Virtual Cycling Reality

By

Bryant Johnson

Chongxin Luo

Gregory Knox

Design Review ECE 445, Senior Design, Spring 2016

TA: Luke Wendt

2 March 2016

Project No. 47

Contents

Part I

Introduction

1 Purpose

The age of Virtual Reality (VR) is fast approaching, but VR experiences are currently beyond the reach of the average consumer. 3D goggle systems and hardware designed to interact with them are prohibitively expensive for the average consumer at $1500 for an oculus rift pc bundle. The Virtual Cycling Reality (VCR) Project seeks to bring the power of VR into the homes of the average consumer. This will be accomplished by creating a 3D video game world in which the user can bike. This world will come to life through a system of sensors, a fan based environmental feedback system, and by harnessing the power of the user's smart phone, coupled with a head mounted display. While other existing virtual biking experiences, such as CycleOps and The Virtual Bike, also attempt to bring a cycling experience into the home, they rely on 2D display measures and lack the environmental feedback that the VCR Project will provide.

2 Objectives

2.1 Goals

- Provide smooth virtual world for the user to experience.
- Track user head movement precisely.
- Accurately sense cycling speed and turning, with low latency user feedback.

2.2 Functions

- Measure rider speed.
- Detect handlebar rotation.
- Track head motion in three degrees of freedom.
- Interpret sensor data and use to update the virtual world.

2.3 Benefits

- Allows the fun of biking during any weather or season.
- Provides a more physically interactive way to virtual bike ride.
- Users can experience and explore a unique world.

2.4 Features

- Fan based wind feedback system.
- Variable resistance simulates different terrain types.

Part II

Design

3 Block Diagram

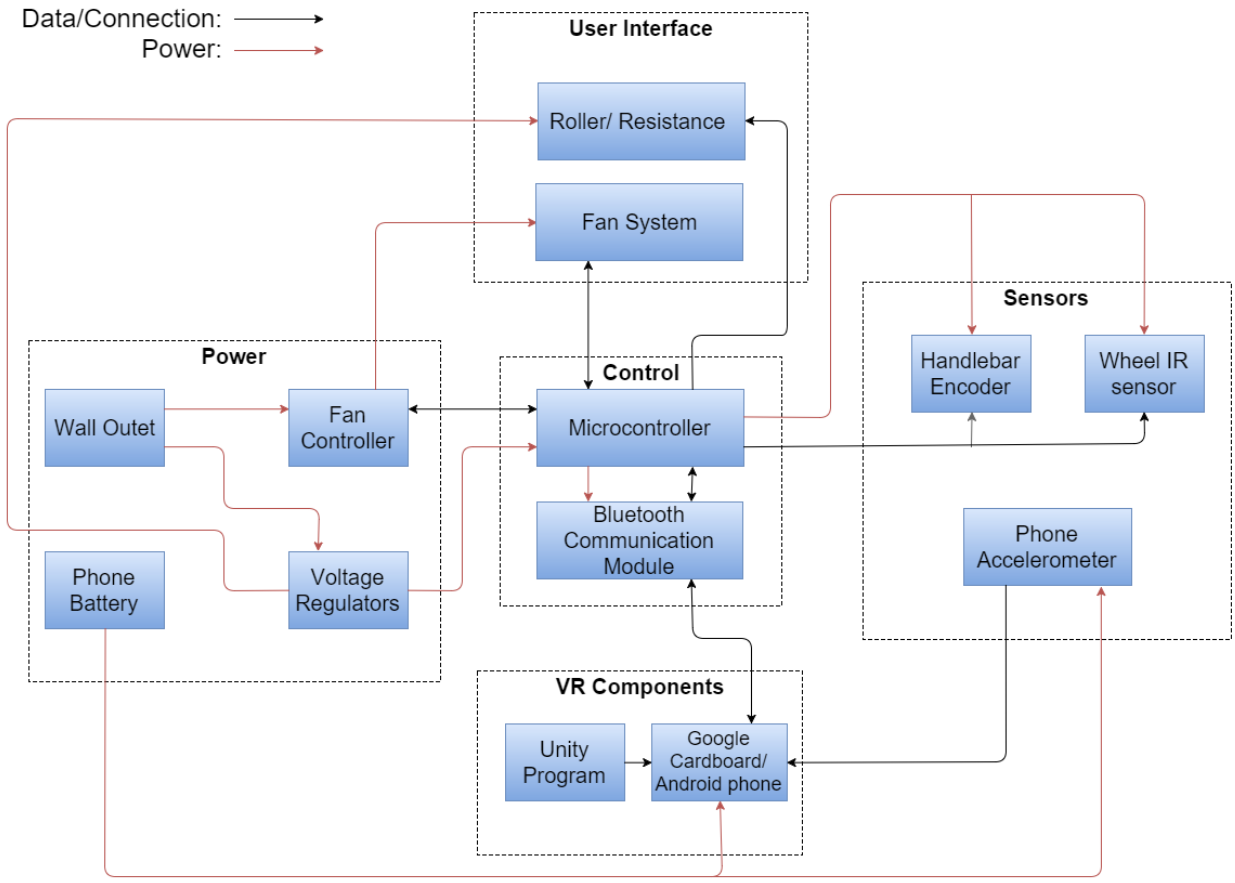

Figure 1: Block Diagram for Virtual Cycling Reality Project

4 Block Descriptions

Shown in Figure 1 above is the Block Diagram for the Virtual Cycling Reality project. The diagram contains five main blocks, which are Power, Control, User Interface, VR Components and Sensors. Each block can be implemented individually, and connecting the blocks in the end will achieve all functionality listed previously. The implementations and functionality of each block are being discussed in the sections below. (whats the functionality, where takes the input, where output goes).
The breakdown of this design splits the work into modules which involve power circuit, sensor/microcontroller design and software engineering. All three technical areas of focus matches with the areas of specialties of project members, which each member is able to put their area of interests into the maximum use.

4.1 Power

4.1.1 Voltage Regulator

Power for this project is acquired directly from the standard wall outlet. A 5:1 step-down transformer will lower the 120 $Vrms$ outlet potential down to 24 $Vrms$ where it will be rectified into dc. An output capacitor filters the pulsed dc signal. The dc voltage will be approximately at the peak voltage of the 24 V_{rms} signal ($24 \times \sqrt{(2)} = 34\ V$). Figure 2 shows the schematic of this rectifier.

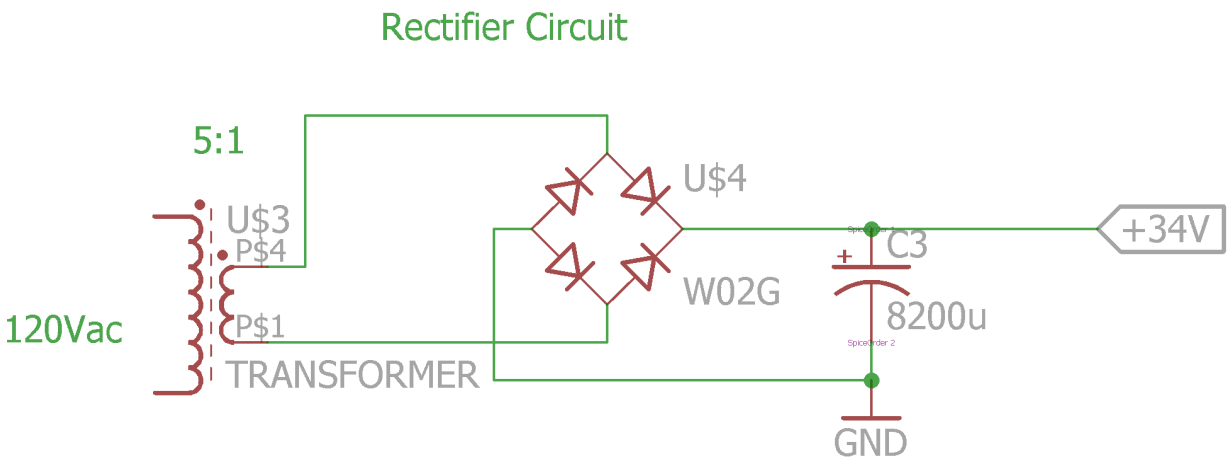

Figure 2: Stepdown transformer and rectifier circuit.

From the rectified voltage two fixed voltage output switching regulators will provide 12 V and 5 V outputs. These regulators will use the ACT4514 switching regulator. This device accepts an input voltage up to 40 V and can output up to 12 V and 1.5 A. Figures 3 and 4 show the two regulator circuits.

