METU Electrical and Electronics Engineering Department

EE463-STATIC POWER CONVERSION I

TERM PROJECT – FINAL REPORT

GROUP 5

Ekin Arda Çömez - 2374791

Ahmetcan Akuz - 2650315

Utku Deniz Altıok – 2457596

Table of Contents

1. Introduction

In this project report there will be detailed discussion of the project, which is a High Voltage DC Motor Driver Circuit. There will be discussion on topology selection, component selection, power analysis, PCB design, test results and closed loop current control.

The three phase rectifier is a vital component of the system, as it converts the incoming AC power from the power grid into nearly a DC power, which is then fed into the buck converter to adjust the voltage level to match the desired voltage level for the DC motor. The rectifier circuit will be a full-wave bridge rectifier, consisting of six diodes, which will convert the three-phase AC input into a near DC output.

The buck converter, also known as a step-down converter, is a non-isolated DC-DC converter that reduces the voltage level of the DC input to the desired level for the DC motor. This converter will be implemented using a high-frequency switching technique and will be controlled by a PWM signal generated by a microcontroller. The microcontroller will also be used to control the current output by varying the duty cycle of the PWM, this control operation will be closed loop control with the help of a current measurement sensor.

This system is designed to have less loss under the high load, such as up to the 2 kW rated power, selection and design implementations will be discussed later in detail. Difficulties that we faced and how we ended up with the solutions also discussed in the relative subtopics. Design ideas and approaches, simulations results, analytical solutions and final test/demo results are given at each subtopic.

2. Topology Selection & Simulation Results

2.1 Topology Selection

We used a three phase rectifier to have a voltage which has less ripple and less THD. After the rectification process we would obtain an average voltage value of $1.35\,V_{line-line}$ which is not a DC but more DC like than a regular 1-phase voltage source. This output will be connected to a DC-link capacitor of $940\mu F$ which has a voltage rating of 450V then the sinusoidal characteristics of the output voltage which is discussed above is filtered to achieve a DC voltage. This DC voltage is then connected to a buck converter to control the voltage with a variable duty cycle PWM signal. A regular buck

converter can be seen at the figure 2.1. Our main topology is the combination discussed above with minor additions such as gate drivers, LDO regulators, microcontrollers and some required resistors, capacitors.

We made some changes to this regular version. First and most important one is changing the position of the mosfet, in the regular version source pin of the mosfet is not grounded and it will be at high voltage, 180V, so to open this mosfet one should apply $180 + opening\ voltage$ In our case the safe opening voltage was 10 V so we should apply 190 V to open this voltage. Or a bootstrap gate driver should be used which means an extra component which needs to be carefully checked and adjusted. But if we move our mosfet to the bottom side which is low side as it can be seen in figure x now the source pin is grounded so we can open this mosfet by only applying a voltage above its threshold value directly.

Another modification which we did is removing the inductor and capacitor at the output. The capacitor is used to filter voltage and the inductor is used to filter current. The DC motor actually does not need a perfect DC voltage or DC current to operate. It works when an average DC current is applied, current itself should not need to be DC but its average should have some DC value. So we can remove the capacitor since we don't need it, but we should have some inductor to have an average DC current. But luckily the motor itself has an armature inductance, with an additional interpole inductance which is connected as series to the armature which makes 24.5 mH of inductance. So we can use this inductance to filter the current as well therefore we won't need an additional inductance to place it at the buck converter.

