ELECTRICAL AND ELECTRONICS ENGINEERING DEPARTMENT

EE463 Static Power Conversion

2024-2025 Fall Project

Current Affairs

Yağız Utku Tiryaki 2517050

Veysel Uysan 2376002

İsmayil Nesibov 2491314

1.Introduction

This report details the design, simulation, and implementation of a controlled rectifier used to drive a DC motor. The aim of this project is to design a DC motor drive with an adjustable DC output of up to 180 V, utilizing a three-phase or single-phase AC grid as input. The project focuses on topology selection, simulation, component selection, controller implementation, and PCB design, along with detailed considerations.

2. DC Motor Specifications

Specs of the all motor windings are measured as follows:

Armature Winding: 0.8 Ω, 12.5 mH

Shunt Winding: 210 Ω, 23 H

Interpoles Winding: 0.27 Ω, 12 mH

Figure 1: DC Motor

3. Design Decisions

3.1 Topology Selection

There are lots of topologies that were provided to us due to several reasons below Three Phase Diode Rectifier + Buck Converter Topology was selected.

After evaluating several topologies, including:

- **Single-Phase Diode Rectifier + Buck Converter**
- **Three-Phase Diode Rectifier + Buck Converter**
- **Other Advanced Converter Topologies**

The chosen topology is a **Three-Phase Diode Rectifier + Buck Converter** for the following reasons:

- **Advantages:**
    - High efficiency with minimal harmonic distortion compared to single-phase designs.
    - Smoother output voltage due to the lower ripple in a three-phase rectifier.
    - Compatibility with the project's adjustable output voltage requirements.
- **Disadvantages:**
    - Increased complexity in gate driver and control circuit design.
    - Higher component count and potential cost.

In line with the requirements of the project, Three Phase Diode Rectifier + Buck Converter topology at **Figure 1 and Figure 2** was better option to choose. Factors that play a role in choosing this topology are Output Stability, Reduced ripple ensures stable motor operation and reduces electromagnetic noise and High Efficiency(The topology achieves high power factor and low energy losses, which are critical for industrial applications).Following this, MOSFET was preferred for the switch in the buck converter part of the system, and IGBT was considered as a side option. MOSFET offers higher switching speeds and lower losses in low voltage applications compared to IGBT, while IGBT is more efficient at higher voltages and lower switching frequencies. But IGBT has the drawback of complex control. The use of MOSFET in the buck converter was preferred for the following reasons high switching speed, low losses(With low RDS(on) and gate charge (Qg) values, both switching and conduction losses are low.

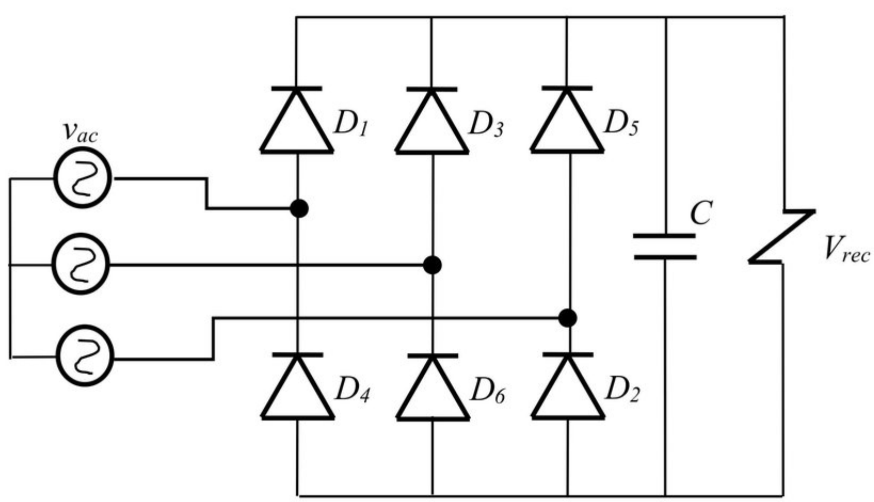

Figure 2: Three Phase Diode Rectifier

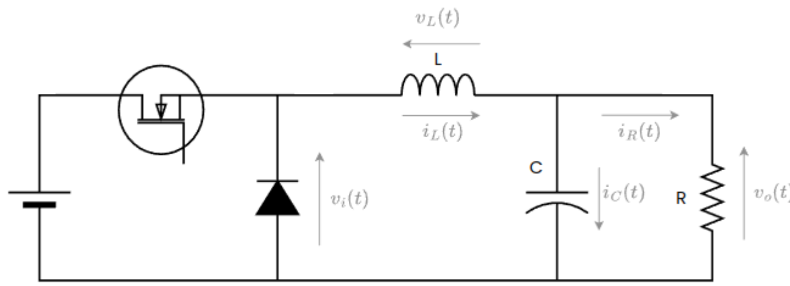

Figure 3: Buck Converter Topology

4. Simulation and Design Procedure

Full Bridge Rectifier

We have implemented our overall design at Simulink software to test our results. Open-loop simulations verified the basic operation of the rectifier and buck converter. These simulations were expanded to include a closed-loop PI controller, ensuring precise speed and voltage control. Non-idealities such as diode resistances, capacitor ESR, and parasitic inductances were incorporated to validate real-world robustness to make design more valid.

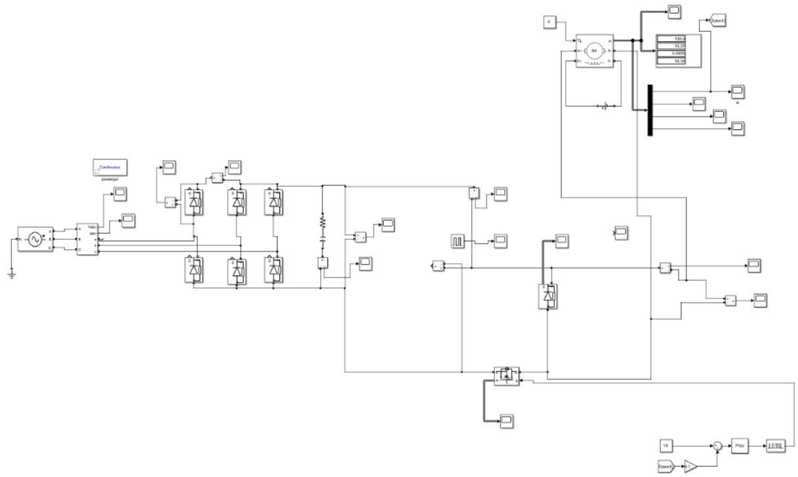

Figure 4: Overall Design at Simulink

At first part of our design Three Phase Diode Rectifier with DC Link Capacitor is connected to the AC Grid. Main purpose of this structure is to converting AC to DC for buck converter. AC Grid is represented as Variac and it is adjusted to the Vrms Ph-Ph of 150V this means that 202.57V at output of three phase bridge rectifier.

