ESET 219
Digital Electronics

Laboratory Report

Final Project
Krisys Robot Design

Author's Name: Mark Perez

Lab Due Date: December 9, 2022

Lab Submitted Date: December 9, 2022

IS THIS LAB LATE? No

All of the information contained in this report is my own work that I completed as part of this lab assignment. I have not used results or content from any other sources or students.

Mark Perez	Mark Perez
Printed Name	Electronic Signature

Section #1:
Introduction

In this project, we are tasked with using our knowledge from multiple previous labs in order to design and produce a robotic car with the capability of following a line through induction sensors, which is known as the Krisys Robot. In terms of the robot itself, we had to ensure that the robot would be able to move not only straight, but also in the left and right directions depending on the inputs the robot were to receive about its location on the track. In order to take these three situations into account, we had to use our knowledge on state machine design to create a state machine for our three-state robotic controller that would allow for the motors to know how to function when transitioning between the states. To create the main design through Multisim for the motor controller, we used our knowledge on various topics from our previous labs and lectures, which include basic logic circuitry, flip-flops, frequency division, and pulse width modulation. Hardware-wise, we used the Basys 3 Board, which acted as the brain of our project since it housed our programmed design files, a sensor board, which used inductors to sense where the robot was on the course, and the motor driver chip, which would send the pulses from our pulse width modulator to each individual motor based on the sensor inputs. Along with the main physical chips, we used two motors with their corresponding wheels and two ball bearing wheels in order to have the robot move with less friction.

In doing this project, we were able to use our knowledge from lecture materials and previous labs in order to design and create a project based on certain criteria, which would mirror a possible situation we would face in our future careers. Another aspect of designing and creating this project that we accomplished was being able to “debug” our design when the program was not outputting the intended outputs, which is a skill needed if we were to ever go into any engineering based career. We also were able to work not only on the virtual design of this project, but also on the hands-on, hardware side of the project, which is unlike what we have done in previous laboratory assignments. Due to timing, we were not able to start on this hands-on design from scratch, but since we were using robotic designs that were already built, we were able to test those designs to look for any areas for improvement, which is another aspect we might face in the real-world since we would most likely be working in groups and collaborating our ideas. All in all, in completing this project, we were able to understand and apply the materials we learned in previous labs, as well as learn several techniques we could possibly use in our future projects and potential careers. Below you will find Figure 1.1, which shows the completed Krisys Robot that I designed and built, as well as Figure 1.2, which shows me with the robot.