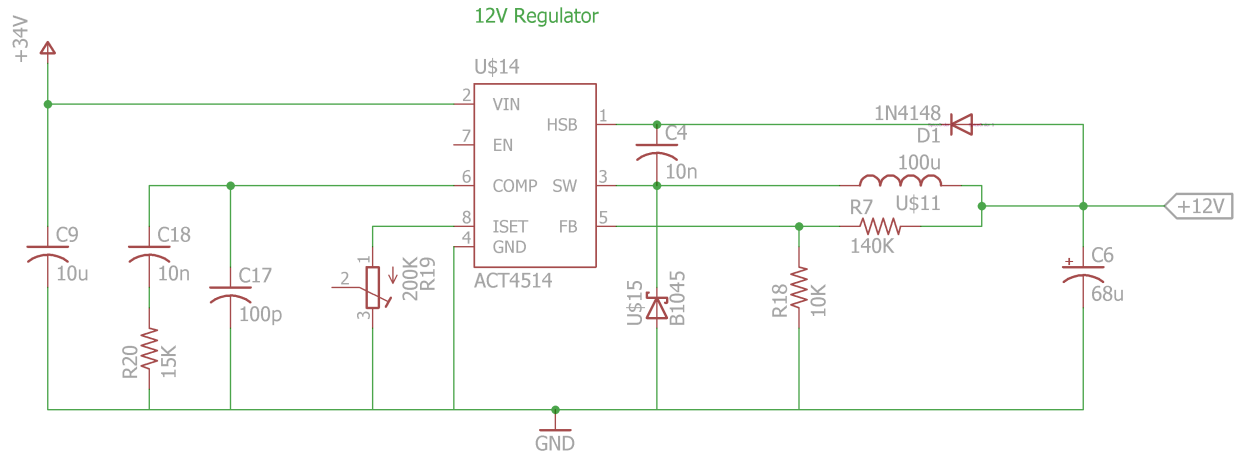

Figure 3: 12 V switching regulator circuit.

Parameter values for these converters are determined using the datasheet. The feedback resistor divider is used to set the output voltage. This also acts to correct for any changes in the output load that would change the output voltage. Equation 1 is used to calculate the second resistor in the divider for the 12 V regulator. R_{FB2} is recommended to be 10 $K\Omega$. A 140 $k\Omega$ resistor is used as a close value to 138.5 $k\Omega$

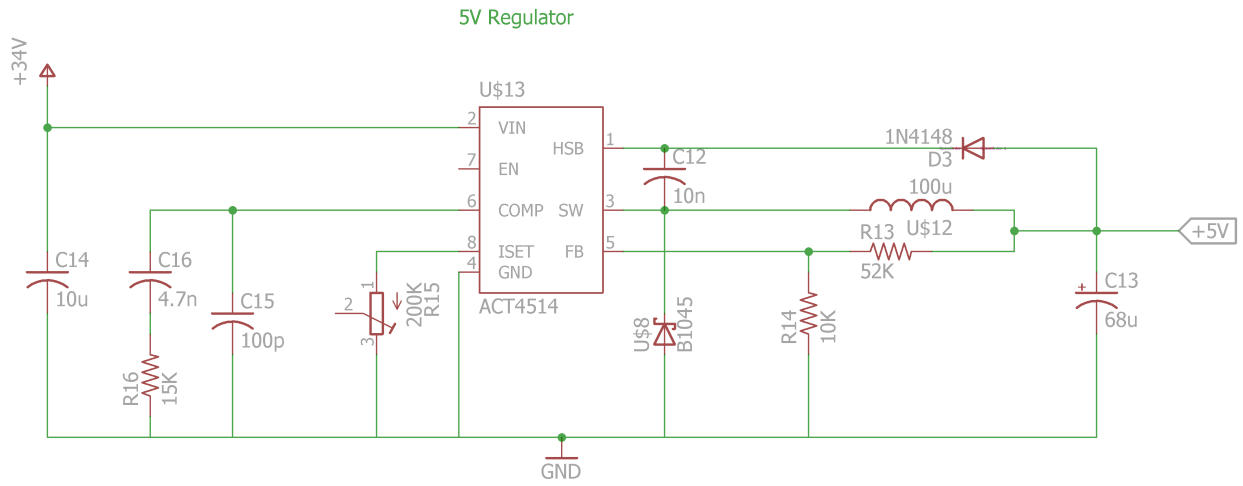

Figure 4: 5 V switching regulator circuit.

$$R_{FB1} = R_{FB2}(\frac{V_{out}}{8.08} - 1) = 10k(\frac{12}{8.08} - 1) = 138.5\ k\Omega[1] \tag{1}$$

The inductor calculation is shown in Equation 2. The calculated value is for the 12 V regulator. The datasheet recommends using $K_{ripple} = 0.3$ and indicates that the switching frequency is 210 kHz. The calculated value of 88.04 μH was substituted with a 100 μH inductor since they are much more common.

$$L = \frac{V_{out} \times (V_{in} - V_{out})}{V_{in} f_{sw} I_{loadmax} K_{ripple}} = \frac{12 \times (34 - 12)}{34 \times 210 \times 10^3 \times 1.4 \times 0.3} = 88.04\ \mu H[1] \tag{2}$$

The input capacitor recommended by the datasheet is a 10 μF ceramic capacitor and the output capacitor is determined by the voltage ripple equation shown in in Equation 3. Equation 4 shows the ripple result using a 68 μF 7 $m\Omega$ ESR Capacitor. The resulting voltage of about 7 mV indicates that the voltage ripple will be extremely small with this arrangement. The parameters for the 5 V regulator were calculated similarly.

$$V_{ripple} = I_{outmax} K_{ripple} R_{ESR} + \frac{V_{in}}{28 \times {f_{sw}}^2 L C_{out}}[1] \tag{3}$$

$$1.4 \times 0.3 \times 7 \times 10^{-3} + \frac{34}{28 \times (210 \times 10^3)^2 \times 100 \times 10^{-6} \times 68 \times 10^{-6}} = 6.99\ mV \tag{4}$$

4.1.2 Fan Controller

Fan speed is determined by the rider speed in the program simulation. Speed control for the dc fans is achieved through a buck converter. The dc fan covers low rider speed situations. The buck converter accepts a PWM signal from the microcontroller and outputs a varying voltage to the dc fans. The circuit for this converter is shown in Figure 5. A high side mosfet driver is required for this circuit to operate. The IRS2124 was chosen for this task.

The equation used to calculate the buck converter inductor value is given in Equation 5. This value

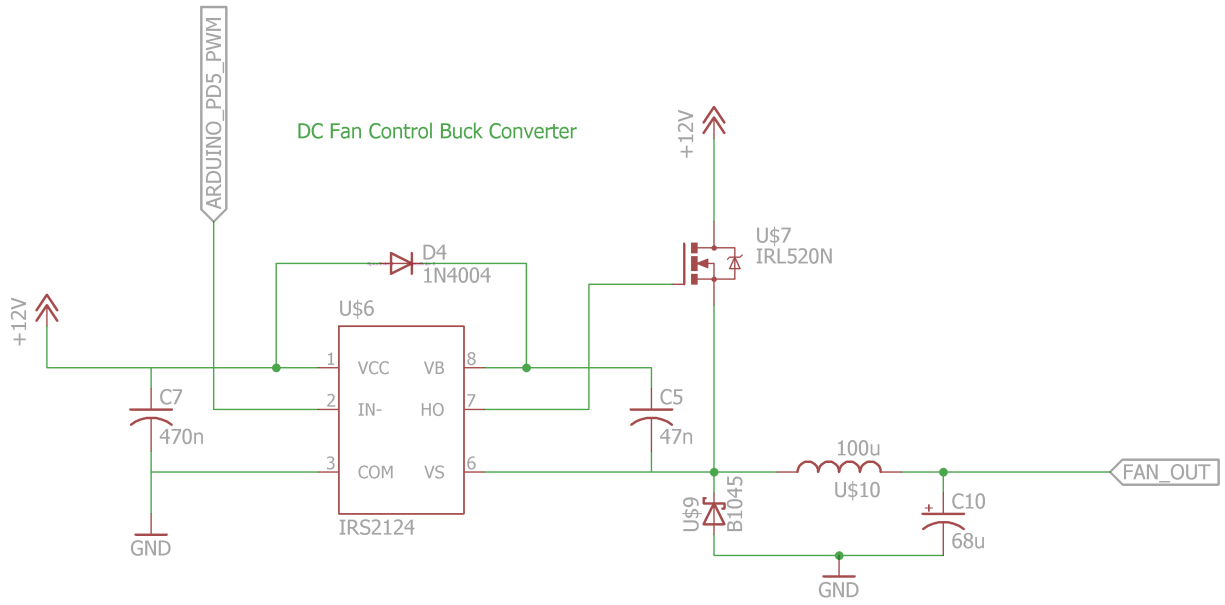

Figure 5: Fan control buck converter circuit.

was calculated for a minimum output voltage of $V_{out} = 6\ V$ which corresponds to a duty ratio $D = 0.5$. The switching frequency was chosen to be the maximum PWM output frequency of the Arduino Uno. This microcontroller can output at 62.5 kHz in fast PWM mode with 8-bit resolution.[2] The inductor ripple current Δi_L was chosen to be 40% of the maximum current of 1.2 A. Equation 6 shows the result of this calculation. The output capacitor value was similarly determined. The equation for this calculation is show in Equation 7. The ripple voltage was arbitrarily chosen to be 1% to find an appropriate capacitance. Equation 8 shows the result of this computation.

$$L = \frac{(V_{in} - V_{out}) \times (D \times \frac{1}{f_{sw}})}{\Delta i_L} \tag{5}$$

$$L = \frac{(12 - 6) \times (0.5 \times \frac{1}{62500})}{0.4 \times 1.2} = 102.86\ \mu H \tag{6}$$

$$C = \frac{\Delta i \times (1 - D) \times \frac{1}{f_{sw}}}{\Delta V_{ripple}} \tag{7}$$

$$C = \frac{0.467 \times (1 - 0.5) \times \frac{1}{62500}}{0.06} = 62.2\ \mu F \tag{8}$$

The calculated values for the capacitor and inductor were used to choose the values of the inductor and capacitor appearing in the circuit in Figure 5. Since 100 μH inductors are common this was chosen as the inductance value. A very low ESR 68 μF capacitor was chosen for this design. A simulation of these parameters was performed using the LTSpice circuit in Figure 6. Figure 7 plots the output values at various duty ratios showing dc voltages close to expected values.

