in Fig. 16 on the bike CAD model.

iii) Touchscreen

An RPi-compatible 7 inch wide touchscreen tablet device will be employed as a secondary display. It will be connected through a dedicated ribbon cable directly to the RPi and will serve as the medium for customizable difficulty selection options as well as the method of interaction for the health monitoring system. The connections for this system are outlined in the diagram for the main microcontroller.

E. Game Controller Subsystem

The game controller will need to be present in order to preserve menu navigation for the user. The bike parts will only replace a couple of buttons on the controller, meaning that using the console otherwise would be impossible. The game controller is also likely going to be the communication device used to send the inputs to the console. The RPi libraries will send control data that is fed through the game controller and to the console. Therefore a standard console gamepad will be required.

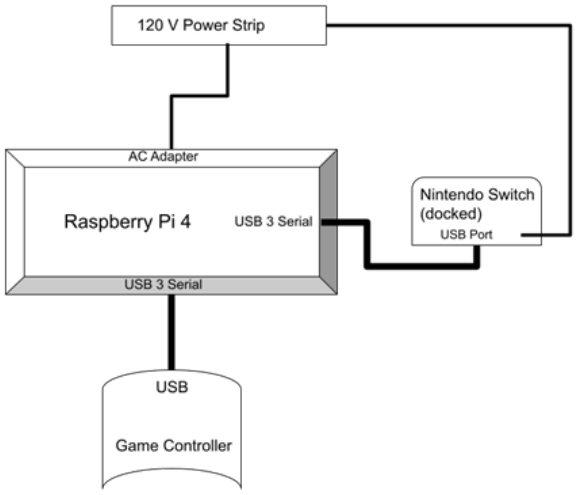

Fig. 15. Game Controller Schematic

F. Power

Much of the system will be powered through 5V pins on the various microcontrollers. The DC Motor and Magnet resistance circuits, tachometer board, steering component, handlebar buttons, and touch screen will all be powered this way as they require GPIO pin connection as the power source. The microcontrollers for each subsystem will be powered from the USB connections they share with the RPi. The console and monitors will be powered from a standard 120V wall outlet via a power strip located inside the microcontroller box.. The RPi will also be powered this way as it has a dedicated AC adapter on board. There are no batteries or external power circuits necessary for this design.

iv. BIKE SYSTEM

A. Pedals Subsystem

This subsystem must accomplish the task of translating the rotational movement generated by the rider pedaling the bike into a controllable resistance via the magnet or DC motor systems. The main constraint is that very little modification should be made to the bike so that the user may still use their bike on the road.

This subsystem is already taken care of for the magnet bike variant. The magnet/flywheel housing comes with a roller that rests against the rear tire and is also what transfers the resistance from the magnet to the tire.

For the DC motor bike, the team has designed a support that will hold the motor/gearbox assembly. This support will act as the means through which the mechanical motion of the pedals is transferred to the DC motor. The details of this support are explored later in the “Frame Subsystem” section.

B. Steering Subsystem

This subsystem is the other main input that the rider should be able to use to steer in-game. This subsystem must allow for full range of motion of the bike’s handlebars while also translating the rotational movement into game data. It must also be modular and removable for ease of use and easy installation. There are two components to this subsystem: the physical framework that allows the bike itself to turn properly as the user intends and the electronic component that will obtain that physical turning data and convert it into digital data to represent the steering control in-game. The digital component, already covered in the controls subsection, will of course be the potentiometer system. The physical frame component has its own set of constraints that will be met to maintain safety and functionality.

The first constraint of the physical frame will be met via the frame suspending the bike. This frame prevents the front tire from contacting the ground and still allows for full range of motion of the handlebars and will be detailed further in the frame subsystem section. The second constraint requires that the potentiometer be mounted onto the bike and connected to the turn shaft in such a way that the steering motions of the user translate to a corresponding physical manipulation of the encoder rod and thus a steering response by the in-game character. For this, a system of gears including a tall one mounted to the rod of the potentiometer meshed with a thinner one mounted to the turn shaft of the bike will be utilized. The gears will be placed such that the turning at the potentiometer rod is the result of the inverse of the user’s direction, but this will be corrected within the translation code. The potentiometer itself will be held by a clamp placed on the front, static portion of the bike frame that encompasses the turn shaft. This design prevents any manipulation of the potentiometer save for the rotating rod, and it keeps the entire apparatus out of the way of the user. A model of this setup is shown below.

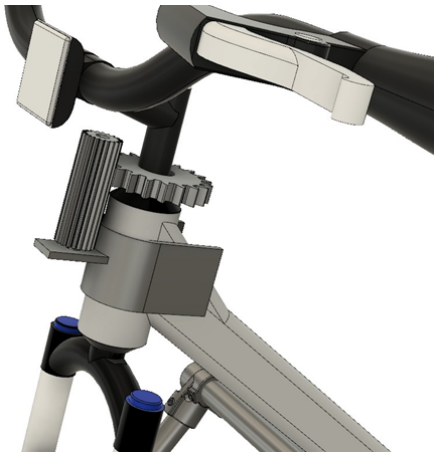

Fig. 16. Prototype CAD Model of the Potentiometer Mounting

C. Frame Subsystem

The frame of the bike must suspend the front and rear wheels off the floor. It must also hold the display that the game will be played on. The frame should be height-adjustable so that various kinds and sizes of bicycle may be attached. The frame should also be foldable or easily disassembled for storage. It should be rigid and wide so that the bike can't be tipped over from any angle. This is to ensure rider safety.

For the purpose of lifting the bike off the ground, the team has chosen to use a hybrid frame. The back wheel will be suspended by a bike trainer, and the rest of the bike will be held up by a frame designed by the team. The bike trainer serves the dual purpose of lifting the back wheel and providing resistance to it for the magnet bike variant. The chosen bike trainer comes with a manually adjustable magnetic roller, but the adjustment will be automated and controlled by a sub-microcontroller. The rest of the bicycle will be supported by a frame built in-house. The plan is to build it out of one inch steel pipe and connect the pipes using slip-on fittings. The display will be mounted to a vertical pipe at the front of the frame using a VESA pole mount. A concept model of this is shown in the next Figure.