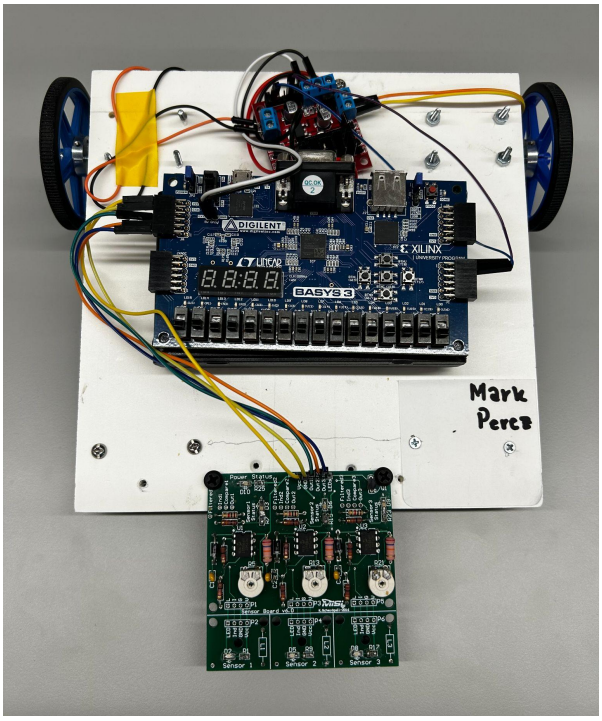

Figure 1.1: Completed Krisys Robot Design

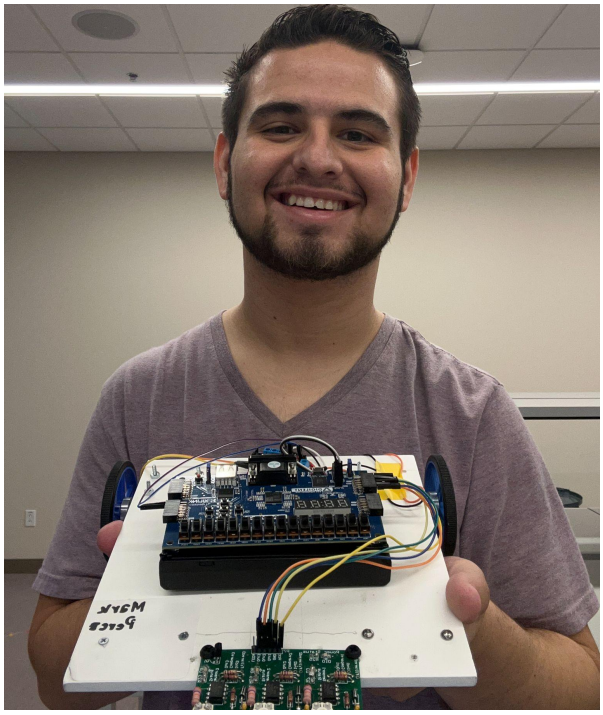

Figure 1.2: Me & The Krisys Robot

Section #2:
Conceptual Block Diagram

Below you will find the conceptual block diagram which was created to show the connections between all of the main hardware components we will be using in order to build our Krisys Robot once we have completed our virtual designs through Multisim. At the center of this design is Basys 3 Board, which as stated above will act as the brain of our robot since it houses the design we created as well as converts the input signals into output pulses. We can see that it will be connected to the Sensor Board via the JA ports, which would allow for the sensor inputs to go through our design we programmed into the Basys 3 Board. The sensor board would use the inductors under the chip itself to sense where the robot is on the course, which is a wired track with a current going through it, and the board would then relay any sensor changes to the Basys3 Board. We also have the Motor Driver chip connected to the JC ports as our outputs, which would allow the outputs from our design to be transferred to the motors. The motor chip would relay the pulses from our pulse width modulator from our design to each of the separate motors, which would allow the robot to move in the left, right, and forward directions. Since we will have the device running without a cord connected to it, we also have a battery pack as our power supply, which is connected to Motor Driver via the VCC and GND pins. Since the motor driver chip is receiving the power directly from the source, we used cables in order to connect the power supply to the Basys 3 Board, which would allow for the power supply to power all hardware components on the Krisys Robot. On Figure 2.1, you can see not only all of the hardware components we used, but you can also see each of the connections we made at each of

the labeled ports in order to have the robot run correctly once the Multisim design has been programmed into the Basys 3 Board. In Table 2.1, you can see the table that was developed, which houses all sensor input combinations that were implemented into our three-state motor controller design. Along with the input combinations that were considered when developing our state machine, the table below also shows what each of those numerical combinations signifies in terms of the sensor that is activated. We also displayed what action the robotic device we built would take in order to stay on the intended course, such as making a right hand movement if the device gets too far to the right to correct itself, and the duty cycle the device would need for each motor to have that intended action occur.