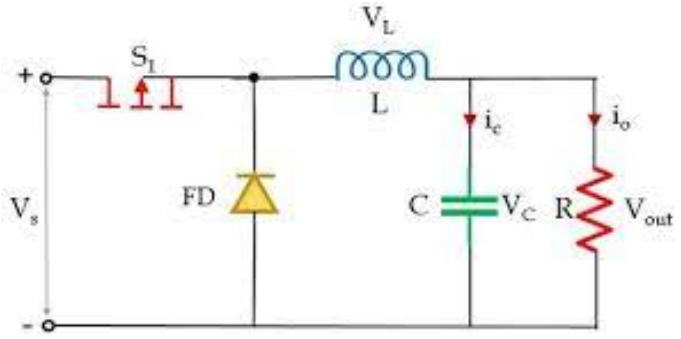

Figure 2.1 A Regular Buck Converter Topology

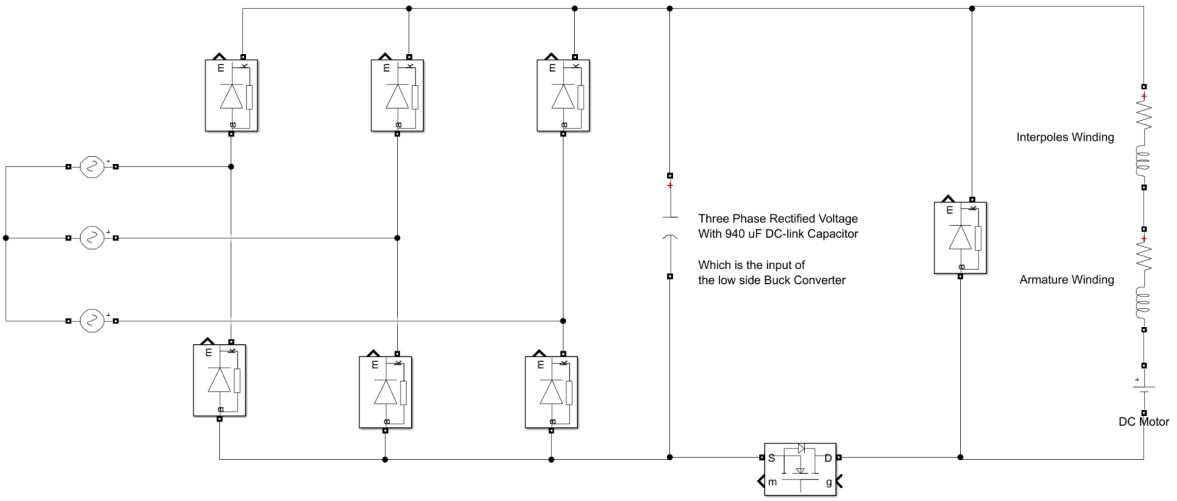

Figure 2.2 The Buck Converter Topology We Revised for the Project

2.2 Simulation Results

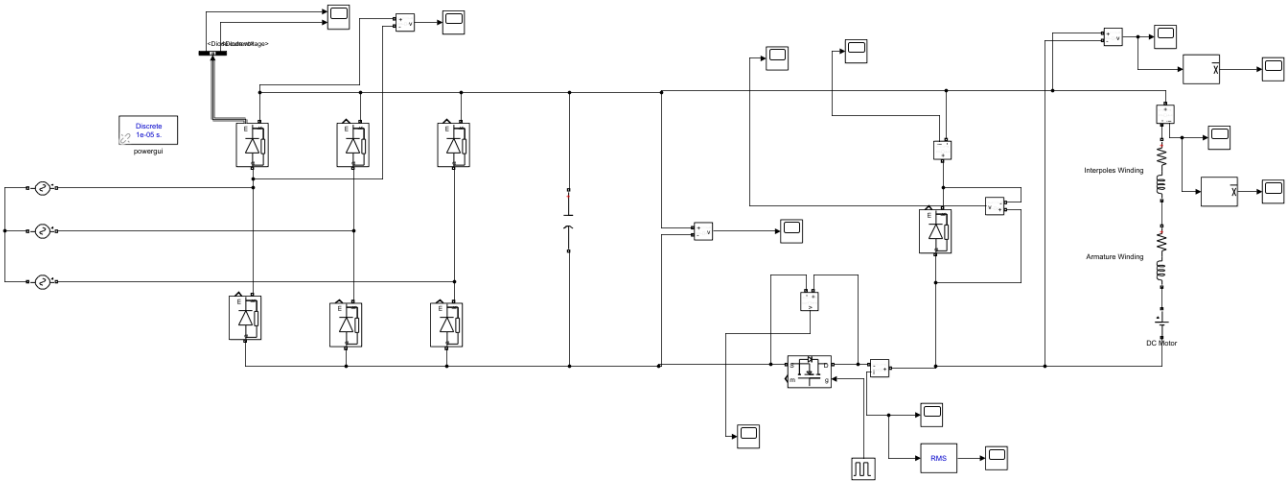

Figure 2.3: Overall Circuit Simulation Schematic For Project

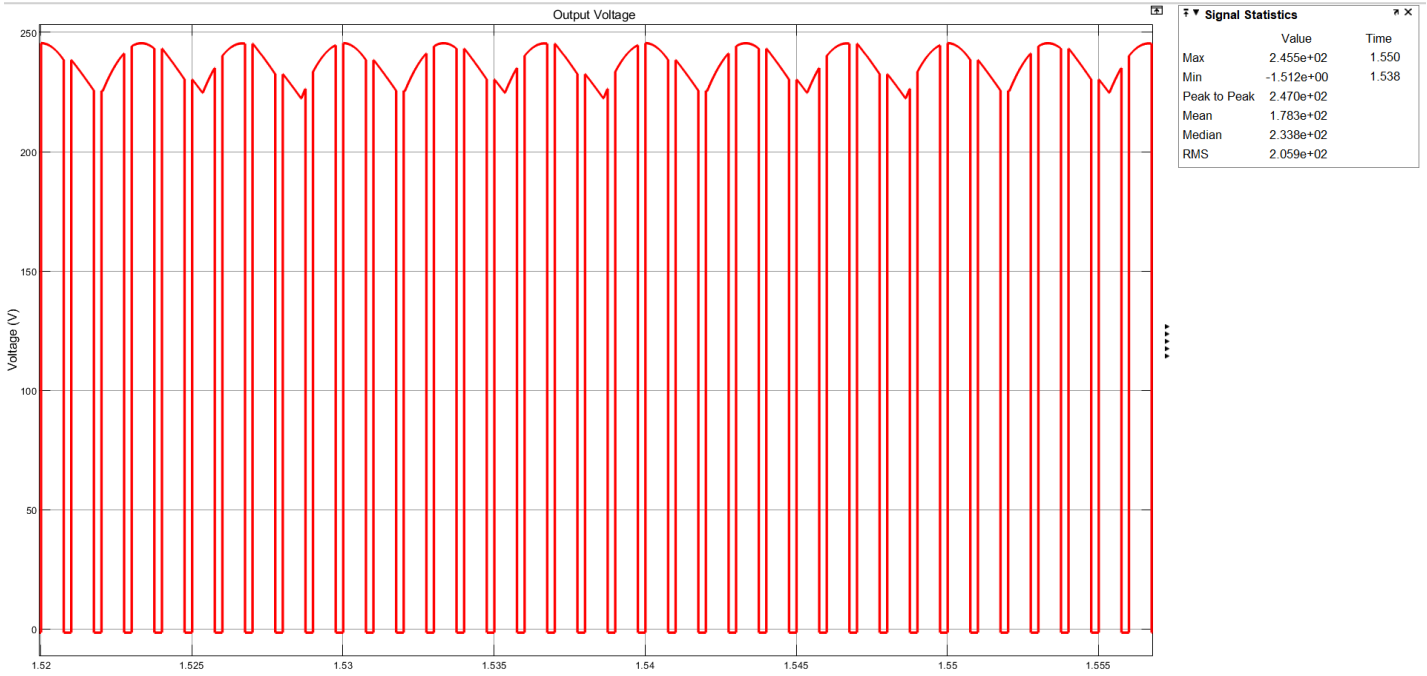

Figure 2.4: Buck Converter Output Voltage

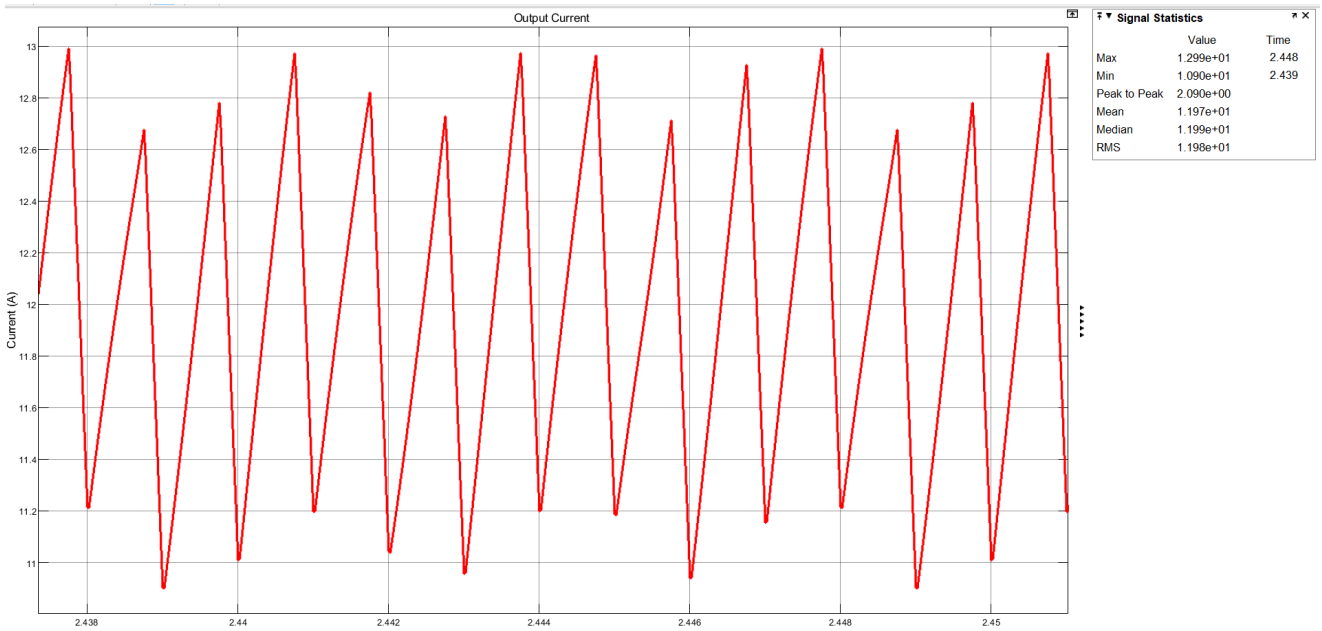

Figure 2.5: Buck Converter Output Current

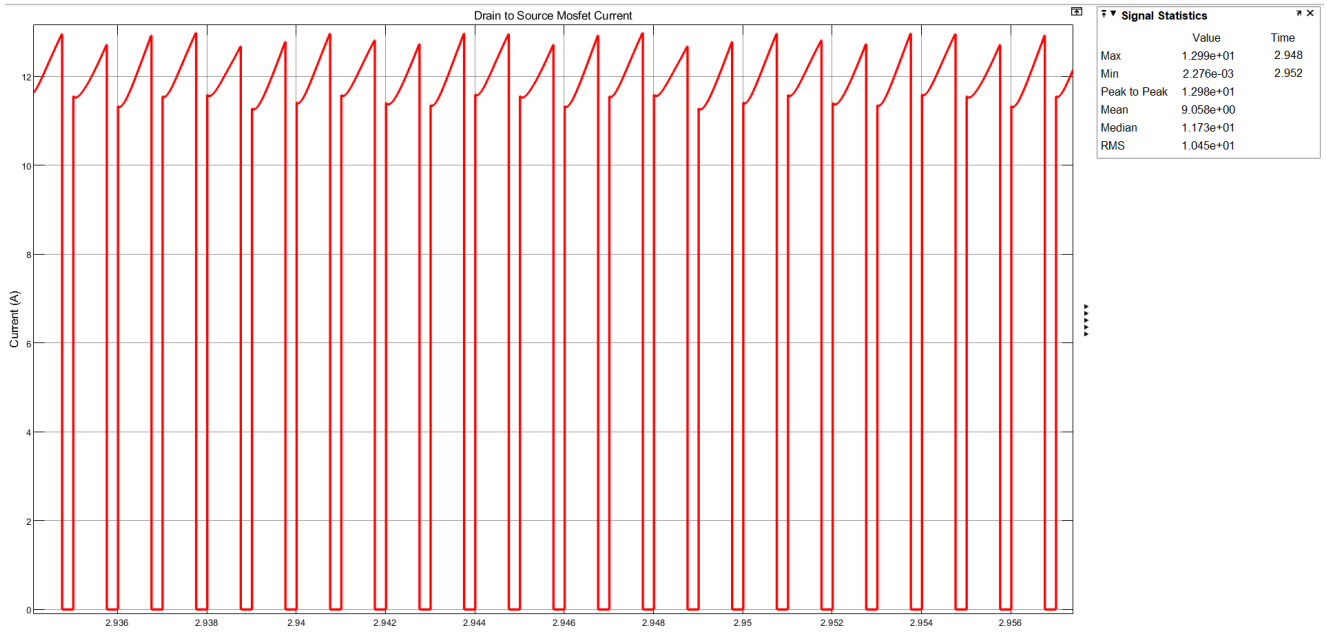

Figure 2.6: Drain To Source Current Of Mosfet

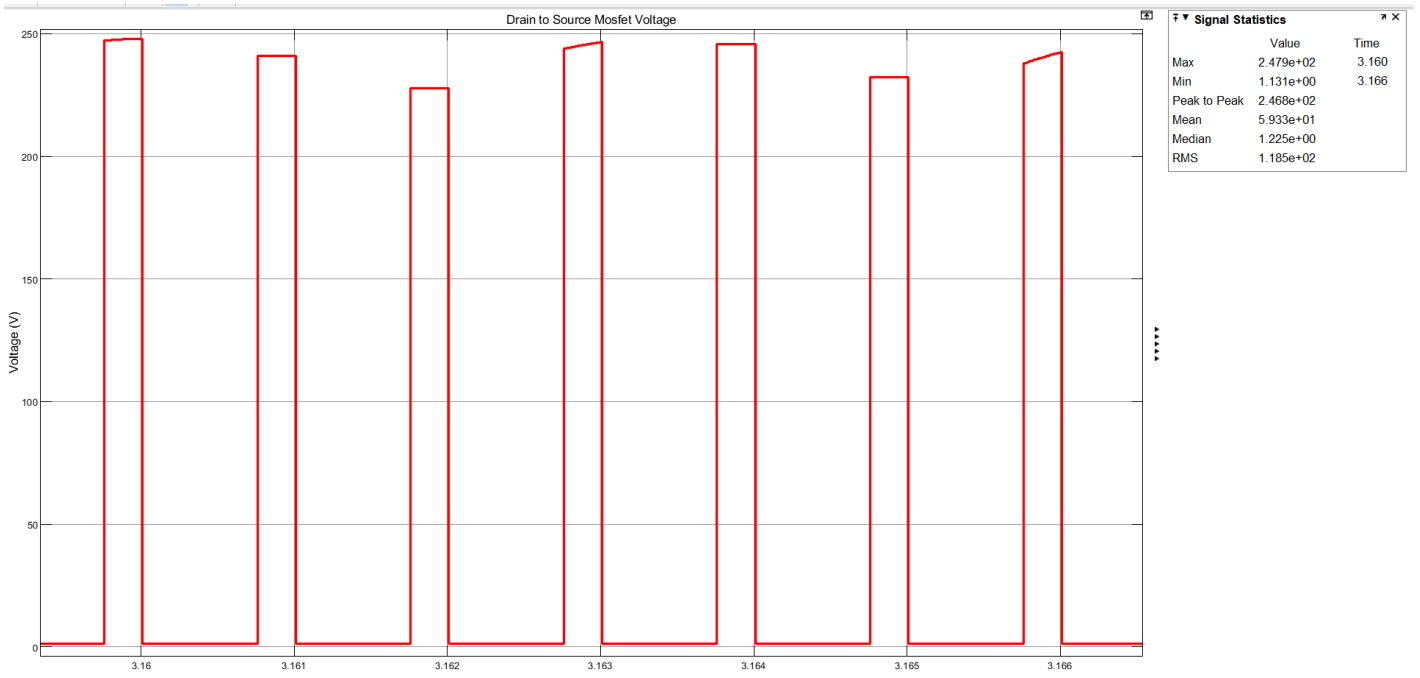

Figure 2.7: Drain To Source Voltage Of Mosfet

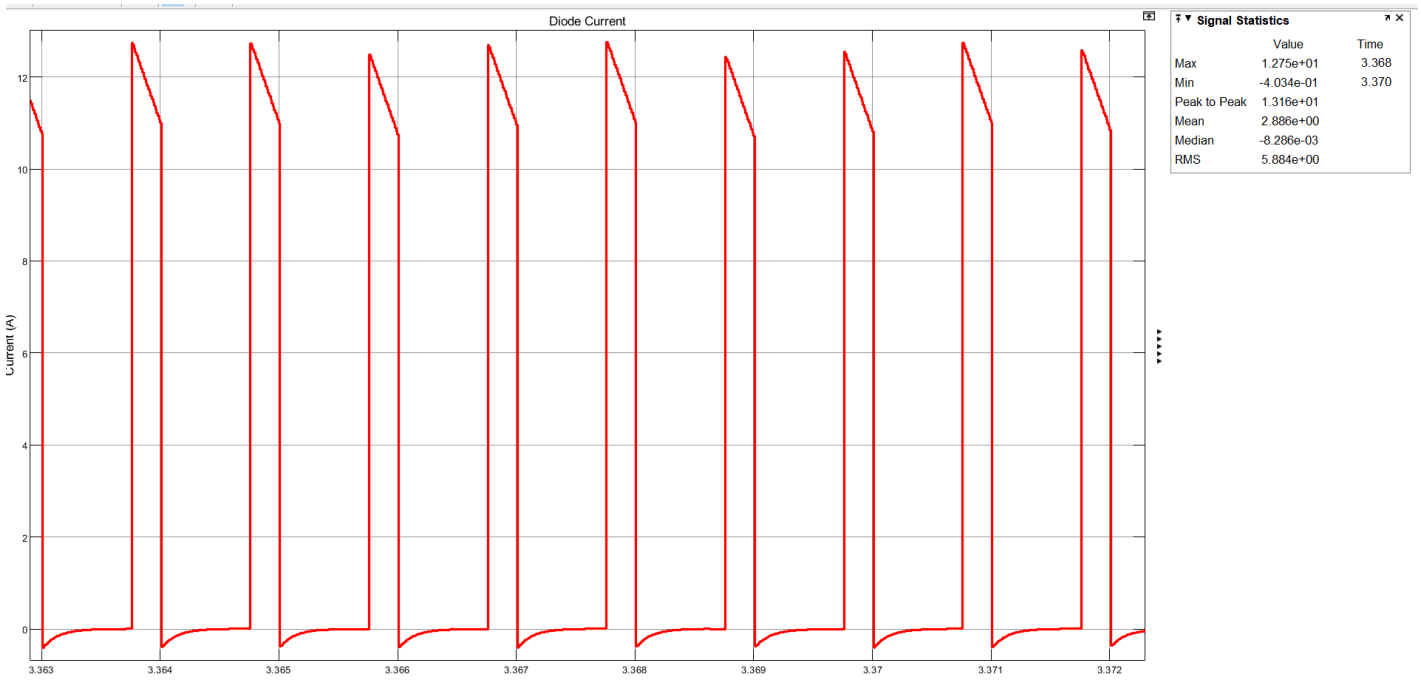

Figure 2.8: Forward Current Of Diode

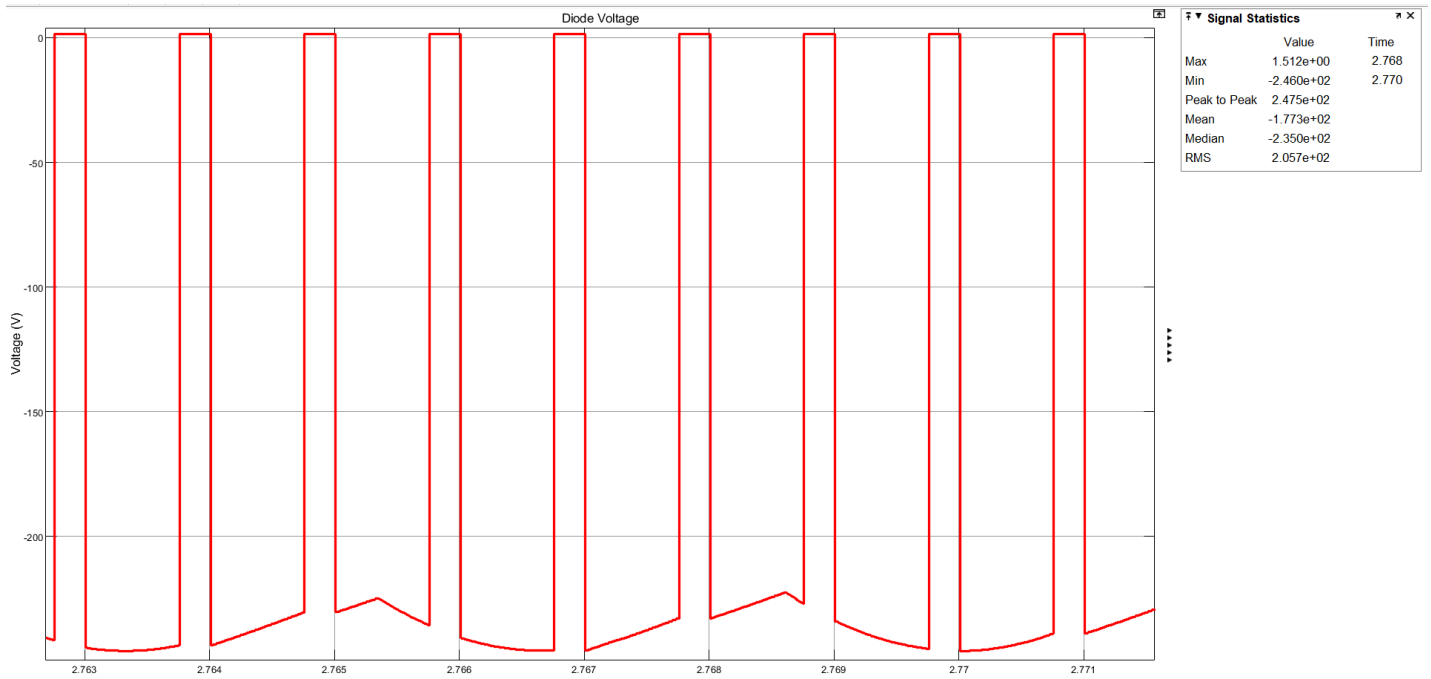

Figure 2.9: Voltage Drop On The Diode

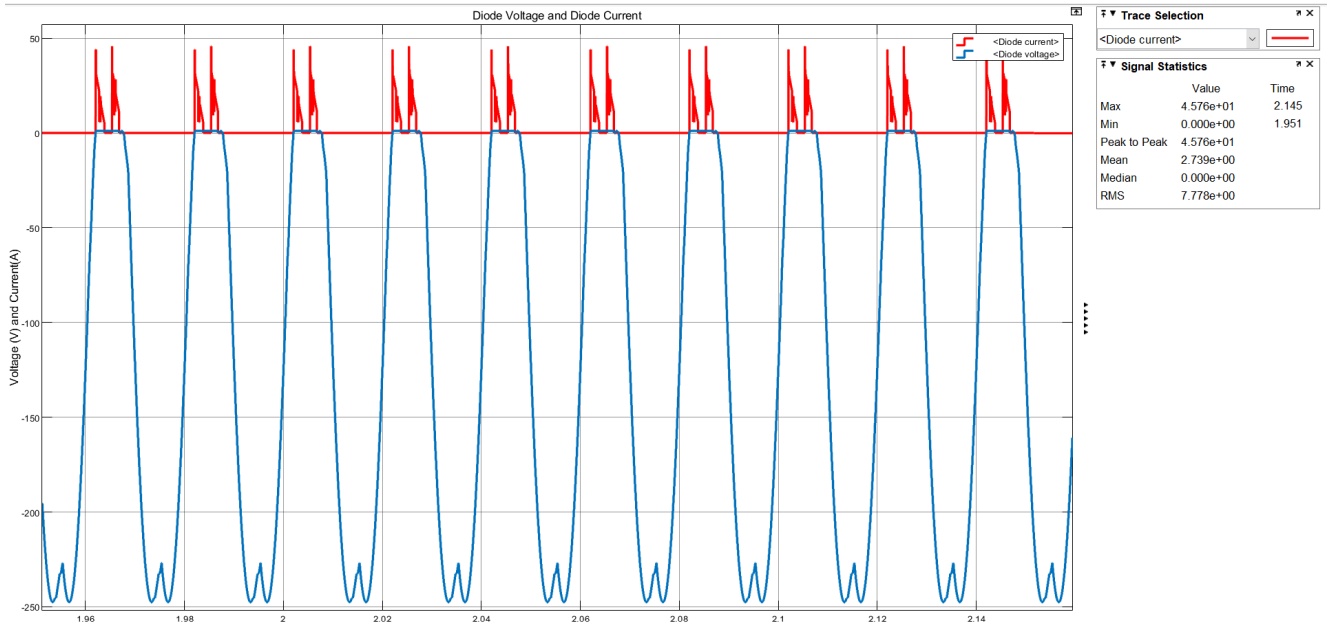

Figure 2.10: Current and Voltage Waveform Of One Diode in Three Phase Diode Rectifier

3. Component Selection

To achieve 2 kW output power for the bonus there should be a required current for it. Maximum applied voltage is given as 180 V as in the requirement list. So for 2kW output supply there should at least be 2000/180=11.2 A current through, let's round this number and call it 12.

Since the armature voltage of the motor will be zero at the start it will draw much more current at the starting phase but we will try to achieve a soft start process so that the motor will start slowly since the voltage applied to the motor will be low, which is achieved with a low duty cycled buck converter, therefore starting current will not be very high compared to the final steady state current after starting the process. After the starting phase since there will be armature voltage, the current drawn will stabilize. The simulation of the starting phase was done at the Simulink and it showed that the maximum required current value at %4 duty cycle is 13 A.

Therefore it can be said that selection of a 20 A maximum rating will be a logical and educated choice. The component selection will be based on this and based on the applications-usage the other parameters will be decided, such as low switching loss for high switched components and low conduction loss for low switched components or voltage ratings should be analyzed carefully.

3.1. Bridge Rectifier

For the full-bridge three phase rectifier, there are lots of compact modules which are small in size but they have all the necessary components inside so instead of using base components separately and constructing the full bridge rectifier. “GUO40-12NO1” will be used which is a compact rectifier with 5 legs, its datasheet is given in the appendix [1]. It has a 1200 V 40 A rating which is suitable for this design and since there is 50 Hz operation at the rectifier side high switching efficient components were not essential.

3.2. MOSFET

A mosfet is a switching component that is the most important component to achieve output voltage lower than applied input voltage. When the mosfet is on, input voltage supplies energy to the DC Motor output side through the three phase diode rectifier and DC link capacitor path. When the mosfet is off, the output side and input side are separated from each other.

Firstly, IXFH30N60X Si mosfet is decided to be used. It has a various advantages for aims of the groups

- Small Rds(on) resistance <= 155 mOhm
- High reverse voltage rating VRRM = 600 V
- Acceptable rise time tr= 43 ns and fall time tf = 33ns

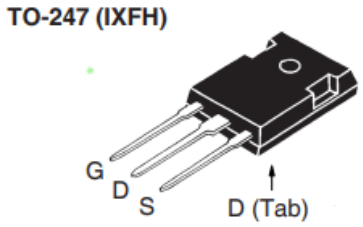

Figure 3.1: IXFH30N60X Si mosfet

Although the IXFH30N60X was a reliable component for the project requirements, it was destroyed two times on the trial part of 180 V output voltage and 12 A output current. The reason for that was, the cables going from Ardunio to the gate

driver circuit were constructing a loop around the PCB at 8 kHz. At this high frequency, an induced magnetic field arises due to the current loop and this field causes induced voltage on the mosfet side. Therefore, the switching mosfet stayed always on because of the induced voltage. Consequently, it was inevitable to harm mosfet.

After the failure of two same mosfets, new mosfets were researched. During this research, IXTQ32N65X switching mosfet was found and it has better properties compared to previous mosfet. They were;

- Smaller Rds(on) resistance <= 135 mOhm
- Higher reverse voltage rating VRRM = 650 V
- Similar rise time tr= 49 ns and smaller fall time tf = 28 ns

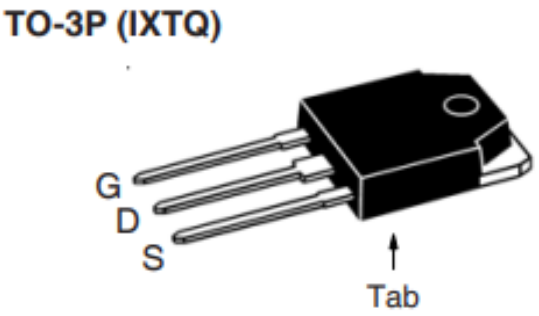

Figure 3.2: IXTQ32N65X Si mosfet

3.3. Freewheeling SiC Diode

A freewheeling diode is commonly used in a buck converter circuit to prevent the inductor from generating a voltage spike when the switching transistor is turned off. When the switching transistor is turned off, the current flowing through the inductor cannot suddenly stop and must continue to flow through a path. The freewheeling diode provides this path, and allows the energy stored in the inductor to be dissipated allowing the current to continue flowing and preventing the voltage spike. This results in a continuous current flow to the output and is necessary to use in the buck converter. To achieve better efficiency in higher frequency and to be able to get the wide bandgap semiconductor bonus we used a SiC diode.