$$V_{dc} = \frac{3 \times \sqrt{2}}{\pi} \times V_{line-to-line}$$

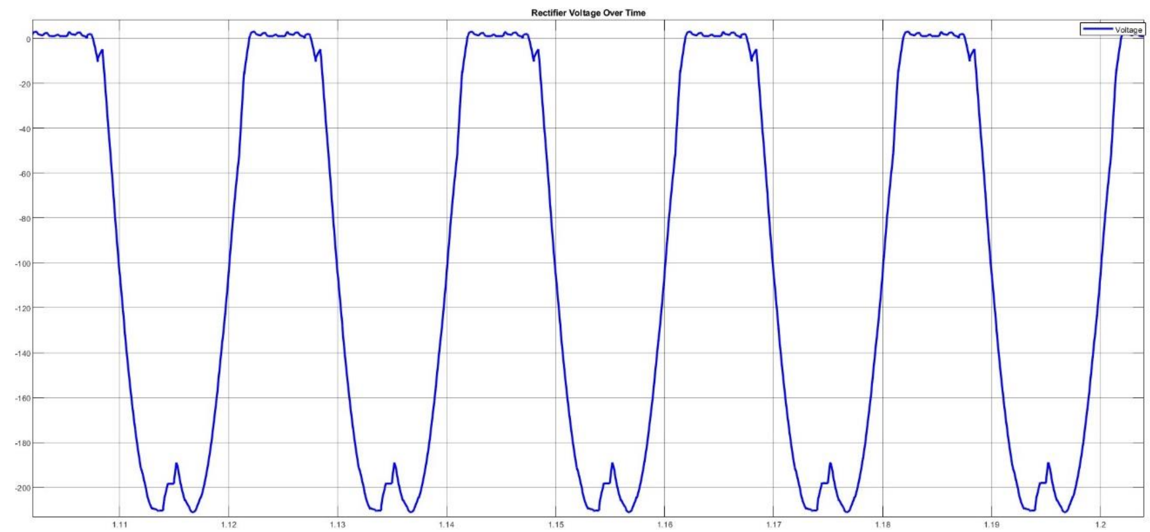

Figure 5: Three Phase Full- Bridge Rectifier Diode Voltage

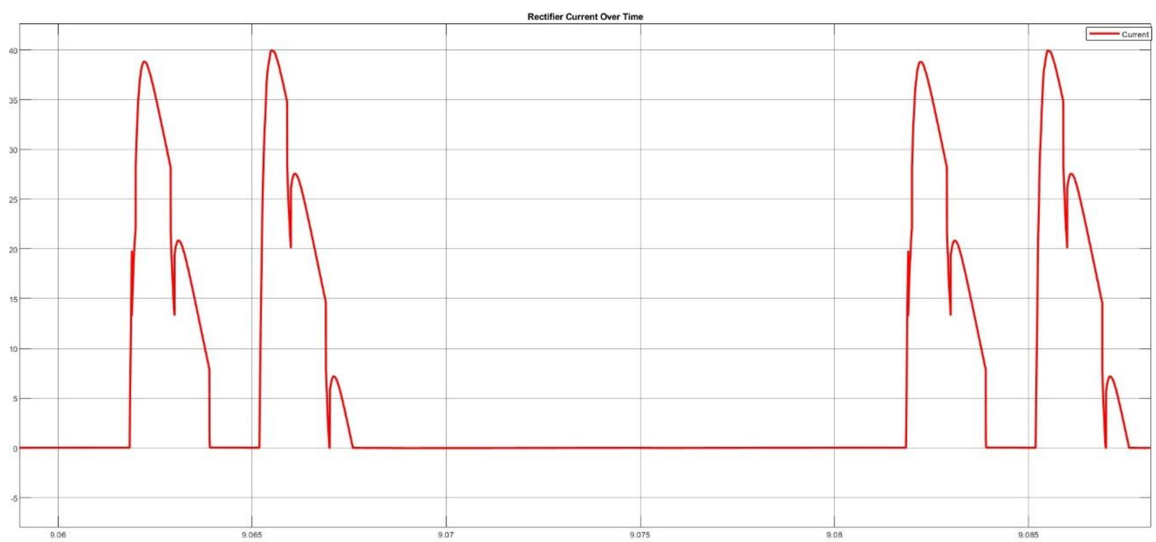

Figure 6: Three Phase Full-Bridge Rectifier Diode Current

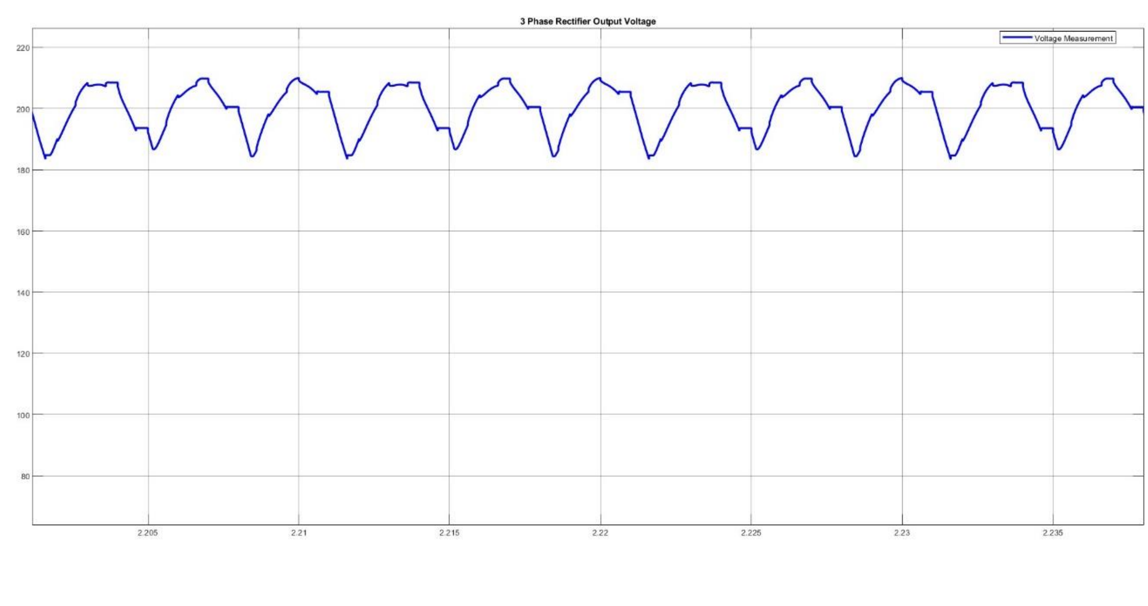

Figure 7: Three Phase Full-Bridge Rectifier Output Voltage

It can be inferred from **Figure 4, 5, and 6** that Three Phase Full Bridge Rectifier worked perfectly by rectifying AC signal to 201.9 Vrms. Also, we can observe that sharp fluctuations in our Full Bridge Rectifier Circuit's output voltage. The rectifier is connected to a buck converter and a motor load, the dynamic behavior of the load (e.g., changes in motor current due to speed variations) can cause fluctuations in the output voltage.

Results and Analysis of Simulations Full Bridge Rectifier:

Output Voltage (Figure 6): The rectifier effectively converted the AC input to a stable DC output. The output voltage simulated and displayed helped to choose DC link capacitor for reducing ripples to negligible levels .

Diode Currents (Figure 5): The current through each diode was ensured the current values were within the rated limits, allowing for the safe and effective operation of the diodes.

Voltage Behavior (Figure 4): The voltage across the diodes shows the ideal switching behavior, which means that the topology of a rectifier is functioning well. The DC link capacitor absorbs any fluctuations of the load dynamics, especially load and very short duration fluctuations created when an electric motor comes on load.

The output of the rectifier, as verified in the simulations, will deliver a stable DC voltage to the output of the buck converter.