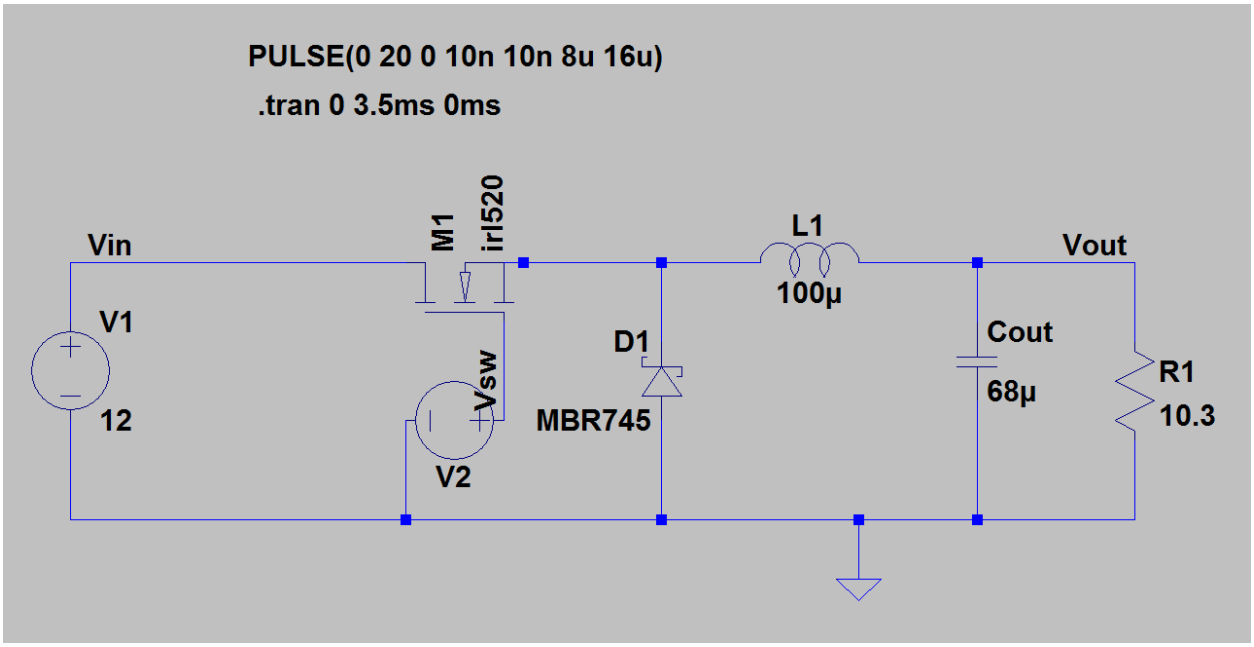

Figure 6: LTSpice buck converter circuit.

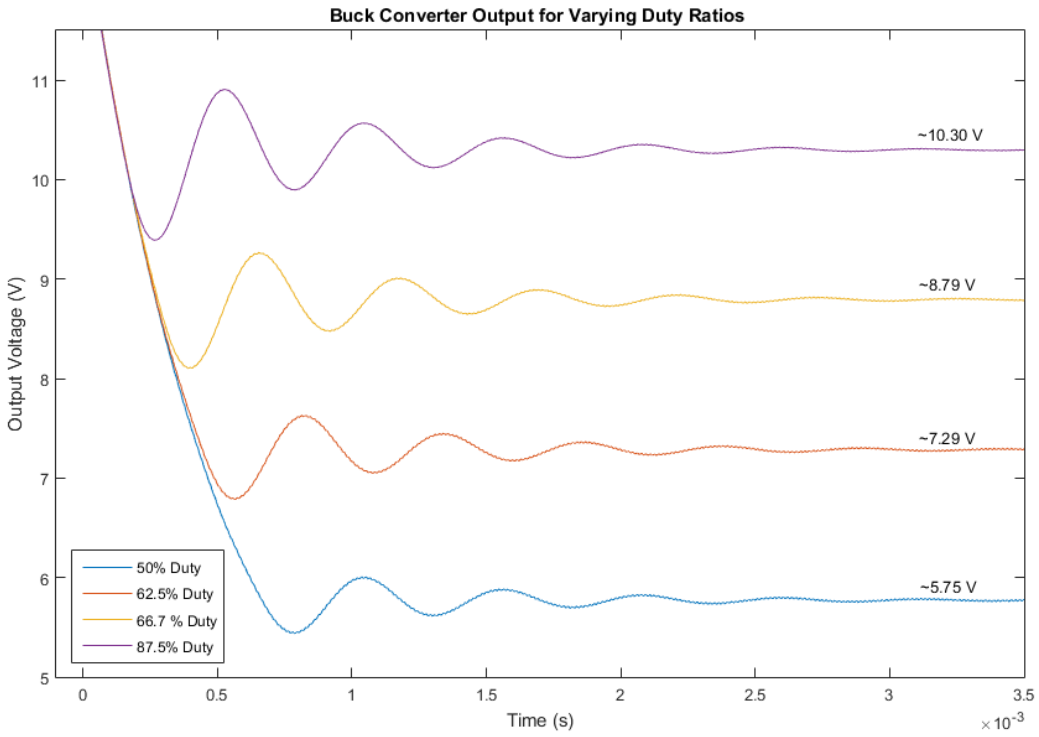

Figure 7: LTSpice simulation results showing expected output voltages.

The ac fan is intended for high rider speed situations and is controlled by a triac circuit. This circuit is shown in Figure 8. The triac blocks the ac voltage until a control signal is provided to the gate. The voltage

can then pass through the triac until the signal reaches zero. The gate is activated with a pulse from the microcontroller. The MOC3012 is used to isolate the high voltage ac signal from the microcontroller. The resistor and capacitor values in Figure 8 are derived from the MOC3012 datasheet.[3]

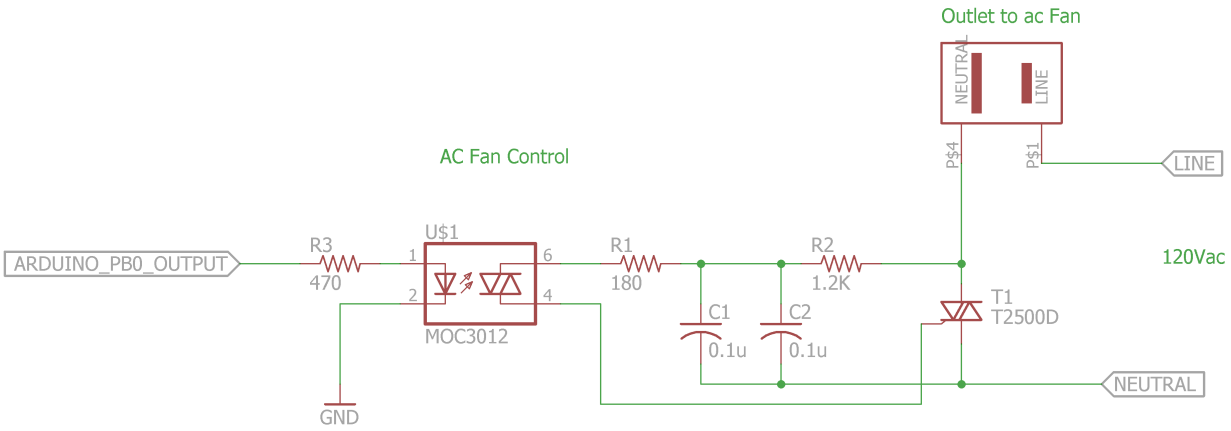

Figure 8: Control circuit for ac fan.

In order for the microcontroller to know when to trigger the triac a zero crossing circuit must be used. The zero crossing circuit is shown in Figure 9. When the ac signal drops below the optocoupler's turn on voltage the diodes no longer conduct and the optotransistor no longer conducts. A brief 5 V signal appears at the microcontroller input.[4] This behavior verified in an LTSpice simulation. The LTSpice circuit is shown in Figure 10.