Why did we choose SiC can be examined in 3 topic

-Higher temperature operation: SiC diodes can operate at higher temperatures than Si diodes, which makes them suitable for use in high-temperature environments.And we tested our circuit in 2kW load so it should be resistant to the high temperatures.

-Faster switching speed: SiC diodes have a faster switching speed than Si diodes, which allows for more efficient power conversion in high-frequency applications. We were going to set our switching frequency to 8kHz instead of the default values of 980 Hz.

-Lower power loss: SiC diodes have lower power loss than Si diodes, which leads to less heat generation and increased efficiency.

We used GEN2 SiC Schottky Diode LSIC2SD065A16A, which its datasheet can be found in the appendix, we used a SiC diode because at first we were thinking to set the switching frequency as 8kHz, and SiC diodes are much more efficient compared to the regular diodes when the switching frequency is high, their switching losses are smaller. But after we faced problems switching at 8kHz,it broke the microcontroller and caused the circuit to burn several times due to the magnetic field induction which is caused because of the high current that is passing around the microcontroller, then we swapped our frequency back to the default value of 980Hz.

3.4. PWM Controller and Gate Driver

In this project, to obtain a digital controller and to generate a PWM duty cycle signal in one device, Arduino Nano was used. We chose Arduino Nano as a microcontroller because it is more compact and also it is solderable to the PCB when we compare the other similar microcontrollers. Arduino also has a very easy to use IDE and lots of built in functions to ease the process. Arduino might not be compatible to use in complex projects which have high standards or needs an extensive control operation, but it is one of the best solutions for our project.

Arduino can get digital and analog input and can give digital and analog input from its respective pins. The most important functionality of the Arduino that it also has a function “analogWrite(DutyCycle*255)” which produces a PWM signal according to the given input. A pinout schematic of Arduino Nano can be seen in the figure x given below.

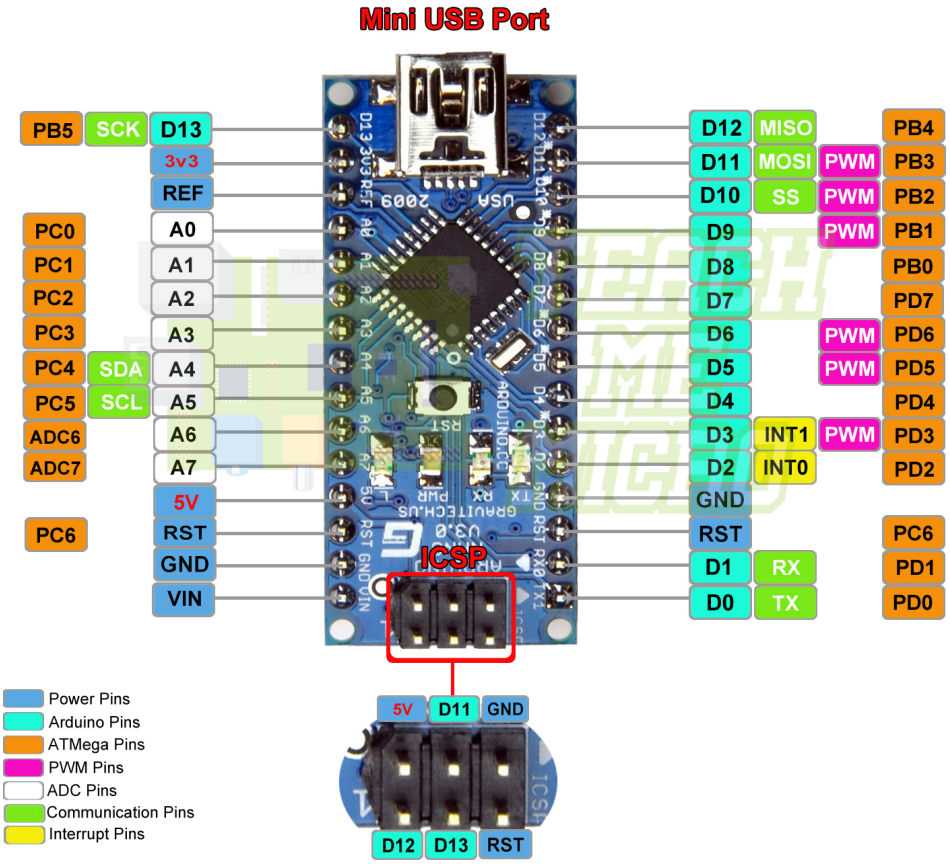

Figure 3.3 Arduino Nano Pinout in Detail

The pins we used in the Arduino Nano are : D5, D8, D9, A0, Vin, 5V and Gnd. We constructed conducting lines according to the pins that we used in our PCB as well. Digital pins can only read 1 and 0, on the other hand analog pins can read values between 0 and 5 V. Default duty cycle frequency of D5 is 976 Hz and this can be adjusted by dividing the crystal oscillator in the Arduino but it can only have multiples of this frequency, it can be 2kHz, 4kHz ,8kHz up to the 64kHz. We firstly used 8kHz for switching frequency but this caused some magnetic field problems which induced voltage causing the arduino to fail, so we changed to the 976 Hz default back.

We used D5 to give a PWM output signal of 980 Hz to our optocoupler, a gate driver which is discussed below, D8 and D9 for digital inputs which will be get from the buttons, buttons are used to give reference current values. A0 is used to read the voltage output value of the current sensor. Vin is used to supply power to the Arduino where this voltage is coming from the 7808. Gnd is used to connect the common ground of the circuit with the Arduino ground. And finally 5V output is used to power on the current sensor. To sum up, Arduino is used to give PWM, read the output current measurement and control the output current with different control mechanisms by changing the duty cycle of the PWM. The Arduino code can be seen at the appendix.

Driving MOSFET from an arduino pin is not possible since the current value supplied from arduino pins and voltage is not enough to open a power MOSFET. Therefore, a gate driver must be used in this topology. TLP250 was the isolated optocoupler gate driver that was used in this project. Its datasheet can be found in the appendix [5]. With the help of LED inside the IC circuit, it gives isolated voltage from input to output. TLP250 also provides the MOSFET enough voltage (up to 35V) and enough gate current (up to 1.5A) to open the MOSFET. The input current range needed to TLP250 to work accurately is 7-10mA.

TLP250 has its own recommended circuit. This circuit is shown in Figure 3.1 for low side MOSFET. There are two resistance for the gate driver and pull-down. The suggestion for TLP250 is 10Ω and 10kΩ respectively. These are shown in Figure 3.1 and R2 and R3. However, to obtain more smooth switching characteristics, switching speed of the circuit was slowed a little. Therefore 20Ω gate driver resistance was used instead of 10Ω. There is also another resistance shown as R1. The reason for this resistance is limiting the current drawing from the arduino.

