ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

Stepper Machine Power Generation: Final Report

Team #46

ZACH DEARDORFF
(zjd3@illinois.edu)
JOOSEUNG KIM
(jsk4@illinois.edu)
JAYDEN CHO
(beomkic2@illinois.edu)

TA: Qingyu Li

May 4, 2022

Abstract

This final report describes our Stepper Machine Power Generation system. We begin by introducing the problem we are trying to solve, and continue with describing our solution. We then define our high-level requirements that our solution needs to achieve to be considered a successful project. We then will go into our subsystem requirements and design, including how well we succeeded in regards to those requirements. Finally, we will conclude with a cost analysis of our project, the ethics and safety of our project, and what we would change in the future for our project.

Contents

1 Introduction

1.1 Problem

Exercise in our era is difficult to fit into our busy schedules. Even when we can find the time to exercise, sometimes we are unmotivated or it comes at an opportunity cost. The pandemic also increased the amount of people working remotely from their desks at their homes. Sitting around is just as much of a killer as smoking is. It has been found that people who sit for more than 13 hours a day are actually at a 200% higher risk of death when compared to people who sit for only 11 hours or less per day [1]. However, we are forced to sit sometimes, and it would be helpful to get some exercise while sitting and doing work.

1.2 Solution

A sitting exercise step machine that could generate electricity by an up and down movement of the legs would be a viable exercise in an office setting, because it leaves the arms free by allowing you to use the machine without your hands. Also, the exercise is not intense, so you are not sweating while sitting or standing in an office. Finally this exercise machine can use the movement of legs in order to generate some electricity, giving users a sense of accomplishment. The machine could also fit under a desk to be convenient to use at work.

The exercise machine we plan on creating is a step motion machine that can be used while sitting. The steps will be able to be converted into electrical energy by connecting the stepper machine to a DC motor. We plan on using this electrical energy to efficiently charge a portable (5V) battery. The idea of charging a portable charger made sense because then the user could still use their phone during the workout/day and after work be able to take the portable charger with them to charge their phone. Charging a portable battery may not be enough incentive, so we also plan on connecting the machine to a computer to be able to remind the user to use the machine throughout the day.

1.3 Visual Aid

Below in Figure 1 is our fully integrated system with the stepper machine, pressure sensor, PCB, and portable charger connected.

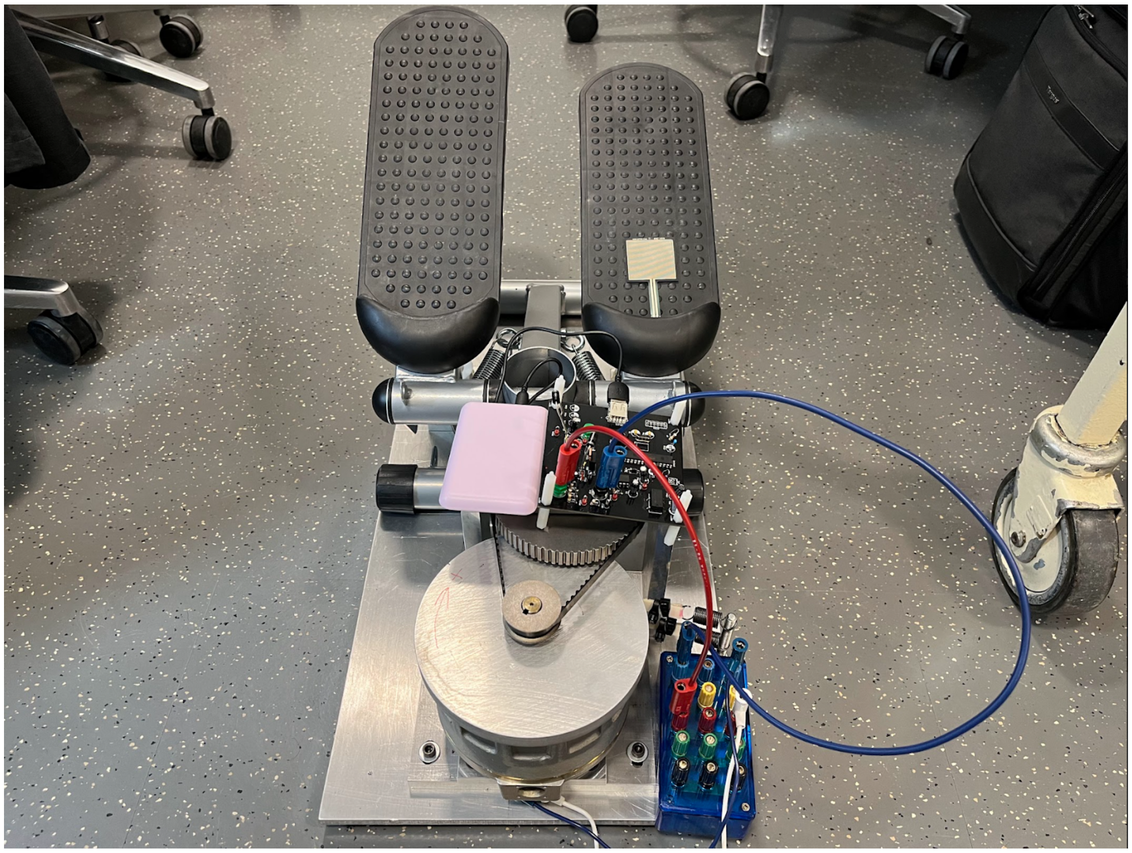


Figure 1: Stepper Machine Power Generation Integrated System

1.4 High-Level Requirements

1) The power electronics, our DC-DC converter, needs to be able to convert the electrical energy generated from the motor and stepper into a constant 5V to supply the output to the portable charger within 5

2) Machine can be used while sitting, and small enough so that it can fit under a desk (About 36 inches deep, around 30 inches tall, and minimum width for a person of around 24 inches [2]).

3) The pressure sensor and computer program system are able to reinforce working out at least 8 times a day (Once every hour of an 8-hour workday).

1.5 Subsystem Overview

There are three major blocks for our project. The first being our mechanical stepper-generator system that is able to produce electricity. The second being our DC-DC converter that is able to take the input from the DC motor and convert it to a constant 5 V to charge a portable charger. Finally, our third major subsystem is our computer application that interfaces with the microcontroller and the pressure sensor to remind the user when they should work out and use the machine. This next section further describes our subsystems and how they interconnect. Below is the block diagram for our project in Figure 2.

1.5.1 Block Diagram

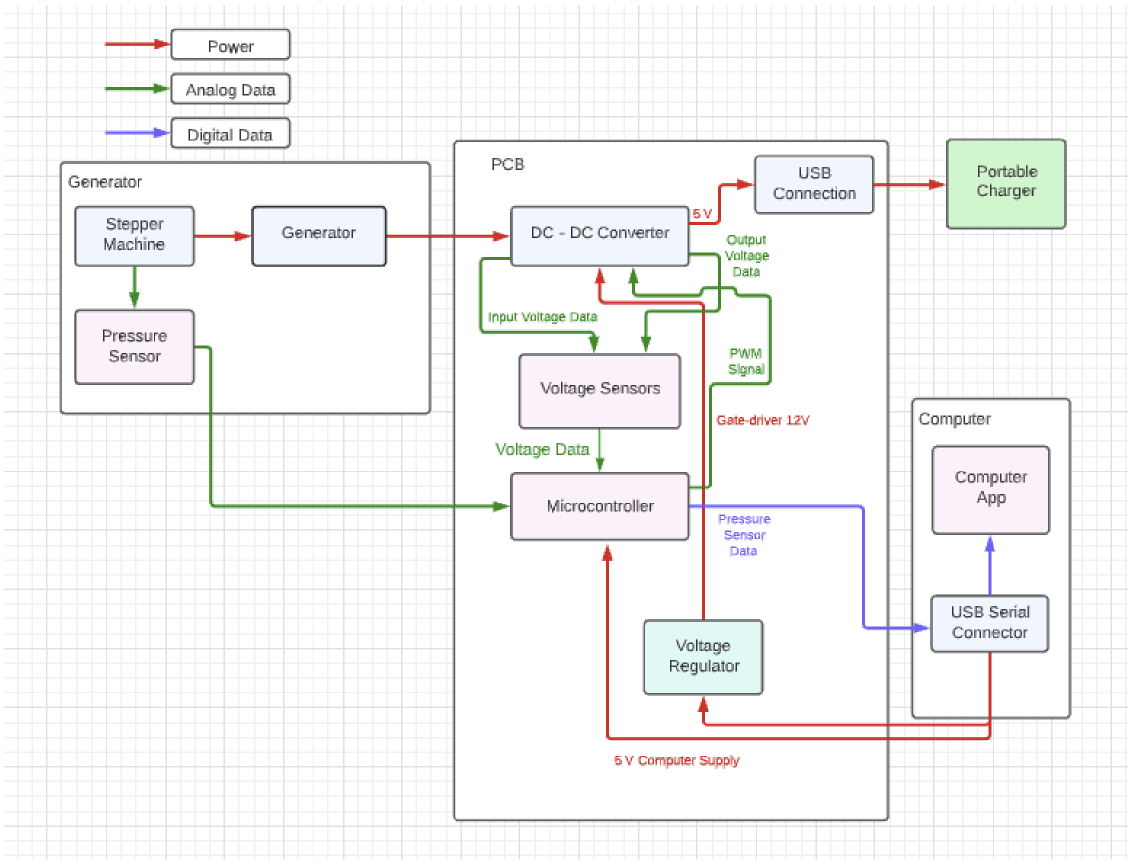


Figure 2: Block Diagram for Stepper Generator System

1.5.2 Mechanical Stepper System

Stepper Machine:

To generate power we needed a mechanical system that can convert the stepping motion into electrical energy. We did this by connecting our stepper machine to a flywheel contraption that was able to turn the DC motor with each step of the machine. With each step of the machine we were able to generate electricity from the DC motor.

Generator:

For our electricity generation we used a Minertia Motor P12-H. This motor is a DC motor that is relatively small and is still able to generate a lot of power. This motor was perfect because it allowed us to make our machine small enough to fit under a desk which was very convenient as designed. As mentioned before, the generator was connected to the stepper using a flywheel contraption that spun the motor with each step. The output of our generator was then connected to the input of our DC-DC Converter and the PCB with wires for the positive and negative terminals.

Pressure Sensor:

We needed a pressure sensor on the stepper machine so that we could be able to tell if the user was using the machine or not. We used a force sensitive resistor for our pressure sensor. This allowed us to use a simple voltage divider circuit to sense if pressure was being applied to the stepper machine. The pressure sensor circuit is connected to an input of the microcontroller. The actual pressure sensor was located on the top of the left step so that when the user put their feet on the machine to workout force would be applied to the sensor and a voltage signal would be sent.

1.5.3 DC-DC Converter and PCB

DC-DC Converter:

The input voltage from the stepper-generator system needs to be converted to a steady $5V$ output. A flyback converter design was created in order to achieve this output. The converter was designed to take in an input of a range between $3-10V$ with an output ripple of 5%. The output of the DC-DC converter is connected to an USB connector that the portable charger can then be plugged into to charge.

Microcontroller:

The microcontroller on the PCB controlled two main things. It received the input and output voltages of the DC-DC converter and then it calculated the duty ratio needed in the Pulse Width Modulation (PWM) signal being sent to the gate driver for the DC-DC converter based on those input-output values. This insured that no matter the input voltage, the output voltage would remain a constant $5V$. It also served to take in the input signals from the pressure sensor and send it to the computer. This was necessary so that our computer program would recognize when the user was actively exercising on the stepper machine. The microcontroller we used on our PCB was the ATMega328P.

This microcontroller was then connected to the computer through a USB Serial Decoder. Unfortunately, the serial decoder part never arrived so we instead for testing had to use an Arduino for testing our PWM control and pressure sensor connection. Luckily, the microcontroller on the Arduino is the ATMega328 so our code would still work as intended for the microcontroller if we were able to communicate with it using the USB serial decoder.

Voltage Sensors:

Voltage sensors are needed to be able to sense the input and output voltages for our DC-DC converter. These voltage sensors are needed so that the microcontroller can accurately control the PWM duty cycle so that the output voltage can stay a constant 5V. The voltage sensors are simple resistor divider circuits that step-down higher voltages to below 5V so that our microcontroller could take the voltages as inputs. The microcontroller took the inputs and then converted them back to the real values through some gain calibration.