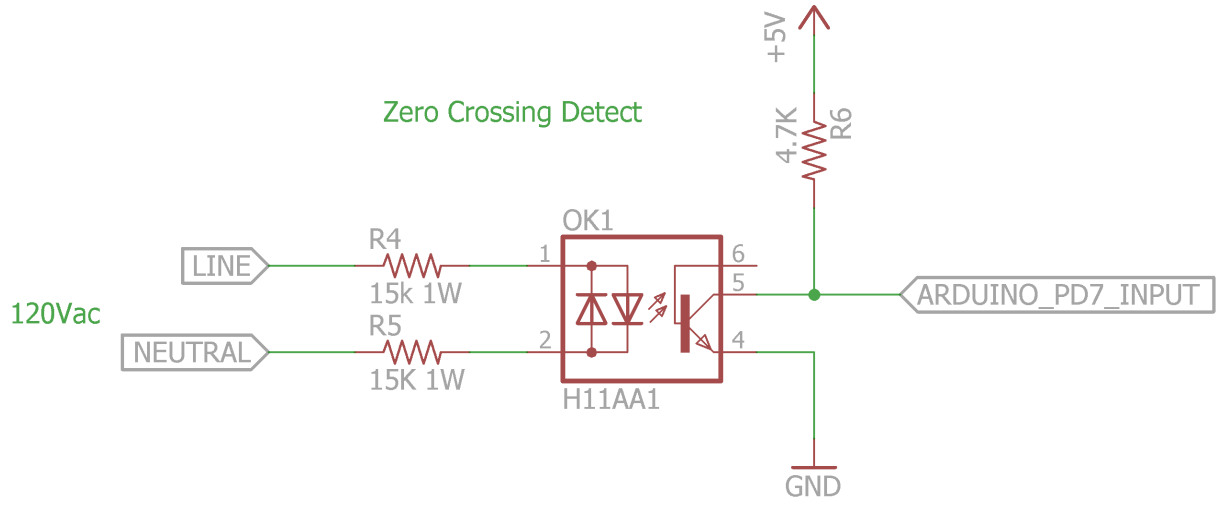

Figure 9: Zero crossing circuit schematic.

The results of the LTSpice simulation are shown in Figure 11. The figure shows that at each zero crossing of the input signal there is a brief 5 V peak at the output to the microcontroller to indicate the zero crossing. Once the zero crossing has been located the microcontroller will wait a certain amount of time depending on the desired speed of the fan and then trigger the triac. A shorter waiting period results in a faster fan speed. Figure 12 shows a close up of one of the zero crossing pulses. The width of the pulse is approximately 1ms long which is sufficient time for the microcontroller to register the pulse.

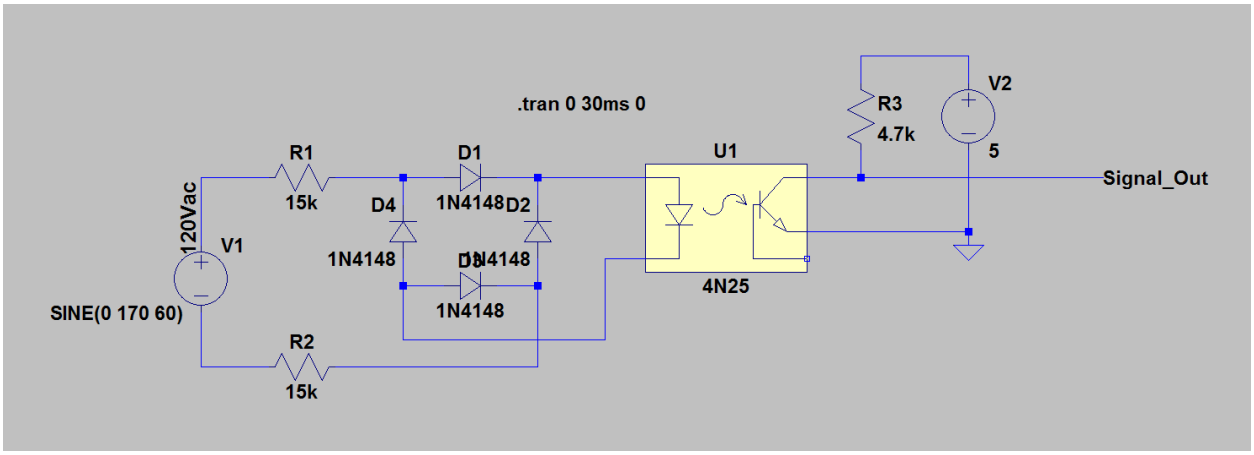

Figure 10: LTSpice Zerocrossing simulation circuit.

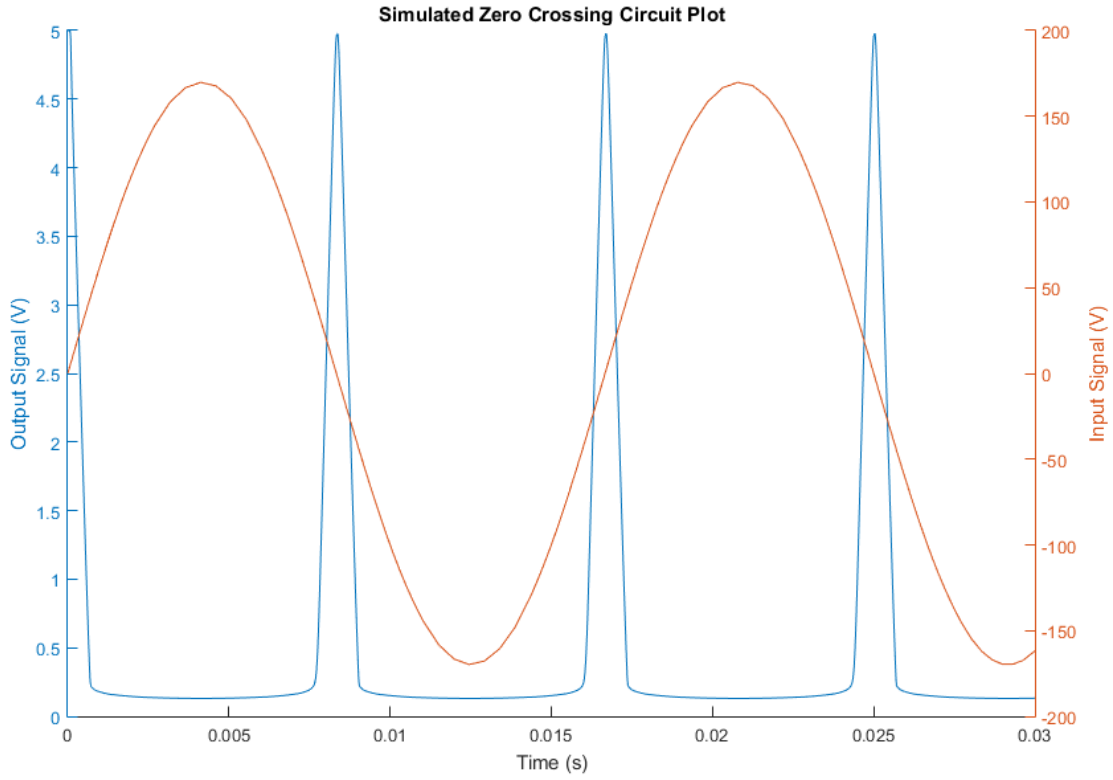

Figure 11: Plot of zero crossing pulses.

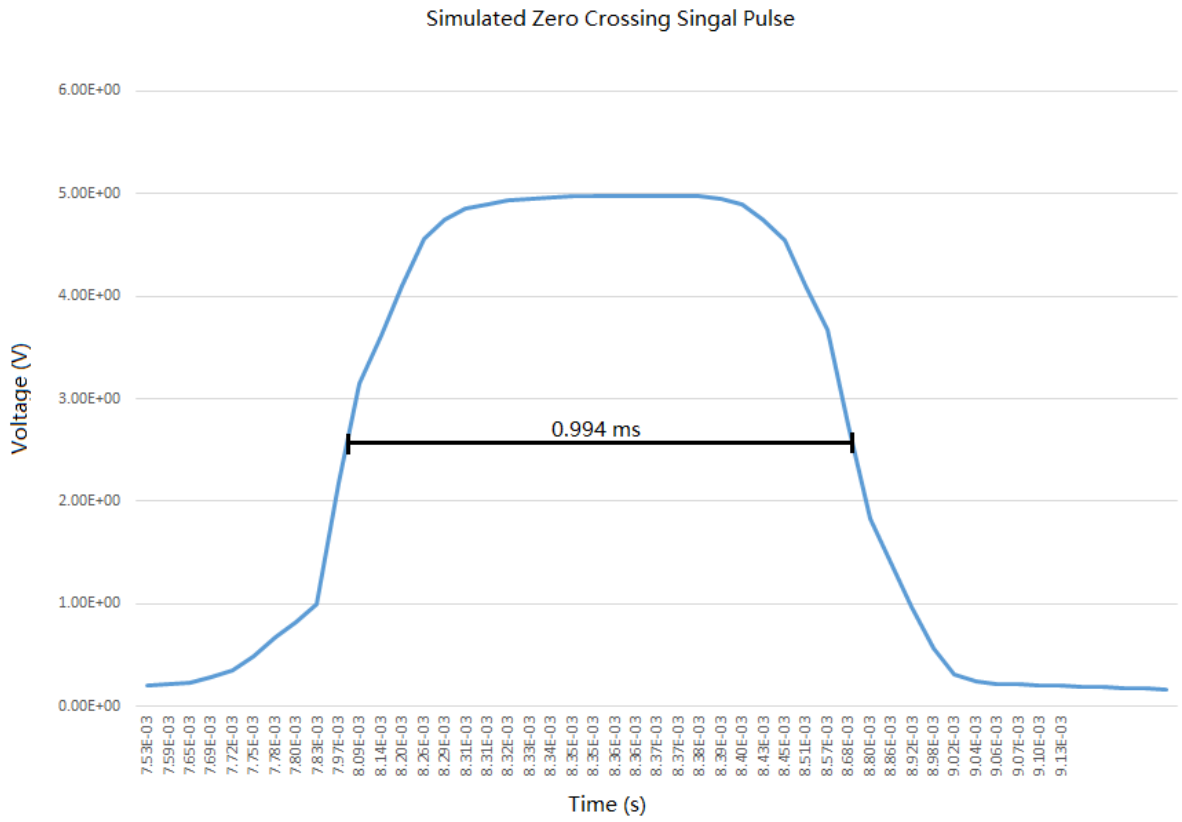

Figure 12: Single zero crossing pulse plot.