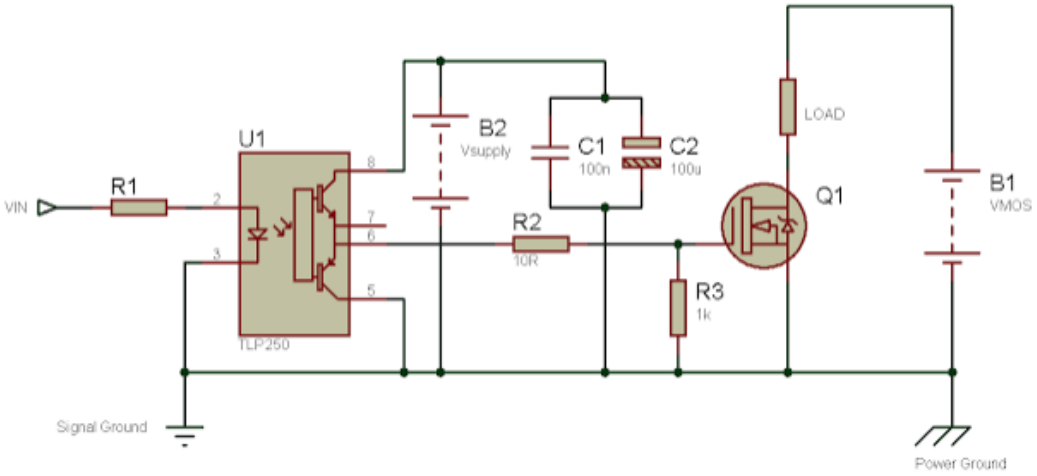

Figure 3.4 TLP250 Gate Driver Circuit Topology

The D5 pin of the arduino is connected to the VIN pin in Figure 3.1. Ground is connected to ground of TLP250. There is an internal diode to transfer power. The forward voltage of the diode is 0.8V. To obtain enough input current level, 510Ω was enough.

3.5. LDO

As a requirement of this project, it would be acceptable to use only one external power supply. On the other hand, there are two different voltage levels for gate driver and Arduino. Arduino should have less than 12V power supply. However, TLP250 does not work properly when Vcc is 12V. Therefore, 15V to 8V LDO was used in this design To obtain this conversion, a 7808CV linear regulator has been chosen.

The recommended circuit for 7808CV is shown in Figure 3.2. To eliminate the noise on input and output voltage, there should be capacitors as recommended in the datasheet. These were added as parallel capacitors to input and output.

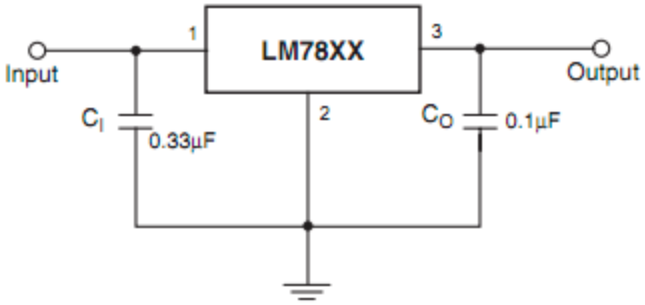

Figure 3.5 7808CV Recommended Circuit Design

3.6. Passive Components

For the DC-link capacitor, two 470uF electrolytic capacitors were used. Moreover, pull-down, TLP250 input and gate driver resistance were chosen as 510Ω, 10kΩ and 20Ω respectively. There are two 100nF, two 10nF and one 330nF capacitors to eliminate noises at input and output.

3.7. Current Sensor “ACS712”

Figure 3.6: ACS712 Current Sensor

The most crucial component for the closed loop control bonus is the measurement device. We used an Arduino-compatible sensor module which is called ACS712. ACS712 has different current ratings as 5, 20, and 30. We chose 30 A because, in the 2kW load, the motor may draw current up to 12 A RMS so it makes 17A peak value, we wanted to stay in the safe range. The working principle of this sensor is that by using the hall effect, the current passing through a conductor will induce voltage at the near side and by reading this voltage which occurs at another conductıor it will approximately measure the current passing through. Using this sensor is beneficial since it isolates the high power side current from the low side logic current. The sensor induces some voltage at the output pins and we can read it by the Arduino Nano analog input pin. Then in the algorithm, there is a mapping function that takes this value and then converts it to the current value.

The main problem in this measurement device is that it takes the instantaneous current value at each cycle of the microcontroller. Our current passing through is an AC triangular-like current so if this sensor measures at all irrelevant parts of the signal all calculations will be wrong. This is the reason that we could not demonstrate our closed loop current control bonus at the demo. We were not aware that we would get an instantaneous value, not an RMS value. To fix this we take the RMS value by collecting enough data at each measure call. There is an ACS712.h library in the Arduino and it calculates the RMS with

respect to the frequency. We are switching at 980 Hz so our frequency of the signal is 980 Hz. After these arrangements, we were measuring the correct enough values.

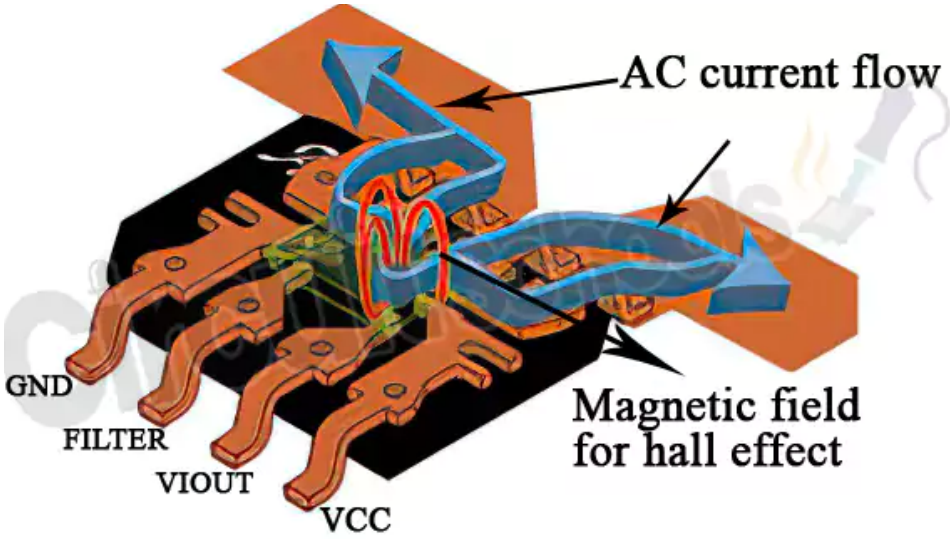

Figure 3.7: ACS712 Current Sensor Working Principle

The main issue with our control process was the instability and non-linearity of the measurement device and the DC motor. The measurement device had unrealistic spikes and offsets, while the DC motor's current was difficult to stabilize and linearize due to the discrete voltage values and non-ideal back emf. These uncontrollable factors were the primary source of problems.

As a solution to the noisy output of the measurement device we could apply a software filter, which is basically a low pass filter function implemented at the code, or an analog filter, which is basically an rc to the analog input pins of the Arduino, to reduce the noise. But again we could not think of such a solution.

4. Power Loss Calculations and Thermal Analysis

All the power loss calculations for mosfet, freewheeling diodes and all the 6 diodes in the three phase rectifier will be done for the 180 V output voltage and 12 A output current case. The reason for this is that maximum losses will arise under these conditions so that maximum junction temperature values happen in this situation. If all components would handle these operating conditions, they would also handle other lower voltage and lower current cases.

IXTQ32N65X Switching Mosfet

Conduction Losses

Conduction losses of the mosfet happens when the current passes through the mosfet which means the switch is on.

Rds(on) = 135 mOhm

D = 0.75

Imosfet(rms) = 10.45 A from the Simulink

Iout(rms) = 11.98 A from the Simulink

$$Pconduction = Rds(on) * Iout^2 * D = 0.135 * 11.98^2 * 0.75 = 14.53\,W$$

or

$$Pconduction = Rds(on) * Imosfet^2 = 0.135 * 10.45^2 = 14.74\,W$$

Both calculations give almost the same result.

Switching Losses

Switching loss is happening during the transient time that MOSFET is either switching from off to on state or switching from on to off state.

Switching frequency (fs) = 1 kHz

Rise Time(trise) = 49 ns

Fall Time(tfall) = 28 ns

Imosfet(rms) = 10.45 A from the Simulink

Vds (max) = 247.9 V from the Simulink

$$Pswitching = 1/2 * (trise + tfall) * Imosfet * Vds * fs = 0.1\,W$$

Therefore, total loss can be calculated as

$$Ploss = Pswitching + Pconduction = 14.84\,W$$

Thermal Analysis

- Ambient temperature is chosen as Tamb = 30 °C
- According to the datasheet of the mosfet junction temperature Tj can vary between -55 °C to 150 °C. We can choose Tj <= 100 °C for safety.
- Junction to cause thermal resistance Rjc = 0. 25 °C/W at maximum from the datasheet of switching mosfet
- Case to heatsink thermal resistance is Rch = 0.25 °C/W typically from the datasheet.

$$Tj - Tamb = Ploss * (Rjc + Rch + Rha)$$

Therefore,

$$Rha = (Tj - Tamb)/Ploss - Rjc - Rch = 4.217\,°C/W$$

Heatsink Requirement

- Thermal resistance of heatsink should be smaller than or equal to $4.217\,°C/W$ to keep junction temperature less than $100\,°C$.
- The package of mosfet is TO-247 from the datasheet.

LSIC2SD065A16A SiC Diode

Conduction Losses

Typical forward voltage drop (Vf) = 1.5 V from the datasheet of the component.

Average forward current (If) = 2.886 A from the Simulink.

$$Pconduction = Vforward * Iforward = 1.5 * 2.886 = 4.329\,W$$

Switching Losses

Vreverse = 246 V from the Simulink

Maximum reverse current (IRM) = 0.05 mA from the datasheet of the component.

On the other hand, reverse recovery time (trr) is not given in the datasheet of Sic Diode since it is Schottky Diode. Therefore, we could ignore the switching loss for SiC Schottky Diode. However, switching loss formula of diode can be found by,

$$Pswitching = 1/2 * Vreverse * I(RM) * fs * trr$$

Therefore, Ploss = 4.329 V

Thermal Analysis

- The ambient temperature is chosen as Tamb = 30 °C again.
- According to datasheet of the diode junction tempearure Tj can varies between -55 °C to 175 °C. We can choose Tj <= 100 °C for safety.
- Junction to case thermal resistance Rjc = 1.2 °C/W at maximum from the datasheet of SiC Diode.

$$Tj - Tamb = Ploss * (Rjc + Rch + Rha)$$

Therefore,

$$Rha = (Tj - Tamb)/Ploss - Rjc - Rch = 14.97\,°C/W$$

Heatsink Requirement

- Thermal resistance of heatsink should be smaller than or equal to $14.97\,°C/W$ to keep junction temperature less than $100\,°C$.
- The package of diode is TO-220-2L from the datasheet.

GUO40-12NO1 Three Phase Diode Rectifier

The 6 diodes on the three phase diode rectifier have the same characteristic properties since they are in the same module. The calculations will be done for one of them. After that, the results will be multiplied with 6 for total loss calculation and thermal analysis.

Conduction Losses

Typical forward voltage drop (Vf) = 1.06 V from the datasheet of the component.

Average forward current (If) = 2.739 A from the Simulink.

$$Pconduction = Vforward * Iforward = 1.06 * 2.739 = 2.9\,W$$

For one diode on the rectifier. Therefore, total conduction loss is ,

$$Pconduction(total) = 6 * Pconduction = 17.4\,W$$

Switching Losses

Vreverse = 247.7 V from the Simulink.

Maximum reverse current (IRM) = 1.5 mA from the datasheet of the component.

On the other hand, reverse recovery time (trr) is not given in the datasheet of this three phase diode rectifier. Moreover, now the frequency of the diode rectifier is taken 50 Hz which is equal to line frequency. Hence, it is very small as compared to switching frequency of 50 Hz. Therefore, switching loss of diodes in the diode rectifier can be ignored. However, switching loss formula of diode can be found by,

$$Pswitching = 1/2 * Vreverse * I(RM) * fs * trr$$

Hence,

$$Ploss = Pconduction(total) = 17.4\,W$$

Thermal Analysis

- The ambient temperature is chosen as Tamb = 30 °C again.
- According to the datasheet of the diode junction temperature Tj can vary between -40 °C to 175 °C. We can choose Tj <= 140 °C for safety.
- Junction to case thermal resistance Rjc = 4.3 °C/W at maximum from the datasheet.
- Case to heatsink thermal resistance Rch = 0.5 °C/W typically from the datasheet.

$$Tj - Tamb = Ploss * (Rjc + Rch + Rha)$$

Therefore,

$$Rha = (Tj - Tamb)/Ploss - Rjc - Rch = 1.52\,°C/W$$

Heatsink Requirement

- Thermal resistance of heatsink should be smaller than or equal to $1.52\,°C/W$ to keep junction temperature less than $140\,°C$.
- The package of diodes is GUFP from the datasheet.

Heatsink Choosement

Although the heatsink requirement calculations were totally done for each component, adequate thermal resistance information was not available on the Ulus or any website in Turkey. They were sold by size only. However, great effort was made to make the right choice and 12 V Fan is used to guarantee no components are burned. As a result, temperature of the components never exceed $72\,°C$ which is very good.