<u>Buck Converter</u>

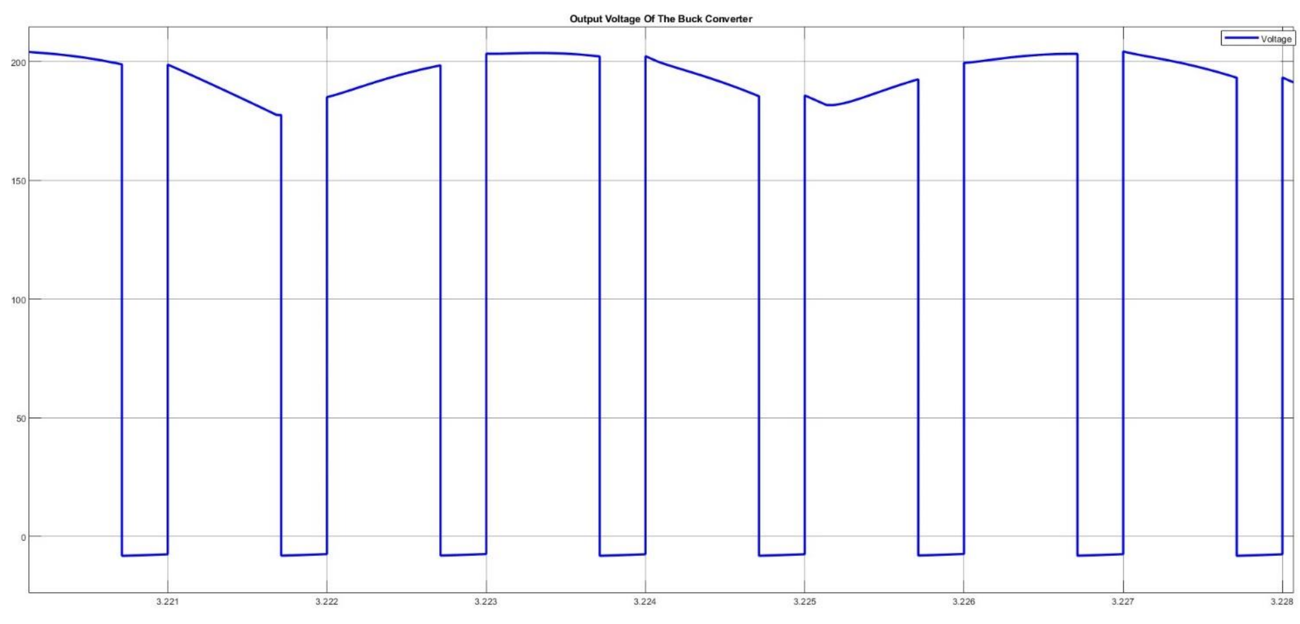

Figure 8: Buck Converter Output Voltage

Overall simulation was tested in the maximum case of driver 147V of output voltage and duty cycle of the 0.85.

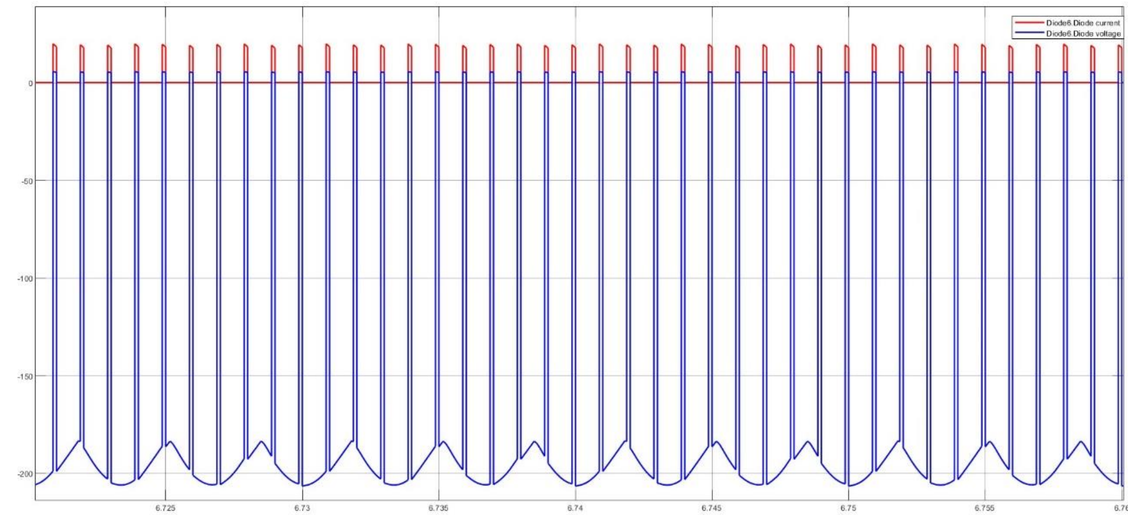

Figure 9: Freewheeling Diode Current and Voltage

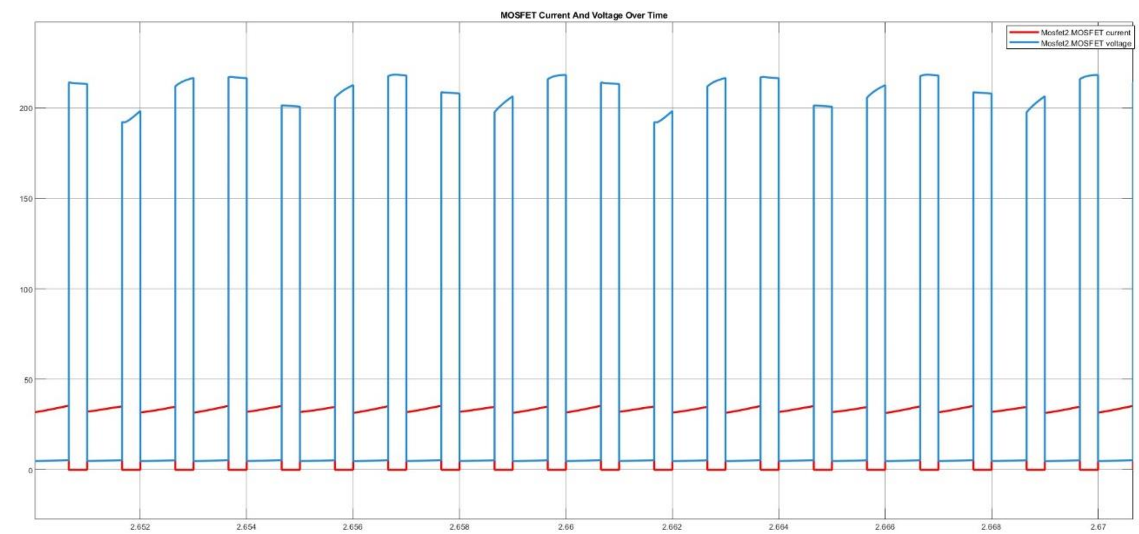

Figure 10: Mosfet Current and Voltage

The buck converter functions by adjusting the rectified DC voltage downward as needed to satisfy the motor's operating requirements. The main switching component MOSFET, was chosen for its low on-resistance, fast switching speed, and effectiveness under the specified voltage and current parameters. This guarantees minimal power losses while in operation. Diode was utilized for freewheeling current, efficiently managing the rapid transitions and minimizing switching losses because of its low forward voltage drop and short reverse recovery time.

Furthermore, the output capacitor, which is a low-ESR electrolytic type, was selected to stabilize the output voltage and further minimize voltage ripple, guaranteeing smooth motor performance.

Simulations of the buck converter confirmed its capability to precisely regulate output voltage, even with the dynamic load conditions presented by the motor. The closed-loop PI controller was essential for ensuring stable output voltage by modifying the duty cycle based on feedback signals.

PI Controller

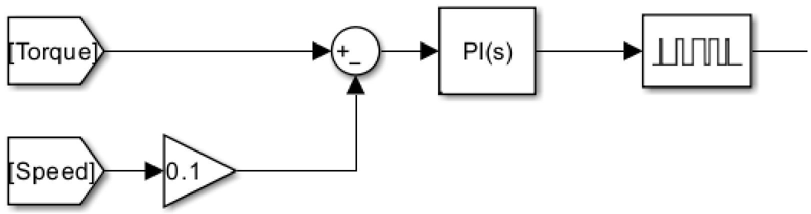

Figure 11: PI Controller On Simulink

In this design, the closed-loop speed control relies on a proportional-integral (PI) controller that adjusts the converter's duty cycle according to the motor shaft speed measurement. A sensor such as an optical encoder or tachometer is mounted on the shaft and outputs a raw signal proportional to the motor's rotational velocity. This signal is fed into a scaling amplifier that transforms it into a range consistent with the PI controller's input. The scaled feedback then undergoes subtraction from the setpoint speed, generating an error that indicates how far the actual speed deviates from the target. This error feeds into the PI block: the proportional term reacts instantly to present deviations, while the integral term accumulates the error over time, eliminating any residual offset. The PI controller's output drives a PWM generator, which produces gate signals for the buck converter's power switch. By increasing or decreasing the duty cycle, the converter regulates the armature voltage, thus controlling motor speed. Proper tuning of the PI gains is crucial to avoid oscillations and overshoot, while ensuring a fast response to disturbances such as varying load torque or supply fluctuations. In a real implementation, additional filtering and fine-tuning handle parasitic effects and measurement noise, and the entire control loop runs on a microcontroller or dedicated drive circuitry, providing robust, stable speed regulation over a wide range of operating conditions.