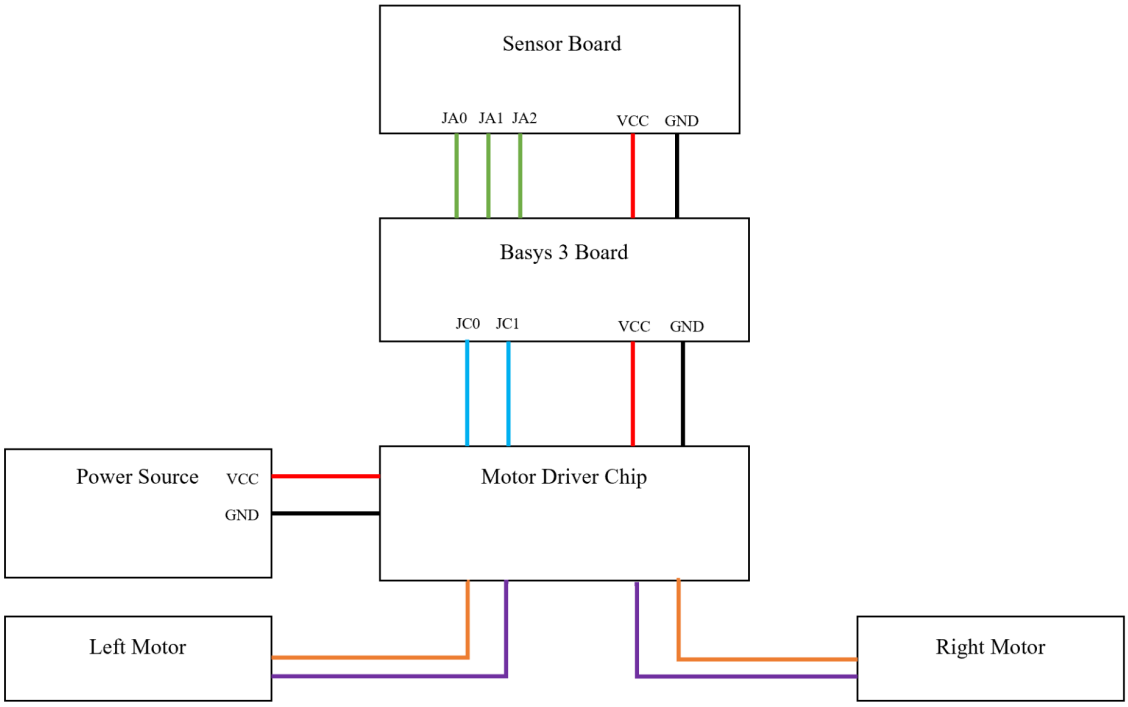

Figure 2.1: Conceptual Block Diagram of Hardware Setup

Table 2.1: Sensor Input Meanings and Duty Cycles

Sensor Input	Meaning	Action	Duty Cycle
101	Middle Sensor Activated	Continue Straight	Left Motor = 56.25% Right Motor = 56.25%
011	Left Sensor Activated	Left Turn Motion Till Middle Sensor Activated	Left Motor = 56.25% Right Motor = 0%
110	Right Sensor Activated	Right Turn Motion Till Middle Sensor Activated	Left Motor = 0% Right Motor = 56.25%

Section #3:
State Machine

In Figure 3.1, you will find the state machine diagram that was developed using the table seen in the previous section, which would allow us to create the design through Multisim that would allow our robot to function properly. On the state machine diagram below, you can see three circular figures, which are the three states we intend the robot to allow to transition between, which are know as S_0, which would allow the robot to go straight, S_1, which would have the robot make a right turn, and S_2, which would have the robot make a left turn. Inside each of those state figures, we can see what would happen to the motors in order to have them work as they are intended to work. Since our design is Active-Low, meaning that low signals would activate the states, you can see that to make the robot go straight, both of the motors would be active. To allow the robot to make a right turn, the right motor must be active while the left motor is inactive, and the vice versa is true if you want the robot to make a left-hand turn.

Along with the states and their corresponding motor combinations, we also have the combinations from the sensors that would allow for the robot to transition between each of the states. These transitions between states are shown in the figure below through the arrows that go from state to state or the arrows that allow for the state to remain at its state. Next to each of the arrows is the combination from the sensor board that would allow for the robot to transition from one state to another along with the duty cycle that would be used for each of the motors that would allow for the robot to easily transition into the state without going off of the track. The inputs for these transitions between states are also active-low, which means that when say one sensor is activated showing that the robot is too far to the right per say then the left and middle sensors would be active while the right would not. Knowing that the sensor board is active low allowed for us to create the combinations that are shown on the design below. As you can see from the design, if you were to want to transition from either the left or right state to the straight state, the sensor board would have to receive that only the middle sensor is activated, which would allow the robot to transition from any state to the straight state. If you were wanting the robot to transition from any state to the right state, then you have to have the right sensor activated on the sensor board, which would allow the robot to balance the robot to get the robot onto the straight sensor. If you wanted to make a left turn, then you would have to have the left sensor activated since the robot would then make a left turn to balance the robot out and allow it to go in the straight direction again.