5. Schematic and PCB Design of Topology

5.1. Schematics of Project

The overall circuit according to component selection and analysis can be seen in Figure 5.1. Since LDO is not an isolated component, isolation of power ground and signal ground is not possible for this design. There is one common ground on the circuit. However, this was modified to eliminate unwanted noises in the second version of PCB.

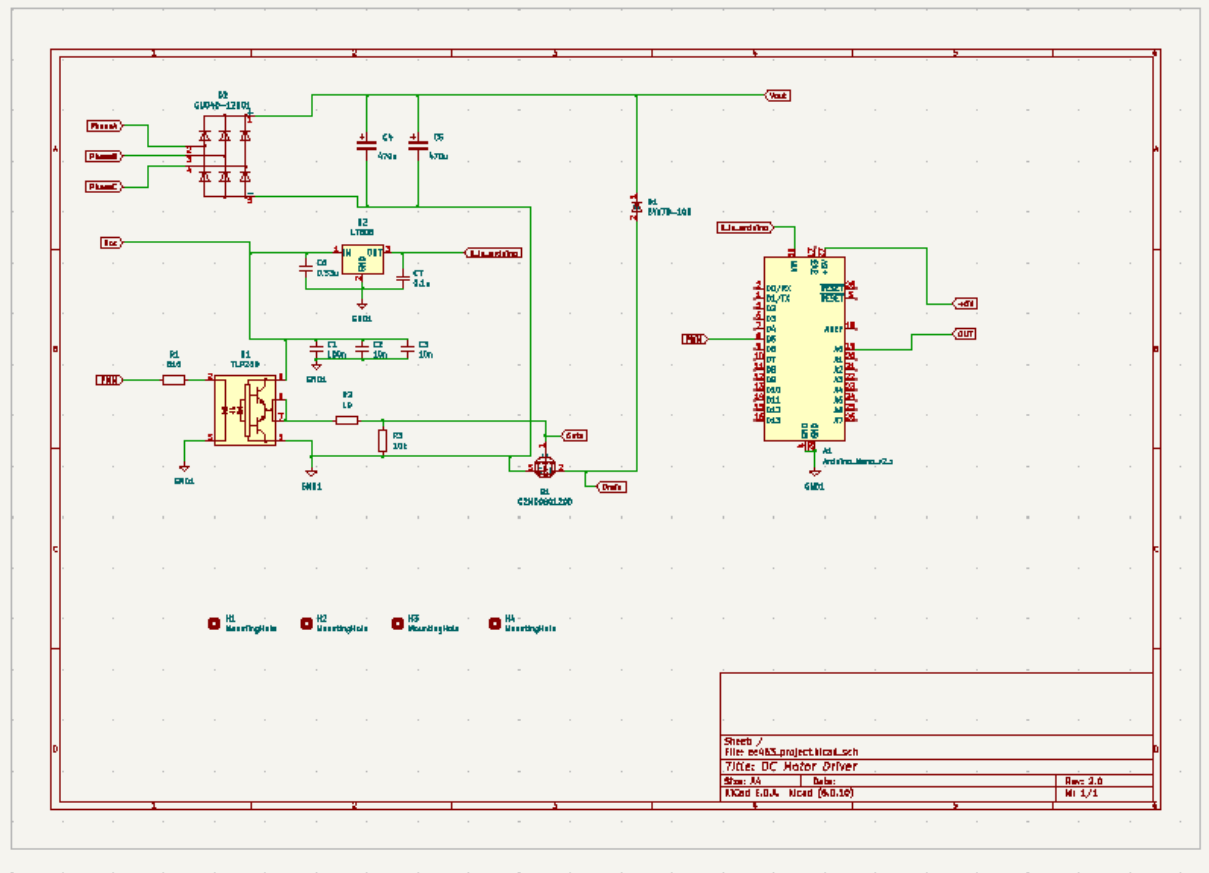

Figure 5.1- Schematics of Overall Topology

5.2. First Design and Problems

The first PCB design of the project can be seen in Figure 5.2. In the first design there is a large ground plane. 3 Phase Full Bridge was placed upper left of the PCB. The positive output of this placed a long copper plane left to right. There are two current loops. One of them is at upper right, when the MOSFET is OFF state. Another is parallel to positive output of the full bridge rectifier. DC-link capacitors were placed between this output and ground signals. At the bottom right, the gate driver and supply circuits were placed. Lastly, arduino was placed at the bottom left of the PCB.

Except for the PCB design, the current sensor to read and control output current sensor was placed at the bottom left of the design. The cable that current sensor reads the current over it was placed from upper right to bottom left. This causes noises at higher load levels since this cable passes over the arduino and small signal lines.

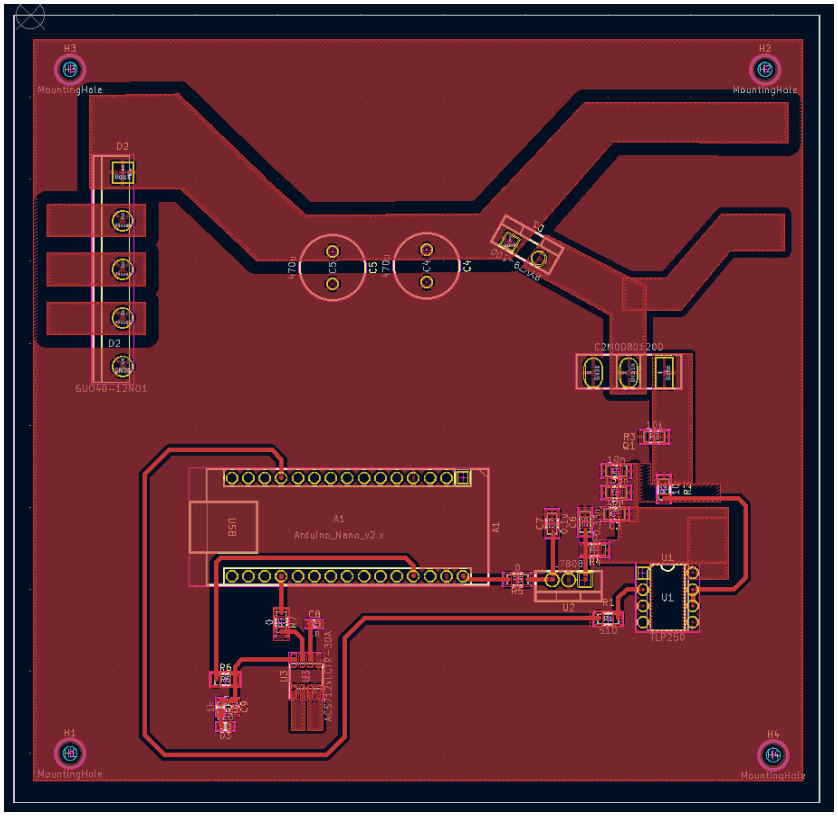

Figure 5.2 - PCB Design of the Project (v1.0)

5.3. Second Design and Improvements

Because of the problems in demonstration, the current sensor's position has been changed. It has been placed at the DC output of PCB. The copper plane of the positive output of DC bus voltage was drawn thicker than before.

To block the effect of power side over signal side, the ground plane was separated as signal ground and power ground and they were connected in one common point with a small copper. This can be seen in Figure 5.3. Finally, spaces between coppers have been increased to decrease risk of clearance. 3D view of the final circuit can be seen in Figure 5.4.

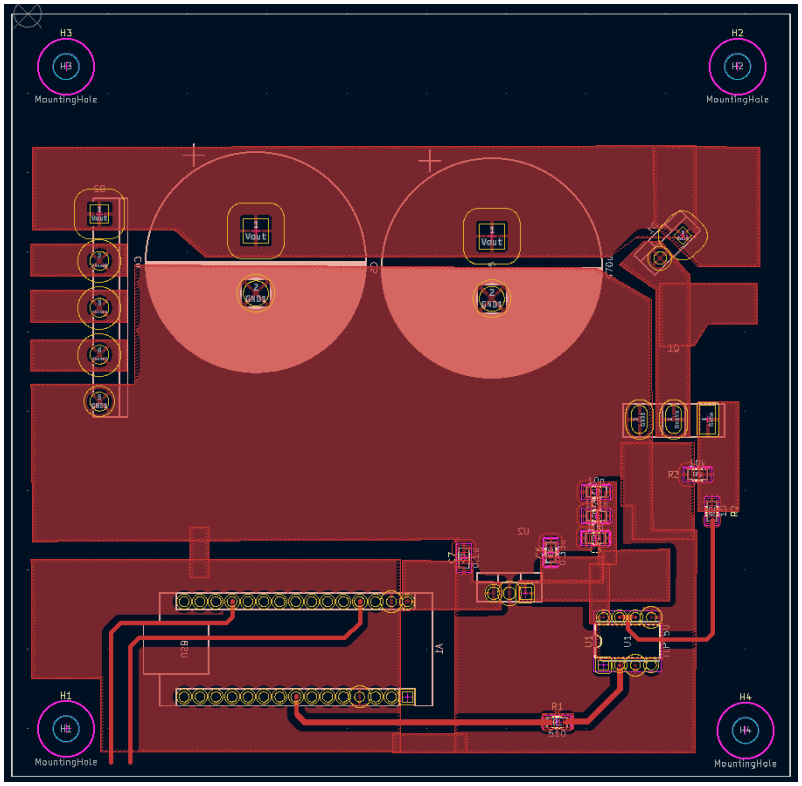

Figure 5.3 - Final PCB Design of the Project

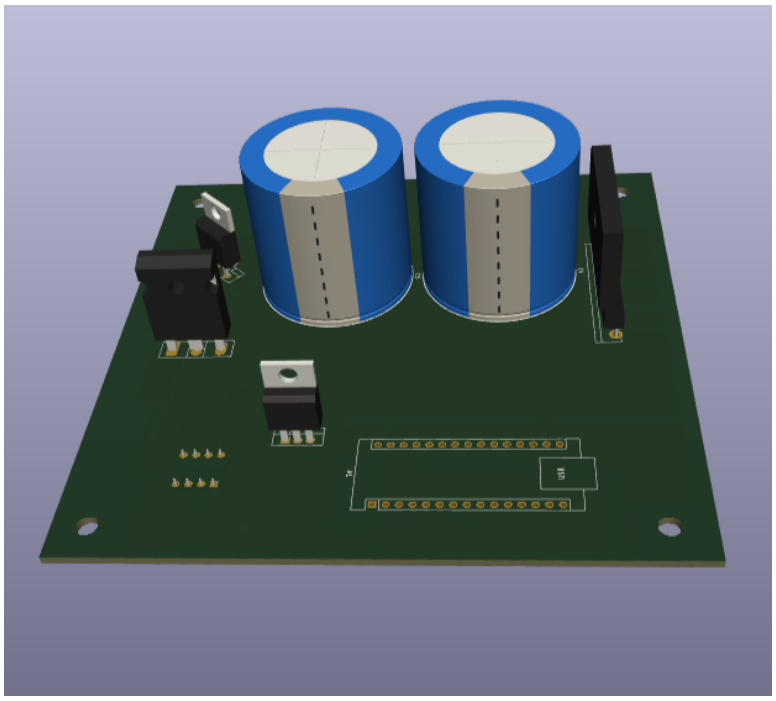

Figure 5.4 - 3D View of the Project

6. Test Results

The project was tested step by step to detect critical errors at the beginning. Firstly, gate driver topology has been tested. The test setup was set up as the recommended circuit of TLP250. This setup can be seen in Figure 6.1. The arduino is connected to red and black cable and PWM was generated by it.

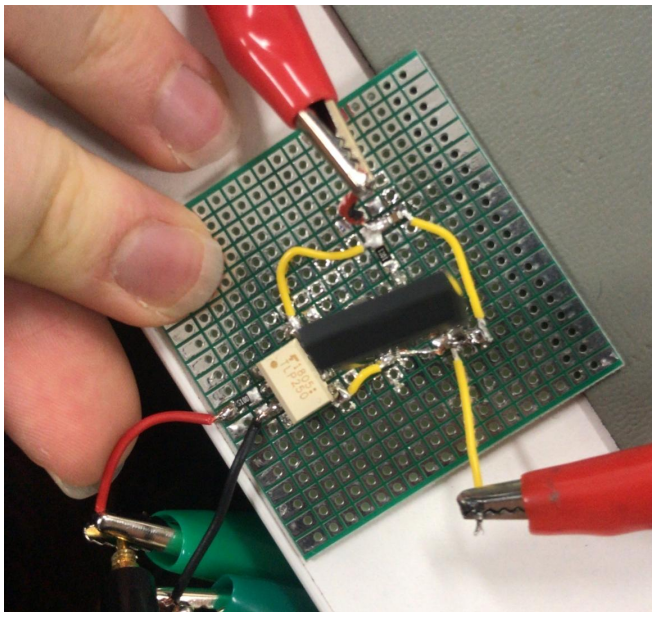

Figure 6.1. Gate Driver Circuit

After successful gate driver test, only buck converter topology of the project has been set. Arduino and the input of test setup were supplied with computer and external power supply respectively. An R load was connected to the output of the circuit to observe response of the circuit. PWM was generated as 20%, 50% and 70% duty cycle. The results are shown in Figure 6.2, 6.3 and 6.4 respectively.