Fig. 17. Prototype CAD Model of the Bicycle Supporting Frame

The bike trainer attaches using a special pin included in the packaging. This pin replaces the one on the bicycle and allows the bike to be mounted using a single clamp. The bike attaches to the pipe frame via a clamp that is screwed tight. The clamp is intended to allow bikes of various sizes and constructions to be mounted to the frame. It will be angle-adjustable and have a rubber interior for extra grip. The required fittings are a slip-on tee, two three-way 90° slip-on elbows, four 45° slip-on wyes, two 90° slip-on elbows, and a wye connector. The triangles on each side are to prevent the bike from falling forward or backward and side to side. The frame is wide enough to allow full range of motion for the front wheel to steer. The clamp is meant to be mounted in a position on the bike that doesn't obstruct the motion of the pedals either. The bicycle, bike trainer, pipe fittings, VESA mount, and display are models that the team acquired online. The bicycle was made by user Omkar Narkar on GrabCAD. The bike trainer was made by user Muhammad Furqan on GrabCAD. The display was made by user LeoDJ on GrabCAD. The pipe fittings and VESA mount were acquired from McMaster-Carr's website.

The motor support mentioned previously is meant to hold the motor/gearbox assembly in a way that allows it to act as a replacement resistance system for the bike trainer stand. The stand comes with a magnetic resistance system that presses against the rear tire. The idea here is that the motor support will mimic the attachment of the existing system. Below is a figure of the proposed motor support.

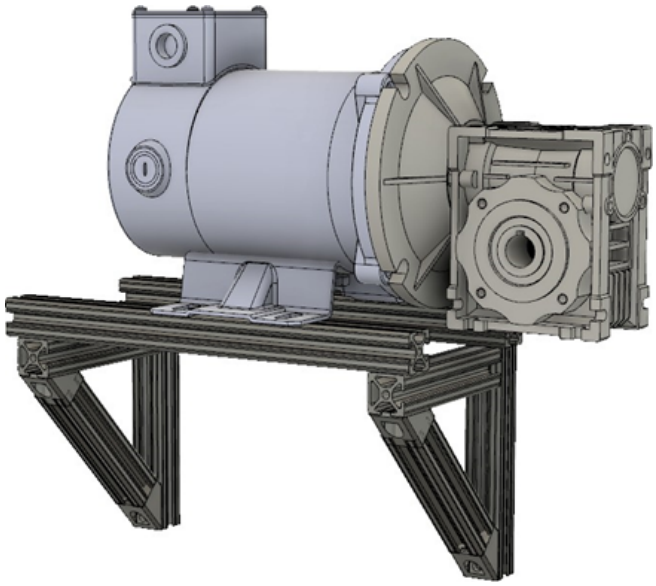

Fig. 18. Proposed DC Motor/Gearbox Support

The motor will mount to T-slotted framing rails that allow it to slide back and forth. This is needed so that the roller attached to the gearbox can be pressed against various tire sizes. The rails have two 45 degree braces underneath that will serve to support the weight of the motor. They will be attached to the rails using concealed brackets meant for fastening T-slotted framing rails together. This assembly will be attached to two mounting plates on the bike trainer stand. One plate has a vertical slot, and the other a horizontal slot. This allows for angle adjustment. The plates are pictured in Figure 19.

Fig. 19. Motor Support Mounting Plates

As for the mounting of all of the computing components to the bike frame, a box has been designed. The box houses the Raspberry Pi, two Arduino Nanos, the DC motor/magnet control circuit box, and the power strip. The team decided to put the control circuits in their own boxes due to the high voltage and current being used in the circuit. This smaller box will be velcro taped to the inside of the larger box for ease of removal. The power strip will also be velcro taped to the inner wall of the larger box. The two Nanos will be attached to the floor of the box using M1.6 standoffs and screws. The Raspberry Pi display has been integrated into the lid of the box, sitting perfectly flush. The Raspberry Pi mounts to the underside of the display. The walls of the box are talls enough so that the Raspberry Pi does not come into contact with the other components inside. The box has a large slot at the bottom for all of the cables to feed through. Below is a figure of the lid and the box with estimates of the component placement inside. The box will be attached to the front pole of the frame, just below the monitor, using zip ties or velcro straps. The box will be 3D printed as well.

Fig 20. Computing Control Box

v. HEALTH SYSTEM

A. Calculating Calories: Permanent Magnet Version

In this subsystem, the RPi 7" touchscreen will receive the input of the user, then send the data to the Raspberry Pi via a Display Serial Interface (DSI) connector to be calculated internally and displayed on the touchscreen after the user ends the session. For the inputs, the customer must enter their age and weight (lbs), and then press start. The start button is the condition for the function timeDuration() to be enabled. When the user presses the stop button, the amount of time elapsed.

(1) will be calculated in this function and then the value returned will be stored in a variable and passed to calculationMET() to calculate calories (2) with a preset value for the Metabolic Equivalent Task (MET). With the stop button as a condition, the display() function will cause both timeElapsed and calories to be transmitted via the DSI connector to the touchscreen display.

Equation for calculating duration:

(1) timeElapsed = stopTime - startTime

Equation for calculating calories with MET:

(2) calories = T*60*MET*3.5*W

Where the values T = time (hours), MET = Metabolic Equivalent of the chosen task (a preset value), and W = weight (kg, converted from pounds to kilograms).

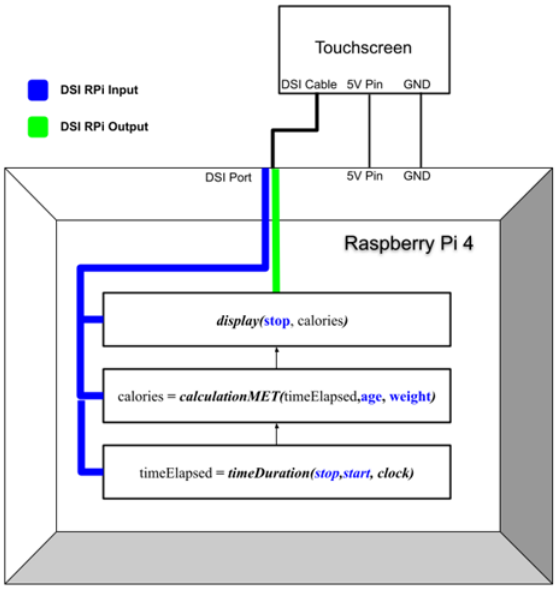

Fig. 21. Magnet Bike Health System

B. DC Motor Version

This subsystem should read the voltage and current from the Arduino Nano and use the values to calculate power and kilocalories, then output the time elapsed and kilocalories to the 7" display for the customer. The Nano has two analog values for inputs to the DC Motor control. The voltage has a range of 5 volts or 1024 in resolution for Analog to Digital Converter (ADC). The current has 4 volts, or the ADC resolution of 819. The following equation that can be used to convert from the ADC reading:

(1) ADC Resolution/System Voltage = ADC Reading/Measured Voltage

Since the given values would be the system voltage, ADC resolution, and the ADC reading, the measured voltage can be solved for. The same can apply for the current, which is represented by the system voltage of 4 volts.