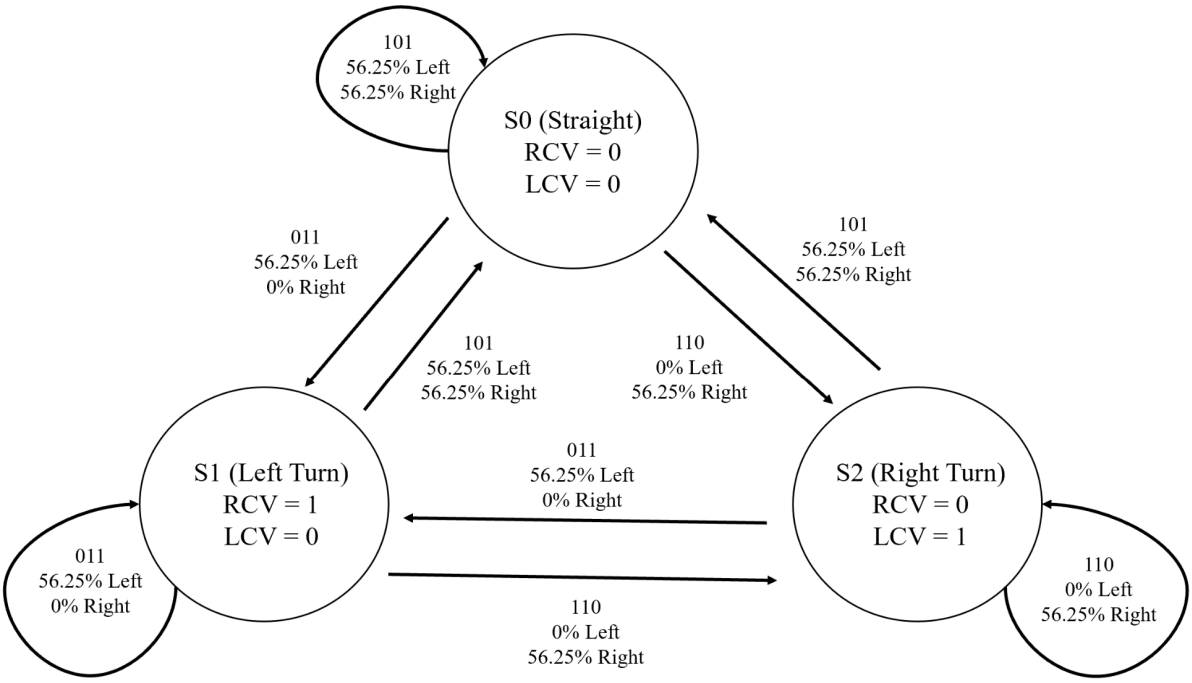

Figure 3.1: State Machine Diagram for Three State Robotic Motor Controller

Section #4:
Schematics

Below, in Figure 4.1, is the schematic block diagram that was used in order to develop our Multisim design for the three-state robotic motor controller. As you can see from the block diagram, we will be using multiple aspects we have discussed and created in previous labs such as the frequency divider and pulse width modulators. In regards to the frequency divider, we can see that the Basys 3 Clock input is connected to the input of the frequency divider, which would allow for us to slow down the high frequency from the Basys 3 Board to a lower frequency. The output from the frequency divider would then be connected to all of the clock inputs on the pulse width modulators and flip-flops on the state machine. In regards to the state machine, the inputs from the state machine would be connected to the JA inputs, which in turn would be connected to the sensor board that would relay the sensor combinations to the state machine. The outputs from the state machine would be connected to the pulse width modulator for each of the motors using several logic combinations. Once we have connected the state machine with the pulse width modulators via logic circuitry, we would choose the output to connect to the JC pins that would act as our motor outputs.

Below you will also find Figures 4.2, 4.3, and 4.4, which show the Multisim circuitry for the frequency divider, pulse width modulator, and the complete circuitry for our three-state robotic motor controller. In order to create the frequency divider, multiple counters and T Flip-Flops were cascaded together in order to create the intended frequency. For the pulse width modulator, we used two devices, a four-bit counter and a four-bit comparator. The four-bit

counter would feed into the A values on the comparator, and we would store a value using Digital High's and Low's for the B-Value. This would allow us to have a certain percentage duty cycle depending on the value we choose to place as our B-Value. The connections between the frequency divider, state machine diagram, and pulse width modulators can be seen in Figure 4.4 below. As you can see from the completed schematic in Figure 4.4, we get signals fed in from the sensor board via the JA pins which goes through our state machine, via the two D Flip-Flops, and the outputs from our state machine are connected to each of the pulse width modulators via logic circuitry.