Figure 6.2 - Buck Converter Output with 20% Duty Cycle

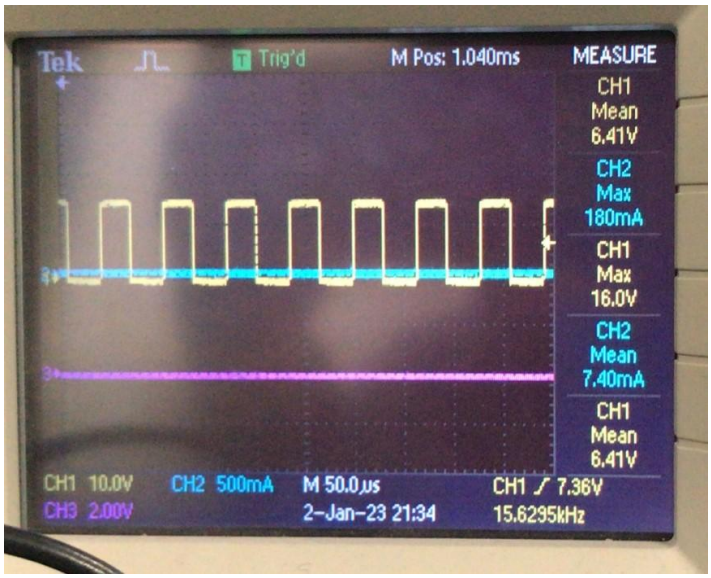

Figure 6.3 - Buck Converter Output with 50% Duty Cycle

Figure 6.4 - Buck Converter Output with 70% Duty Cycle

After successful buck converter tests, PCB has been designed and prepared to test. Firstly, gate driver and buck converter tests were repeated with PCB. Then variac was connected to 3 phase input of the circuit. Light load tests were done. Test has begun with 50V DC bus voltage. After the observation of good results, DC bus voltage was increased to about 180V with light load. The output resistance of the circuit is about 180Ω and the mean output current was obtained as 800 mA. The test setup of this circuit can be seen in Figure 6.5.

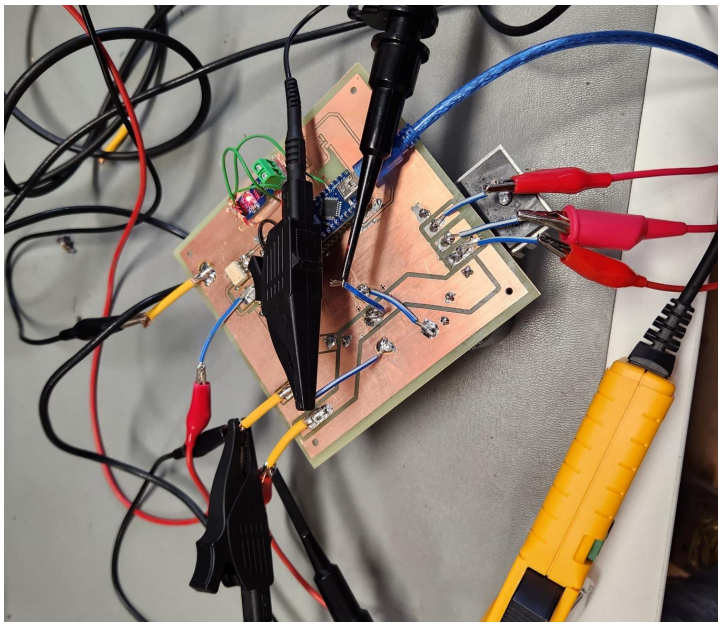

Figure 6.5. R Load Test Setup of Project

The DC bus voltage was measured from CH1. Output voltage and output current were observed in CH2 and CH3 respectively. Then, load was increased to 3A and DC bus voltage was decreased to 115V. The value of R load is about 40Ω. The results can be shown in Figure 6.6. Since the switching frequency of the setup was 7.8kHz, the system in CCM.

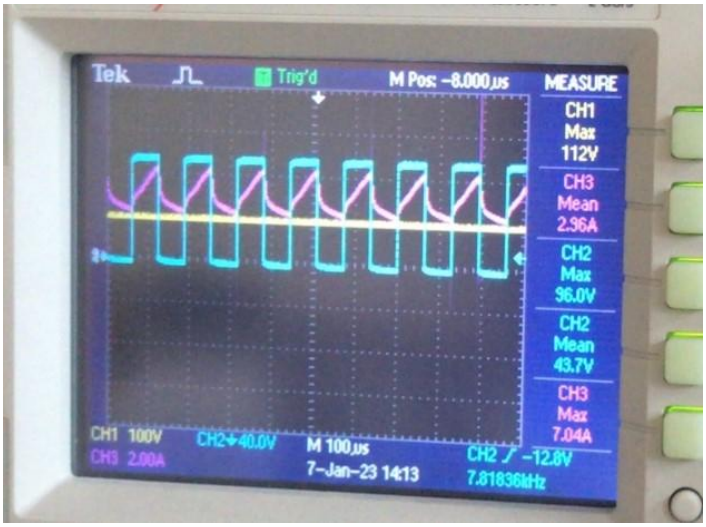

Figure 6.6 - The results of circuit with high load

Before the test with a DC motor, L load was added to the circuit to simulate the system like a DC Motor connected scenario. The result can be seen in Figure 6.7. The output current was observed as DC current because of high inductance at the output and around 3.02A.

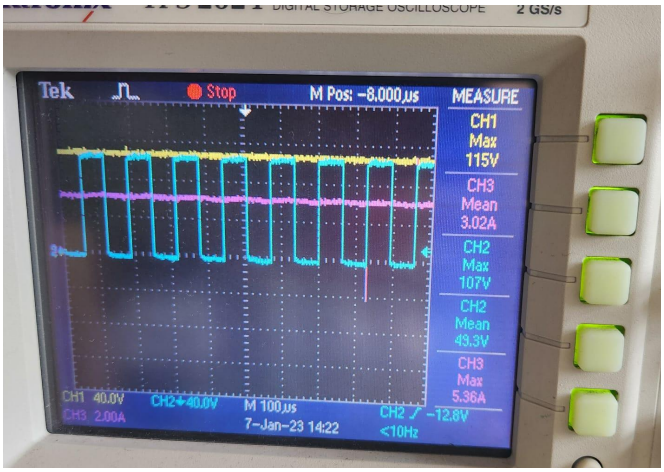

Figure 6.7 - The results of circuit with R-L Load

After these results, motor was driven by this circuit with no load and $50V_{ll}$ voltage. This test was also successful. The circuit was boxed and the current sensor was added to the system. The second motor test failed after the boxing process on the first demonstration day. The output current and voltage results can be seen in Figure 6.8.

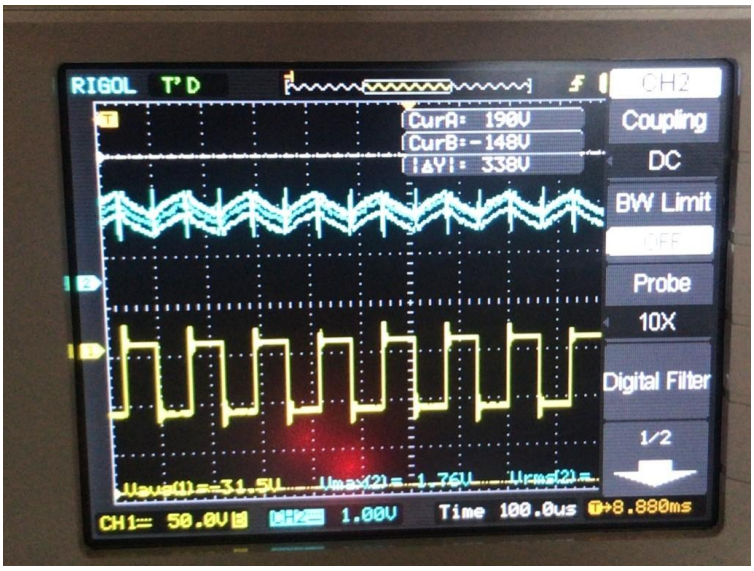

Figure 6.8 - Output Voltage (CH1) and Current (CH2) of System at DC Motor Test

To understand and solve the problem, the first overall test was repeated with light load. The result of the test can be shown in Figure 6.9.

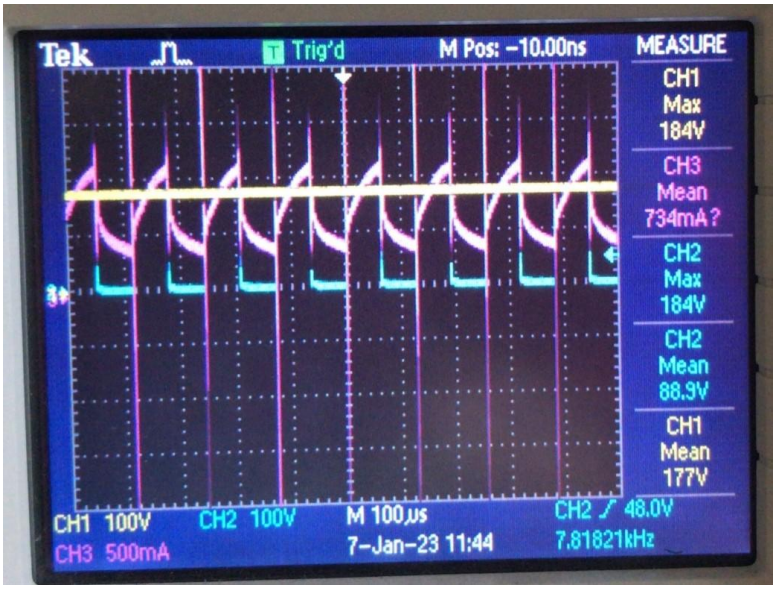

Figure 6.9 - Overall Ciruit Test with 180Ω R Load

There were high level spikes on the output current when the system was tested again. These spikes can be seen in Figure 6.9. The PCB and box placement were updated. The cables passing across the circuit were shorted and the current sensor was placed next to the output of the circuit. The fan was also added to the 2kW output test.The final box layout can be seen in Figure 6.10. The switching circuit was also decreased to 980Hz.

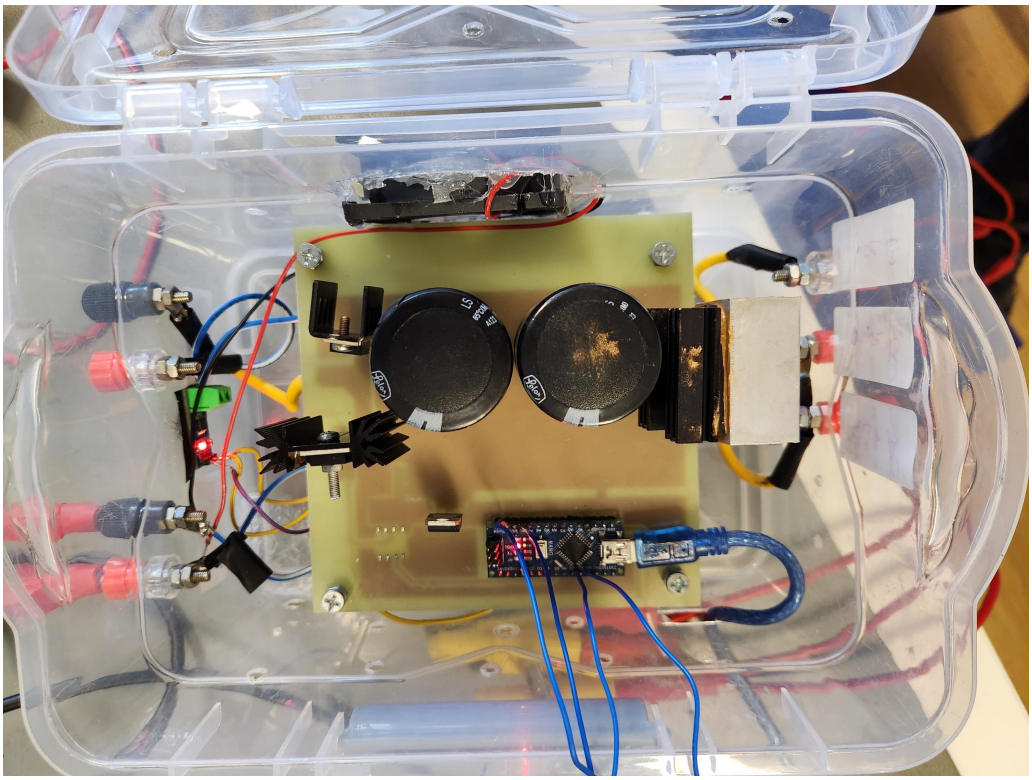

Figure 6.10- Final Box Layout of the Project

After these improvements, DC Motor was tested again. The system drove the motor properly. The no load output current and voltage results can be seen in Figure 6.11. Since the switching frequency of the circuit is 980 Hz, the system operates at DCM. The mean of the current was 2.01A and DC bus voltage was 180V.