Gate Driver

The main purpose of the using gate driver in our design is that microcontroller gives us PWM signal which has low magnitude to drive mosfet switch. So that is why gate driver is used to amplify microcontrollers output to drive switch of buck converter. There are some constraints we had in this design such that main power source is used and only power source is left to drive both microcontroller and gate driver. Another critical point is that gate driver should be suitable both microcontroller and gate driver.

5. Component Selection

Full-Bridge Rectifier Diode

PRIMARY CHARACTERISTICS	
$I_{F(AV)}$	60 A
V_R	300 V
V_F at I_F	0.85 V
t_{rr} typ.	28 ns
T_J max.	175 °C
Package	TO-247AC 3L
Circuit configuration	Single

Figure 12: Rectifier Diode And Characteristics

For the diode-based rectifier stage, we selected the **VS-60APH03-N3** device, rated for a forward current $\left(I_F(AV)\right)$ of **60 A** and a reverse blocking voltage (V_R) of **300 V**. These ratings were chosen to match the expected load current and the DC bus voltage level following three-phase rectification, which can approach 220 V (plus a safety margin for transient overvoltages). By selecting a 300 V part, we avoid excessive overdesign while ensuring sufficient voltage headroom. The diode's 60 A average current rating similarly provides a comfortable margin above the approximate 50–60 A peak current observed during start-up, as seen in the simulation waveforms.

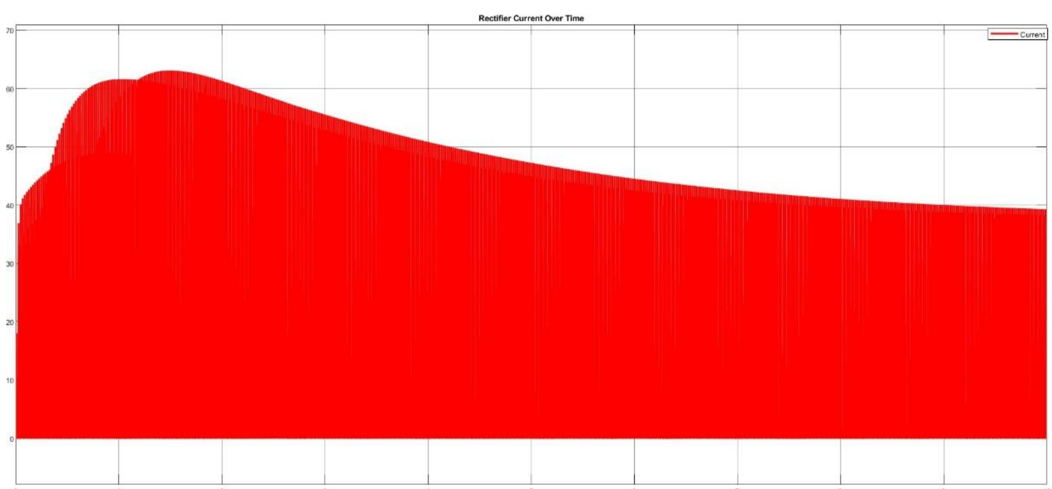

Figure 13: Rectifier Current Over Time

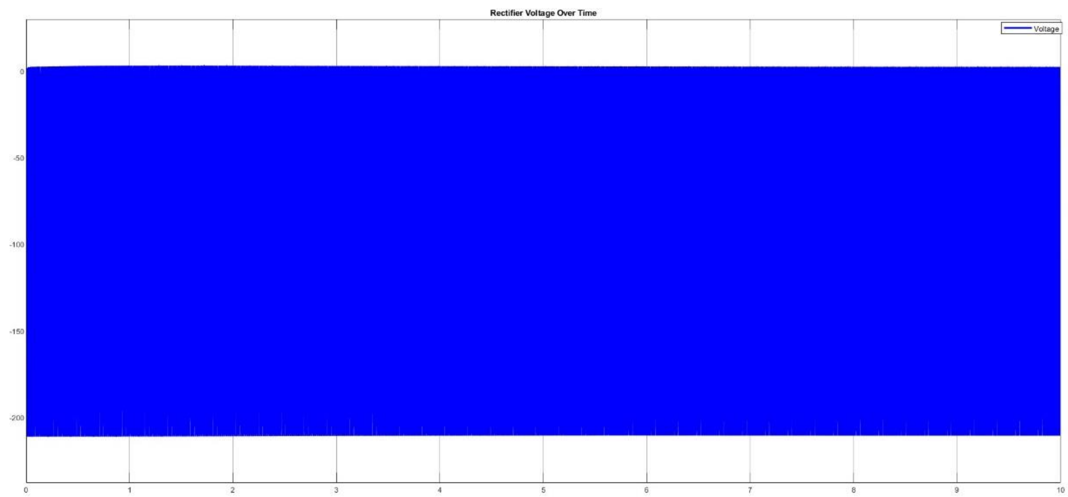

Figure 14: Rectifier Voltage Over Time

The red waveform in the attached figure shows the **rectifier current** over a 10 s window. Immediately after power-up, the current peaks above 60 A for short intervals, then gradually tapers off as the DC motor accelerates and the system stabilizes. The diode must handle these high initial surge currents without exceeding its maximum junction temperature of 175 °C. Its forward voltage drop of roughly **0.85 V** (at the rated current) contributes to the conduction losses, which remain manageable given the design's cooling provisions.

Meanwhile, the blue waveform illustrates the **rectifier output voltage**. Under normal operation, it remains near the 210–220 V level, never exceeding the diode's 300 V reverse rating. The diode's typical recovery time (t_{rr}) of **28 ns** is sufficiently fast for this application, minimizing switching losses and voltage spikes during commutation. Overall, the VS-60APH03-N3 provides an appropriate balance of current capacity, reverse voltage headroom, and switching speed for the targeted 60 A, 220 V rectifier design without incurring unnecessary overdesign.

Free Wheeling Diode

The freewheeling diode, critical for ensuring continuous current flow during the MOSFET's off-state, was chosen with the same careful consideration. As we use it in buck converter, we need to our diode compatible with operating frequency of 10kHz, so that low switching losses. From Figure 8, it can be inferred that our freewheeling diode has peak reverse voltage of 207.8V and maximum current that passes through diode is 20.5A and mean of 4.6A. So, **STTH6003** is selected as freewheeling diode.

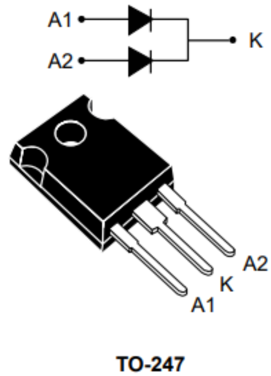

Figure 15: STTH6003

STTH6003 is suited for switch mode power supply and high frequency DC to DC converters. Packaged in TO-247, this device is intended for use in low voltage, high frequency inverters, free wheeling operation, welding equipment and telecom power supplies.

Symbol	Value
$I_{F(AV)}$	2 x 30 A
V_{RRM}	300 V
V_F (max.)	1 V
t_{rr} (max.)	55 ns

Figure 16: Device summary of STTH6003

<u>MOSFET</u>

Figure 17: IRF300P227

IRF300P227 MOSFET was selected for the buck converter stage operating at 20 kHz and carrying approximately **30 A** of continuous current under steady-state conditions. During motor start-up, short-term currents can reach **40 A**, but once the system reaches a stable speed and torque, the current settles around **30 A**. Key technical specifications and maximum ratings of the MOSFET at 25 °C are summarized in table below.