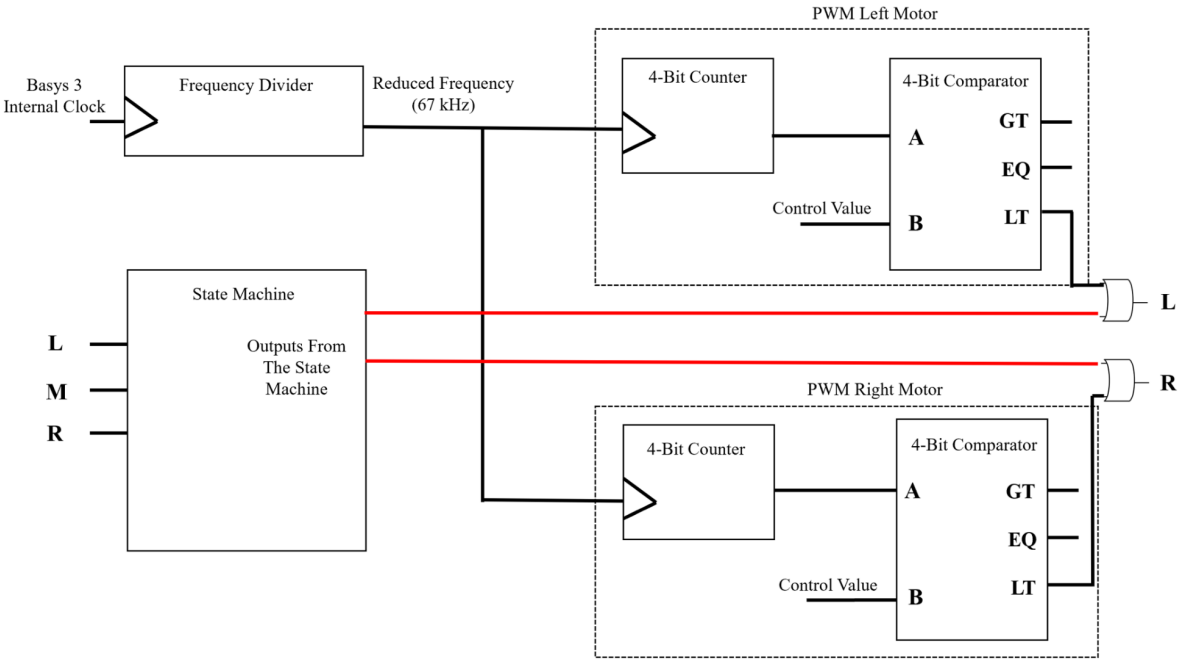

Figure 4.1: Block Diagram For Multisim Design of Three-State Motor Controller

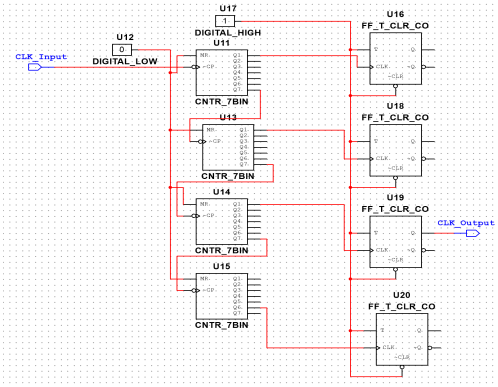

Figure 4.2: Frequency Divider Schematic

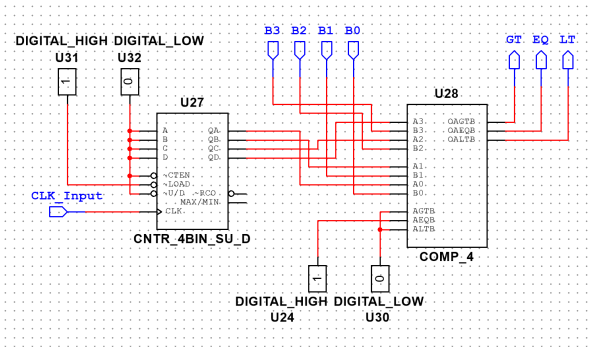

Figure 4.3: Pulse Width Modulator Schematic

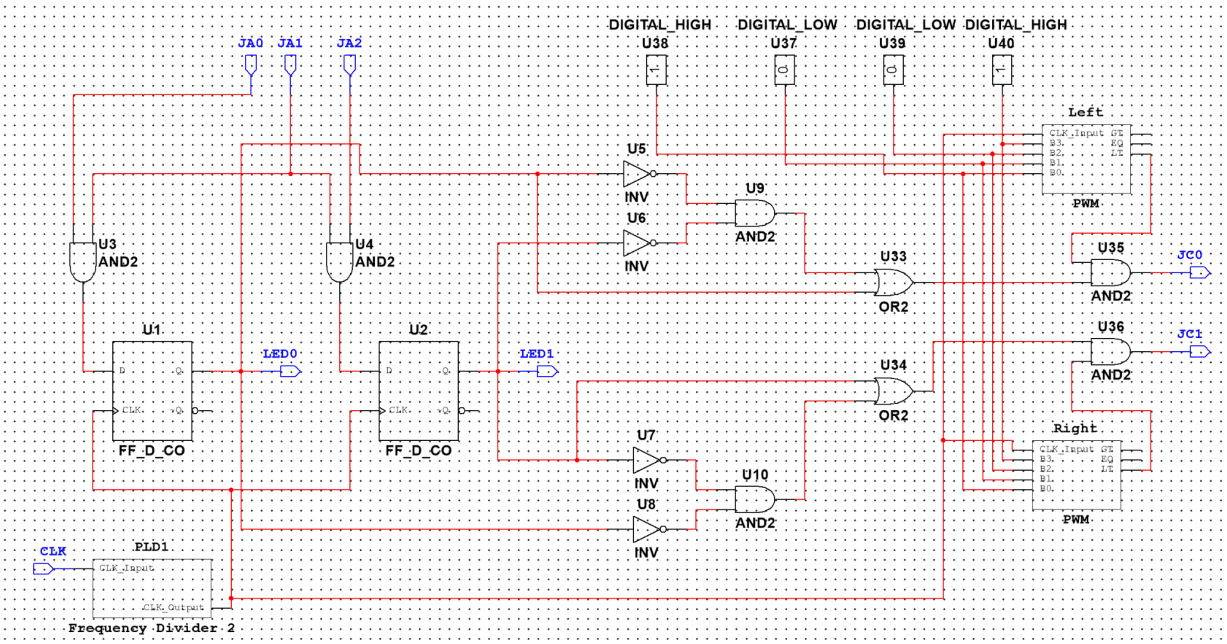

Figure 4.4: Complete Multisim Design for 3-State Robotic Motor Controller

Section #5:
Testing & Optimization

In regards to the initial testing that was performed in this project, we first needed to ensure that the state machine design we developed using our state machine diagram and equations was working correctly. In order to do this, we used LEDs that we placed on the outputs of our two flip-flops and switches for the three inputs, as well as the frequency divider on the clock input since we would need to slow down the Basys 3 Boards internal clock to view the changes. We then programmed just the state machine portion into our Basys 3 Board, which allowed us to use the switches and LEDs to test each input and examine the outputs. Once we designed the pulse width modulators and connected it to the state machine design, we used the oscillator in the lab to see the change in pulse width as the switches went from state to state.