Voltage Regulator:

A voltage regulator is needed to power our gate driver chip that is a part of our DC-DC converter circuit. This voltage regulator needed to take an input of 5V (the power being supplied to the rest of the board from the USB serial decoder connection) and output a constant 12V to supply our gate driver chip. We used the 'PDS1-S5-D12-M' as our voltage regulator since it can take input from 4.5V to 5.5V and have voltage output of -12V or 12V.

1.5.4 Computer Application

A key requirement for our machine is that it is able to remind users to workout throughout the day. For this we designed a computer application that took in data about the pressure sensor from our microcontroller through the serial decoder. The computer application would first ask the user for an input for how long they want to workout. Then a workout timer would appear and would only countdown when the user was applying pressure to the pressure sensor and using the machine. Once the workout was completed a new "rest" timer would appear that would be an hour long usually. This allowed the user to rest and then be reminded to workout again once their rest was over. This would cycle eight times a day to ensure that the user completed multiple workouts at different times of the day. This ensured that the user wasn't simply sitting the entire day.

2 Design

The Design section will focus on the design work we had to do for all of our major subsystems. The major equations and simulations we performed will be discussed as well as major decisions we made in the design.

2.1 Stepper-Generator Mechanical System

To first begin designing our mechanical stepper-generator system we first needed to figure out the minimum rpm needed for the motor to actually be able to produce a usable DC voltage. To do this we connected our DC motor to a machine that could spin the motor at varying rpm. From this testing we were able to graph the voltages generated at lower rpm. This graph can be seen below in Figure 3. From Figure 3 we are able to see that at the minimum we would need to spin our motor at around 200 rpm to be able to use the voltage from the DC motor.

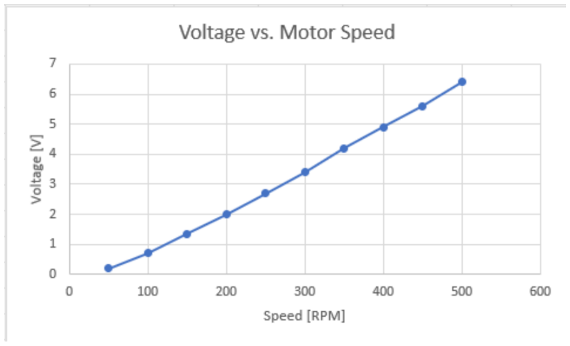


Figure 3: DC Motor at low rpm

Now that we had the minimum rpm that was needed we were ready to update the machine shop with the necessary specifications for the stepper machine. After a few weeks the machine shop was able to complete our stepper-generator system with a contraption that was able to turn a flywheel with every step of the stepper machine. The flywheel was then able to turn the DC motor fast enough to exceed the minimum rpm we needed.

2.2 Flyback (DC-DC) Converter

There were many options to choose from for the design of our DC-DC converter. We needed a converter that would be able to both step-up and step-down the output voltages from the stepper-generator system. The reason we needed a converter that could both step-up and step-down voltages is because when we began testing the stepper-generator system we found that at low stepping speeds we would get voltages below 5V that would need to be stepped up to 5 V and at high stepping speeds we would generate voltages above 5V that would then need to be stepped down to 5V. We had two choices for converters that we had previously learned about that could both step-up and step-down voltages [3]. The first was a buck-boost converter. This converter was a relatively simple design that cascaded a buck converter with a boost converter where two different duty ratios would be able to control what the output voltage would be. We found that having two different duty ratios to control would be hard to keep track of for our control system so we decided to look further into our options. We found that our second option a fly-back converter had two major advantages. It was able to be controlled using one duty ratio with only one switching MOSFET. With only one switching MOSFET this would also mean there would be less switching losses in the system which meant it would be more efficient than the buck-boost converter. The second inherent advantage to the fly-back converter was that it isolated the input and output sides of the circuit due to its use of a transformer in the circuit. We ended up deciding to use the flyback converter because of these two reasons. In Figure 4 the circuit for our flyback converter can be seen.

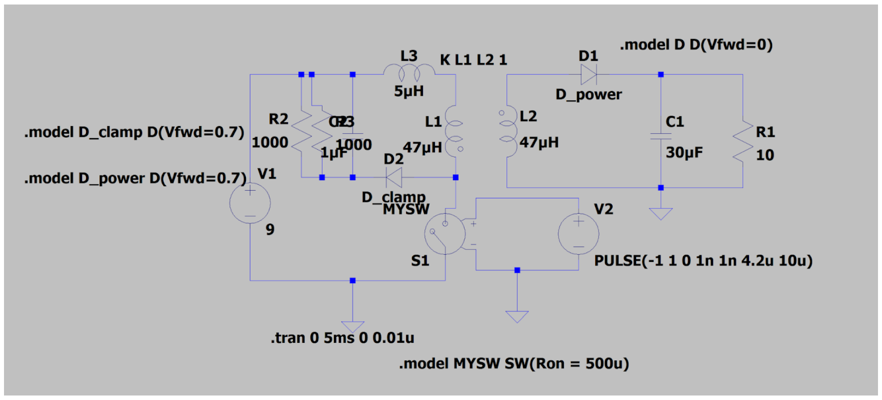


Figure 4: Flyback Converter Schematic

The way a flyback converter works is that a PWM signal with a certain duty ratio drives when the MOSFET switch on. This causes the input side of the circuit turns on and off. When the MOSFET switch is on the input voltage is able to be sent through the transformer which is able to store that energy as an inductor. When the switch is off the diode on the right side of the circuit can turn on and the voltage is sent to the output. The output capacitor needs to be large enough to remove any ripple in the output voltage. There are inherent inductances in the wiring of the circuit which can cause the current to run away with sudden changes in voltage. This is why a clamp circuit is needed within the circuit to clamp down on sudden changes in the input voltage. There were two options for a clamp circuit in our converter; a zener diode clamp circuit and a RC clamp circuit. Ultimately, we choose a RC clamp circuit, which can be seen on the left side of the circuit, because it dissipates less power when compared to the zener diode option. The duty cycle is what we have control over to effect the input-output voltage relationship. Equation 1 shows how the input-ouput voltage relationship with respect to the duty cycle. The variable n is the turns ratio for the transformer, which in our case is simply 1. It can be seen that when the duty ratio is greater than .5 the voltage can be stepped up and when it is less than .5 the voltage can be stepped down.

$$\frac{V_{out}}{V_{in}} = \frac{nD}{1-D} \tag{1}$$

The major parts of the flyback converter we needed to design for our specific input and output voltage specifications included the clamp circuit, the output capacitance, the magnetizing inductance for our transformer, and our switching device ratings [4]. We designed our converter with the specifications of the input ranging from 1.5V to 9.5V and an output of 5V with a 5% tolerance. The first thing we designed was our RC clamp circuit. To do this equations 2 and 3 below were used to find the value of the resistor needed for the clamp circuit. Where D is duty ratio, T is the time for one cycle ($1/f_{sw}$), and L_m is the magnetizing inductance that will be discussed next.

$$t_{reset} = \frac{V_{in} * DT}{V_{in} - V_C} \tag{2}$$

$$R_C = \frac{V_C^2 * 2 * L_m * f_{sw}}{V_{in}^2 * D^2} \tag{3}$$

Equation 2's purpose is to find the voltage that V_C needs to be so that it can be used to find the resistor value. V_C is the value of the voltage across the capacitor and it needs to be larger than V_{in} so that the clamp circuit can be successful. The value for t_{reset} was chosen to be $1 * 10^{-6}$ seconds so that the clamp was faster than (1-D)T. The math was worked out in a lab notebook that can be seen in Appendix A, Figures 24 and 25. The resistor value needed was found to be around 528.75 Ohms from equation 3. In our design we decided we would use two 1.1k Ohm resistors in parallel to roughly achieve this value. Using two high value resistors in parallel actually helped our power dissipation a bit. Then to find the capacitor value we simply need $R_C * C$ to be greater than 10T which is equal to $1 * 10^{-4}$ seconds. A 1μF capacitor was chosen to achieve this in our design.

Next we designed the magnetizing inductance L_m that would be needed for our circuit. It is important to have a high enough inductance to ensure that your circuit doesn't run in discontinuous conduction mode. To solve for this we used equation 4 below to find a minimum value of L_m that could be used.

$$L_m = \frac{V_{in} * D * (1-D)}{2 * f_{sw} * I_{out}} \tag{4}$$

Using the above equation we were able to find that the minimum value of L_m that should be used should be around 21 μH. The math carried out for this calculation can be seen in the lab notebook found in Appendix A, Figure 26. We found a 1:1 transformer for our design that had an inductance of 67μH which we decided would be perfect for our design.

We then needed to design our output capacitor to ensure that our ripple was small enough that we were inside our tolerance of 5% of our 5V goal. Equation 5 below is how we designed our output capacitor. ΔV is the maximum ripple we want to have which in our case is half of 5% of 5V or .125V and f_{sw} is the switching frequency of our circuit which is 100kHz.

$$C \geq \frac{P_{out} D}{V_{out} \Delta V f_{sw}} \tag{5}$$

From this eqaution we were able to find that the minimum output capacitor we needed would be 2.56 μF. We decided that for our actual capacitor we would use should be a lot higher to ensure that our ripple is much higher which is why we increased our final capacitor to be 30μF.

We performed some simulations for our converter to ensure that it worked as intended with ideal components. In Figure 5 a simulation is showed where the input in red is seen to be a constant 6 V and the output of our converter does in fact become 5V with a very small ripple. This shows that our converter works as intended in simulation. In Appendix A, Table 1 shows the duty ratios that were used in simulation to achieve an output of 5V for differing input voltages.

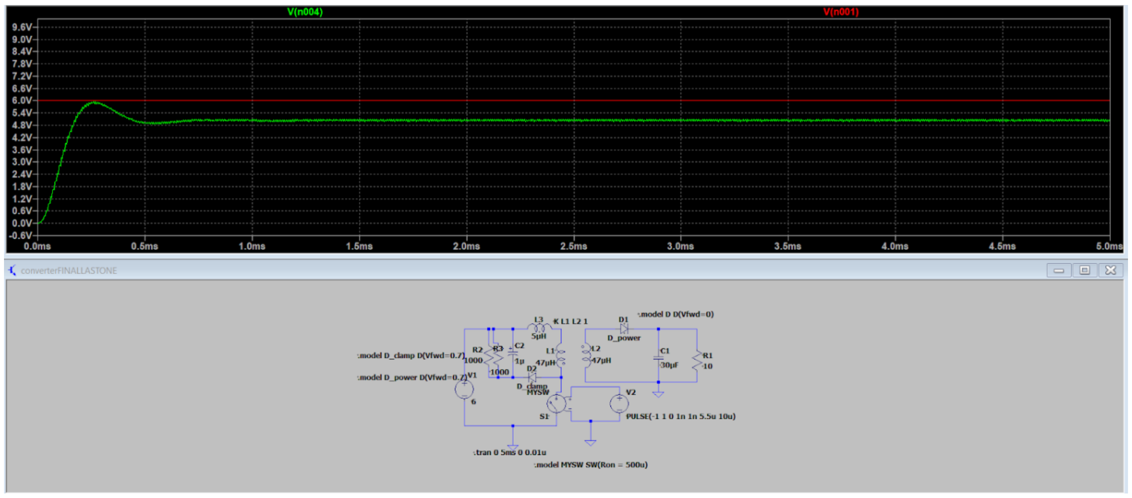


Figure 5: Flyback Simulation: Input of 6V

2.3 Pressure Sensor

For the pressure sensor we needed to figure out how we would be able to send a single to the microcontroller if someone was on the pressure sensor or not. This turned out to be a relatively easy thing to do since the pressure sensor we choose was simply a force sensitive resistor. This meant that we could simply use a resistor divider circuit where the pressure sensor resistance would be so high when no pressure was being applied that the microcontroller would see 0V as an input. Then when pressure was applied to the force sensitive resistor the resistance would be low so the microcontroller would see a higher voltage around 5V. This allowed us to use 5V as an ”on” voltage and the 0V as an "off" voltage. The circuit we designed for the pressure sensor can be seen below in Figure 6.