4.2 Control

4.2.1 Microcontroller

The microcontroller in this project acts as the connecting bridge between sensors and the simulation. An Arduino Uno will be used for for this section of the design. The power source for this unit comes from the voltage regulator and will be operating at 12 V. It takes input from the wheel IR sensor and the handlebar encoder. After probing the sensors the data collected will be sent Google Cardboard module using the Bluetooth Communication Module (BTCM). In the case of a changing terrain in the simulation the Google Cardboard module will send a new resistance value to the microcontroller via the BTCM. In response, the microcontroller then sends a corresponding signal to the Roller/ Resistance system in order to change the pedalling resistance, thus providing feedback to the user.

When designing the microcontroller this issue of latency was a large concern. Due to the nature of our project the critical path must be kept as short as possible. Based on the documentation given for our microcontroller , the execution time for an analog read from one of our pins is 100 μsec. Since we are polling from the potentiometer on the handlebar as well as the IR sensors attached to the wheel. Thus, reading from these sensors would take a combined 200 μsec. The Bluetooth communication module operates at a bandwidth of 1 Mbps. After reading from the pins a packet containing the sensor information will be sent to the phone. Estimating the packet size to be around 1 Kb we can assume a total latency of 1 ms. As seen in Equations 9 and 10. The critical path of our design involves a signal be sent back from the phone to the

microcontroller as well as time for phone processor computation (estimated at 500 μs). The summation of these delays is 2.7 msec and can be seen in Equations 11 and 12. These values are well within the bound for an responsive user experience.

$$BT\ Packet\ Latency = \frac{Packet\ Size}{BT\ Bandwidth} \tag{9}$$

$$BT\ Packet\ Latency = \frac{1\ Kb}{1\ Mbps} = 1\ msec \tag{10}$$

$$Total\ Latency = Sensor\ Probing + BT\ Latency * 2 + Phone\ Computation \tag{11}$$

$$Total\ Latency = 200\ \mu s + 1\ msec * 2 + 500\ \mu s = 2.7\ msec \tag{12}$$

4.3 User Interface

The User Interface module contains all components that interact with the user. The User Interface system has two major functionality. First of all, it gives a platform for user to use physical actions to control simulation characters. Secondly, it gives the user physical feedback according to the simulation environment. The User Interface system contains two main components, which are the front fan system and the rollers/resistance system. The user experiences these systems through riding a bike. The bike is being modified in order to achieve all functionality for this project.

4.3.1 Fan System

The fan control system gives realistic physical wind feedback to the user. For low speed fan operations a set of up to six 120mm 12v dc computer fans will be used. For high speed operations a 120 V ac fan is used. Feedback for the dc fans will be done using the sense pin on the fan. Pulses from this pin will be read by the mircocontroller to determine RPM of the fan.

4.3.2 Roller/Resistance System

The Rollers/Resistance system is located on the rear wheel of the bike, and it gives direct feedback to user from the simulation. The rear dropout of the bike is supported by a bike stand while the roller system is applied to the rear wheel. The roller system is incorporated on the bike stand as a part of the bike trainer. The resistance works by moving a magnet in proximity to a metal roller. The position of this magnet is controlled by a steel cable which will be moved using a servo controlled by the microcontroller.

4.4 VR Components

The VR Components for this project contains two main parts, the software part of the VR Components is supported with Unity5 program, and the hardware part of the VR components is supported with Google Cardboard.

4.4.1 Google Cardboard

Google Cardboard is being used as the head mount display for this project. It provides the user with a 3D virtualization environment. The processing power of Google Cardboard is supplied by an Andriod phone

(Galaxy S5). The mobile processor takes inputs from on chip phone accelerometer, and microcontroller via Bluetooth(BT). All sensor input information will be used as controller input for the Unity Program. The corresponding field of view will be displayed onto the phone screen.

4.4.2 Unity Program

As the software side of the project, a VR simulation is built with the Unity5 program. User will experience a simulation of riding a bike in the virtual environment, all physical inputs from the bike will be transformed into control inputs, and the certain simulation elements such as the slope and resistance will be used as the output to the bike for user feedback. The simulation program will be running on the smart phone directly.

4.5 Sensors

There are total of three sensors being used in the project in order to track the user's physical movement. The sensor signals are being integrated into simulation inputs.

4.6 Handlebar rotary Encoder

In order to effectively measure the angle of the handlebars, a handlebar rotary encoder will be placed on the head tube of the bike. The voltage reading from the rotary encoder will be probed by the microcontroller in order to calculate the degrees of rotation of the handlebar. The voltage readings will need to be tuned to ensure that voltage ranges read can be quantized into approximate handlebar positions.

4.7 Phone Accelerometer

The built in accelerometer on the smart phone is being used to track the user's head movement. Its input directly feeds into the Unity Program, which is being used to control the simulation field of view. Due to the limitation of the accelerometer, only rotation movement will be support, the linear movement will not be support for the VR experience.

4.8 Wheel IR Sensor

The IR Sensor is installed on the bike stand. The IR sensor functions to record the speed of the rear wheel, the sensor inputs is being used as control input for the simulation as well as the control input for the front fan system. The IR sensor contains both IR emitter and IR receiver, and the emitter and receiver are installed on the opposite side of the rear wheel. The spokes of the bike that passing between emitter and receiver will interupt the IR signal and produce certain signal fluctuation frequency. The frequency of IR sensor readings will be used to calculate the speed of the bike.