Table 2 Maximum ratings (at T_J=25°C, unless otherwise specified)

Parameter	Symbol	Conditions	Values	Unit
Continuous Drain Current	I_D	T_C = 25°C, V_{GS} @ 10V	50	A
Continuous Drain Current	I_D	T_C = 100°C, V_{GS} @ 10V	35	
Pulsed Drain Current ①	I_{DM}	T_C = 25°C	189	
Maximum Power Dissipation	P_D	T_C = 25°C	313	W
Linear Derating Factor		T_C = 25°C	2.1	W/°C
Peak Diode Recovery ③	dv/dt	T_J = 175°C, I_S = 20A, V_{DS} = 150V	6.0	V/ns
Gate-to-Source Voltage	V_{GS}	-	± 20	V
Operating Junction and Storage Temperature Range	T_J T_{STG}	-	-55 to + 175	°C
Soldering Temperature, for 10 seconds (1.6mm from case)	-	-	300	
Mounting Torque, 6-32 or M3 Screw	-	-	10 lbf·in (1.1 N·m)	-

Table 3 Thermal characteristics

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Junction-to-Case ⑦	$R_{\theta JC}$	T_J approximately 90°C	-	-	0.48	°C/W
Case-to-Sink, Flat Greased Surface	$R_{\theta CS}$	-	-	0.24	-	
Junction-to-Ambient	$R_{\theta JA}$	-	-	-	40	

Figure 18: Mosfet Ratings

Continuous Drain Current (I_D): 50 A (@ 25 °C, V_{GS} =10 V)
Under nominal conditions, our system draws about 30 A steady-state. The 50 A rating provides a comfortable safety margin.

Drain-to-Source Voltage (V_{DS}): 200 V or higher (depending on the datasheet version)
After three-phase rectification, the DC bus can reach around 210–220 V. Because the MOSFET is placed in a buck configuration (and we leave some additional voltage margin), the chosen device's voltage rating is adequate.

Gate-to-Source Voltage (V_{GS}): ± 20 V
We plan to drive the gate at about 10–12 V, so the ±20 V limit is sufficient from a driver protection standpoint.

Thermal Characteristics ($R_{\theta JC}$, $R_{\theta JA}$):

- Junction-to-Case ($R_{\theta JC}$): ~0.48 °C/W (typical)
- Junction-to-Ambient ($R_{\theta JA}$): ~40 °C/W
  With a properly sized heatsink, the overall thermal resistance from junction to ambient can be lowered further, enabling reliable long-term operation at around 30 A.

2. Waveforms and Steady-State Values

MOSFET Current Waveform:

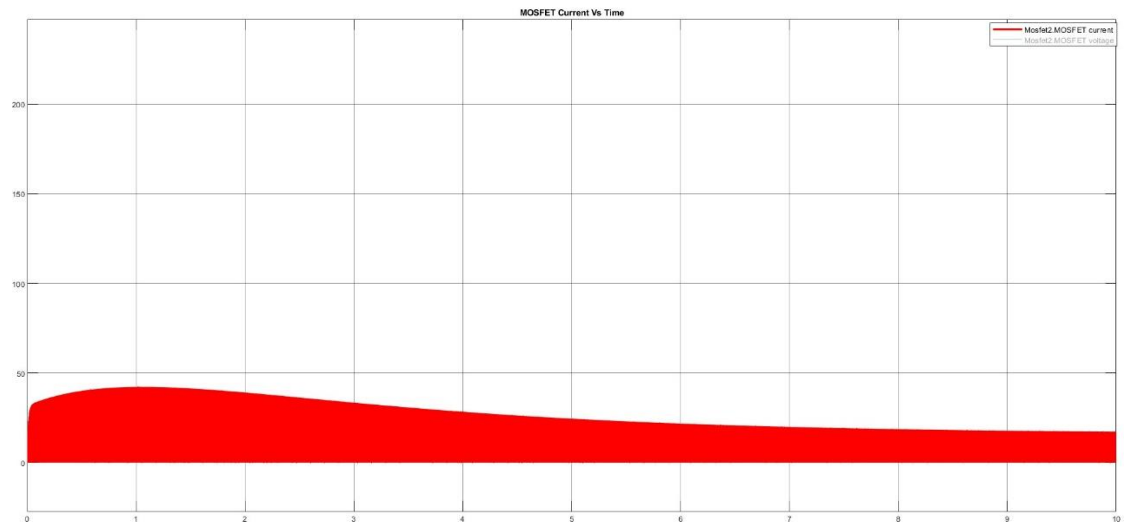

Figure 19: Current Vs Time Of The MOSFET

At the 20 kHz switching frequency, the current averages around 30 A with a ripple in each switching period. During start-up, short peaks of up to ~40 A occur, but once the motor is at nominal speed and torque, the current stabilizes around 30 A.

MOSFET Drain-Source Voltage

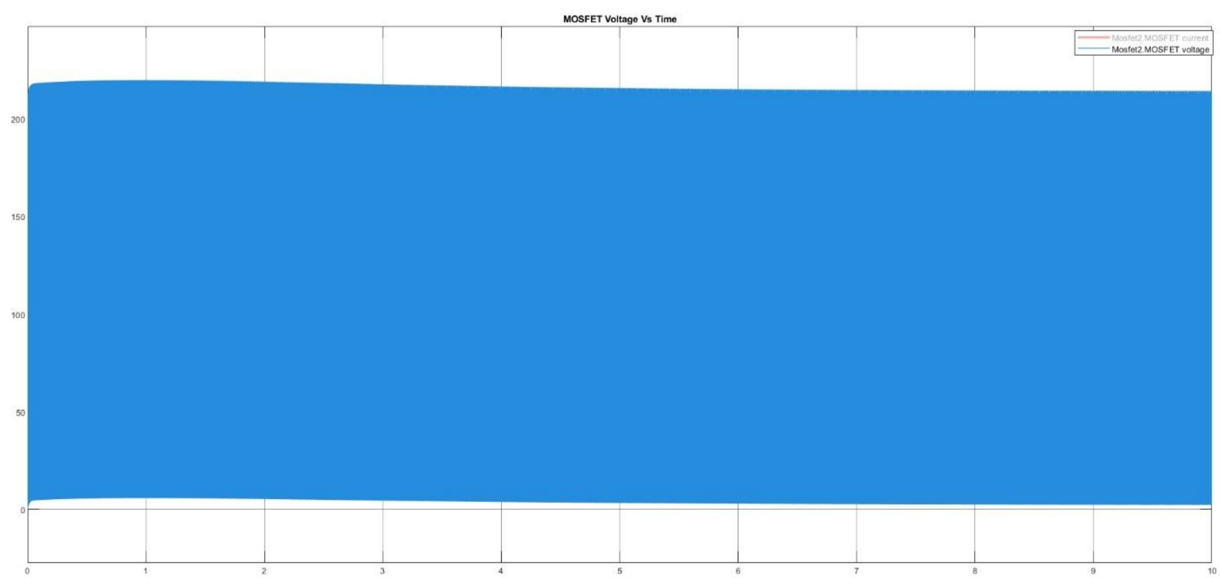

Figure 20: Voltage Vs Time Of The MOSFET

When the MOSFET is off, the drain-source voltage rises to nearly the DC bus level (around 200 V). During conduction, the voltage drop reduces significantly (on the order of a few volts or less, depending on $R_{DS(ON)}$ and the current). The waveform in Figure 2 resembles a rectangular pulse, transitioning between near-bus voltage and the low conduction voltage.

3. Power and Thermal Analysis

Conduction Losses (I^2R): Assuming a moderate $R_{DS(on)}$ (e.g., 5–10 mΩ), at 30 A the conduction loss is $P_{cond} \approx 30^2 \times 0.005\,\Omega = 4.5\,W$. This is well below the MOSFET's listed maximum power dissipation (313 W at 25 °C), although actual derating with temperature must be considered.