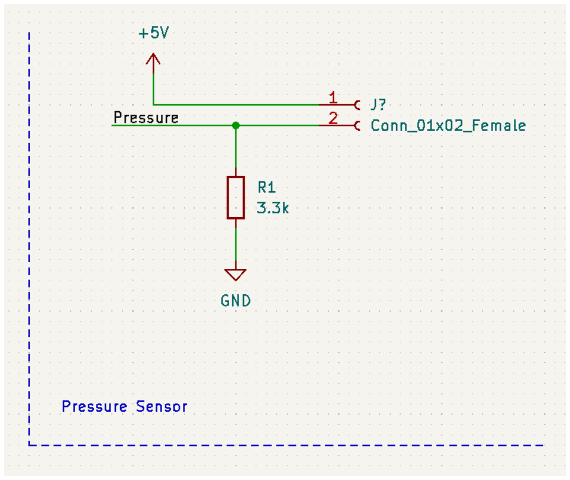


Figure 6: Pressure Sensor Schematic

It can be seen that we choose a 3.3k ohm resistor as our other resistor in our voltage divider circuit. This choice was made because we wanted it to be a lot smaller than the resistance when no pressure was being applied while also large enough to have low power dissipation through the resistor. It should be noted that changing this resistor value changed how sensitive the pressure sensor circuit could be in estimating the force on the pressure sensor. However, since we simply needed to know when any force was being applied this resistor value worked well because it was responsive enough to tell when even relatively small pressures where being applied to the circuit.

2.4 Voltage Sensors

We needed voltage sensors for the input and output of our converter so that the microcontroller could use the voltage values to determine the duty ratio needed to output a constant 5 V. The input and output voltages are necessary as inputs to our PWM control algorithm the microcontroller will be running to control the output voltage. Since the input and output may be greater than 5.5 V which is the tolerance of the microcontroller we decided we would need voltage dividers to map the higher voltages to voltages that are within the tolerance of our microcontroller. To do this we used equation 6 below to decide which resistor values would be needed to map the voltages correctly [5].

$$V_{out} = \frac{V_{in} * R_2}{R_1 + R_2} \tag{6}$$

Using equation 6 we were able to determine that R_1 would have a value of 10k Ohms and R_2 would have a value of 30k Ohms. The values were chosen to be very high so that there would be limited power loss through them. The mathematical workout of these values can be seen in the Appendix A in Figure 23. These values are slightly different than the values in the design document because when we were finally able to test our stepper-generator system we found that the input voltage values were able to get higher than we originally thought they would be able to. Below the simple voltage sensor circuit can be seen in Figure 7.

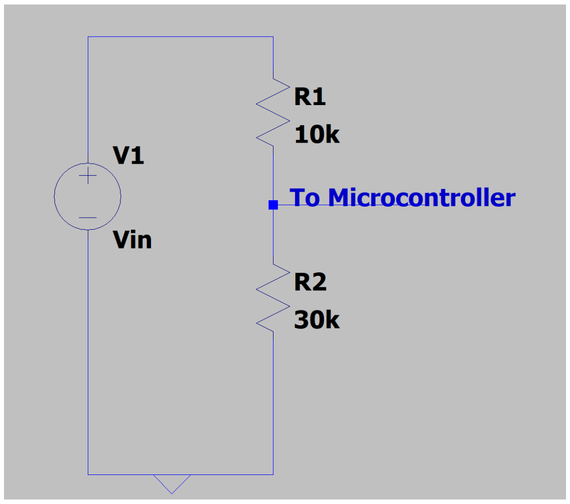


Figure 7: Voltage Sensor Schematic

2.5 PWM Voltage Controller

A PWM signal had to be created by the microcontroller and sent to the gate driver, after which the gate driver had to send that signal to the MOSFET. The PWM signal sent to the gate of the MOSFET in the DC DC converter needed to be controlled, so that the output of the converter would consistently be 5V. In order to ensure this, our preliminary plan was to find the difference between 5V and the voltage outputted and use that error to get our duty ratio closer to the correct value. We chose to use a PI controller because it was very responsive and had little error. This was important for our design because the duty ratio needed to be changed quickly and accurately as it would affect our tolerances at the output.

$$V_{error} = 5 - V_{out} \tag{7}$$

$$P = K_p V_{error} \tag{8}$$

$$I = I + (K_i T_{sampling} V_{error}) \tag{9}$$

$$Duty = P + I \tag{10}$$

From there we constrained the duty ratio, so that the controller did not run off with it if the voltage values exceeded our tolerances of 1-10V. Furthermore, we had to also find the gains, K_p and K_i. While we knew the ball park of what these values were going to be, a lot of it was guess and check in order to find out what the best values would be.

2.6 Computer Application

One of the high-level requirements for the computer application was being able to send a reminder to users so that users could remember work out throughout the day. However, in addition to the reminder system, we decided to include an user friendly interface and program that could help users work out efficiently and accurately. Just like how workout machines let people know how much time they worked out for, we allowed users to choose how much time they want to work out for. The whole flowchart of computer application design can be seen in Appendix A Figure 18. To reach our goal, we could use various kinds of coding languages such as C++ and Java. Among these coding programs, we decided to use Python as our main programming method, because Python had two advantages: coding flexibility and various UI components with Python gui.

Since we were only able to code/control the microcontroller via Arduino IDE, we coded the microcontroller in Arduino IDE so that the serial monitor prints out 1 when pressure is applied to the pressure sensor and prints out 0 when the pressure is not applied. To transfer the pressure sensor Boolean information to the Python gui, we used a program called 'CoolTerm' that can synchronously save the serial monitor value to a text file. The picture of Arduino IDE code can be seen in Appendix A, Figure 19. In the main Python program, the code reads the Boolean text file and determines if the users are actively using the stepper machine. In addition, the program also sets up a daily workout cycle for users. Again the control loop for this workout cycle can be seen in Figure 21 of Appendix A.

2.7 PCB Design

With some research on how to connect the serial decoder we would be using to the microcontroller, how to connect the pressure sensor, and how to connect the USB to our PCB we were able to design our complete PCB [6] [7]. Our PCB contains the circuits for our microcontroller, converter, and pressure sensor as well as ways to connect the portable charger USB and serial decoder. Our completed PCB and the circuit schematic can be seen in Appendix A Figures 16 and 17 respectively.

3 Verification

The purpose of this verification section is to discuss the testing and verification of the major subsystems of our project. Our requirement and verification tables can be seen in Appendix B. This section has been split up into the major subsystems of our project, and each section will discuss the testing of that subsystem. In bold are our major requirements that we will be discussing the completion of for each subsystem.

3.1 Stepper-Generator Mechanical System

The stepper system needs to be able to fit under a desk.

The stepper system does indeed fit under a desk and is well within the specified measurements for the verification. The user can use the machine comfortably sitting as intended. As a bonus, the machine can even be used with only one leg and still generate electricity.

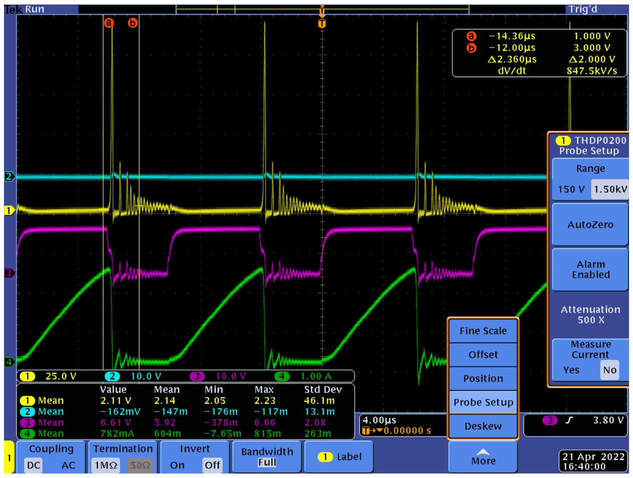
The stepper-generator system also is able to generate electricity as intended from the mechanical stepping motion. The machine generates different voltages based on the user's stepping speed. Table 2 in Appendix A shows the voltages, current, and power being generated depending on differing step speeds. This was really cool to see because we were able to succeed in generating electricity from our own stepping motion.

3.2 Flyback (DC-DC) Converter

Needs to output 5V with a tolerance of ±2.5%.

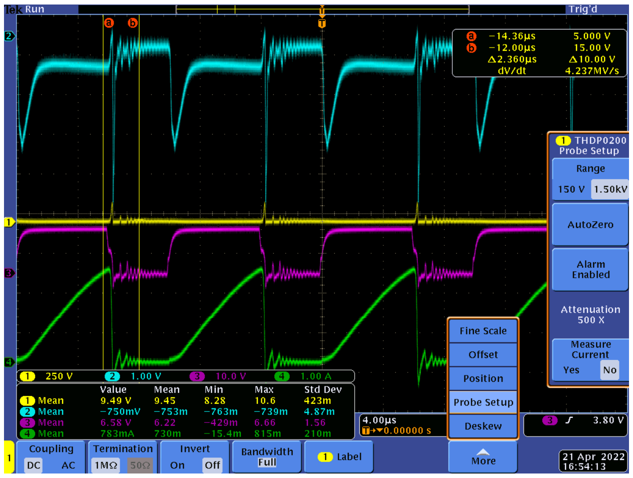
Unfortunately, when we began testing our flyback converter was unable to work as intended on our fully soldered PCB. Although, the converter worked as intended in simulation we ran into a few problems when we tested with our non-ideal components. Figure 8 below shows the waveforms we were seeing when we first began testing our flyback converter with a constant DC input and using a signal generator for our PWM signal. This figure shows that the output voltage is 0 which is not intended. It can also be seen that the input voltage is very noisy and is only averaging around 2 V even though we were inputting 6V to our input. This will further be explained when we look at the next set of waveforms. The purple and green waveforms for this figure are actually working as expected because the MOSFET is switching as intended and the current is flowing only when the MOSFET is on.

Figure 9 now includes the RC Clamp capacitor voltage in blue and this is how we found out our problem with our flyback circuit on our PCB. As can be seen in Figure 9 the capacitor voltage actually stays at its high voltage even after the switch is off. This is not intended operation and actually shows that the RC clamp circuit is not successfully clamping the voltage as intended. We believe that this is why our input voltage waveform is messed up in Figure 8 due to the fact that it is actually competing with the capacitor voltage that is staying high. This was the major reason that our flyback circuit failed. We believe our capacitor wasn't large enough to fully clamp the voltage due to non-idealities



```
a  −14.36µs    1.000 V
b  −12.00µs    3.000 V
   Δ2.360µs    Δ2.000 V
   dV/dt       847.5kV/s

1  25.0 V    2  10.0 V    3  10.0 V    4  1.00 A

          Value     Mean     Min       Max      Std Dev
1 Mean    2.11 V    2.14     2.05      2.23     46.1m
2 Mean    −162mV    −147m    −176m     −117m    13.1m
3 Mean    6.61 V    5.92     −378m     6.66     2.08
4 Mean    782mA     694m     −7.65m    815m     263m

4.00µs    T→▼0.00000 s    3  ∫  3.80 V
```

Figure 8: Flyback Waveforms 1 (Blue-Output Voltage, Yellow- Input Voltage, Purple Gate Voltage of MOSFET, Green - Input Current

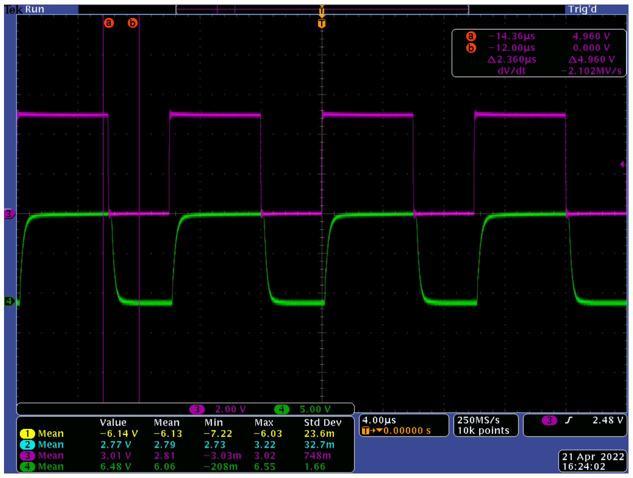


```
a  −14.36µs    5.000 V
b  −12.00µs    15.00 V
   Δ2.360µs    Δ10.00 V
   dV/dt       4.237MV/s

1  250 V    2  1.00 V    3  10.0 V    4  1.00 A

          Value     Mean     Min       Max      Std Dev
1 Mean    9.49 V    9.45     8.28      10.6     423m
2 Mean    −750mV    −753m    −763m     −739m    4.87m
3 Mean    6.58 V    6.22     −429m     6.66     1.56
4 Mean    783mA     730m     −15.4m    815m     210m

4.00µs    T→▼0.00000 s    3  ∫  3.80 V
```

Figure 9: Flyback Waveforms 2 (Blue-RC Clamp Capacitor Voltage, Yellow- Output Voltage, Purple Gate Voltage of MOSFET, Green - Input Current

that we didn't originally take into account. Something that further heightened our non-idealities in our circuit was the fact that we had a problem soldering our diodes onto our PCB. The problem was that our footprint was too small for the large diode wires. Our solution was to solder smaller wires onto the ends of the diodes to have those wires then soldered to the board. Our soldering solution can be seen in Figure 20 in Appendix A. This introduced a lot of parasitic inductance into our circuit that we did not originally account for, and we believe this also contributed to our circuit failing as it did because we would have needed a larger capacitor value to balance this extra inductance.