5 Software Flowcharts

5.1 Microcontroller Control Flowchart

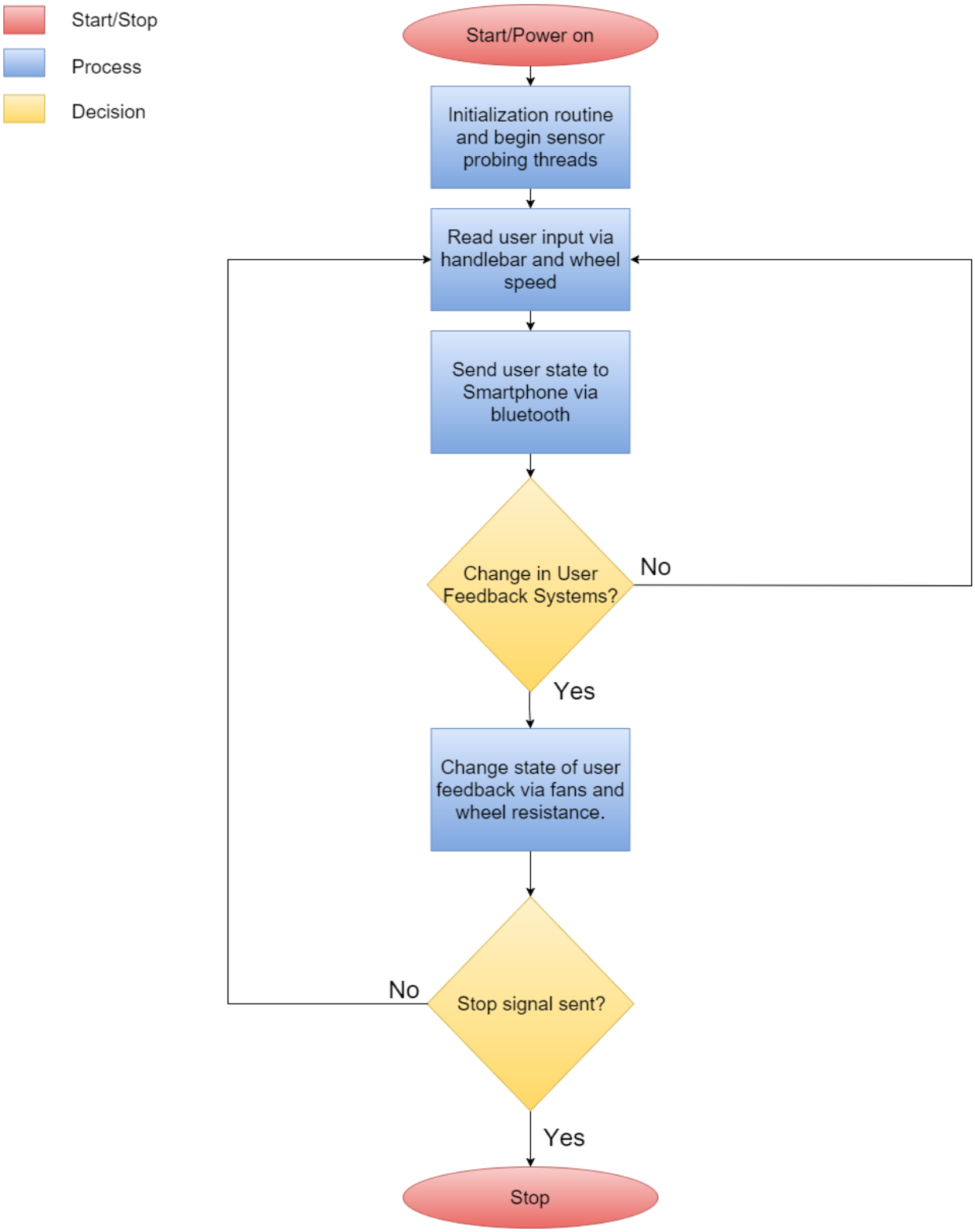

Figure 13: Microcontroller Control Flowchart

5.2 Simulation Software Flow Chart

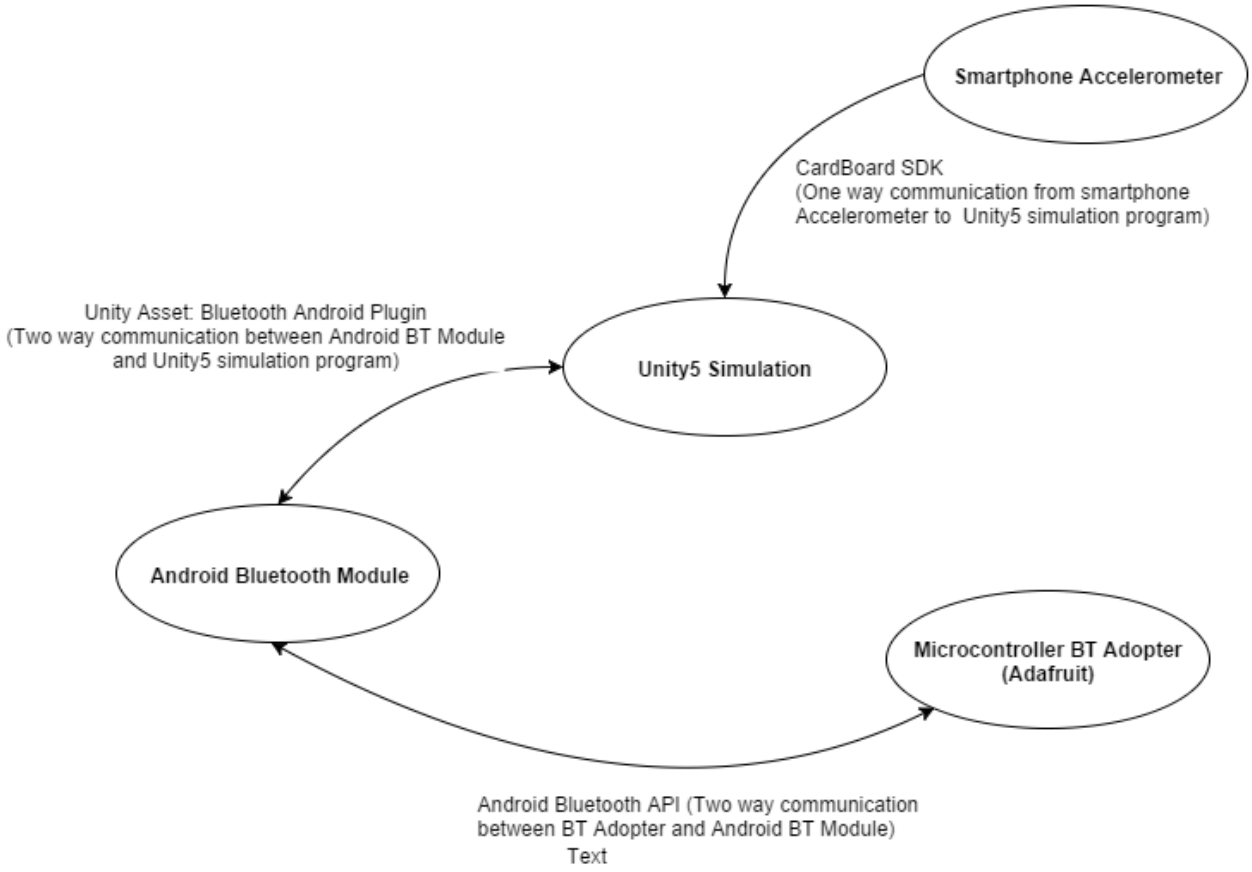

Figure 14: VR Component Logic Flow Chart

5.3 Software module description

5.3.1 Unity5 Simulation program

Simulation program is being built on Unity5 Engine. It contains functions for simulation environment rendering, shading, and texture apply. The Unity5 simulation is the center computation unit for VR simulation software. It takes inputs from both smartphone accelerometer and Android Bluetooth Module, and integrating the input signals into simulation control.

5.3.2 Google Cardboard SDK

Google Cardboard SDK is being used to pare simulation program with Google Cardboard. It uses Android API to extract data from smartphone accelerometer, and directly send the input into the Unity5 simulation program. The Google Cardboard SDK functions as the communication connection between smartphone accelerometer and Unity5 Simulation software.

5.4 Unity Bluetooth Andriod Plugin

The Unity Bluetooth Andriod Plugin is being used in order to integrate smartphone Bluetooth reception into Unity5 simulation program. It connects the smartphone Bluetooth module with Unity5 simulation

software, and the communication goes in both directions which Unity5 simulation program both sending and receiving data to android Bluetooth module via Unity Bluetooth Andriod Plugin.

5.4.1 Android Bluetooth API

Android Bluetooth API works as the data connection between android Bluetooth module and Microcontroller Bluetooth Adopter. The data communication between Microcontroller Bluetooth Adopter and Android Bluetooth Module is established in both ways via Android Bluetooth API.

The communication between Microcontroller BT Adopter and Android BT Module is using Andriod Bluetooth API[5].
The communication between Android BT Module and Unity5 Program Simulation is using Unity Asset (Bluetooth Android Plugin)[6].
The communication between Unity5 Program Simulation and Smartphone Accelerometer is using CardBoard SDK[7].

Part III

Requirements and Verification

6 Components, Requirements, & Testing Verification

Components Requirements	*Testing Verifications*
Voltage Regulator 1. 12 Volt regulator supplies current up to 1.4 A with maximum ripple of ±1%. 36 V maximum input. 2. 5 Volt regulator current up to 1 A with maximum ripple of ±1%. 36 V maximum input. 3. Buck converter supplies load up to 14 W at 100% duty ratio and operates down to 50% duty ratio. Maximum voltage ripple of ±3%	1. (a) Attach 8.5 Ω power resistor as regulator load. (b) Attach oscilloscope probe across load. (c) Supply 36V to regulator from DC supply. (d) Verify voltage remains between 12.6 V and 11.4 V. 2. (a) Attach 5Ω power resistor as regulator load. (b) Attach oscilloscope probe across the load. (c) Supply 36V to regulator from DC supply. (d) Verify voltage remains between 5.05 V and 4.95 V. 3. (a) Attach 10.3 Ω power resistor at converter output. (b) Attach oscilloscope probe across the load. (c) Supply 12V to converter from DC supply. (d) Provide 50% duty ratio 62.5 kHz from microcontroller. (e) Verify voltage remains between 6.3 V and 5.7 V.

Microcontroller 1. Read analog voltages between 0 V and 5 V within ±0.15 V variance. 2. Round trip time for signal sent via Bluetooth less than 25 ms.	1. (a) Using a 5 V voltage supply, drive a testing circuit containing a variable resistor. (b) Verifying voltages across the resistor with a multimeter in comparison with values read from the microcontroller on the same circuit. 2. (a) Sending a test signal from microcontroller to the phone, begin timer. (b) Phone immediately sends acknowledge signal back to the microcontroller. (c) Stop the timer upon reciving acknowledgement signal from the phone.
Bike 1. Support user with maximum of 120kg body weight. 2. Support maximum biking speed of 30km/h.	1. (a) Measuring out 120kg of weight with a scale. (b) Place the weight directly onto the bike with weight mount. (c) Check the stability of the system. 2. (a) Installing bike speedometer in order to measure the wheel speed of the bike. (b) Testing member riding the bike, increasing the wheel speed gradually until reaches 35km/h. (c) Check the stability of the system.
Bike Trainer/Roller Resistance System 1. Driving servo rotating into 3 positions with 20 degrees rotation variance between each position. 2. Driving servo can apply at least .15 Nm of torque.	1. (a) Using microcontroller and write several testing angle with 20 degress difference to the servo. (b) Measuring the servo rotation angle with a protractor. 2. (a) Bring the servo to a neutral position. (b) Attach a weight of .1 kg to a lever arm of .15 m to the servo. (c) Attempt to drive the servo to rotate said lever arm.