Switching Losses : At 20 kHz, there are turn-on/turn-off transitions that add to the total power dissipation. The exact switching loss depends on the MOSFET's parasitic capacitances, di/dt, and the driver and snubber circuitry. Under typical conditions for this power level, total switching losses plus conduction losses can bring overall dissipation to around 10–15 W (depending on specific conditions).

Thermal Management and Cooling: With $\boldsymbol{R_{\theta JC}}$ **= 0.48 °C/W**, heat transfer from the MOSFET junction to its case is relatively efficient. Using a well-sized heatsink (plus possible thermal interface material) can keep the case temperature at a safe level. The choice of heatsink (defining$R_{\theta CA}$) must keep the junction temperature below the maximum 175 °C. For instance, at around 10 W total dissipation,$10\ \mathrm{W} \times \left(R_{\theta JC} + R_{\theta CS} + R_{\theta SA}\right) < \left(175\ ^{\circ\mathrm{C}} - T_{\mathrm{ambient}}\right)$must be satisfied.

DC Link Capacitor

Another step of the design is to choose DC link capacitor for the output of three phase full bridge rectifier. Main purpose of this is reduce ripple by filtering out harmonics at the output. So it has critical effect on the design procedure. **Type 550C Ultra Ripple Aluminum 1000u** capacitor is chosen which has low ESR of 0.04ohm.

Figure 21: Type 550C Capacitor

Controller

The **Arduino Uno** was chosen as the platform for implementing the PI controller and generating PWM signals due to its simplicity, flexibility, and compatibility with the project's requirements. The Arduino Uno, based on the ATmega328P microcontroller, provides sufficient processing power to implement a discrete PI control algorithm efficiently while maintaining the real-time responsiveness needed for motor speed regulation. Its built-in PWM capability allows it to generate precise control signals for the MOSFET in the buck converter. Additionally, the Arduino's extensive library support and ease of programming significantly reduce development time. Its ability to interface seamlessly with sensors, such as current and voltage feedback sensors, ensures accurate data acquisition for closed-loop control. The availability of multiple I/O pins further allows integration with external hardware, making it a cost-effective and robust solution for this application.

Figure 22: Arduino Uno microcontroller

Gate Driver and DC Battery

For drive switching device MOSFET, we need gate driver such that can be supplied same source with Arduino UNO. Microcontroller is supplied with 12V DC Battery in Figure 14**. MIC4421/4422 9A-Peak Low-Side MOSFET Driver** in Figure 15 can be supplied with 12V DC also generate output larger than 10V which is enough to apply gate of mosfet to drive it.

Figure 23: DC Battery

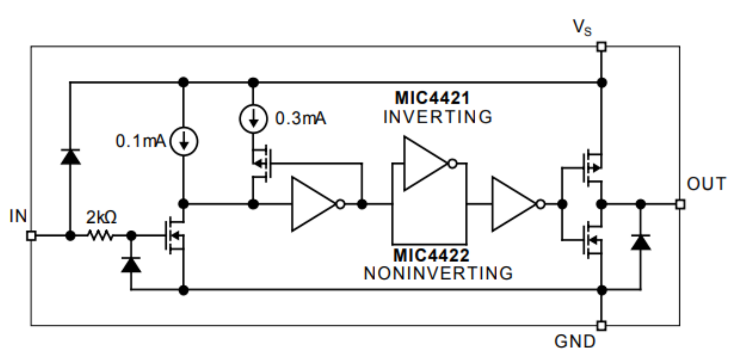

Figure 24: MIC4421/4422 Mosfet Driver

Speed Sensor

To control closed loop system we need to get feedback from system's speed, and it can be achieved by using speed sensor of **Hall-Effect Sensor A3144**

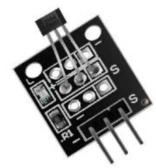

Figure 25: Hall-Effect Sensor Module

It attaches a small magnet to the motor shaft. Then Hall-effect sensor detects the magnetic field as the shaft rotates, generating digital pulses for each rotation. Digital signal (high and low pulses) proportional to the rotation speed.

6. PCB Design

After designing the circuit and selecting the components, we are now ready to design a PCB board where our circuit will be built upon. In PCB design process, we use KiCad program which is an open-source program for PCB Design. In the PCB design process, we follow three steps: Schematic design, footprint design and PCB design.

In the schematic design part, we build our circuit on KiCad as seen in figure 26. While building this circuit, we use the real components that we selected because when we design PCB board, we need to know the exact dimensions of the components. Not all the components are available in KiCad's libraries. In this case, we can build a custom symbol and custom footprint for our component. Also, we can use similar components that match their size. For example, we select IRF300P227 as our MOSFET, but this component does not exist in KiCad libraries. However, IRFP4468PbF exists. So, I first compare their dimensions from their datasheets. When I find that they have exactly same dimensions, I used that instead of building a custom symbol and footprint.

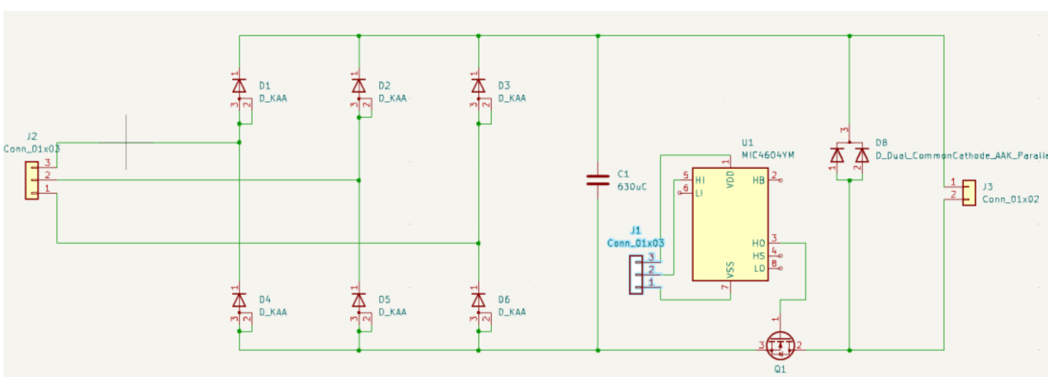

Figure 26: Schematic Design of the circuit in KiCad.

After schematic design, we add footprints for each component. Footprints are the 2 and 3 dimensional drawings of the components as you can see in figures 27 and 28. To build a PCB, we need to know the size of each component.

Figure 27: Footprint of the MOSFET

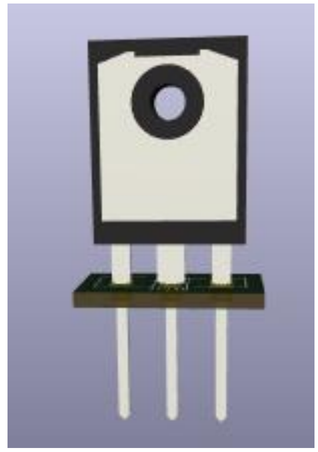

Figure 28: 3-D image of the MOSFET

After schematic and footprint design, we can now work on PCB design. When we open PCB editor, we can see all the components with their footprints. Then we adjust each component as how we want to lay them on the board, as in figure 29. After this, we need to draw root tracks.