Example Arduino Code to Demonstrate ADC Reading:

```
//Defining pin as input
pinMode(A0, INPUT);

//Read analog value of A0 and store in y
int y = analogRead(A0);

//Printing would show the value in ADC resolution
Serial.println(y)
```

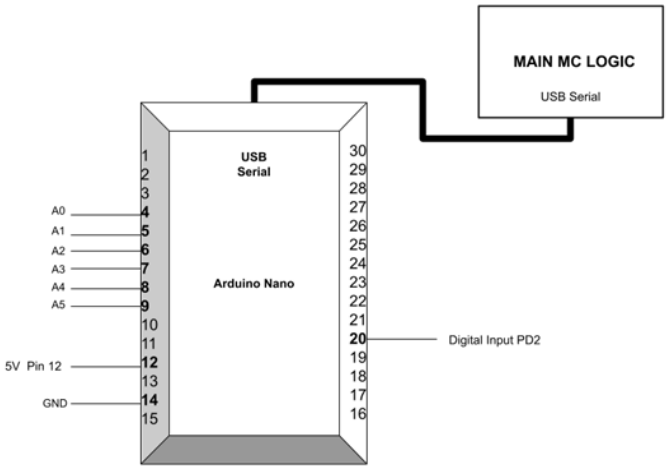

Fig 22. Motor Bike Health System Data Connections

In order to communicate with the Raspberry Pi (labeled as Main MC LOGIC), the Nano uses a serial USB port. There are libraries that help the Raspberry Pi identify the Nano and access its analog pins. In order to avoid a common issue of having to manually enter the Arduino's serial device name, there is an available "sketch" project for Arduino and a library for RPi that has APIs (Application Programming Interface) in C and Python. The link for this solution follows:

https://www.uugear.com/uugear-rpi-arduino-solution/

In addition to the serial USB communication, the user input for start and stop via the DSI of the 7" RPi touchscreen will be input to the system. Once the user presses stop, the RPi function caloriesPower() should use the time computed from a PLL clock, the time elapsed, as well as the current and voltage transmitted from the Arduino Nano serial USB port to compute the power. The code structure flow can be demonstrated from the following equations:

(1) Power = Current * Voltage

The voltage and current will be measured every second in a loop in conjunction with the time elapsed to calculate power.

(2) Kilocalories/Hour = Power * 0.860421

The power will be used as a variable to a second equation to calculate kilocalories per hour.

(3) kcal = Kilocalories/Hour * timeElapsed

The third equation in the function will compute the total kilocalories burned in a session.

Once the user presses the stop button, the arguments to the caloriesPower() function will be the kilocalories and the time elapsed. These variables will be displayed via the DSI connector to the RPi 7" Touchscreen.

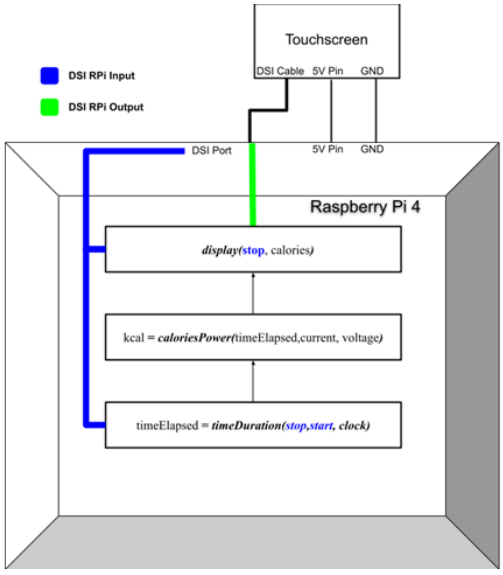

Fig 23. Motor Bike Health System

Example: Pseudo Code for Time Duration

```
timeDuration = functionTime(startVariable,stopVariable)

If (startButton)

{
      startVariable = I2C_data (at time stamp)
}

If(stopButton)

{
     stopVariable = I2C_data (at time stamp)
}

functionTime(start, stop)
{
      Time = stop - start
      (Return Time)
}

If(stopButton)
{
  (display timeDuration)
}
```

Fig 24. Pseudocode

vi. Ethics and Broader Implications

This project aims to avoid any copyright issues by following guidelines provided in Nintendo documentation. Nintendo explicitly states that game ROMs downloaded from the internet are copyright infringements. However, the company has claimed that the use of a backup copy made from an owned copy of the game does not pertain to ROMs downloaded from external sources via the internet. They do not allow selling or distribution of the game from a copied license in this way. The emulated game copy will be used for testing purposes and not for commercial use or exchange. Though there are no legal precedents or trials relating to the individual use of emulators or ROMs, after testing is completed, an officially licensed copy of the game will be used for the final state of the project in order to avoid any legal issues.

The Mario Kart Bike will be designed with consideration of environmental impact. The main housing material will be made of recycled filament for 3D-printers to house buttons. In addition to using environmentally friendly materials, the bike will implement reusing parts such as used bike brake handles to mimic realistic breaking for the player.

The exercise bike will still comply with any regulations or safety standards regarding exercise equipment as we will not be modifying the frame or operational parts in any way that could compromise the integrity or effectiveness. Also, analysis and testing of safe design and operation of the resistive methods will be conducted to prevent physical harm to the user. These safety regulations have determined many design choices. For example, the resistive capacity will be limited to follow existing resistance safety standards on commercial exercise bikes, and the rate of change of resistance will be tuned within the MC interfaces to prevent sudden stops or adjustments that may lead to injury. Also, instead of using a conventional bike that is converted into stationary function, a traditional static exercise bike will be employed and modified to simulate the steering function of a regular bike. This is to allow for greater stability and easier mounting of the motor/magnet systems, again in the interest of preventing injury to the user.

The team conducted an architectural risk analysis on the bike. The consensus is that there are few attack surfaces based on the design. This is evident because there are no wifi or bluetooth connections, although bluetooth is a possibility if it is added in later versions. However, there are serial data connections between the microcontrollers of each subsystem. To mitigate risk, the serial data will be encrypted via code. The team identified the cable linking the controller to the switch as a trust boundary. Unfortunately, the switch data cannot be encrypted. This boundary will be protected with a barrier to house the cord so that the switch data is protected.