Fan Controller 1. Zero crossing circuit detects all zero crossings and provides pulse width of at least 0.05 ms. 2. Ac fan operates at four distinct speeds seperated by 20 Vrms ± 5 Vrms. 3. Dc fan operates at three distinct speeds. Maximum speed is at 11,000 RPM and each speed setting is 200 RPM ± 50 RPM from the last setting.	1. (a) Attach oscilloscope probe across digital output. (b) Provide wall power to circuit. (c) Confirm on oscilloscope that time between pulses are consistent and are about 8 ms apart. (d) Measure width of pulse on oscilloscope and verify that it is at least 0.05ms across. 2. (a) Attach power meter probes across. (b) Provide circuit with wall power. (c) Activate microcontroller test routine to test 4 different time delays. (d) Record the rms voltage supplied to the fan for each time delay setting. (e) Verify each speed setting provides between 15 and 25 Vrms difference from the last. 3. (a) Attach oscilloscope probe across dc fan pulse output setting. (b) Provide 12V to buck converter circuit. (c) Activate microcontroller test routine to test 3 different duty ratios. (d) For each setting record fan sense pin pulses on oscilloscope. (e) Measure time between pulses and invert the time and multiply by 60 to get the RPM. (f) Verify that the RPM is at 1100 at maximum output and verify each setting is between 150 and 250 RPM different from the last.
Unity Simulation Program 1. Achieve minimum of 30 frames per second.	1. (a) Run Unity Simulation program with testing personal riding the bike. (b) Verify with frame rate displayed on the screen.

GoogleCardboard VR System 1. Achieve 1 hour of continuous usage with fully charged battery.	1. (a) Fully charge the test phone. (b) Run Unity Simulation program and start the timer. (c) Record the time when the test phone turns off due to lack of power.
Handlebar Encoder 1. Maximum rotation of 120 degrees in both directions (clockwise and counter clockwise) in reference to neutral handlebar position. 2. Angle resolution of 5 degrees. 3. Angle accuracy with $\pm$ 3 degrees.	1. (a) Calibrate encoder using neutral handlebar position (Neutral position is when the the handlebar is orthogonal to the frame, and the front and back wheels are in line). (b) Rotate handlebar 120 degrees clockwise, verify microcontroller readout with protractor. (c) Rotate handlebar 120 degrees counterclockwise, verify microcontroller readout with protractor. 2. (a) Calibrate encoder using neutral handlebar position (Neutral position is when the the handlebar is orthogonal to the frame, and the front and back wheels are in line). (b) Rotate handlebar 5 degrees in both clockwise and counterclockwise direction. (c) Observe microcontroller readout changes. 3. (a) Calibrate encoder using neutral handlebar position (Neutral position is when the the handlebar is orthogonal to the frame, and the front and back wheels are in line). (b) Rotate handlebar 5 degrees in both clockwise and counterclockwise direction. (c) Verify angle reading accuracy with protractor measurements. (d) Rotate handlebar 120 degrees in both clockwise and counterclockwise direction. (e) Verify angle reading accuracy with protractor measurements.

Phone Sensors 1. Detect rotational acceleration with error of 5% in all three axis. (Yaw, Pitch, Row) 2. Drifting error within 25 degrees in a period use of 20 minutes.	1. (a) Running testing program, set the default orientation. (b) Rotate phone 360 degrees of the x-axis, verifying with default orientation. (c) Rotate phone 360 degrees of the y-axis, verifying with default orientation. (d) Rotate phone 360 degrees of the z-axis, verifying with default orientation. 2. (a) Running testing program, set the default orientation. (b) Run the testing program for 30 minutes. (c) Return to default orientation, measure angle drifted.
Wheel IR Sensor 1. Detect wheel speed of the bike with accuracy of $\pm$ 2 km/h. 2. Minimum speed detection achieve 3 km/h. 3. Maximum speed detection achieve 30 km/h.	1. (a) Install bike speedometer to measure the speed of bike wheel. (b) Testing personal ride the bike, with speedometer reading of 2 km/h. (c) Verifying bike wheel speed reading from microcontroller. (d) Testing personal ride the bike, with speedometer reading of 30 km/h. (e) Verifying bike wheel speed reading from microcontroller. 2. (a) Install bike speedometer to measure the speed of bike wheel. (b) Testing personal ride the bike, with speedometer reading of 3 km/h. (c) Verifying bike wheel speed reading from microcontroller. 3. (a) Install bike speedometer to measure the speed of bike wheel. (b) Testing personal ride the bike, with speedometer reading of 30 km/h. (c) Verfying bike wheel speed reading from microcontroller.

7 Points Summary

Module	***Requirement***	***Points***
Power	***Requirement***	15
Control	***Requirement***	10
VR System	***Requirement***	10
Sensors	***Requirement***	10
User Interface	***Requirement***	5

Part IV

Tolerance Analysis

7.1 Simulation Latency

To meet reliability goals, the latency for the microcontroller system needs to be kept under reasonable limit. For any VR experience, a total system latency of $50 milliseconds$ will feel responsive, but still noticeable lagging, and 20 milliseconds or less will provide the minimum level of latency deemed acceptable. Despite the latency due to mobile processor rendering, the latency of the microcontroller signal processing needs to be limited under 50 miliseconds. The latency of the sensor signal plays an important part. Both smart phone accelerometer and handlebar encoder output are read constantly, however the rear wheel IR sensor requires at least two IR reading peaks in order to calculate the speed of the bike. The IR sensor systems need to specifically designed in order to meet this requirement. The diameter of the bike rear wheel is $0.66\, meters$ ($25''$ Bike), therefore the circumference of the wheel is $c = \pi * Diameter = \pi * 0.66\, m = 2.07\, m$. Given a minimum initial speed of 5 km/h, the rotational speed of the wheel is $\omega = c/v = 2.07/5000 = 0.00105\, hr/rotation = 1.49\, s/rotation$. This gives total amount of 30 reflecting positions on the wheel in order to give out IR readings every $50\, milliseconds$. Therefore, the spokes of the rear wheel will be put into use, and each spoke will be used as an IR reflecting position for the IR sensor.

7.2 Accelerometer drifting error

Due to the smartphone accelerometer is the only sensor that being used to contribute to the head tracking, it's function and correct data output is essential for the VR experience. The smartphone accelerometer itself contains a drifting error, and without outer sensor source correcting, this drifting error adds up and makes significant impact to the VR simulation. The figure 15 below shows an accelerometer drifting error within 1400 seconds. The data was collected by Oculus developing group. The drifting error without correction shows over 80 degrees of drifting over 1400 seconds period of usage is not acceptable for this project. Therefore, a software correction algorithm is being added into the simulation in order to reduce the drifting error. By considering the natural of bike riding, the user is spending most time looking forward (focusing on the road). Taking advantage of this behavior, the program will automatically recenter the simulation camera if the data input from accelerometer stays relatively unchanged. The recenter algorithm also considers the speed of the bike, and current situation, and it will not recenter when user stops on the road and looking around in the simulation. In addition to self correcting algorithm to solve the accelerometer drifting problem, an additional physical reset button is being added for user to recenter their simulation camera in any situation[8].

drifting error.png

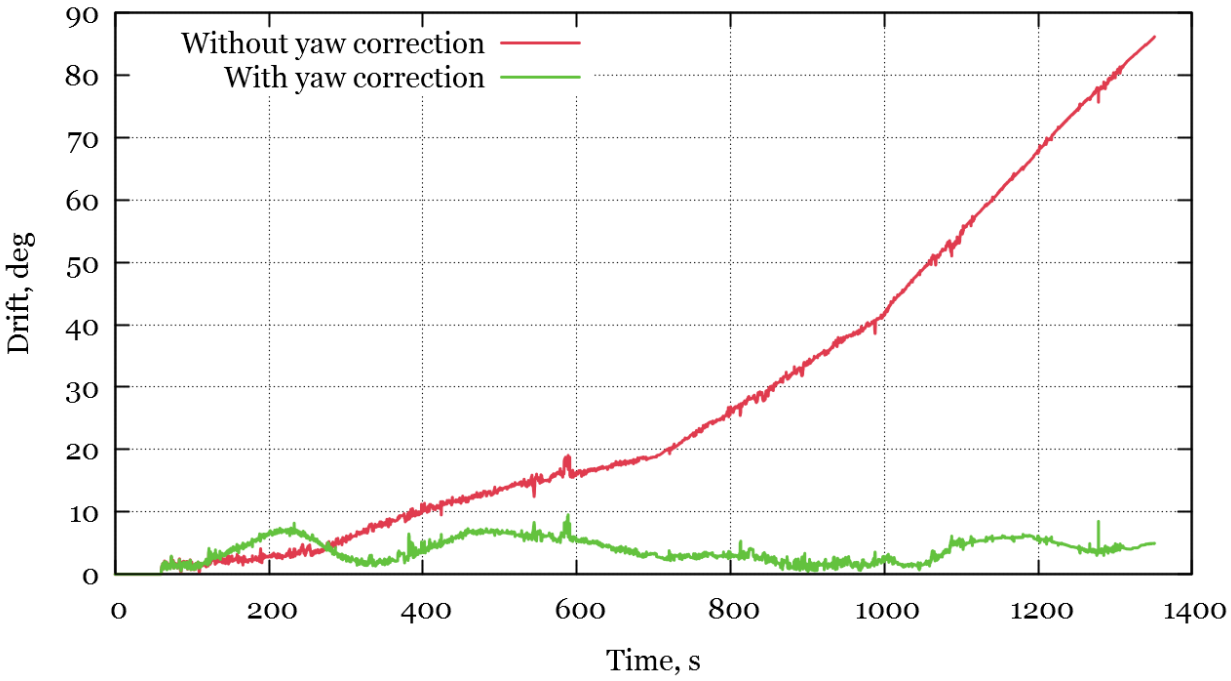

Figure 15: Accelerometer drifting error

Part V

Cost Analysis

8 Cost Analysis

8.1 Labor

Labor	**Cost**
Wage Per Hour * 2.5	$78.75
Hours Per Week (hours)	20
Number of Weeks (Weeks)	16
Number of Workers (Person)	3
Total	$75,600.00