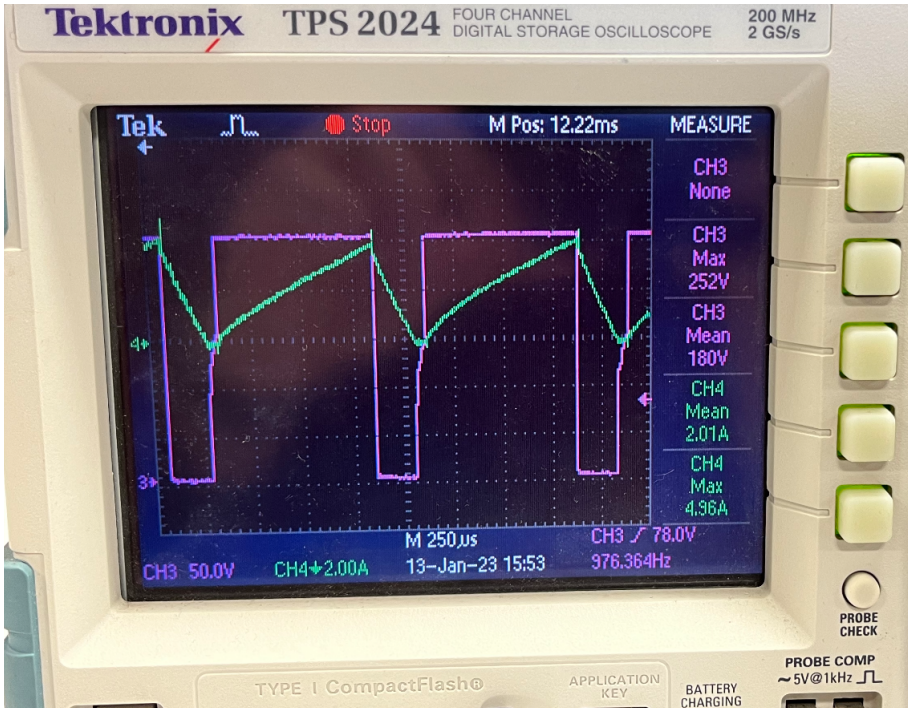

Figure 6.11 - Current and Voltage Characteristic of DC Motor with No Load

Then, the load of the motor was increased step by step up to 2kW. After the system worked at 2kW for a short time, the load was decreased to around 1.5kW and waited until the water was boiling.

The current and voltage of the motor can be shown in Figure 6.12. The output current was around 7.87A and output voltage was 167V. The input and output power results can be seen in Figure 6.12. Input power was 1.40kW and output power was 1.34kW. From these results, efficiency of the system can be calculated as 95.6%.

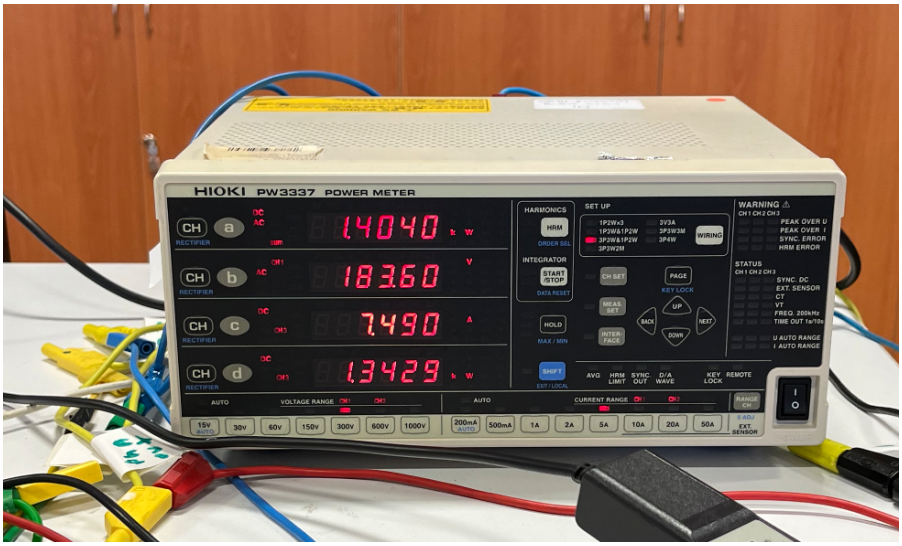

Figure 6.12 - Input and Output Power of DC Motor Driver with Load

The temperature levels from the thermal camera the system keeps working at around 1.5kW can be seen in Figure 6.13. Since the duty cycle was 75%, the MOSFET was conducted more than diode. Therefore the highest temperature level is at MOSFET with 72°C. Second one was the heatsink of the 3 phase rectifier. This is because six diodes are placed in a strict body. So, dissipation of heat is a little bit hard with respect to one diode with its own case.

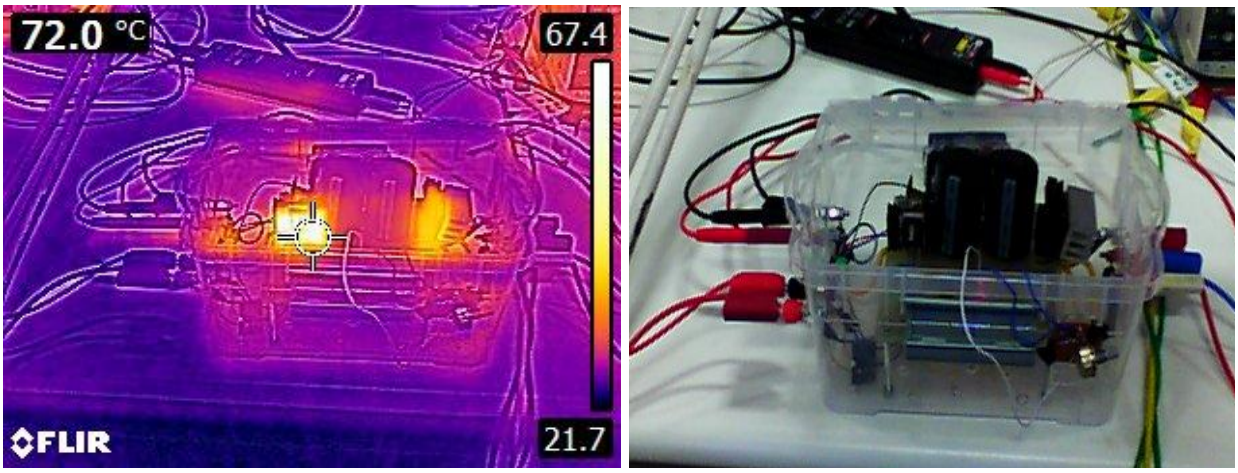

Figure 6.13 - Temperature Levels of the Project

7. Closed Loop Current Control

In this project we tried to get the closed loop current control bonus by using Arduino Nano and ACS712 current measurement module which were discussed above. We could not demonstrate at the EE463 demo but we could demonstrate at the EE407 demo. The main reason we couldn't show at the EE463 demos was that we didn't think we should get RMS. We used two different types of control: a PI control and increment-decrement control.

7.1 Increment-Decrement Control

Increment-Decrement control is the simplest form of all control methods. The basic idea is the duty cycle is increased when the error, which is $I_{ref} - I_{out}$, is positive, and it is decreased when the error is negative. Note that the control does not depend on the amount of error, but it only depends on the sign of the error. Hence, its implementation is easier than other control methods such as PID, fuzzy, etc.

There was also a “dead band” or “idle region” where the controller didn't try to change the output. We set our reference values and respected deadband gap values from the Arduino code as whatever we want.

Advantages:

- Simple to apply
- Inexpensive
- Only two or three (dead-band) states

Disadvantages:

- Steady-state error
- The controlled parameter will continuously switch around the setpoint and if the hysteresis is not correctly set, the deviation from the setpoint could be quite significant
- It may cause the actuator to switch on and off very fast and break it

7.1 PI Control

PID control is the most used control method among. Its logic depends on the error calculation and manipulation of the output depending on the error. The letters P, I, and D represent the proportional, integral, and derivative terms respectively. These terms are expressed in the following equation.

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{d}{dt} e(t) \right)$$

The proportional term multiplies the error with the proportional constant Kp. The integral term collects the cumulative value of the error to compensate for the steady-state error of the P term and multiplies it with the integral constant Kp/Ti. The derivative term calculates the rate of error change as if it is trying to predict the effect of the error and multiplies it with the derivative constant Kp*Td. The most used control methods using these terms are P, PI, and PID control.

In this project, a PI controller is chosen to be implemented. The derivative term will not be used as it increases the noise elements caused by the current sensor and other disturbance factors. Also, among the motor (which is basically an RL load) control methods, PI control is the most preferred control method.

Advantages:

- No steady-state error
- Simple structure
- Decreases noise components

Disadvantages:

- Wind-up can be seen due to the integral term
- Narrower stable region
- Decreases the speed of the response
- Hard to implement when plant can not be identified precisely

In this project we implemented both increment-decrement control and PI control together but separately, one can choose the control method just by commenting or uncommenting the function at Arduino IDE. The first method was more error prone and stable and safe to run it with the DC motor, since any errors lead to burn the components. But it's not a very high quality and academic control method. PI control is a better control method overall but in our case it was very slow and unstable it shows some harmful spikes.

We applied these control methods and recorded the data of the reference current and output current, then by a tool which is called “ArduSpreadSheet " we created .asv files and plotted them by using Matlab. Plots can be seen below and codes can be seen at Appendix II.

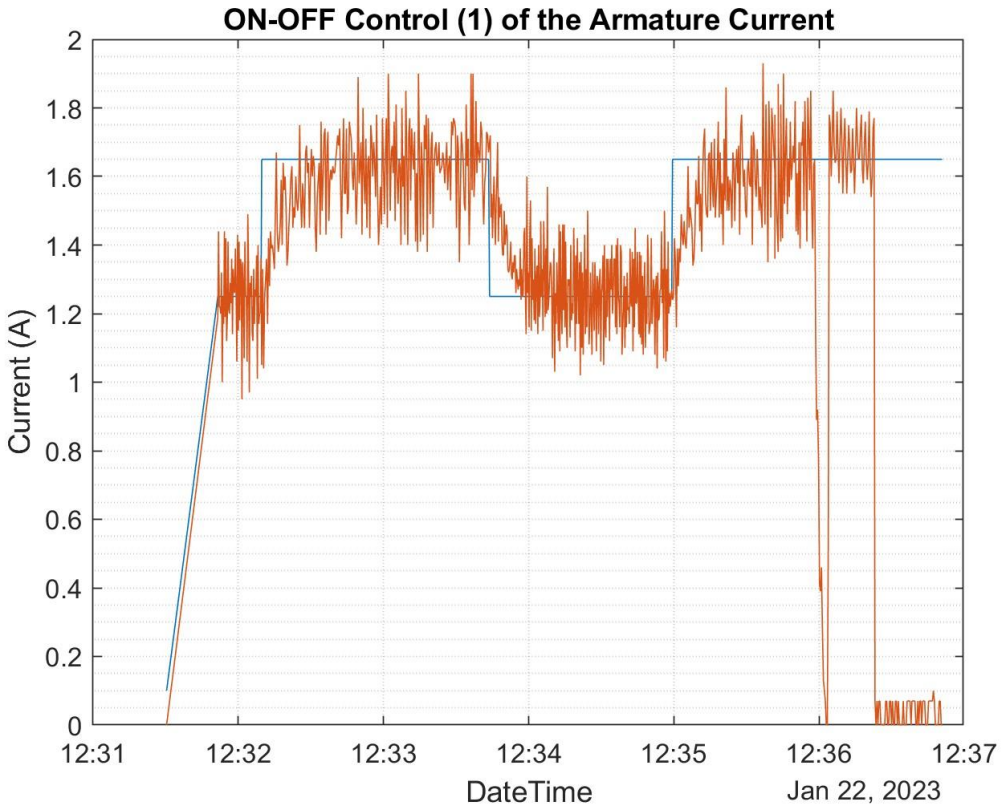

Figure 7.1 - Increment-Decrement Control Result 1 (Blue line is reference)

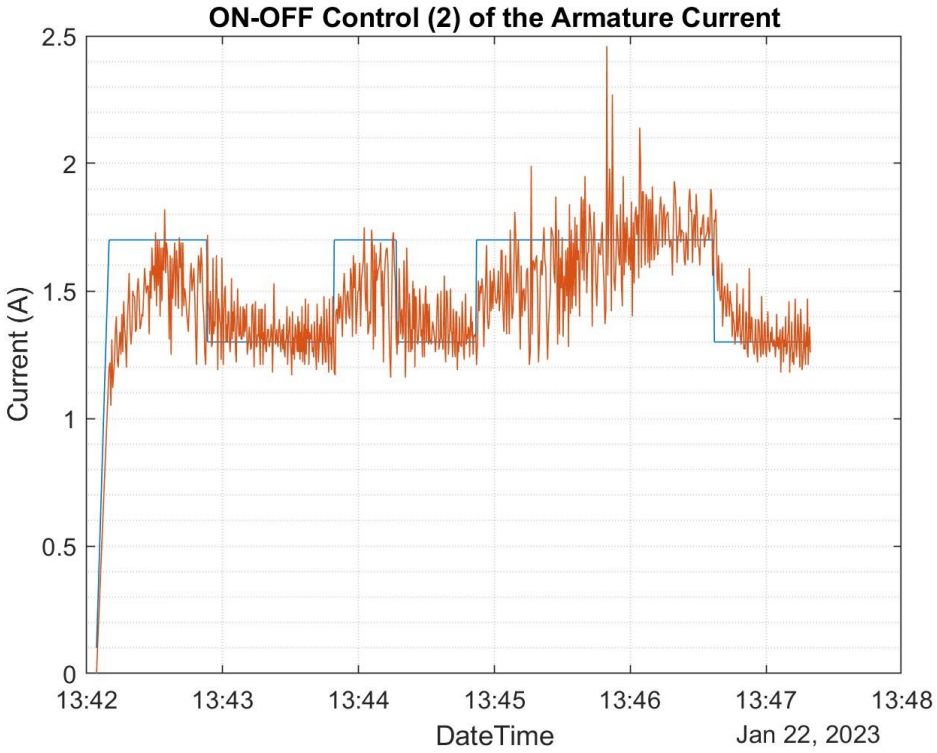

Figure 7.2 - Increment-Decrement Control Result 2 (Blue line is reference)