There are additional risks to the system's performance that pertain to user safety. To protect the bike system from being manipulated by the user, components with safety features were selected. For example, the team chose a DC motor with a junction box to house wiring. This decision was made to prevent the user from harm via electric shock.

V. RESULTS AND DISCUSSION

A) Pedals Subsystem:

The purpose of this subsystem is to translate the rotational movement of the bike's pedals to the DC motor resistance system. The main constraint is that no major modification can be made to the bike so that it may still be rideable on the road. The design outlook has changed since design phase 2 was written. It has evolved into an easier method, while still maintaining the specs and constraints.

To test this subsystem, a rider pedaled the bike and it was demonstrated that the bike tire roller was able to transfer the motion through a flexible shaft to the DC motor sitting on the floor. No modifications were made to the bike besides adding a couple of bolts to the rear axle so that it may sit in the bike trainer more securely.

Various riders have indicated that having the pedals attached to the DC motor does not hinder gameplay, but it

does add a noticeable amount of resistance even when the motor is not being braked. Therefore, it can be concluded that the subsystem is successful and no further improvements are necessary for base functionality of the subsystem.

The following video demonstrates the experiment:
https://drive.google.com/file/d/1FQCbuubpCnskCKTeCOQMqnzUGAGgylcD/view?usp=sharing

B) Steering Subsystem:

The purpose of this subsystem is to transfer the rotational movement of the bike's handlebars to a potentiometer that will convert it into game data. The constraints are that it must be modular and removable when not in use. The design has changed slightly since phase 2, but the basic idea is the same. Two gears translate the motion to the potentiometer, and the whole assembly is removable via screws. Although, the gear attached to the bike doesn't have to be removed as it doesn't hinder normal operation of the handlebars.

To test this subsystem, a rider turned the handlebars 180 degrees at varying speeds and forces. The gears can turn further than 180, but no sane rider would need to do that in-game. It has been found that the steering control could be tuned to provide more steering per degree of rotation, but that will be fixed in software. As far as the physical subsystem goes, the design is successful and no changes need to be made for base functionality.

The following video shows a demonstration of the experiment:
https://drive.google.com/file/d/1FfzsU2vSYcDkufuFrZ7AIU6jiKNoKl8x/view?usp=sharing

C) Frame Subsystem:

The purpose of this subsystem is to suspend the bike and the rider off the ground. This is to ensure that the front and rear wheels can move freely to serve functionality for other subsystems. It also must hold the game's display for an arcade-style gaming experience. The constraints are that the frame must be height-adjustable, foldable and/or easily disassembled, and rigid/wide enough to support any rider and prevent tipping over.

The design of the frame was not changed from phase 2 except for the way the DC motor is mounted. It is no longer secured to the bike frame. It instead sits on the floor with rubber feet. The rest of the frame was built to spec. To test the strength of the frame, a rider stood on the bike, leaned both ways, and pedaled. The monitor mount was also nudged to test for wobble and stability. The results were good. The frame did not exhibit any significant wobble. The monitor can be wobbled by hand, but it does not move while the rider is pedaling. The bike is held off the ground securely. The only way that the constraints were broken is that the front wheel had to be removed. However, it is easily reattached with two bolts and it even makes steering easier and more fluid to have no front wheel.

The following video shows the experiment:
https://drive.google.com/file/d/1FirlIeNljJ6o62IekmWVKg16TLf-UYEZ/view?usp=sharing

D) DC Motor Resistance Subsystem:

The purpose of the DC motor is to provide dynamic resistance to the pedals of the bike and therefore the rider. The main constraint is that the system must be self-contained and unable to harm anyone using it. This DC motor outputs high voltage and current so this is imperative. Several precautions have been made to prevent shock.

The purpose of the MOSFET is to provide a way to control the braking of the motor. However, through extensive testing, it has been found that the selected MOSFET will not work for two reasons. One is that the transistor is rated to dissipate 160W while switching. It has been found that the motor can generate up to 260V and 1.2A, which obviously leads to a power much higher than 160. The second reason is that this is a typical n-channel MOSFET, meaning its intended use is as an on/off switching device. By operating it in the linear region, we are introducing a catastrophic amount of internal heat. This has led to two of the MOSFETs internally shorting and becoming unusable. Additionally, the chips meant to measure the current and voltage generated at all times are not functional. In phase 2, it was thought that the chips could be powered by the DC motor. However, that is not the case. The op amps and hall sensor need a dedicated power source to be functional. Therefore, the measurements conducted in the experiments below were done by a DMM.

The solution that has been implemented (for now) is to instead put a relay across the motor leads. This is not a permanent solution, but it is what we used for the experimentation that will be mentioned below. The relay serves the same purpose, which is that we can control the high power motor with a low power device (the Arduino). However, the resistance is not dynamic. Because it is a relay, it can ONLY be treated as a switching device. We have implemented a program that randomly turns the relay on for 30 ms, which creates a noticeable jump in resistance for a brief moment. These jumps in resistance happens randomly every 15-45 seconds. This adds an element of exercise on top of what is already present. Driving the motor in an open state already leads to a considerable amount of resistance, along with the random jolts.

Again, the above solution with the relay is only temporary. However, some experimentation has been done using the temporary system. To test the performance of the motor, we had a rider run 4 races. In 3 of them, we measured the voltage generated by the motor. In the fourth, we measured the current generated. The reason we only did

one trial of current measurement is because the motor is open for the majority of the race due to the relay. That means that theoretically, zero current will be generated. We did one trial to prove that. For the voltage trials, it was found that an average of 140-160 Volts is steadily generated throughout the race. Further, because "zero" current is being generated, that means a very small amount of power was generated (microWatts). The results have been plotted below.