Our PCB wasn't fully a failure however, because we were actually able to see that our gate driver was able to drive our MOSFET switching as intended. In this that also meant that our voltage regulator was correctly stepping up the 5V source to 12V to power our gate driver chip. In Figure 10 our MOSFET switching voltage can be seen in green and it can be seen when compared to our PWM signal in purple that the MOSFET is switching ON when it is supposed to.



a	−14.36µs	4.960 V
b	−12.00µs	0.000 V
	Δ2.360µs	Δ4.960 V
	dV/dt	−2.102MV/s

	Value	Mean	Min	Max	Std Dev
1 Mean	−6.14 V	−6.13	−7.22	−6.03	23.6m
2 Mean	2.77 V	2.79	2.73	3.22	32.7m
3 Mean	3.01 V	2.81	−3.03m	3.02	748m
4 Mean	6.48 V	6.06	−208m	6.55	1.66

Figure 10: MOSFET Switching Voltage

3.3 Pressure Sensor

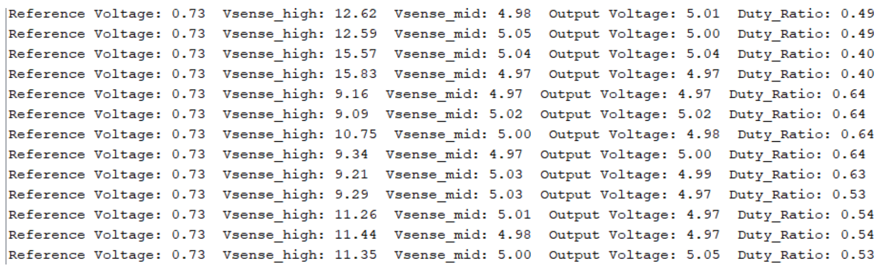
Must be able to sense pressure of force above 10kg.

To test the pressure sensor, we connected constant 5V voltage supply to pressure sensor circuit and observed the voltage output across 3.3k ohm resistor. As we can see in Appendix A Figures 21 and 22, when the pressure was not applied, average voltage output was around -150mV which is close to 0V. However, when we applied enough force, we got a voltage output of 4.4V. As a result, we could see that the voltage output of pressure sensor is high only when the pressure is applied to the sensor. We also ensured that the pressure sensor was able to sense a weight above 10kg by placing a 10kg weight on it and ensuring that it worked as intended.

3.4 PWM Voltage Controller

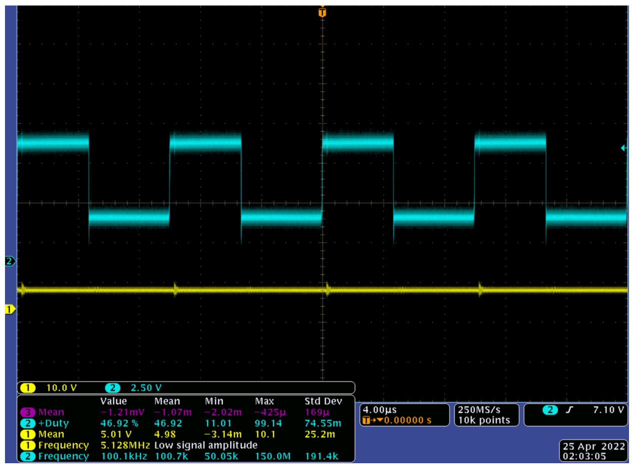
Must take in voltage at the input and output of the DC-DC converter and create the appropriate duty cycle for the converter. (PWM capability)

The values of the gains were found through a series of guess and check. We knew that the K values would be around 0.5 for the K_p and 0.01 for the K_i Undershooting the values caused the output voltage to lag and so often we wouldn't meet the tolerance values, while overshooting these values caused the duty ratio to oscillate between a higher duty ratio and a lower duty ratio as the controller kept overshooting the actual value. The values that we found, $K_p = 0.01$ and $K_i = 0.003$ created a controller that was very responsive and accurate. While we were unable to verify if this worked on our flyback converter, we were able to use a buck converter from last semester. Because the control aspect is not dependent on which converter we use, we knew that if it worked on our buck converter it would have worked on our flyback converter if it was operational. The tests were done by uploading the code onto an Arduino as it was not possible to connect the buck converter to our microchip as we did not have a serial decoder. The Arduino took in the output and input voltage, calculated the duty ratio, and send the correct waveform to the MOSFET. We were able to verify this by looking at the output voltage as we varied the input voltage. The outputs below in Figure 11 show the different input voltages that we tested and an output voltage of around 5V with varying duty ratios. Figure 11 is an example that our Arduino code was able to read inputs from voltage sensors which was a low-level requirement for our voltage sensor circuit. Figure 12 simply shows the PWM control signal that our Arduino created and how the output voltage was able to stay at a constant 5V.



```
Reference Voltage: 0.73  Vsense_high: 12.62  Vsense_mid: 4.98  Output Voltage: 5.01  Duty_Ratio: 0.49
Reference Voltage: 0.73  Vsense_high: 12.59  Vsense_mid: 5.05  Output Voltage: 5.00  Duty_Ratio: 0.49
Reference Voltage: 0.73  Vsense_high: 15.57  Vsense_mid: 5.04  Output Voltage: 5.04  Duty_Ratio: 0.40
Reference Voltage: 0.73  Vsense_high: 15.83  Vsense_mid: 4.97  Output Voltage: 4.97  Duty_Ratio: 0.40
Reference Voltage: 0.73  Vsense_high: 9.16  Vsense_mid: 4.97  Output Voltage: 4.97  Duty_Ratio: 0.64
Reference Voltage: 0.73  Vsense_high: 9.09  Vsense_mid: 5.02  Output Voltage: 5.02  Duty_Ratio: 0.64
Reference Voltage: 0.73  Vsense_high: 10.75  Vsense_mid: 5.00  Output Voltage: 4.98  Duty_Ratio: 0.64
Reference Voltage: 0.73  Vsense_high: 9.34  Vsense_mid: 4.97  Output Voltage: 5.00  Duty_Ratio: 0.64
Reference Voltage: 0.73  Vsense_high: 9.21  Vsense_mid: 5.03  Output Voltage: 4.99  Duty_Ratio: 0.63
Reference Voltage: 0.73  Vsense_high: 9.29  Vsense_mid: 5.03  Output Voltage: 4.97  Duty_Ratio: 0.53
Reference Voltage: 0.73  Vsense_high: 11.26  Vsense_mid: 5.01  Output Voltage: 4.97  Duty_Ratio: 0.54
Reference Voltage: 0.73  Vsense_high: 11.44  Vsense_mid: 4.98  Output Voltage: 4.97  Duty_Ratio: 0.54
Reference Voltage: 0.73  Vsense_high: 11.35  Vsense_mid: 5.00  Output Voltage: 5.05  Duty_Ratio: 0.53
```

Figure 11: PI controller



	Value	Mean	Min	Max	Std Dev
3 Mean	−1.21mV	−1.07m	−2.02m	−425µ	169µ
2 +Duty	46.92 %	46.92	11.01	99.14	74.55m
1 Mean	5.01 V	4.98	−3.14m	10.1	25.2m
1 Frequency	5.128MHz	Low signal amplitude			
2 Frequency	100.1kHz	100.7k	50.05k	150.0M	191.4k

Figure 12: PI controller

3.5 Computer Application

When activated, the computer all must take in the amount of time the user wants to work out every hour

When the PCB is connected to user's computer through the USB cable and the program is ran, the computer application successfully takes in the amount of time the user wants to work out every hour. The picture of the program taking in the user input can be seen below in Figure 13. We specifically designed our application to take up a small portion of the users screen so that the user could continue working while also seeing the timer.

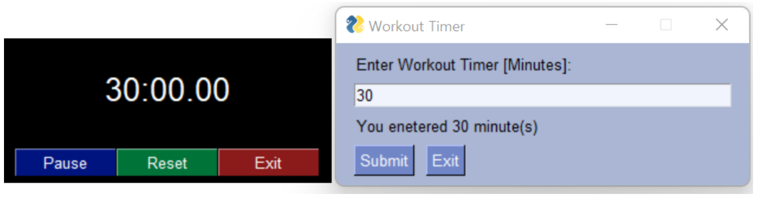


Figure 13: Workout Timer User Input

When the user touches the pressure sensor, the computer app must start the timer

By printing out the pressure sensor Boolean value and saving the values as a text file synchronously, we were successfully able to read when users are touching the pressure sensor. Based on the Boolean value, the timer is started; When users actively use the machine, the timer counts down and it pauses when users are not using the machine. Figure 14 shows an example of the timer counting down when pressure is applied.

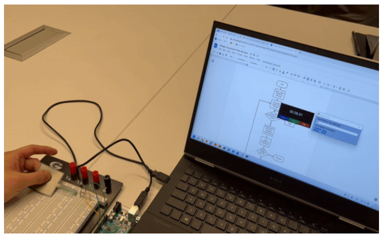


Figure 14: Timer Counting Down as Pressure is Applied

Must be able to send notification to remind users to work out after an hour break

When users are done with the workout and the countdown timer hits 0, a rest timer set to an hour appears so that users can take enough rest time before going back to the workout. Users are then reminded through a pop up message to go back to their workout when the rest timer finishes. After running the workout cycle eight times, the program is done and halted. Detailed pictures of how program runs such as picture of rest timer and pop up notifications can be seen in Figure 15.

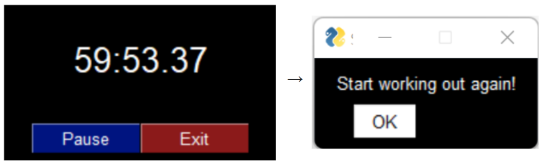


Figure 15: Rest Timer into Notification to workout again

4 Costs

Cost of Project Parts

Part Cost Table can be seen in Appendix C.

The total cost for parts came out to be approximately $**194.38**.

Labor Costs

Average Starting Salary of Electrical Engineering graduates is $79,714 in 2018-19 [8].

Full-Time job works 1920 hours per year assuming that a person works 8 hours a day, from Monday to Friday, and gets 8 holidays a year and 12 vacation days [9].

Average hourly wage = $79,714/1920hr = $41.52/hr

$41.52/hr × 2.5 × 100 hr = $10,379 per person

Total Labor Cost: $10,379 × 3 = **$31,138**

Machine Shop Costs

SCS Machine Shop has construction and repair cost as $36.65/hr + Materials[10]. Assuming the Machine Shop works on our project for 15 hours, the total machine shop cost is $36.65/hr × 15hr = **$549.75**.