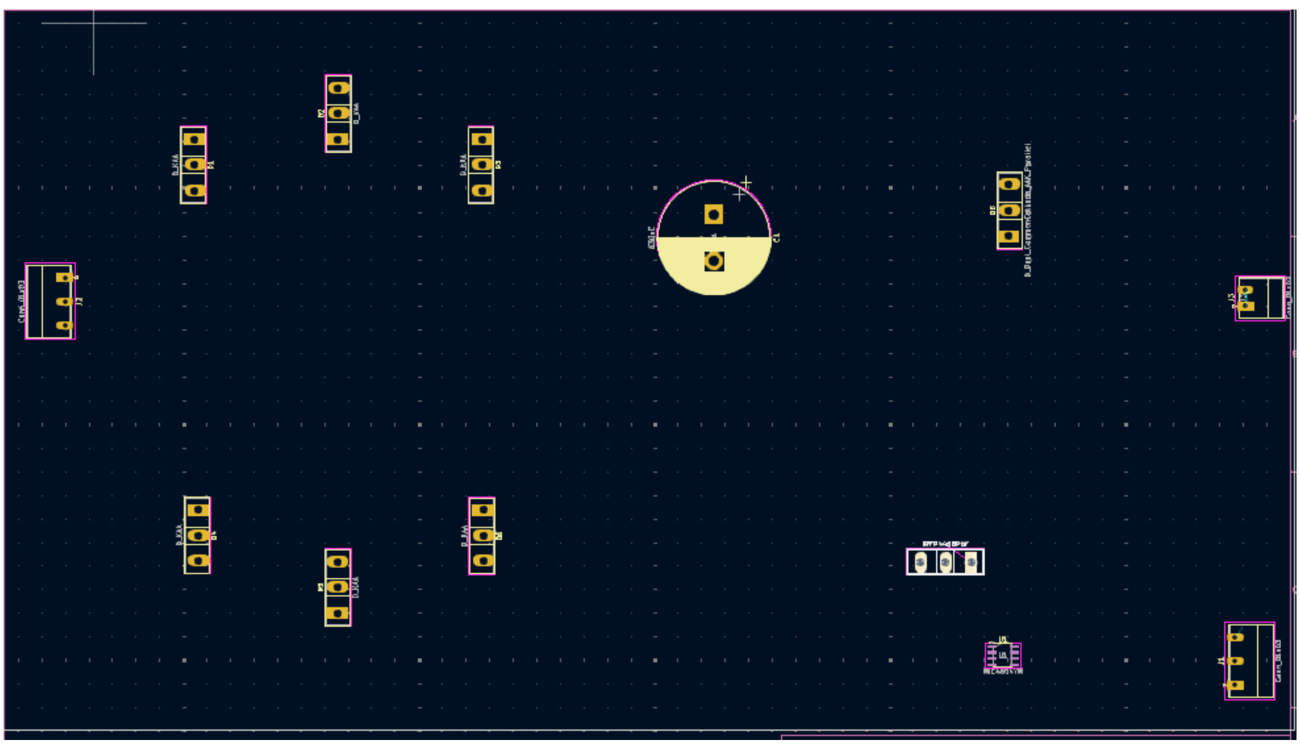

Figure 29: Components on PCB editor.

Root tracks are the path where current can pass through. Before drawing root tracks, we first need to decide on their width and depth. This decision depends on how much current will flow on them. Depending on the current flow, the width of the root tracks changes. To decide on the required root track width, we use KiCad's Calculator Tool. When we first calculate the required root track, it come up more than 80 mm. This is too big. To overcome this problem, we first change the thickness of the traces. By doing so, required width drops but still too big. Then, we introduce multiple layers so that current flows in parallel and required width drops to reasonable size. In our design, we introduced 8 layers. In figure 31, traces on the first two layers are shown. In figure 32, all traces on all layers are shown. The 3-D images of the circuit can be seen in figures 33 and 34.