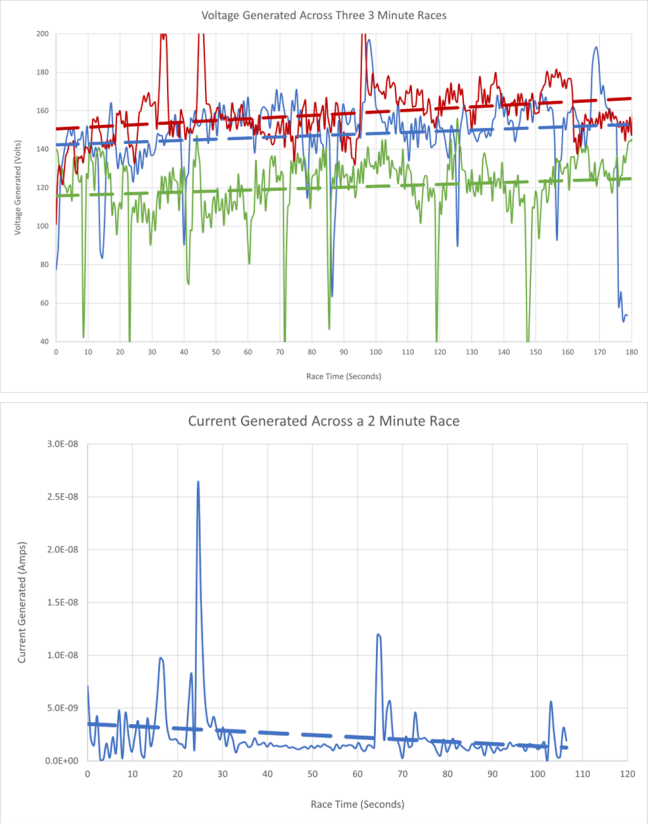

Fig 25. Voltage and Current Generated from DC Motor

The dashed lines represent the average across the data spread. The voltage graph has 3 trials on one plot, each taking about 3 minutes. The rider pedaled at various speeds in each trial, which is evident by the slight difference in the average voltage generated when comparing the three. The large dips in the lines were caused by the random jolts from the relay being turned on. The reason the voltage dips so much is because when the jolt happens, the RPM of the motor drops significantly, leading to less voltage generated.

With this, we would consider the results of the experiment successful. The relay worked as intended and added a considerable amount of exercise to the riding experience. It has also been shown that a considerable amount of voltage can be extracted from the system.

In the future, the relay will be replaced with a linear mode MOSFET with high power dissipation. A linear mode transistor is meant to operate in the linear region, which means the high internal heat will be reduced. Along with higher power dissipation, this should lead to a system that functions as originally intended. Additionally, the op amps and hall sensor will be given a dedicated power source and their functionality will be tested later. Once those improvements have been made, we will run the same trials again and compare the results.

A video was not recorded for this experiment because the operation of the resistance system has been showcased in the video for the Pedals Subsystem.

E) Steering Subsystem

The steering subsystem consists of a 10k Ohm potentiometer and a 12-bit ADC chip with an onboard gain amplifier. The potentiometer is connected to the 5V power and ground of the RPI, and its output is fed into the first analog channel of the ADC. The ADC is also powered by the RPI, and its digital output is returned via I2C. Steering is mechanically translated from the user turning the handlebars by a 1 to 1 gear ratio with one piece around the turn shaft of the bike and the other around the knob of the potentiometer. The potentiometer is mounted such that the middle position of the range is facing directly forwards away from the user of the bike. It has 135 degrees of physical rotation available in each direction.

It was originally expected that the user could only practically operate the handlebars in a range of 90 degrees left or right of the center point. This limited our range of the potentiometer to only 180 degrees out of 270 total degrees available or 2/3 of the base range. The ADC chip measuring 12 bits gives 4096 partitions of measurement possible. Over a range of 5 volts, this works out to be 0.0012207 Volts per partition. Because we are only using 2/3 of the voltage range for controls, we expected about 2730 partitions or 3.333 Volts of practical range. The Nintendo Switch controller analog signal for horizontal movement only scales up to 4096 partitions as well, meaning we still planned to have 2/3 of the range of standard controller steering. Given the already high precision of the controller, we had concluded that the output of the ADC should be directly mapped to steering data from the switch, and that the difference in range would be insignificant for standard play. The physical design of the gear system should ideally result in the same turning distance and speed as turning the potentiometer by hand via the knob.

The steering implementation ended up being relatively close to the specifications. It has the highest priority of all control signals because of its necessary precision and responsiveness. In terms of the range of fidelity the user has when playing the game, the 180 degree range allows access to the 500th to the 3500th partition in the range. However, the 12-bit ADC chip has peculiar gain options that prevent exactly 5 volts from being utilized as the range limiter. The closest choices were either 4.096 V at a gain of 1 or 6.144 V at a gain of 2/3. Selecting the 2/3 gain option allowed us to

use the full range, but the value of the voltage affected by each partition changed, so the voltage range the user has access to practically in the 180 degree range is 0.75 V to 5.25 V. Fortunately, the potentiometer and the RPi have no issue with this difference in range and function just as well as predicted. The steering is mapped one-to-one from the returned digital bit value to the horizontal analog left stick input (steering stick in MarioKart). This was possible because, coincidentally, the analog stick in a Nintendo Switch controller also has 12 bits of fidelity for the horizontal position. The slight loss in range on either end of the steering has not proven to be significant to performance due to the gradual steering process in-game. Functionally, this subsystem works exactly as planned. Subjectively, a different approach to the mapping might be beneficial in the future. While pushing an analog stick all the way to one side is relatively easy and makes sense for a hard turn, turning bike handlebars all the way to one side while pedaling has proven to be awkward and difficult to adjust to while playing. An exponential form of mapping, where the partition value being mapped is increased faster as the user turns, should solve this problem. One other issue has appeared recently with the connection between the ADC and the RPi where the I2C occasionally loses connection, but this is most likely due to the connection being between jumper wire headers and not permanently soldered in place.

Video Testing:
https://drive.google.com/file/d/1hmFfyGo91mm45ISRFt6g9mDjMVWMiVlL/view?usp=sharing

Once a better mapping for the steering is developed, it would be beneficial to test it and compare to the current mapping to see if the quality can be improved subjectively. If there could be a basic exponential function set up to correctly increase the rate of steering as the user turns further to each direction, we can tune it after repeated play-testing. Additionally, another test with different ADC gain settings selected might be beneficial depending on the result of the new mapping design. It could help add just a bit more fidelity to the turning itself.

F) Tachometer Subsystem

The tachometer subsystem is the method to convert the pedaling motion from the user into in-game character driving. It consists of the IR object sensor connected to one of the Arduino Nanos with code running that constantly calculates the RPM based on an equation involving a count of the number of things detected, the times per minute a sample is taken, and the number of things that are sensed on the rotating object. It uses this value to determine the acceleration of the character in-game.

The acceleration specification calls for a way to pick-up the pedaling, or more generally the attempted forward motion from the bike user. This data must be communicated over serial data from the Arduino Nano to the RPi and ultimately used to determine the acceleration of the in-game character. Ideally, it would correspond in speed such that the speed of the user equals the speed of the driving "kart". The component that does this must be fast enough to keep up with the approximately 90 – 120 RPM that could be seen at max from the user while pedaling.