Total Project Cost

Machine Shop Costs + Labor Costs + Project Parts Cost

194.38 + 31,138 + 549.75

Total Cost of Project = $31,882.13

5 Conclusion

There were two major challenges we faced during the final integration of our project that inhibited us from fully completing our project as intended. The first challenge we faced was that our flyback converter did not work as intended due to the RC clamp circuit not performing as intended. Although the complete converter didn't work as intended we were still able to see parts of it working such as the gate driver circuit. Also, with a past class's [3] buck converter we were able to completely show that our output voltage control algorithm was able to work as intended. The second challenge we faced was that our USB serial decoder for the microcontroller never came in. This inhibited us from fully communicating with our microcontroller, and instead we were forced to use an Arduino for our microcontroller. Although this hurt our complete integration it is a small set back because the code we used for the Arduino could simply be used for the microcontroller as well if we were able to communicate with it on our PCB. This is the case because the microcontroller we used is actually the same microcontroller that is the brains of the Arduino. Although we faced these two challenges we believe that both could be solved relatively easily in the future. Overall, we still would consider our project a success because we were able to successfully complete a majority of our requirements. We also completed 2 of our 3 high-level requirements. As it is our machine does successfully solve our problem of sitting too much through our computer program that reminds the user to workout and use the machine throughout the day. It was also very cool to see our mechanical stepping motion being able to create usable electrical power.

5.1 Future Work

If we were to continue working on this project we would initially focus on solving the two challenges we faced at the end of our design. The first problem of the flyback converter we believe could be solved by increasing the capacitance in the RC clamp circuit. We would then continue to make sure that our flyback converter had no further bugs in it. The second problem of the missing serial decoder and the limited integration of the microcontroller could be solved by simply getting a new serial decoder to arrive on time. We believe that with the serial decoder we would be able to tie our PCB and computer application together relatively easily. Another elegant solution to this problem we thought of was instead using Bluetooth to communicate with the computer instead of the USB and serial decoder. This would solve our serial decoder problem and would actually alleviate some of the excessive wiring we have to connect our project to the computer. Finally, another thing we would want to add to our project is some noise-cancelling cushions to the bottom of the stepper because the stepping got noisy when going fast. Overall, we believe our project could have a strong future ahead of it if these problems were solved.

5.2 Ethics

Our biggest safety concern was our battery usage in our project. Since we are charging a portable charger battery through the use of our converter, there is an inherent risk of overcharging the battery. Overcharging can cause the cathode of the battery to become unstable, producing carbon dioxide, eventually leading to a failure and possible fire [11]. One way we thought of preventing this risk from happening was by potentially having a switch to the USB connector that would open if the microcontroller sensed the output of the DC-DC converter above our wanted tolerance of 5%. Also, a flyback converter inherently isolates its input and output. This means that the generator is isolated completely from our battery so if something goes wrong on either end it will not affect the other side. Finally, since we are simply connecting the USB charger to the actual portable charger itself the portable charger will also have safety mechanisms inside it to prevent overcharging of the portable charger battery.

Inherently in using a generator there is a danger of connecting the generator to the converter with the wrong polarity. This could put people in danger because it could cause our circuit to short and overheat. Luckily there was a simple solution to this danger. We simply inputted a diode at the input of our converter. This made it so that whenever the cathode voltage was higher than the anode voltage the diode will act as a switch and separate the power supply from the rest of our circuit [12]. We also made sure to color code the positive side of the generator and the positive side of our converter to ensure that this problem never occurred during testing.

In addition to battery, converter, and generator safety, there also are ethical issues we have to keep in mind for this project. It was important for us to follow the 7.8 IEEE Code of Ethics. Especially, since the project was worked as a team, team members held ”paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment”, and “seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest, and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others”[13]. All the team members acted responsibly and respected each other throughout the course of our project. We also did not engage in any kind of harassment and discrimination against each other [14].

References

[1] S. Scutti. “”Yes, sitting too long can kill you, even if you exercise”.” (2017, September 12), [Online]. Available: https://www.cnn.com/2017/09/11/health/sitting-increases-risk-of-death-study/index.html (visited on 02/22/2022).

[2] S. Randall. “”Standard Desk Dimensions Layout Guidelines”.” (2021, November 29), [Online]. Available: https://upgradedhome.com/desk-dimensions/ (visited on 02/10/2022).

[3] A. Banerjee. “”Power Electronics Laboratory ECE 469”.” (2020-21), [Online]. Available: https://powerece469.web.illinois.edu/wp/ (visited on 03/01/2022).

[4] T. Hudson. “”How to Design a Flyback Converter in Seven Steps”.” (2021), [Online]. Available: https://www.monolithicpower.com/en/how-to-design-a-flyback-converter-in-seven-steps (visited on 03/01/2022).

[5] Electrical4U. “”Voltage sensor: What is it and how does it work?”” (2021, January 31), [Online]. Available: https://www.electrical4u.com/voltage-sensor/ (visited on 02/24/2022).

[6] JIMBLOM. “”Force Sensitive Resistor Hookup Guide”.” (2016), [Online]. Available: https://learn.sparkfun.com/tutorials/force-sensitive-resistor-hookup-guide/discuss (visited on 03/24/2022).

[7] Jaycar. “”USB Serial converter. Arduino Tool: USB-Serial Converter”.” (n.d.), [Online]. Available: https://www.jaycar.com.au/usb-serial-converter (visited on 02/24/2022).

[8] G. E. O. of Marketing and Communications. “”Salary averages Electrical Computer Engineering”.” (n.d.), [Online]. Available: https://ece.illinois.edu/admissions/why-ece/salary-averages (visited on 02/21/2022).

[9] J. Harness. “”Show to calculate number of work hours in a year”.” (2019, February 11), [Online]. Available: https://bizfluent.com/how-5896117-calculate-number-work-hours-year.html (visited on 02/21/2022).

[10] M. Shop. “”Machine Shop Costs”.” (n.d.), [Online]. Available: https://scs.illinois.edu/resources/cores-scs-service-facilities/machine-shop (visited on 02/21/2022).

[11] Jrrichar. “”Battery Safety”.” (2020, October 4), [Online]. Available: https://www.jaycar.com.au/usb-serial-converter (visited on 03/30/2022).

[12] R. Keim. “”Reverse Polarity Protection: How to protect your circuits using only a diode”.” (2018), [Online]. Available: https://www.allaboutcircuits.com/technical-articles/how-to-protect-your-circuits-using-only-a-diode.html (visited on 03/28/2022).

[13] IEEE. “”IEEE Code of Ethics”.” (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 02/08/2020).

[14] ACM. “”The code affirms an obligation of computing professionals to use their skills for the benefit of society”.” (n.d.), [Online]. Available: https://www.acm.org/code-of-ethics (visited on 02/10/2022).

Appendix A

Table 1: Duty Ratios for Varying Input Voltages

Input (V)	1.5	3	4.5	6	7.5	9
Duty Ratio	NA	.78	.666	.55	.48	.43

Table 2: Stepper Power Generation

Speed	Max Voltage	Average Voltage	Average Current	Max Power
	(V)	(V)	(A)	(W)
Slow Stepping	2.5	1.5	.200	.38
Medium Stepping	4.5	3	.509	3.1
Fast Stepping	9.5	7.1	.900	8