Although when we tested the design on the hardware components, it was able to follow most of the track, we did need to try to alter the design of the hardware and Multisim design in order to optimize its functions. In regards to the Multisim design, the main area that needed optimization was the control value, or B-Value, that I was going to use in my completed design. The first B value that I used was the binary value 1011, which is 11 in decimal values, but during testing, it seemed too fast to follow the course accurately. Over multiple iterations of testing the robot, I decided that the binary value of 1001, which is 9 in decimal values, tested quite well since it was still quite quick but it was still very accurate when it came to following the track on both sharp and subtle curves. In regards to the optimization done on the hardware portion of the robot, the two main areas that needed improvements were the sensitivity of the inductor sensors

and the number of ball bearing wheels and their placement. To optimize the sensitivity of the inductors, we used a flathead screwdriver to adjust the sensitivity of the sensors via the potentiometer for each sensor. For each test of the sensors, we changed how sensitive it was and observed whether the changes had a positive or negative impact on the performance of the robot. Another aspect that optimized the performance of the robot was changing the number and placement of the ball bearing wheels. Since we were using pre-designed devices, most had only one ball bearing wheel, but during testing, I noticed that the ball bearing would get caught on the tape holding the wire to the floor. In order to fix this issue and optimize the performance of the robot, I opted to add an additional ball bearing wheel and place them at each corner of the board. After testing the robot with the additional ball bearing and new placement, there were far less times that the robot would stall due to friction due to the tape and wheels.

Section #6:
Completed System

Below in Figure 6.1 you will again see a completed visual of the final design that was used to successfully complete and follow the track based on the inductor sensors. One of the biggest areas of improvement that was made between the primary and final stages of testing was the addition of a second ball bearing wheel and their placement on our robot. Below in Figure 6.2, you can see the placement of both of the ball bearing wheels on our final design for the robotic motor controller. The main issue with using only one of the ball bearing wheels and having it in the center was that it would cause friction between the tape holding the wire track to the floor and the ball bearing, which would lead to our vehicle stalling on important turns that would interfere with the accuracy of our device. Along with visuals showing several views of our final design, in Figures 6.3 and 6.4, you can see visuals showing us working on the physical design of our robotic motor controller. In Figure 6.3, I was working on trying to screw the sensor board onto the top portion of the robot in order to have the inductor sensors at a certain distance from the floor to ensure they will not cause any interference with the robot's movements while still being able to sense the current from the wire track. In Figure 6.4, I was working on both the new placements of the ball bearing wheels as well as the swapping out of a faulty motor. In the case of the ball bearings, new positions were measured and drilled into the board, and the two ball bearing wheels were attached in their new locations using screws. In regards to the faulty motor, we were working with parts that were previously soldered by past students, so the robot I was building had a faulty motor that had to be replaced. Once those changes were made, we were left with a fully functional robotic motor controller robot, which is again seen in Figure 6.1.