The IR sensor component selected for this subsystem does not inherently measure RPM, however connecting to the Nano and running the number of object detections by the component through an equation can result in a relatively accurate RPM number. This data has been successfully acquired with this method and can be consistently obtained from the user. The RPM data has also been successfully sent via serial data to the RPi and used to determine when the user is accelerating or not. Functionally, this system works to act as the "A" button (accelerate in MarioKart) and the game is fully playable with it as such. However, the original specification of mapping the RPM numerically to the desired speed of the user is not achievable in the way intended in the initial design. The acceleration in MarioKart is essentially either stop or go, determined by holding the "A" button. Although the user can tap "A" repeatedly to simulate a slower level of acceleration, it is not possible to do this fast enough with the control system now without the character in-game stuttering. In other words, this system is not quite sensitive enough to smoothly vary the speed of the character in a way that looks and feels like the normal operation. Subjectively, this does not affect playability much, and holding a constant acceleration along with quickly slowing down when necessary is still possible, and racing competitively this way is still an option.

Video Testing:
https://drive.google.com/file/d/1hhtGuFOJsfW5j8NDNWvJnVqN-W55GKUF/view?usp=sharing

Below is some RPM data from 1 minute of pedaling. The test begins from a stop, starting at a normal pedaling cadence and then gradually increasing to monitor the sensitivity and consistency of RPM. The middle spike is from a sudden jump in RPM caused by a quick increase in pedaling speed to test the responsiveness of the code. This test was repeated 3 times with the same pattern of speed change each time. From this data and context, the system does a good job of capturing consistent and usable RPM data.

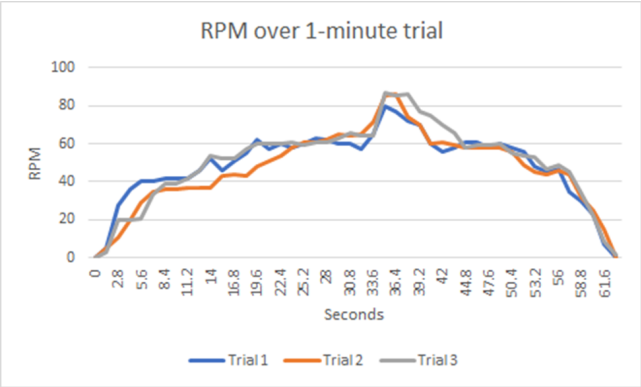

Fig 26. Tachometer Experimentation

A follow-up experiment that may be worth doing is to compare the RPM measured while playing the game to that of a normal bike pedaling on the road. Any difference found could help in understanding the level of exercise the user is getting during the game compared to just riding on a track. Also helpful would be another experiment involving the PWM of the "A" button, this time using the built-in "tap" function found in the "Joycontrol" software. After looking at the code behind it more, it seems like it will cause less stuttering than my original approach.

G) Handlebar Buttons

The buttons subsystem is simply a method of allowing the user to use items and drift in-game using easily accessible brake-style buttons. The original design for the external control buttons was to have standard normally open switches attached to the handle bars to serve as the item-throwing and drifting controls in-game. The buttons would be connected directly to the RPI via GPIO and, when pressed, would return the corresponding button press as the controller button input that it was meant to simulate. It is meant to be quick and responsive enough to feel like a regular switch controller while being conveniently located on the bike handlebars so that it could be pressed during focused playing. The buttons should be able to input separately but also at the same time. Lastly, you should be able to do both the press and the hold functions of a button like a controller.

The buttons currently implemented in our design are essentially bike brake levers with an embedded normally-closed switch that is opened (signaled) when the lever is pressed down. They work as expected, and are assigned to the ZL and ZR (throw item and drift in-game) buttons on a standard Nintendo Switch controller. They can achieve both press, press & hold, press together, and press & hold together operations. They have the lowest priority in the control reading because they do not need as quick of a response as steering or acceleration. The video in this section shows their operation in the button test menu. Functionally, this subsystem is also a success as the buttons serve their intended purpose well. Subjectively, they are well positioned on the bike, easy to press during gameplay, and quick enough in responsiveness not to notice any delay. However, these particular brake handles tend not to settle back to the closed position all the way when released, which sometimes results in the button being held open for longer than intended. Stronger return springs may solve this problem.

Video Testing:
https://drive.google.com/file/d/1hl8dL9uW5UdxS6Jiiq3ZJV202Tk_pT7L/view?usp=sharing

The buttons do not need much more experimentation, but working to fix the springs sticking occasionally will require testing once it is addressed.

H) Controls and Interfacing

This subsystem consists of the method of translating, restructuring, and packaging the control data from the other subsystems to work on the Nintendo Switch console as a simulated controller. It is based on the open-source "Joycontrol" Github package used for simulation Nintendo JoyCons and Pro Controllers

This subsystem is arguably the most important. It is intended to serve as the communication between all of the bike control options and the Nintendo Switch/Mario Kart. Basically, the specification here is that the controls need to be quickly and accurately forwarded by the RPi from the various systems to the Switch and consequently the game. Some less vital specifications also called for the return of vibration data from the Switch and for a typical controller to also be connected so that the user can easily navigate the menus of the game.

The implementation of this system required the most detailed work out of all the others and, consequently, also has the most issues. The method of translation here is done through a modified version of the "Joycontrol" program. Originally, it was meant to connect to the Switch via Bluetooth, mimic the ID of a Switch controller, and then receive controller "input" from a terminal operated on the RPi. It has now been changed to work with both GPIO signals, I2C connection, and serial data lines for the controller input. The program packages the control data the same way, but now the bike controls can be used for input. In terms of basic functionality, that part of the system is successful. The game can be accurately played through at least an entire race with full control. The return of vibration data is very basic in the default program. There only exists a return signal to tell the controller when vibration is enabled, so the intention to use it as a motor control is not currently functional. Also, because the connections are now done over Bluetooth, a second controller cannot be used for the same user, meaning those portions of the subsystem are not operational at this point. Additionally, the program has

proven to be quite prone to different issues that result in disconnection from the Switch. Loss of connections with the I2C, random Bluetooth drops, terminal freezes due to memory leaks or CPU throttling, and issues with the program running too slowly have all occurred during the testing of this system. Functionally, it works when it needs to, but there is a lot of room for improvements in both the code and the hardware involved in the translation and transmission of data. Subjectively, it works well enough to play 1 to 5 races on average without issues. The occasional error mid-race is annoying, but the novelty of playing the game on a bike helps to make up for it.