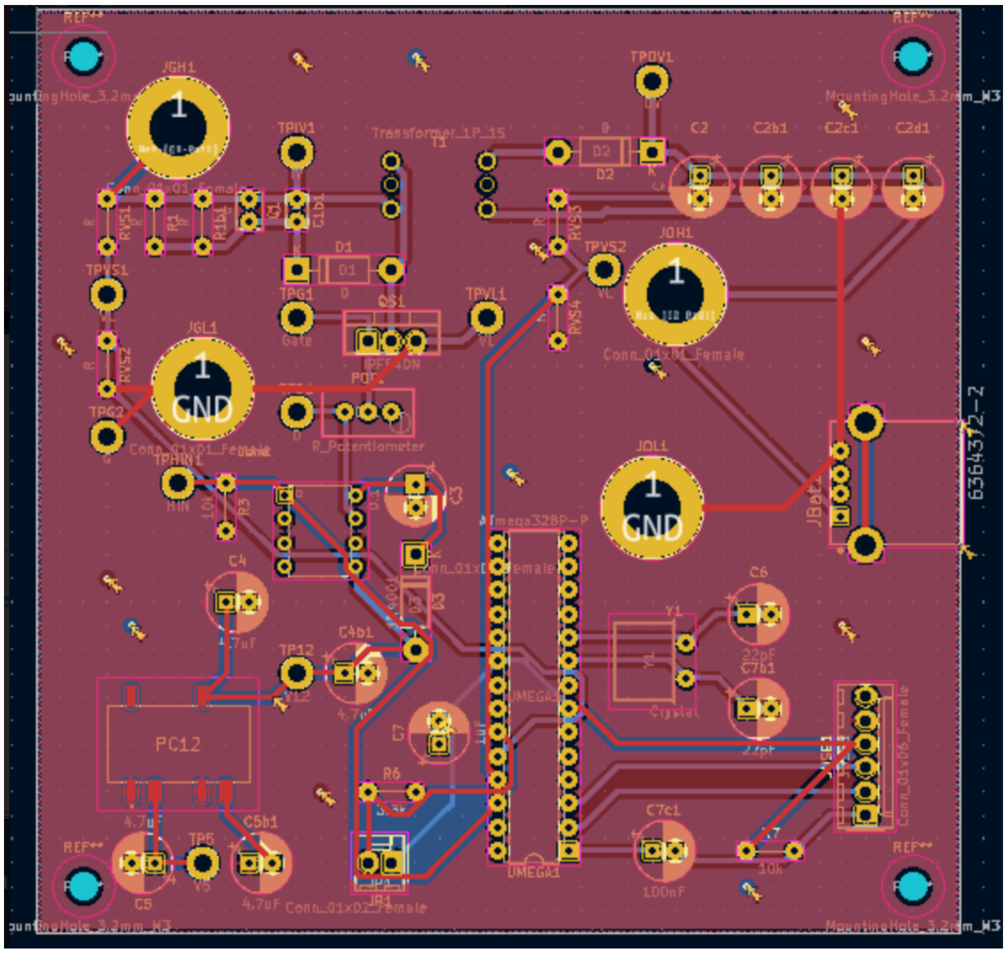


Figure 16: PCB Design

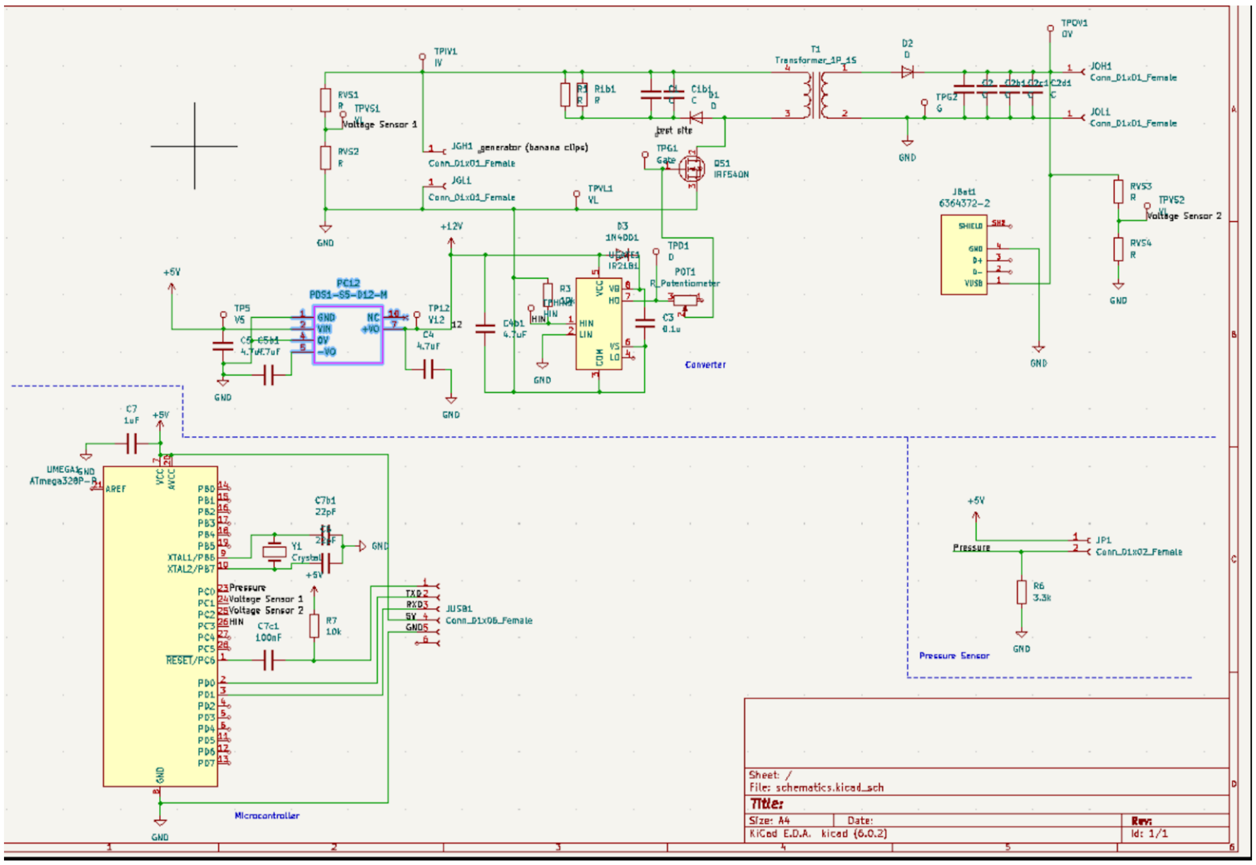


Figure 17: Complete PCB Schematic

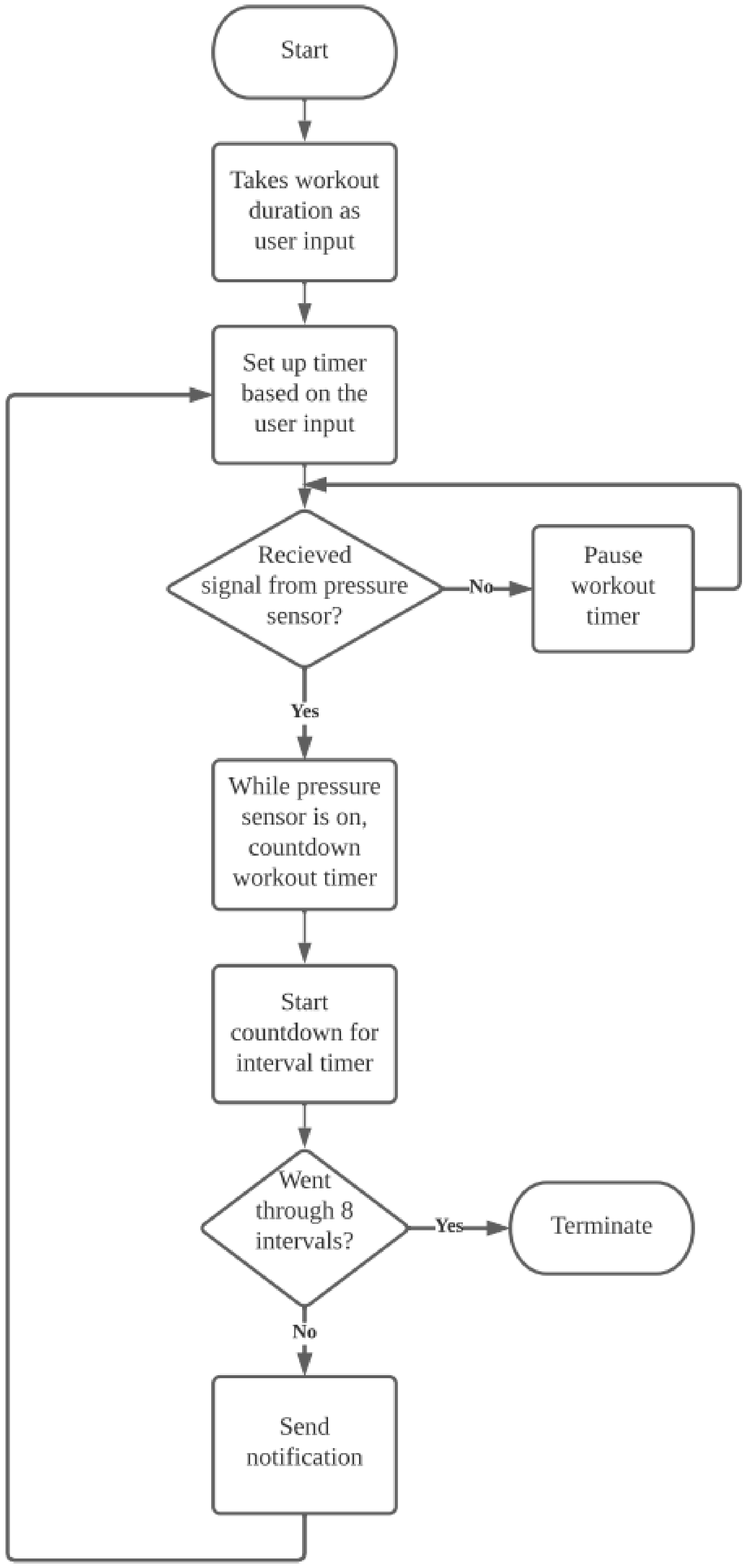
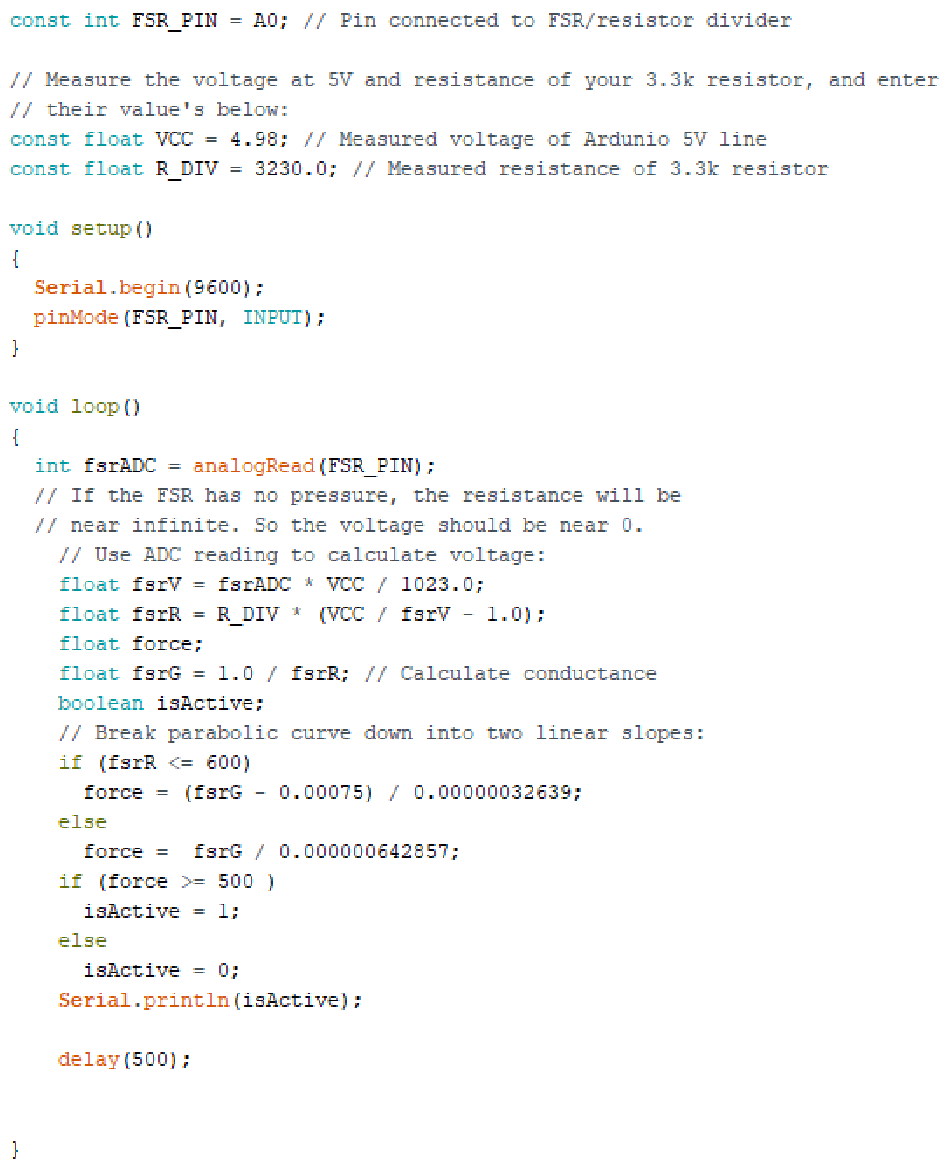

Start
Takes workout duration as user input
Set up timer based on the user input
Recieved signal from pressure sensor?
No
Pause workout timer
Yes
While pressure sensor is on, countdown workout timer
Start countdown for interval timer
Went through 8 intervals?
Yes
Terminate
No
Send notification

Figure 18: Flow Chart for Computer Program Code



```
const int FSR_PIN = A0; // Pin connected to FSR/resistor divider

// Measure the voltage at 5V and resistance of your 3.3k resistor, and enter
// their value's below:
const float VCC = 4.98; // Measured voltage of Ardunio 5V line
const float R_DIV = 3230.0; // Measured resistance of 3.3k resistor

void setup()
{
  Serial.begin(9600);
  pinMode(FSR_PIN, INPUT);
}

void loop()
{
  int fsrADC = analogRead(FSR_PIN);
  // If the FSR has no pressure, the resistance will be
  // near infinite. So the voltage should be near 0.
    // Use ADC reading to calculate voltage:
    float fsrV = fsrADC * VCC / 1023.0;
    float fsrR = R_DIV * (VCC / fsrV - 1.0);
    float force;
    float fsrG = 1.0 / fsrR; // Calculate conductance
    boolean isActive;
    // Break parabolic curve down into two linear slopes:
    if (fsrR <= 600)
      force = (fsrG - 0.00075) / 0.00000032639;
    else
      force =  fsrG / 0.000000642857;
    if (force >= 500 )
      isActive = 1;
    else
      isActive = 0;
    Serial.println(isActive);

    delay(500);


}
```

Figure 19: Arduino IDE code

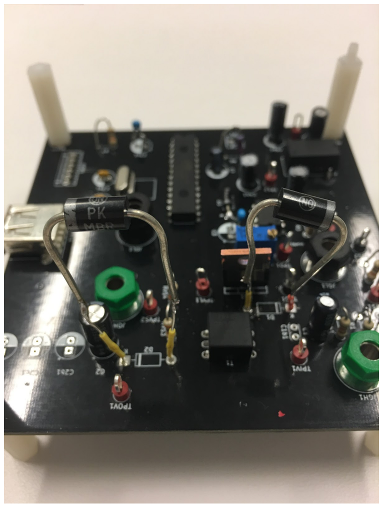


Figure 20: PCB Diode Soldering

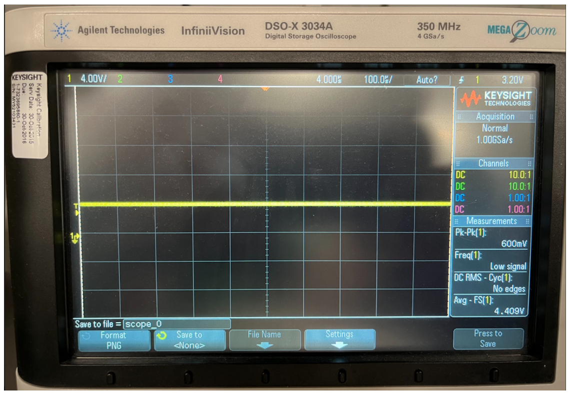

Agilent Technologies InfiniiVision DSO-X 3034A Digital Storage Oscilloscope
350 MHz 4 GSa/s
4.00V/
4.000s 100.0s/ Auto? 3.20V
Acquisition Normal 1.00GSa/s
Pk-Pk(1): 600mV
Freq(1): Low signal
DC RMS - Cyc(1): No edges
Avg - FS(1): 4.409V

Figure 21: Pressure Applied (High Voltage)