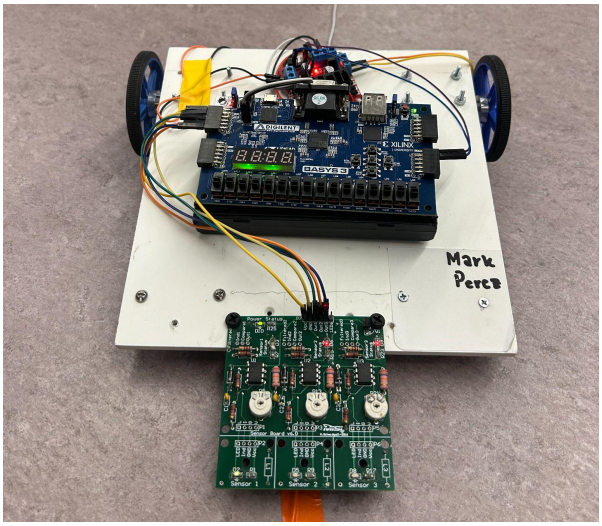

Figure 6.1: Completed Krisys Robot Design

Figure 6.2: Ball Bearing Wheel Positioning

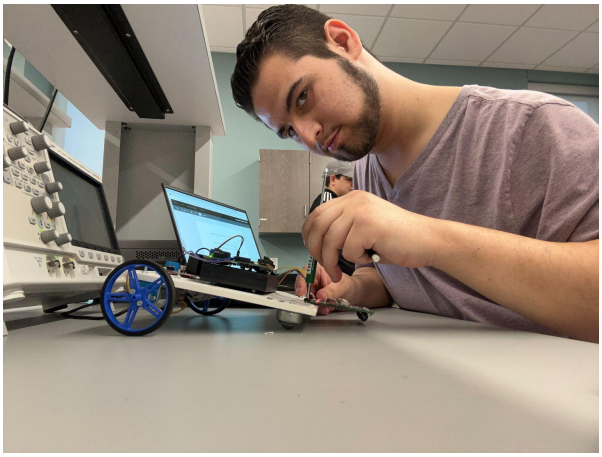

Figure 6.3: Working on Sensor Board Placement

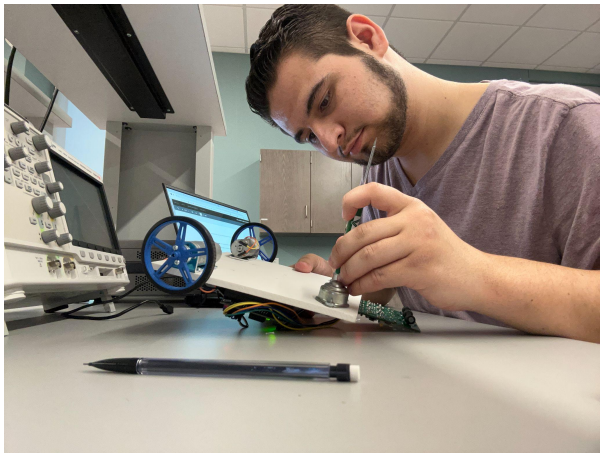

Figure 6.4: Working on Wheel Placement and Motors

Section #7:
Conclusion

All in all, this project allowed us to test our knowledge on the concepts we have learned over the semester in both the laboratory experiments as well as the lectures. In completing the entirety of this project, we were able to see how everything we learned from the beginning of the semester can be used cohesively to create various different systems, in this case the motor controller for our Krisys Robot. Although this is just one example of a system that can be created using the knowledge we have learned in this course, the possibilities are completely endless after learning about the processes for creating, designing, and implementing a state machine, which was the main topic used in this project. In doing this project, we also learned several techniques in order to test and optimize the performance of our program and robot, which are “debugging” techniques that could be used in later courses or any future jobs in our careers.

Although our project was able to successfully run the course, there were problems we had to overcome in order to have a successful design and completed project. One of the biggest problems that we encountered when designing the operational system had to be ensuring that the control values we chose were right for the track we were going to have our robot follow. Through the testing of many different control values, we were able to settle on the control value that offered the robot the best performance speed-wise while also being accurate and precise on the course. Another aspect that I encountered difficulty in when it came to designing the operating system was developing the state machine itself since the state machine essentially was the most important portion of this project. Since we were dealing with two motors and three inputs from the sensors, I encountered difficulty obtaining the equations that would be necessary to develop my circuitry using reduced formulas. In order to obtain the correct equations, I found online portals that were able to guide me through using the Karnaugh Mapping methods in order to obtain the necessary equations.

Section #8:
Recommendations for Improvements/Enhancements

Looking back at the process for our successfully designed Krisys Robot, there were many ways that this project could have been improved to have our robots that we create run smoothly and efficiently. One of the biggest ways I found that this project can be improved to allow the robot to run with less error is to change the minimum number of states from three states to five states. I feel that by adding the two states where the robot is between the middle and right sensor and between the middle and left sensor would allow the robot to be more precise in its movements and allow for less errors during testing. I also feel that there should be a specific wire to be used for the track that can be fastened to the ground without the use of tape since the tape that was securing the wire track to the floor was causing our robot to stall due to friction. Lastly, in order to better improve this project, the addition of a phase where the robot could go in reverse if ever lost from the track would help the robot function to its fullest potential.

When looking at the project as a whole after its completion, the ups and downs of this project are very evident. One of the best aspects that I encountered doing this project was seeing the completed robot work successfully on the track despite the previous errors I was facing. Seeing my project successfully run the track after spending weeks on end working on the design and construction of the robot only ensures that this will not be the last engineering project I will design in my lifetime. The only bad part I feel I encountered during the entirety of this project had to be the sensor board since it wasn't always working the way it was intended to work. There were instances where the inductor sensors would stop working or glitch causing the robot to glitch and stray from the track, so I feel that if the sensor board were to be swapped for something that is more reliable, our robots would be better equipped to run the track accurately.