Video Test:
https://drive.google.com/file/d/1iGuEhgO6bs-T5icZ7CG_KcLFKqcZzPuW/view?usp=sharing

There is plenty of potential for more experiments on this subsystem. Testing the real-time delay of the controls empirically, testing the control load limits to see how much data can be sent at one time before slowdown occurs, and gathering more subjective input from a wider user-base would all be beneficial experiments to consider in the future. Focusing first on finding out the reasons behind the occasional connection issues experienced by the system will be critical.

I) Sound and Display

This subsystem represents the method of displaying the video and audio portions of the game conveniently for the user. The Switch dock provides the HDMI output needed for these signals in the system. This subsystem simply requires a monitor to display the game in a position that the user can easily see and hear from. It originally called for two monitors and an HDMI splitter, but the second bike design was scrapped for now, so only one display system is needed.

The implementation of this subsystem was straight-forward. An HDMI-capable monitor with speakers was obtained, connected to the Switch, and mounted to the frame directly in front of the bike. It is sturdy, well-placed, and close enough to comfortably see and hear the activities of the game. Functionally and subjectively, it works perfectly for what it is needed for.

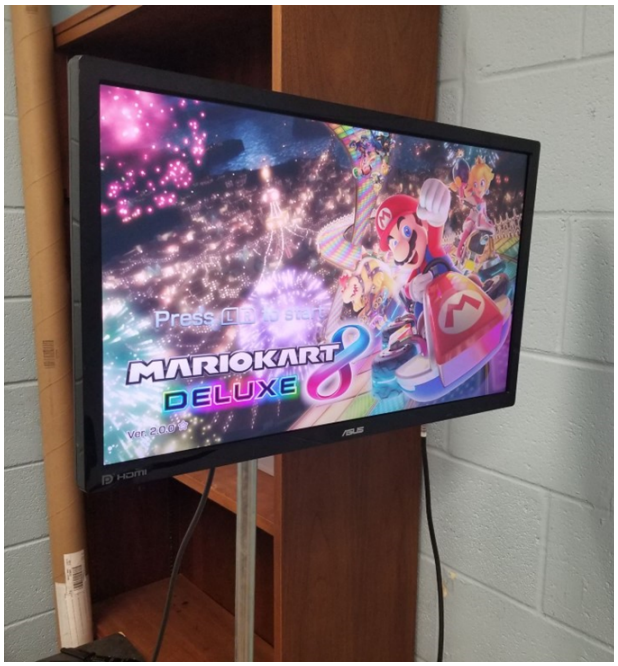

Fig. 27: Display Test

It has been concluded that this system does not need any more experimentation.

J) Deliverables

The team expected to produce two variants of the MarioKart bike: one with a DC motor resistance system and one that used a permanent magnet. In the end, only the DC motor variant was built due to the unforeseen difficulties associated with controlling the motor that slowed the general progress of the bike. Beyond that, every other design goal that the team set to accomplish was implemented in the final version. The bike works with every control needed to navigate the menu of the game, select a race, character, and bike design, and to play competitively in a standard race. Each of the control functions pertaining to a race are implemented in high enough fidelity such that the average user can consistently win a game. Additionally, we were able to contain everything running the controls and the game safely in the board-box out of the way of the user, protecting them from any electrical harm. The physical frame and motor resistance system work well enough to provide stability for even the most fervent users, and it provides enough of a workout to actually notice a burn by the end of a race. Plus, the functional health system provides a reasonable estimate of just how many calories were burned during the race.

VI. TIME FRAME AND ESTIMATED COST

The overall time constraint for this project was to design and build the system within one academic school year broken into two semesters. The first semester involved the

design portion, while the second semester consisted of the build and experimentation. The goals for the project are stated in the specifications section of the introduction. These were decided upon as a group with the intent to reach our goal of a fun and practical exercise system in a high quality and user-friendly manner, and from there the specs were attributed to various subsystems and the work was split accordingly to design those subsystems. The team met on a weekly basis to ensure that each member was on track and to bring updates about their progress. At times, work had to be redistributed to other team members if the load was unbearable. The Gantt chart in figure 27 lays out the design tasks from the first semester.

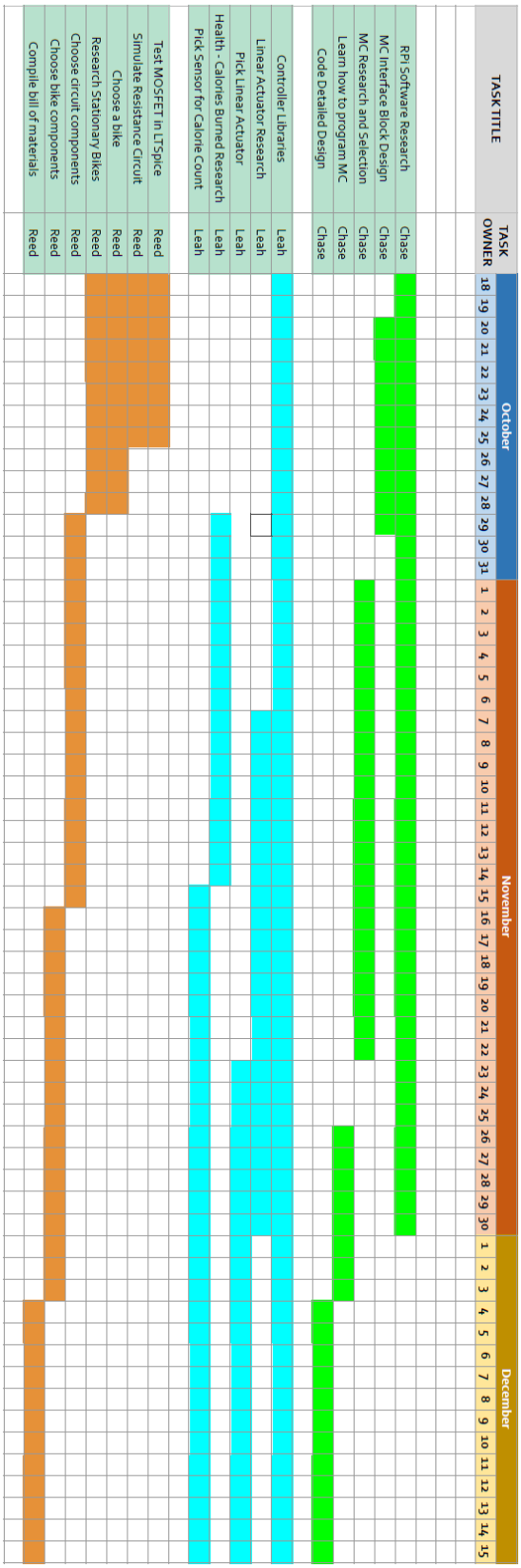

Fig 27. Design Timeline

As for the cost of the project, the team estimated at the very beginning that the production per unit cost would be around $500-1500. From the bill of materials, the actual production cost was just over $1500 for the parts alone. The major contributors to this cost were the motor and the metal frame pieces. If man-hours and salary were factored in as NRE cost, this project would have cost much more to produce and maintain.