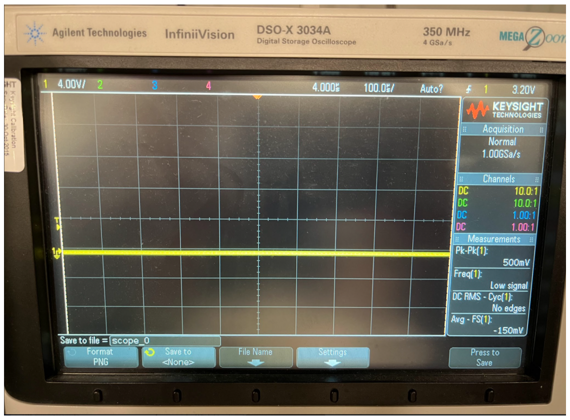
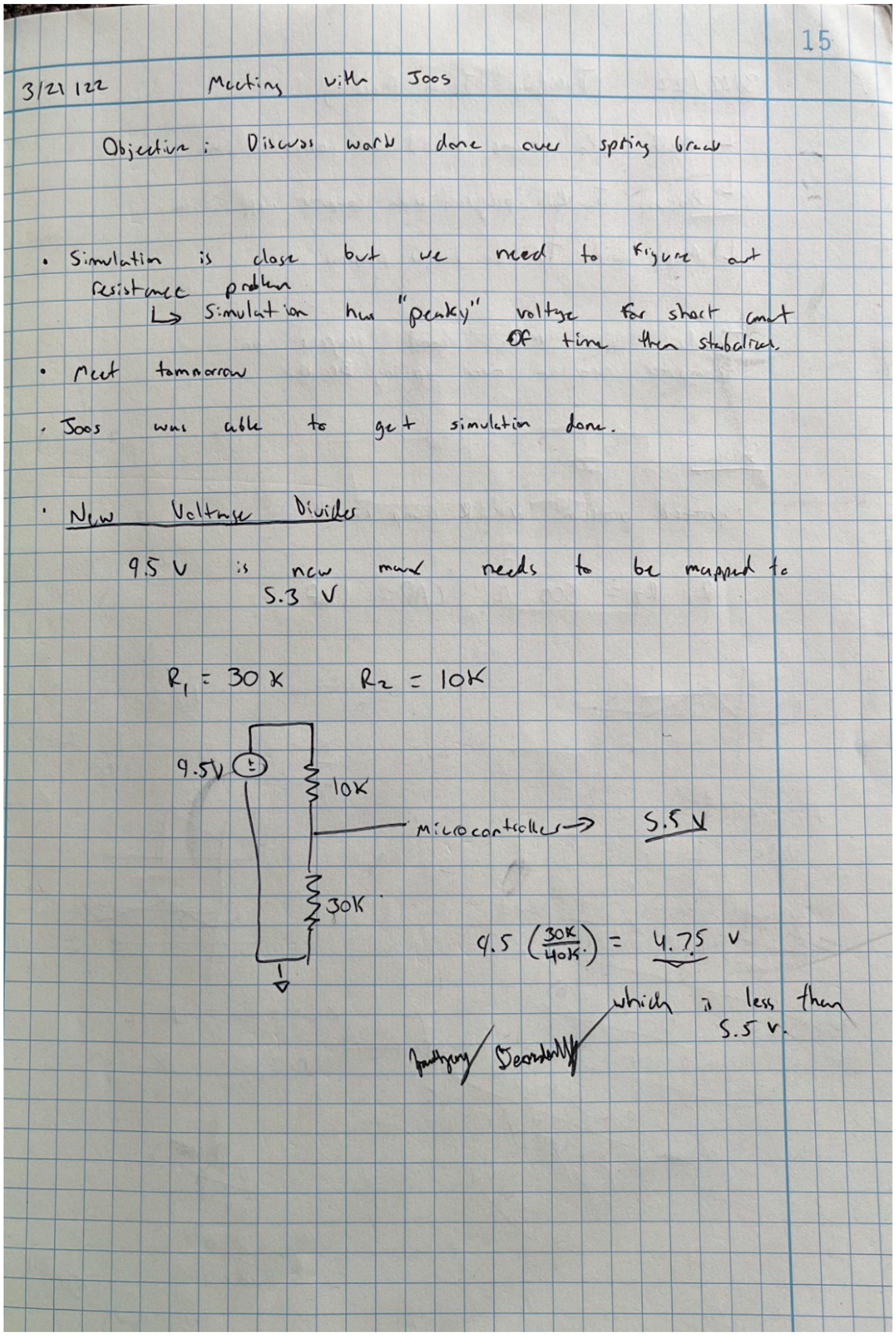

Agilent Technologies InfiniiVision DSO-X 3034A Digital Storage Oscilloscope
350 MHz 4 GSa/s
4.00V/
4.000s 100.0s/ Auto? 3.20V
Acquisition Normal 1.00GSa/s
Pk-Pk(1): 500mV
Freq(1): Low signal
DC RMS - Cyc(1): No edges
Avg - FS(1): -150mV

Figure 22: No Pressure on Sensor (Low VOltage)


15

3/21/22 Meeting with Joos

Objective: Discuss work done over spring break

- Simulation is close but we need to figure out resistance problem
  - ↳ Simulation has "peaky" voltage for short amount of time then stabilizes.
- Meet tomorrow
- Joos was able to get simulation done.
- <u>New Voltage Divider</u>

9.5 V is new max needs to be mapped to 5.3 V

$R_1 = 30K$ $R_2 = 10K$

9.5V

10K

microcontroller → <u>5.5 V</u>

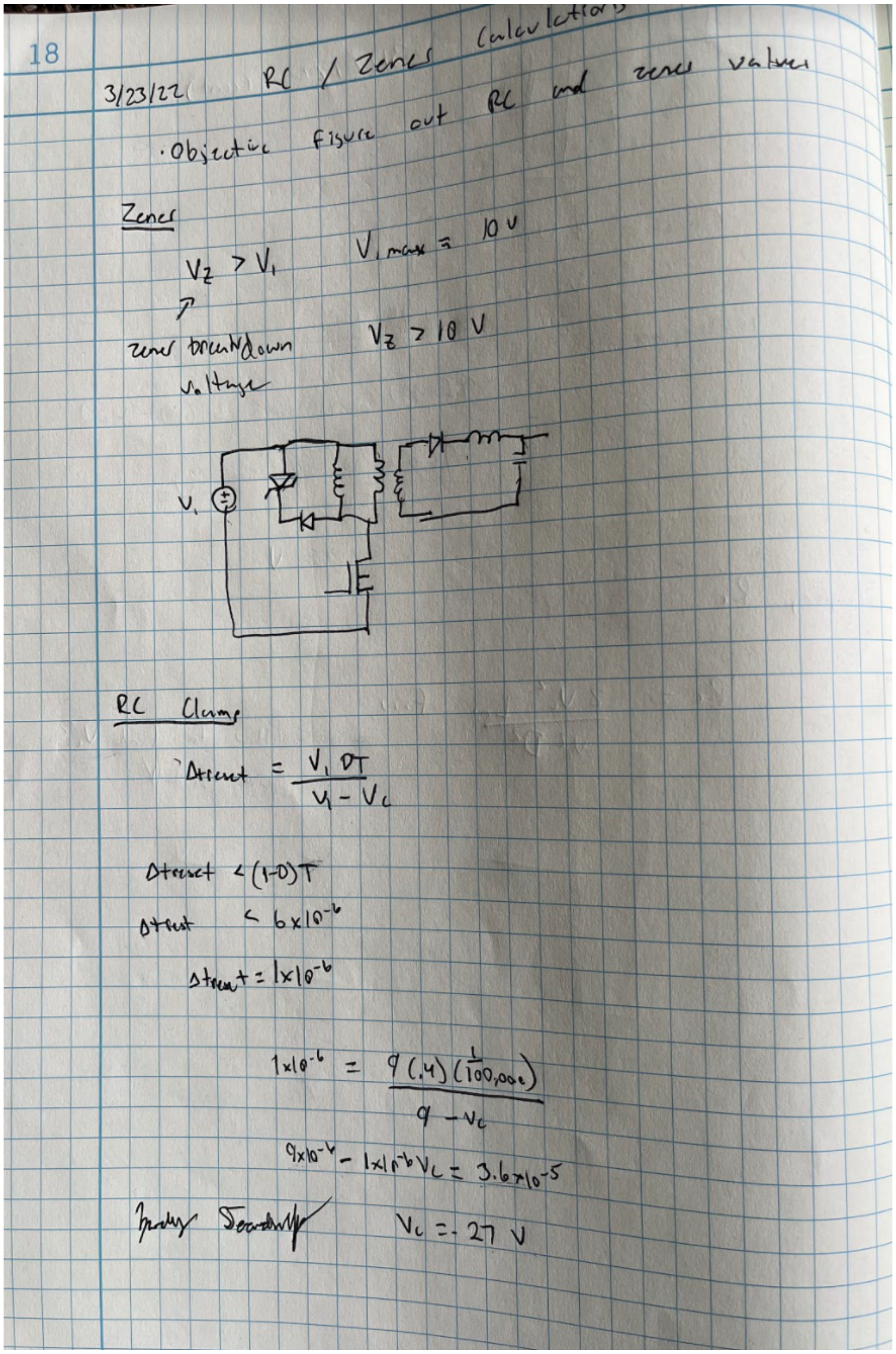
30K

$$9.5\left(\frac{30K}{40K}\right) = \underline{4.75}\ V$$

which is less than 5.5 V.

[illegible] [illegible]

Figure 23: Resistor Divider Lab Notebook



18

3/23/22 RC / Zener Calculations

· Objective figure out RC and zener values

Zener

$V_Z > V_1$ $V_{1\,max} = 10\text{ V}$

zener breakdown voltage

$V_Z > 10\text{ V}$

V_1

RC Clamp

$$\Delta t_{reset} = \frac{V_1 DT}{V_1 - V_c}$$

$\Delta t_{reset} < (1-D)T$

$\Delta t_{reset} < 6\times10^{-6}$

$\Delta t_{reset} = 1\times10^{-6}$

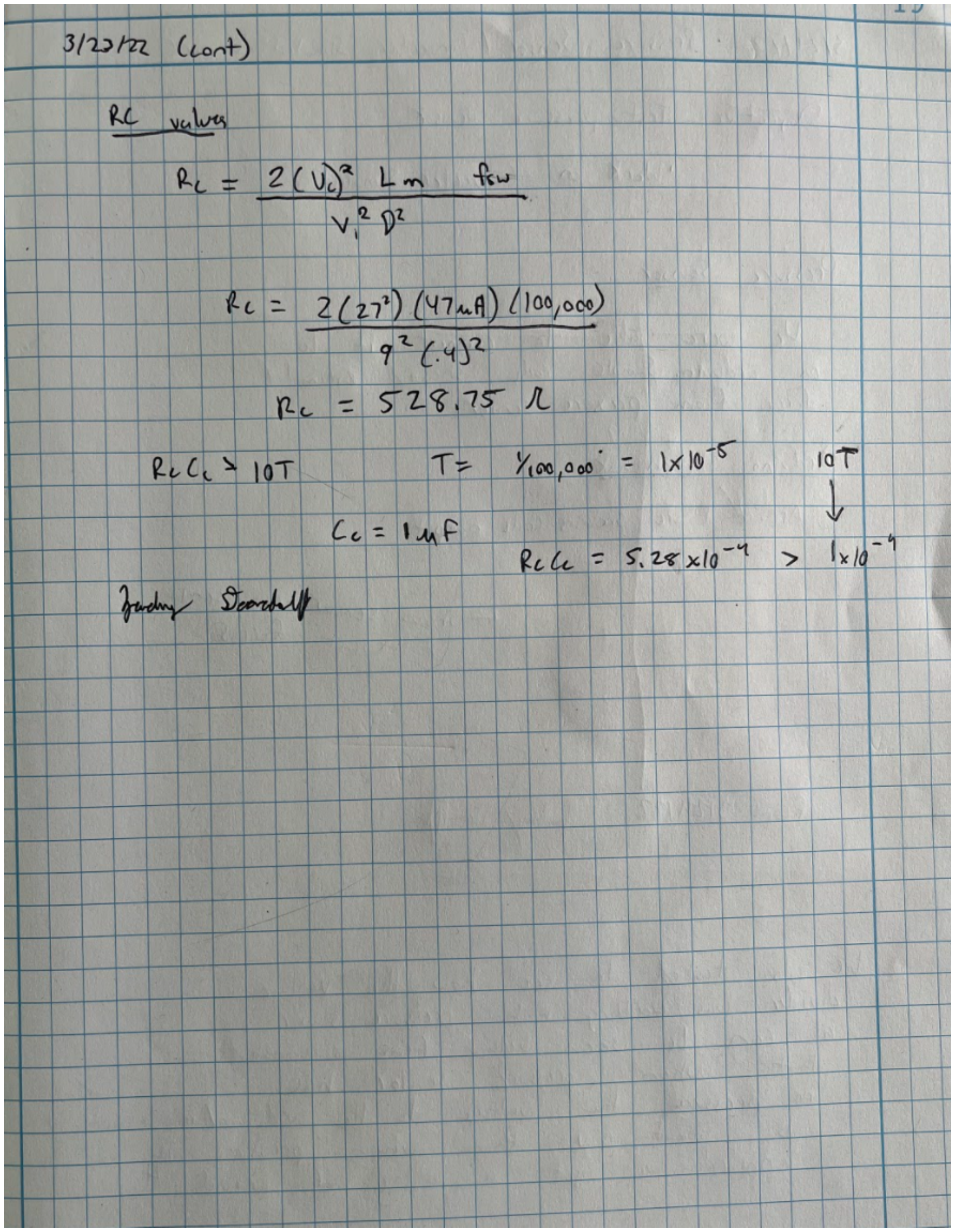
$$1\times10^{-6} = \frac{9(.4)(\frac{1}{100{,}000})}{9 - V_c}$$