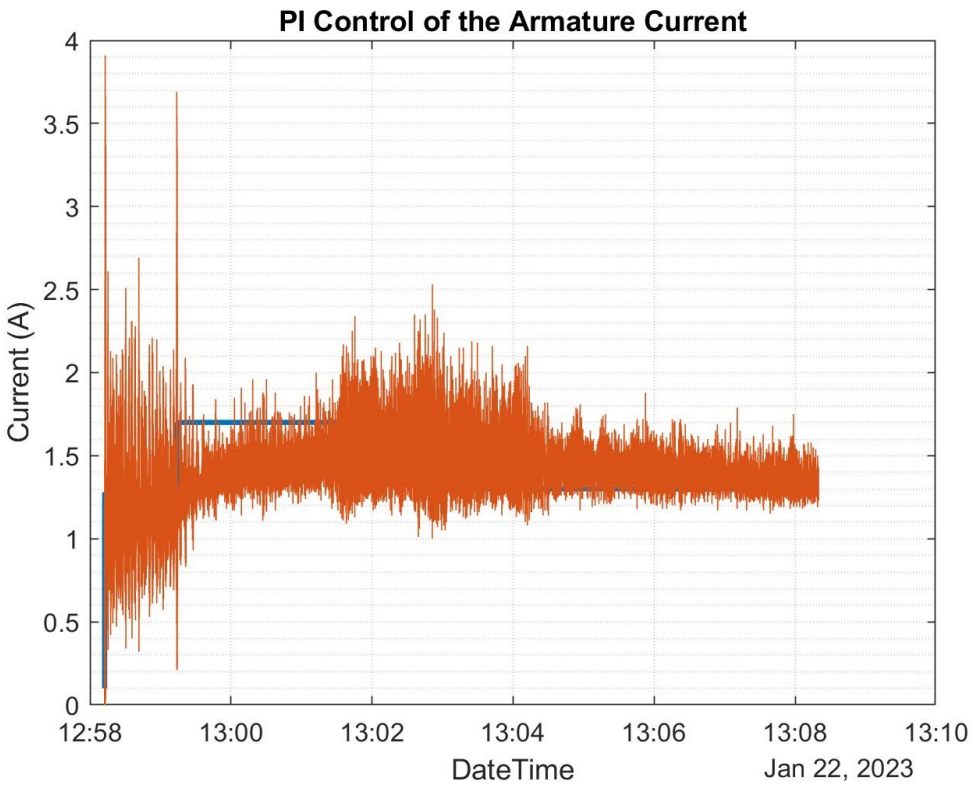

Figure 7.3 - PI Control Result with all the noises (Blue line is reference)

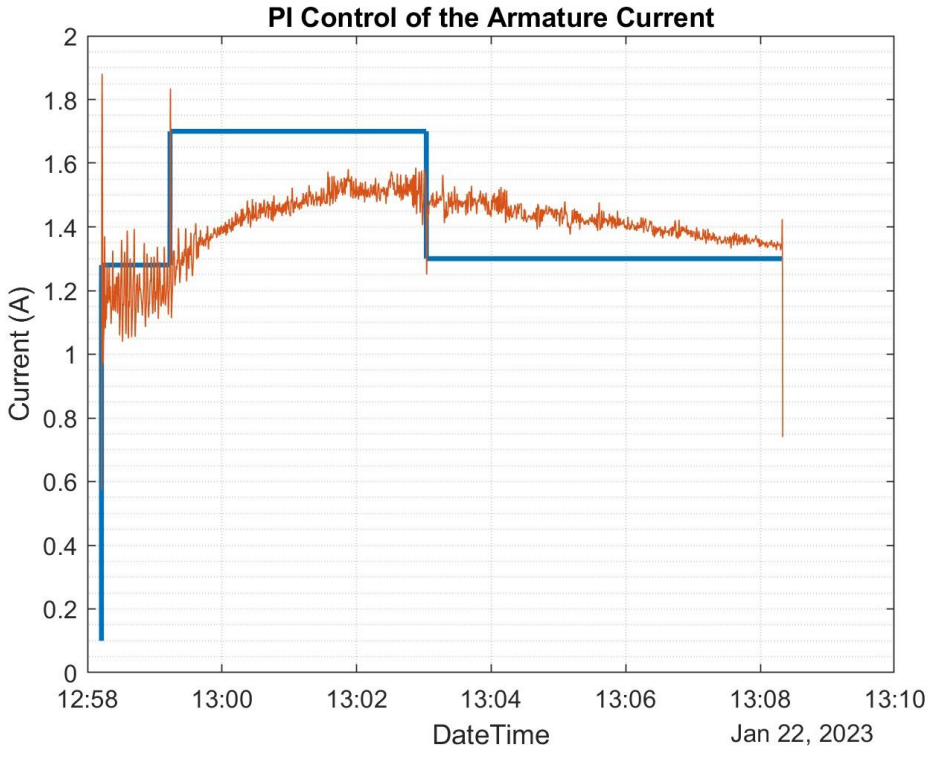

Figure 7.4 - PI Control Result when filter is applied (Blue line is reference)

8. Conclusion

This report summarizes the comprehensive EE463- Term Project. This project covers everything from the beginning to the end of the course and conducts toward repeating many topics and observing the theoretical knowledge we learned in the practical field.

AC to DC rectifying concept is applied through the three phase diode rectifier topology. While choosing this topology, size of the printed circuit board is considered and a rectifier module is preferred instead of constructing the topology with 6 different diodes. At the output of the rectifier, DC link capacitor used to smooth DC input of the Buck Converter

DC to DC converting concept is applied with Buck Converter topology which makes the output voltage lower than input voltage with the help of adjusted duty cycle. In the design of Buck Converter, the switch situation is decided according to input coming from Arduino Nano. Arduino Nano controlled duty cycle according to wanted output voltage and current. Moreover, gate driver circuit is used to amplify gate current of mosfet since Arduino Nano can give 100 mA at maximum which is inadequate. In addition, LDO is also used to convert DC supply input voltage to required Arduino Nano voltage.

In this design, the principle of going from simple to complex was applied. Firstly, opening the mosfet with the help of Arduino Nano that is connected to computer and gate driver is tried. After the success of this operation, DC Supply input is applied and LDO is used to supply required Arduino Nano voltage. Vcc of the gate driver is also taken from DC Supply. After the success of this operation also, PCB design was started. In the component selection part, current and voltage ratings and important times such as fall time, rise time, reverse recovery time are considered carefully.

Great lessons have been taken from a mistake that caused the circuit to fail. The importance of separating the signal path and power path to prevent interference with each other is understood since they are working with too different voltage levels as magnitude. In addition, importance of not touching to capacitor after the turn off power supplies since capacitor would be still charged.

In conclusion, this project was a great experience about practical applications of Power Electronics. The project group members learned a lot of things from their instructors and friends.

9. Appendix I

SiC Diode https://cdn.ozdisan.com/ETicaret_Dosya/592402_208243.pdf

Optocoupler https://pdf.direnc.net/upload/tlp250-datasheet.pdf

L7808 https://pdf.direnc.net/upload/l78-974043.pdf

Si MOSFET https://cdn.ozdisan.com/ETicaret_Dosya/626007_145179.PDF

3-Phase Rectifier https://cdn.ozdisan.com/ETicaret_Dosya/582454_9707777.pdf

keysan.me

10. Appendix II

Arduino Code

```
#define PWM_PIN 5
#define BUTTON_1 8
#define BUTTON_2 9
#define SENSOR_PIN A0
#define CURRENT_LIMIT 8

#include <ACS712.h>

ACS712 sensor(ACS712_30A, SENSOR_PIN); // Sensor declaration

int a = 0; // A dummy variable declared for the soft start to happen
float b = 0.0;
int curDuty = 0;
 // A variable that holds the current duty cycle (hold a value in the range of 0-255 for analogWrite)
double AmpThrough = 0.0; // The current passing through the motor
double RefAmp = 0.1; // Reference for control

// variables for PI control
unsigned long StartTime;
unsigned long EndTime;
float dt = 0.01;
double error = 0;
double integral = 0;
int output = 0;
#define Kp 22  // Proportional gain
#define Ki 2  // Integral gain


void soft_start(){ // It increases the duty cycle each seconds by a small amount up to 60
  int ss_duty = 0; // A variable holding the soft start duty cycle (0-255)
  while(ss_duty < 60){ // The code below runs while the duty cycle is under 60
    delay(1000);
    ss_duty = ss_duty + 3; // Slightly increase the duty cycle at each loop
    ss_duty = constrain(ss_duty, 0, 61);  // Limit the duty cycle to be under %25
    analogWrite(PWM_PIN, ss_duty);
    curDuty = ss_duty; // The current duty cycle variable is updated
    a = curDuty;
  }
}

void on_off_control(){
  AmpThrough = sensor.getCurrentAC(980) - 0.1; // Read the rms value of the current
```

```
  if (RefAmp == 1.70 && AmpThrough < CURRENT_LIMIT)  //
  {
    if (AmpThrough > (RefAmp + 0.08)){
      curDuty = constrain(curDuty - 2, 1, 191);  // Max %75 duty
    }
    else if (AmpThrough < (RefAmp - 0.08)){
      curDuty = constrain(curDuty + 2, 1, 191);  // Max %75 duty
    }
    analogWrite(PWM_PIN, curDuty);
    print_variables();
    delay(400);
  }
  else if (RefAmp == 1.30 && AmpThrough < CURRENT_LIMIT)  // 1 A
  {
    if (AmpThrough > (RefAmp + 0.05)){
      curDuty = constrain(curDuty - 2, 1, 191);  // Max %75 duty
    }
    else if (AmpThrough < (RefAmp - 0.05)){
      curDuty = constrain(curDuty + 2, 1, 191);  // Max %75 duty
    }
    analogWrite(PWM_PIN, curDuty);
    print_variables();
    delay(300);
  }
  else if (AmpThrough > CURRENT_LIMIT) {
    analogWrite(PWM_PIN, 0);  // Resets the duty cycle to zero
    delay(50000); // Wait for 10 seconds
    soft_start(); // Soft start again
  }
}

void pi_control(){
  StartTime = micros();
  AmpThrough = sensor.getCurrentAC(980) - 0.1; // Read the rms value of the current
  if (AmpThrough > CURRENT_LIMIT) {
    analogWrite(PWM_PIN, 0);  // Resets the duty cycle to zero
    delay(50000); // Wait for 10 seconds
    soft_start(); // Soft start again
  }
  error = RefAmp - AmpThrough; // Error is the difference between the reference and output current
  integral = integral + error*dt; // The definition of the integral term
  b = 60+Kp*error + Ki*integral; // PI control
  b = constrain(b, 1, 191);
  curDuty = int(round(b));
  delay(50);
  analogWrite(PWM_PIN, curDuty);
  print_variables();
  EndTime = micros();
  dt = abs(EndTime - StartTime)/1000000.00; // Find the step time
```

```
}

void print_variables(){
  Serial.print(RefAmp); // Print RefAmp
  Serial.print('\t');
  Serial.print(AmpThrough); // Print AmpThrough
  Serial.print('\t');
  Serial.print(curDuty / 255.0); // Print curDuty
  Serial.print('\n');
}

void setup() {
  pinMode(PWM_PIN, OUTPUT); //PWM pin
  pinMode(BUTTON_1, INPUT); //BUTON 1
  pinMode(BUTTON_2, INPUT); //BUTON 2
  Serial.begin(9600);
  analogWrite(PWM_PIN, 0);
  // If you are not sure that the current through the sensor will not leak during calibration - comment out
this method
  sensor.calibrate();
  delay(100);
}

void loop() {
  if (a<60){
    delay(20000);
    soft_start();
    print_variables();
    AmpThrough = sensor.getCurrentAC(980) - 0.1; // Read the rms value of the current
    RefAmp = AmpThrough;
  }
  else{
    if (digitalRead(BUTTON_1) == 1){  // Button 1 is pressed
      RefAmp = 1.70;
    }
    if (digitalRead(BUTTON_2) == 1){  // Button 2 is pressed
      RefAmp = 1.30;
    }
    /* Comment out the control method that you don't want to use */

    on_off_control();
    //pi_control();
  }
}
```

MATLAB Code

```
%%on off 1
figure()
plot(onoff1.VarName1, onoff1.VarName2)
hold on
plot(onoff1.VarName1, onoff1.VarName3)
grid minor
xlabel('DateTime')
ylabel('Current (A)')
title('ON-OFF Control (1) of the Armature Current')
%%on off 2
figure()
plot(onoff2.VarName1, onoff2.VarName2)
hold on
plot(onoff2.VarName1, onoff2.VarName3)
grid minor
xlabel('DateTime')
ylabel('Current (A)')
title('ON-OFF Control (2) of the Armature Current')
%pi 1
a = 1;
b = 4532;
figure()
plot(pi1.VarName1(a:b), pi1.VarName2(a:b), LineWidth=2)
hold on
plot(pi1.VarName1(a:b), pi1.VarName3(a:b))
grid minor
xlabel('DateTime')
ylabel('Current (A)')
title('PI Control (1) of the Armature Current')
%pi 2              lowpass(pi2.VarName3(a:end), 1/5000, 1/100)
a = 1;
b = 10000;
figure()
plot(pi2.VarName1(a:end), pi2.VarName2(a:end), LineWidth=2)
hold on
plot(pi2.VarName1(a:end), pi2.VarName3(a:end))
grid minor
xlabel('DateTime')
ylabel('Current (A)')
title('PI Control of the Armature Current')
```