VII. ETHICAL AND PROFESSIONAL RESPONSIBILITIES

To make sure that this project contributes to the environment, materials used for the bike will be made from recycled 3D printer filament. This will reduce the number of new materials needed for the bike.

Above all else, this project considered public safety and societal impacts. As engineers, there is a responsibility to protect the consumer from injury. In this project, research was conducted to determine the range of resistance that is safe for the human body. Also, it was tuned to be sure the resistance change was not so sudden that it would cause issues during actual high-speed play. The bike was designed to be safe and easy to use for both elders and children with the adjustable mounting points and stable frame design, which was especially important with it potentially being advertised as a consumer product.

VIII. LESSONS LEARNED

From a technical standpoint, not much went wrong on this project. The team was very fortunate to find the JoyControl library as well as other forums to assist in the controls design. However, there were many oversights with the dynamic resistance system. For one, the gearbox that was purchased was both unfit for this application and unnecessary. Worm gears cannot be back driven, and a speed reduction was not even required for the motor because of the chosen attachment point at the rear wheel. Further, the DC motor control circuit could have been designed more carefully. More research needs to be done on linear mode operation of MOSFETs. It is important to understand that while the datasheet may promise a component can handle certain maximums, that is not usually a well-defined statement. In real-world applications, devices will always underperform compared to their spec sheet. Additionally, there were some small structural failures with some of the mounting pieces after a great number of use cycles. In the future, more permanent and structurally stable mounting choices would be better from a production standpoint.

From an organizational standpoint, it is important to recognize the strengths and weaknesses of each team member early on. Fortunately, this team had a good spread of knowledge and was able to adapt and overcome challenges.

IX. CONCLUSIONS

This project saw many changes throughout its course. What started as a way to exercise and have fun rapidly expanded into a complex project. Each element of the design led to a “fork in the road”, where several options were available. If this project were to be repeated by another team, this team has no doubt that it would look completely different. The original problem that was presented can be solved in endless ways. This team is very satisfied with the final edition, and they are excited to see what the future holds. One day, each piece will be redesigned and implemented in a way that is better than before. This project has huge potential to become a consumer product, but the road to that stage is long.

X. APPENDIX

A) Individual Contributions

- Reed Hester - I held the role of lead electronics and structural designer. Essentially, every piece visible on the stationary bike was a decision made by me. I also designed the dynamic resistance system and its many versions. Overall, I held a largely technical role, with some minor leadership and organization as well.
- Chase Griffin - I had the role of lead controls programmer. I handled the main Python program on the RPi that allows the signals from all of our sensors and external components to be translated into controller data that the Nintendo Switch could recognize and use on the game. I also programmed each of the Nanos for their respective function in the motor control and tachometer subsystems. Lastly, I served as a sort of co-leader with Reed in handling group documents, meetings, and general project scope decisions.
- Leah Faulkner - I designed the health system and its GUI, coded touchpad navigation, made modifications to the board control box, and conducted subjective experimentation.

B) Relevant Course Knowledge Required

- Object-oriented programming
- Event-driven programming
- Data structures
- Electronics/circuits
- Programming in C++ and Python
- PCB design

C) Knowledge Gained

While this project was deeply seated in relevant ECE topics, there was also a structural and mechanical element

that was new territory. Design choices were made based on feedback from other disciplines and over-estimation. For example, steel pipe was chosen for the frame rather than aluminum because the team was certain that steel is stronger than aluminum. However, we lacked the structural knowledge necessary to know if aluminum would have been suitable. Further, the transmission of motion from the rear tire to the DC motor took many hours of research due to the mechanical nature of that topic. Lastly, the team learned a lot about project management in general including how to fairly structure the workload and timeline of the project and how to deal with sudden technical and personal issues in the scope of the project.

REFERENCES

[1] "Adult obesity facts," *Centers for Disease Control and Prevention*, 07-Jun-2021. [Online]. Available: https://www.cdc.gov/obesity/data/adult.html. [Accessed: 19-Sep-2021].

[2] J. E. Donnelly, J. J. Honas, B. K. Smith, M. S. Mayo, C. A. Gibson, D. K. Sullivan, J. Lee, S. D. Herrmann, K. Lambourne, and R. A. Washburn, "Aerobic exercise alone results in clinically significant weight loss for men and women: Midwest Exercise Trial-2," *Obesity (Silver Spring, Md.)*, Mar-2013. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3630467/. [Accessed: 19-Sep-2021].

[3] P. F. Staff, "What are the different types of exercise bikes?," *Primo Fitness*, 23-Feb-2019. [Online]. Available: https://primofitnessusa.com/what-is-the-best-exercise-bike/. [Accessed: 19-Sep-2021].

[4] "Eddy Currents," *K&J Magnetics*. [Online]. Available: https://www.kjmagnetics.com/blog.asp?p=eddy-currents. [Accessed: 19-Sep-2021].

[5] "Why is it harder to turn a motor/generator with shorted terminals?," *Electrical Engineering Stack Exchange*. [Online]. Available: https://electronics.stackexchange.com/a/438153. [Accessed: 19-Sep-2021].

[6] "What is the MOSFET: Working and Its Applications," *ElProCus*, 02-Sep-2020. [Online]. Available: https://www.elprocus.com/mosfet-as-a-switch-circuit-diagram-free-circuits/. [Accessed: 19-Sep-2021].

[7] "Optical and Photo Tachometers," *e2b calibration*, 21-Jul-2020. [Online]. Available: https://e2bcal.com/optical-photo-tachometers/. [Accessed: 19-Sep-2021].

[8] A. Rizwan, "What is an emulator?," *EmulatorClub*, 20-Aug-2020. [Online]. Available: https://emulatorclub.com/what-is-an-emulator/. [Accessed: 19-Sep-2021].

[9] B. Ryerson, "IR sensor for obstacle avoidance KY-032 (AD-032)," IR Sensor Obstacle Avoidance Keyes KY 032, 21-May-2021. [Online]. Available: http://irsensor.wizecode.com/ [Accessed: 27-Jan-2022].