$9\times10^{-6} - 1\times10^{-6}V_c = 3.6\times10^{-5}$

[illegible]

$V_c = -27\text{ V}$

Figure 24: Clamp Circuit Notes 1



3/23/22 (cont)

RC values

$$R_c = \frac{2(V_c)^2 \; L_m \; f_{sw}}{V_i^2 D^2}$$

$$R_c = \frac{2(27^2)(47uA)(100{,}000)}{9^2(.4)^2}$$

$$R_c = 528.75 \; \Omega$$

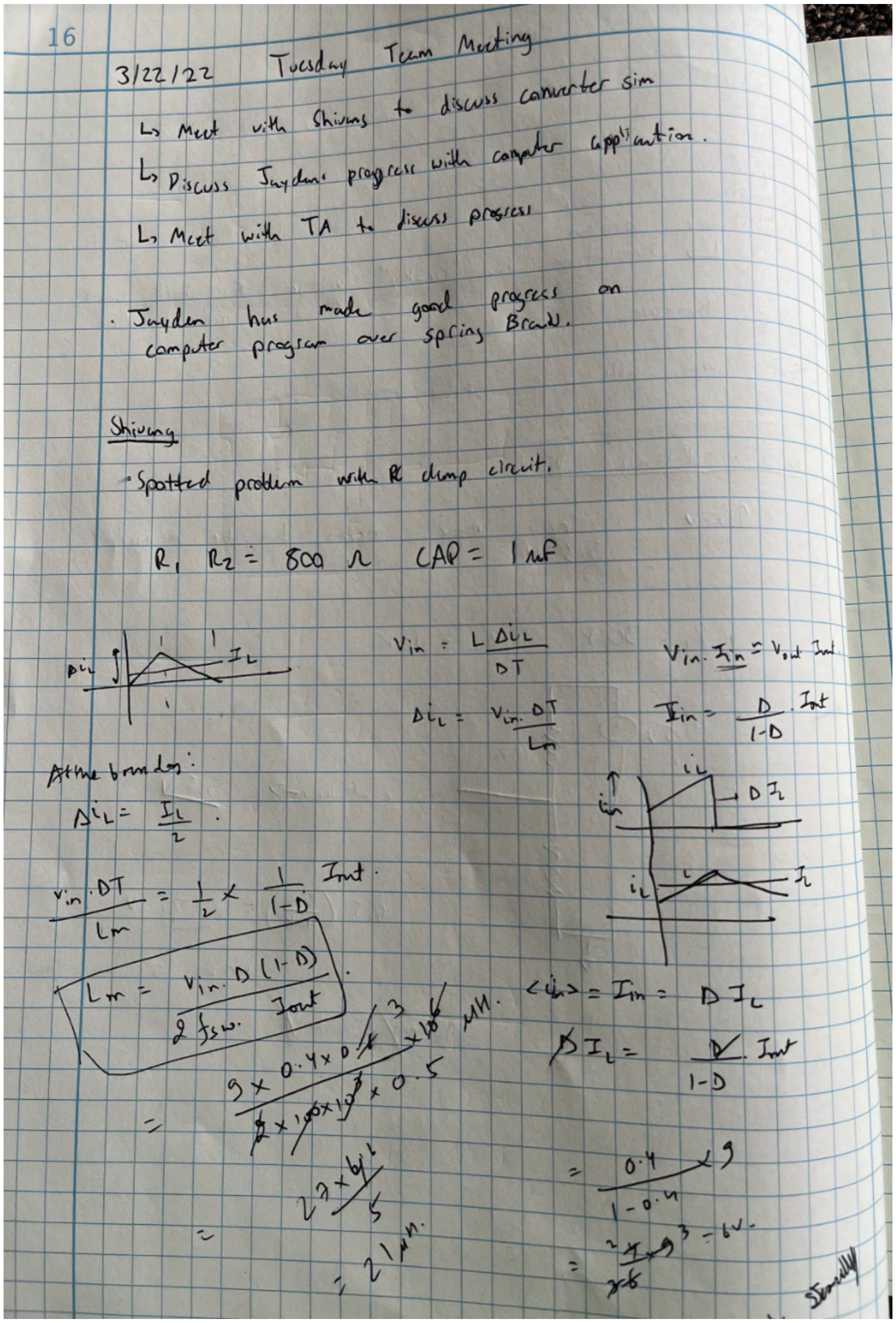
$R_cC_c > 10T$ $\quad T = 1/100{,}000 = 1\times10^{-5}$ $\quad 10T \downarrow$

$C_c = 1uF$

$R_cC_c = 5.28\times10^{-4} > 1\times10^{-4}$

[illegible]

Figure 25: Clamp Circuit Notes 2



16

3/22/22 Tuesday Team Meeting

↳ Meet with Shivang to discuss converter sim

↳ Discuss Jayden's progress with computer application.

↳ Meet with TA to discuss progress

· Jayden has made good progress on computer program over spring Break.

Shivang

· Spotted problem with R clamp circuit.

R_1 R_2 = 800 Ω CAP = 1 nF

$V_{in} = L \frac{\Delta i_L}{DT}$

$\Delta i_L = \frac{V_{in} \cdot DT}{L_m}$

$V_{in} \cdot I_{in} = V_{out} I_{out}$

$I_{in} = \frac{D}{1-D} \cdot I_{out}$

At the boundary:

$\Delta i_L = \frac{I_L}{2}$.

$\frac{V_{in} \cdot DT}{L_m} = \frac{1}{2} \times \frac{1}{1-D} I_{out}$.

$$\boxed{L_m = \frac{V_{in} \cdot D(1-D)}{2 f_{sw} \cdot I_{out}}}.$$

$= \frac{9 \times 0.4 \times 0.\cancel{6}^{3}}{\cancel{2} \times 100 \times 10^{3} \times 0.5} \times 10^{\cancel{6}}$ μH.

$= \frac{27 \times 6 \times 10^{1}}{5}$

$= 21$ μH.

$\langle i_{in} \rangle = I_{in} = D I_L$

$\cancel{D} I_L = \frac{\cancel{D}}{1-D} \cdot I_{out}$

$= \frac{0.4}{1-0.4} \times 9$

$= \frac{\cancel{4}^{2}}{\cancel{6}_{3}} \times 9^{3} = 6$ V.

[illegible]

Figure 26: Magnetizing Inductance Notes

Appendix B

Table 1: RV Stepper Machine

Requirement	Verification	Requirement Met?
The stepper system needs to be able to fit under a desk.	1. Measure the width and height of the machine. 2. Verify that depth < 36 inches, height < 30 inches, width < 24 inches	Yes

Table 2 : RV for Pressure Sensor

Requirement	Verification	Requirement Met?
Must be able to sense pressure of force above 10 kg.	1. Without connecting to PCB first construct circuit below on testing breadboard. 2. Connect 5 V DC supply to the pressure sensor circuit. 3. Read voltage across the 3.3 ohm resistor 4. Apply pressure and make sure that voltage increases. 5. Apply weight of around 10 kg to ensure that the sensor can handle sitting person weight.	Yes

Table 3: RV DC-DC Converter

Requirement	**Verification**	**Requirement Met?**
Needs to output 5V with a tolerance of ±2.5%.	1. Use a variable DC voltage supply to the converter input. 2. Connect a 3ohm (10% tolerance), 30 ohm (10%), 100ohm (10%) to the output. 3. Measure the output with an oscilloscope 4. Change the values of the DC voltage supply from 1V to 10V. 5. Check that the output is 5V ±2.5	No

Table 4: RV Voltage Sensors

Requirement	**Verification**	**Requirement Met?**
Needs to be able to sense voltage within plus or minus 2.5%.	1) Attach a known DC supply to the voltage sensor circuit from 0 - 8 V. 2) Probe the voltage going to the microcontroller to see what the true reading should be. 3) Ensure with simple test code that the microcontroller is correctly sensing the voltage within the tolerance range.	Yes

Table 5: RV Voltage Regulator

Requirement	**Verification**	**Requirement Met?**
Must output a voltage of 12±0.3V	1. Connect the voltage regulator to the 5 V USB supply. 2. Probe the output of the voltage regulator from the output pin. 3. Ensure that the output is within the specified range.	Yes

Table 6: Microcontroller

Requirement	**Verification**	**Requirement Met?**
Must take in data from the pressure sensor and send a boolean value to the computer through the USB.	1. Check that there is a voltage from the register output of the microcontroller. 2. Run test code to ensure that the computer is receiving the data from the microcontroller about the pressure sensor.	Yes
Must take in voltage at the input and output of the DC-DC converter and create the appropriate duty cycle for the converter. (PWM capability)	1. Connect an oscilloscope to the gate drivers. 2. Produce a DC voltage at the inputs for the microcontroller (1V to 10V) 3. Run control code mode for the microcontroller. 4. Ensure that the microcontroller is producing correct PWM by probing the input to the gate driver circuit. 5. Change input DC voltage and ensure that microcontroller changes PWM signal correctly.	Yes

Table 7: RV Computer Application

Requirement	**Verification**	**Requirement Met?**
When activated, the computer app must take in the amount of time the user wants to work out every hour	1. The user inputs a workout duration. 2. Verify that the application can display the amount of time the user imputed and have a countdown.	Yes
When the user touches the pressure sensor the computer app must start the timer.	1. Check that the timer counts down when the user applies force on the sensor. 2. Check if the timer stops counting down when the force is not applied	Yes
Must be able to send notification to remind users to work out after an hour break.	1. Verify that the interval counter appears after the duration timer is done. 2. Verify that the app sends out notification when the interval timer hits 0.	Yes

Table 8: RV USB Port

Requirement	**Verification**	**Requirement Met?**
Must be able to supply 5±0.5V.	1. Measure the differential voltage of the USB pins from Vcc to GND (This is the voltage the USB will be supplying to the PCB). 2. Verify that the output is within our tolerance of ±0.5V.	Yes
USB must transmit information from the microcontroller to the computer.	1. Write and apply dummy code with LED circuit to the microcontroller that won't hurt the system. 2. Test the code to ensure that the microcontroller is able to send data to the computer through the USB data pins and that a simple LED can blink.	No

Appendix C

Table 1: Part Cost Table

Part	Link	Part Number	Quantity	Unit Cost
Force Sensitive Resistor	link(Sparkfun)	FSR 406	1x	$11.25
Female Connector for Pressure Sensor	link(Mouser)	474-COM-14194	1x	$1.95
Std USBType A Connector	https://tinyurl.com/yeur43tb (TE connectivity)	6364372-2	1x	$1.10
USB 3.0 Cable Type A Male to Male	link	N/A	1x	$3.96
5V to 12V DC DC Converter	https://tinyurl.com/2e3yhcr3 (Digi-Key)	PDS1-S5-D12-M	1x	$5.67
Microcontroller	ECEB supply shop	ATMega328-P	1x	$6.40
USB Serial Connector	link	Serial Connector Module	1x	$19.95
USB A to USB-B	Link	0887328702	1x	$3.35
16MHz Crystal	link	16 Mhz crystal and 20 pF caps	1x	$.75
Portable charger	link			$6.00
Stepper Machine	N/A	N/A	1x	$50.00
Motor	Yaskawa Electronics	P12H-DB11	1x	About $45
Approximate Flyback Cost (diodes, MOSFETS, Caps, resistors, gate driver, transformer)	https://www.digikey.com/short/n5rffnjc	N/A	1x	About $45
Rough Total Cost for Parts				$194.38