8.2 Parts

Description	***Manufacturer***	***Part Number***	***Cost***	***Number Ordered***	***Total Cost***
Bike	Dynacraft	n/a	$109.97	1	$109.97
Samsung Galaxy S5	Saumsung	n/a	$467.99	1	$467.99
IR Reflectiv e Sensor	Adafruit	2167	$1.59	2	$3.18
Google Cardboard	Knox Lab	KNOX V2	$15.00	1	$15.00
Arduino Uno	Aruduino	n/a	$24.95	1	$24.95
ATMega328	Amtel	n/a	$3.70	2	$7.40
Bluetooth Adaptor	Adafruit	nRF8001	$19.95	1	$19.95
Bike Trainer	FDW	n/a	$79.99	1	$79.99
Servo	Hiteck	hs311	$7.99	1	$7.99
Fan(4 pack)	Coolermaster	R4-S2S-124K-GP	$12.99	1	$12.99
Rotary Encoder	Allied	RV4NAYSD103A	$7.81	1	$7.81
High side driver	Infineon Technologies Americas Corp.	IRS2124STRPBF	$2.31	2	$4.62
Voltage regulator	Active-Semi	ACT4514SH-T	$0.72	5	$3.60
Transformer	White Rodgers	90-T40F3	$11.99	1	$11.99
N Channel power mosfet	Fairchild	FQP30N06L	$0.95	3	$2.85
10 uF capacitor	TDK	FK20X7S1H106K	$1.06	2	$2.12
8200 uF capacitor	Nichicon	LGU1H822MELB	$3.89	2	$7.78
140 kOhms resistor	Yageo	MFR-25FBF52-140K	$0.10	3	$0.30

Optocoupler	Everlight	H11AA1M	$0.55	2	$1.10
Diode	Fairchild	1N4148	$0.10	3	$0.30
68 uF capacitor	Nichicon	RR71C680MDN1	$0.66	5	$3.30
Power entry module	Qualtek	762-18/002	$9.46	1	$9.46
100 uH inductor	Bourns	6100-101K-RC	$0.77	4	$3.08
Total Components:					$807.72

8.3 Grand Total

Components	$807.72
Labor Cost	$75,600.00
Grand Total	$76,407.72

Part VI

Schedule

9 Schedule

Week	Task	Member Responsible
2/8/2016	Working on project proposal Requirement and Verification table and Tolerance analysis section	Chongxin
2/15/2016	Order necessary components VR: Research for all required APIs and SDKs for the software implementation Power: Create initial power converter design	Greg Chongxin Bryant
2/22/2016	Control: Create microcontroller control flow VR: Creating simulation character control Power: Design fan control circuit	Greg Chongxin Bryant
2/29/2016	Assemble base bike controller Power: Build and test prototype voltage regulator on breadboard. VR: Creating simulation environment	Greg Bryant Chongxin
3/7/2016	Control: Create routine for reading from handlebar angle VR: Research for BT communication API Finalize first pcb design	Greg Chongxin Bryant
3/14/2016	Power: Write code for Fan controller VR: Writing BT communication driver for unity5 program Control: Create routine for reading from wheel IR sensor	Bryant Chongxin Greg
3/21/2016	Build roller resistance breaking system Power: Fan assembly and testing VR: Write BT communication driver for Android system	Greg Bryant Chongxin
3/28/2016	Control: Microcontroller Bluetooth Testing and Debugging PCB assembly and debugging. VR: Integrating BT signal into simulation control control	Greg Bryant Chongxin
4/4/2016	Add sensors and microcontroller to bike Control: Testing and debugging control system VR: Testing sensors BT input, integrating with simulation control	Bryant Greg Chongxin
4/11/2016	Prepare for mock demo Finalize bike controller with VR Control: Optimive routines to achieve low latency feedback	Bryant Chongxin Greg
4/18/2016	Testing and debugging with bike controller Calibrating bike controller with VR for best performance Create circuit housing for pcb and sockets.	Greg Chongxin Bryant
4/25/2016	Prepare final paper Prepare for final presentation Optimize user interface and Resolve issues	Chongxin Bryant Greg

5/2/2016	Finish final paper Final presentation Lab Checkout	Chongxin Greg Bryant

Part VII

Ethics and Safety

10 Safety Statement

The members of Virtual Cycling Reality strive to adhere to the safety rules and procedures outlined in the IEEE code of ethics. During the project lifecycle it is our goal to maintain a safe and productive work environment while minimizing potential risk to ourselves and equipment. This goal shall be accomplished by extensive communication/clarification within the group and also with course staff concerning lab and equipment safety. As Electrical and Computer Engineers in the process of designing a product it is imperative that we as a group understands the possible hazards and dangers that might arise. Therefore we as a group shall maintain the habit of researching all tools and components, as to better understand the risks associated with using them. Below are some of our groups objectives as we seek to accomplish our goal.

1. Research and understand the risk of components used in our design.
2. To identify those risks, consider an appropriate response, and to note the response in our safety manual.
3. Properly cite the designs and works of others.
4. To inform course staff in the event of broken or malfunctioning equipment.
5. Implement safe and cautious testing of our design, including:
   (a) Not eating or drinking in the lab.
   (b) Wearing appropriate attire when working in the lab.
   (c) Ensure that all equipment is turned off and put away correctly before leaving the lab.
   (d) Prevent damage to ourselves and our equipment by ensuring that circuits are grounded and checking the safety rating on our equipment.
6. To ask questions about unfamiliar tools or equipment to a course staff member, prior to using said items.
7. Be able to recognize a state of emergency and to seek help from emergency responders.
8. Saftey warning are being added into the VR simulation, which states with any uncomfortable feelings such as motion sickness, disorientating, and vision blurry should stop simulation immediately.
9. Protection gears are being provided during simulation, which include safety helmet, knee pads and elbow pads.
10. VR simulation should only be experienced with at least one supervisor present. The supervisor is being charged on modifying entire system stability and giving user any physical help during emergency.

To conclude, the objectives and procedures mentioned here were conceived in adherence to the IEEE code of ethics. Throughout the semester we will strictly comply to these guidelines in order to ensure the completion of our project in a safe and secure manner.

11 Ethical Consideration

1. All designs and functionality of this project is being designed with the complete physical safety of the consumers in mind. We will, to the best of our ability, reduce the risks of all physical harms.
2. The VR simulation is being designed with the safety and user experience in mind. It is being tested fully to minimize all possible uncomfortable experiences to the consumers.
3. Safety protections will be provided to the consumers when using the VR demo, including safety helmet, knee pad set, elbow pad set.
4. VR simulation contains user friendly information, the simulation is being designed to be suitable for users from variety backgrounds.
5. Due to the natural of VR simulation, people that are under age 12 are not being recommended to using the VR simulation. The maximum time usage of the simulation is recommended keep under 20 minutes per usage.

Part VIII

References

References

[1] Act4514 datasheet. [Online]. Available: http://www.alldatasheet.com/datasheet-pdf/pdf/430740/ACTIVE-SEMI/ACT4514.html

[2] Arduino forum. [Online]. Available: https://forum.arduino.cc/index.php?topic=310753.0

[3] Moc3012 datasheet. [Online]. Available: http://www.ti.com/lit/ds/symlink/moc3012.pdf

[4] Zero-crossing detectors circuits and applications. [Online]. Available: http://www.bristolwatch.com/ele2/zero_crossing.htm

[5] Android bluetooth api. [Online]. Available: http://developer.android.com/guide/topics/connectivity/bluetooth.html

[6] Unity android bluetooth plugin. [Online]. Available: https://www.assetstore.unity3d.com/en/#!/content/15179

[7] Google cardboard api. [Online]. Available: https://forum.arduino.cc/index.php?topic=310753.0

[8] Magnetic calibration. [Online]. Available: https://developer.oculus.com/blog/magnetometer/