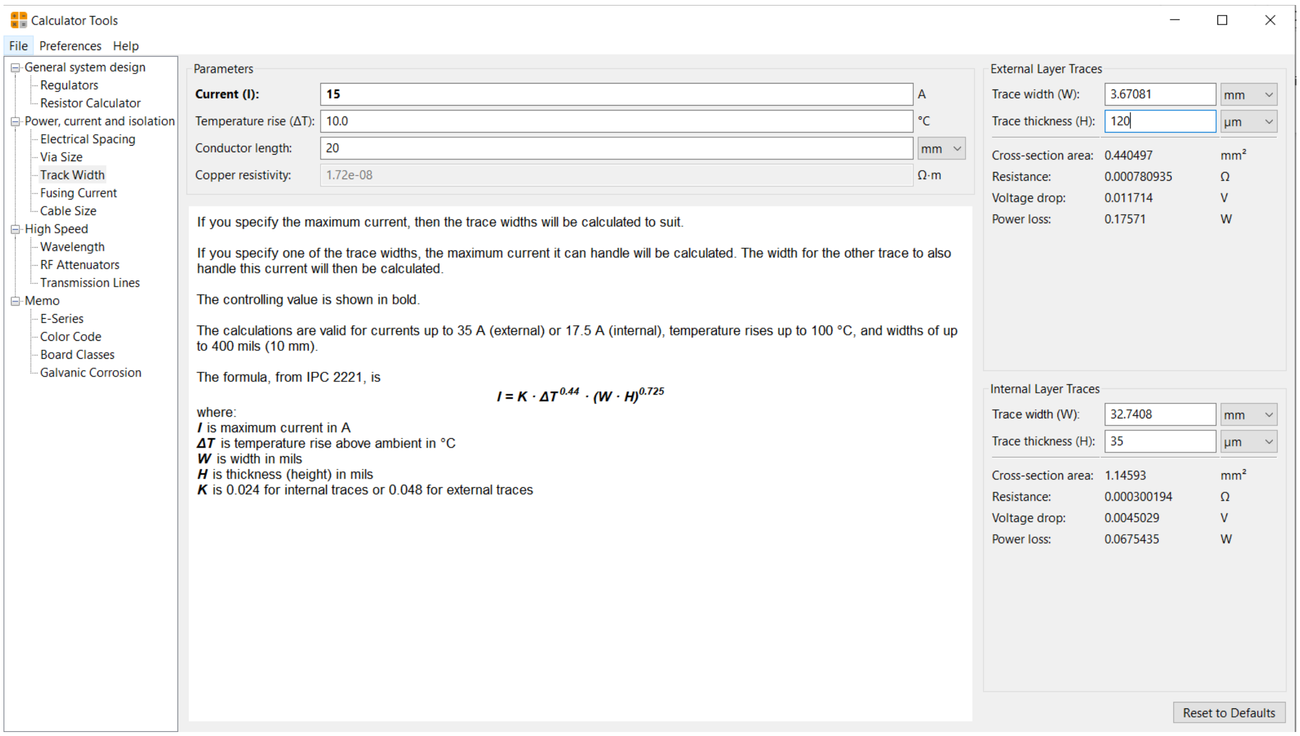

Figure 30: KiCad's Calculator Tool

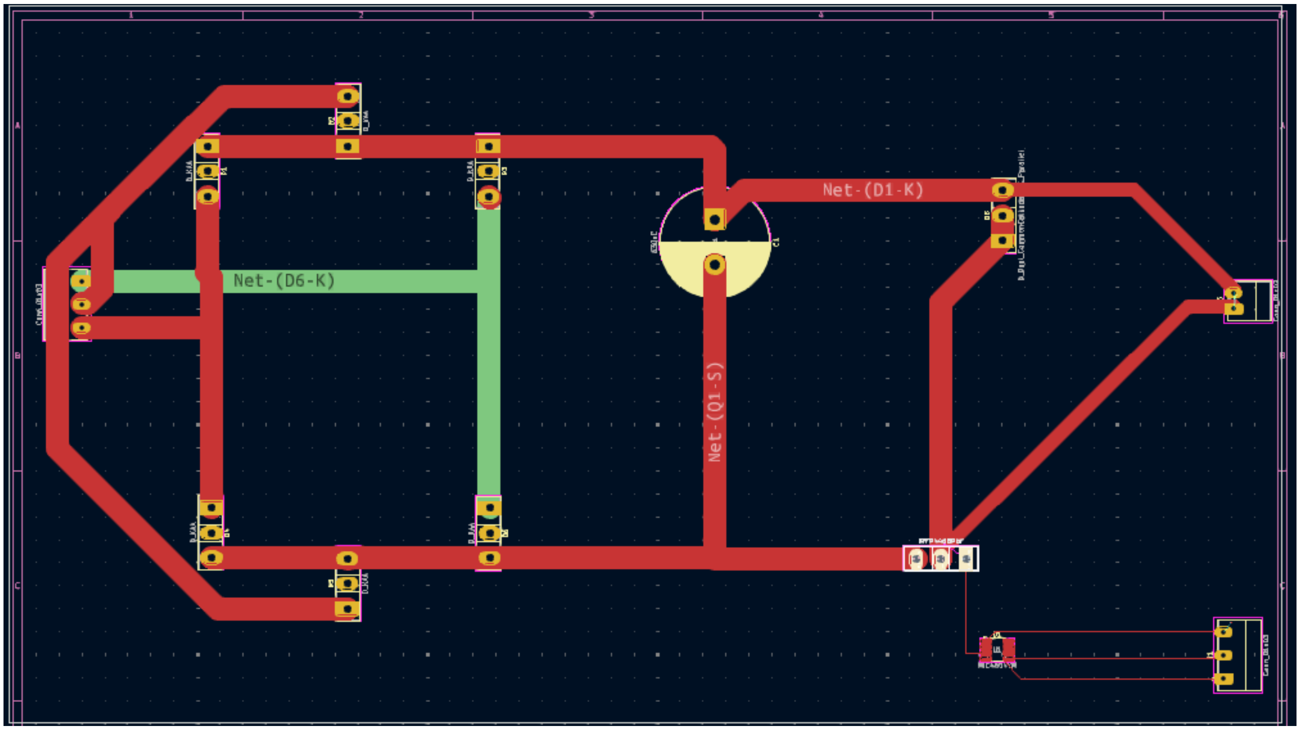

Figure 31: Root tracks on first two layers.

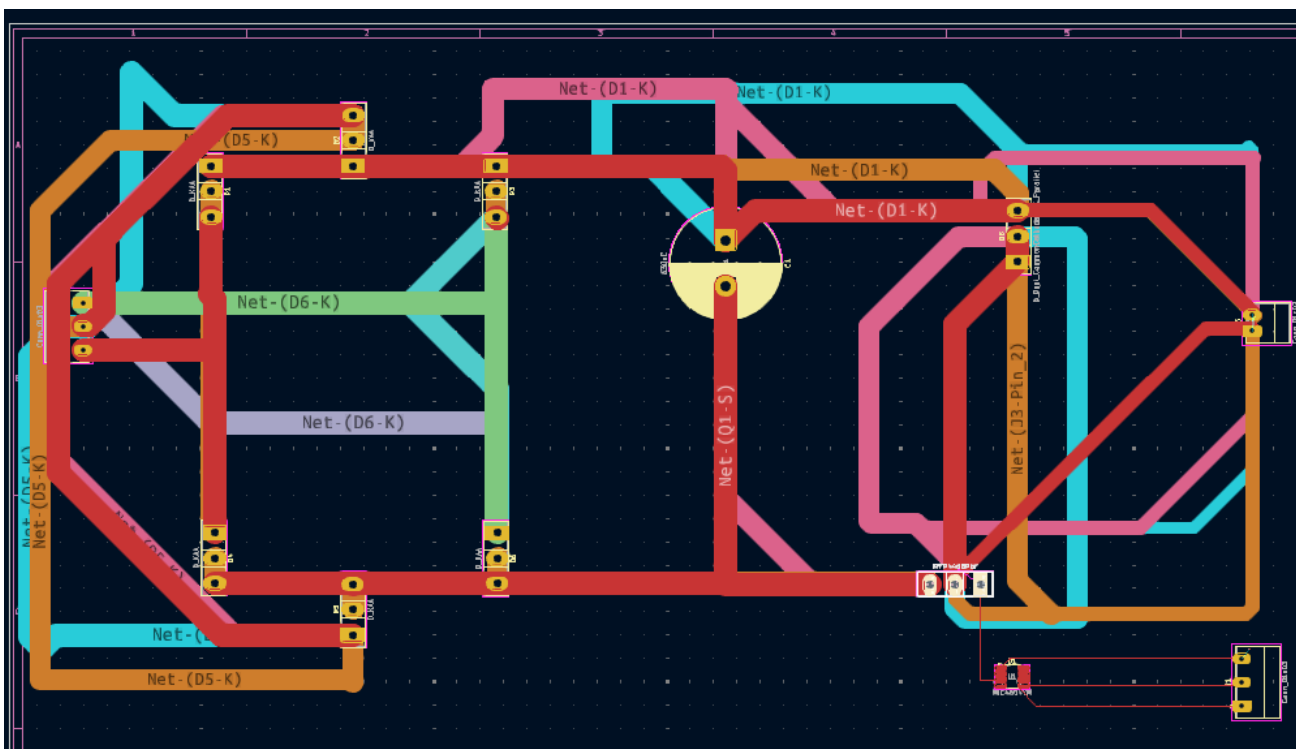

Figure 32: Root tracks on all layers together

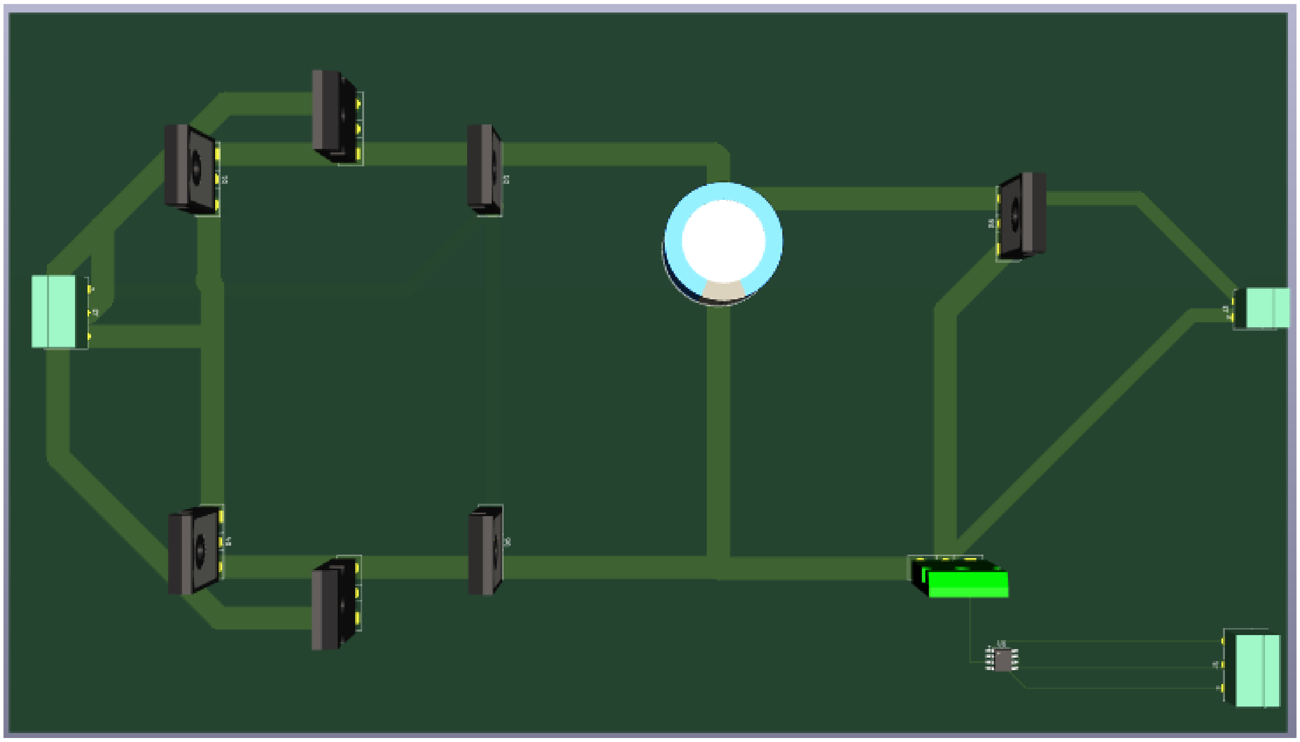

Figure 33: Top view of the circuit on 3-D modelling.

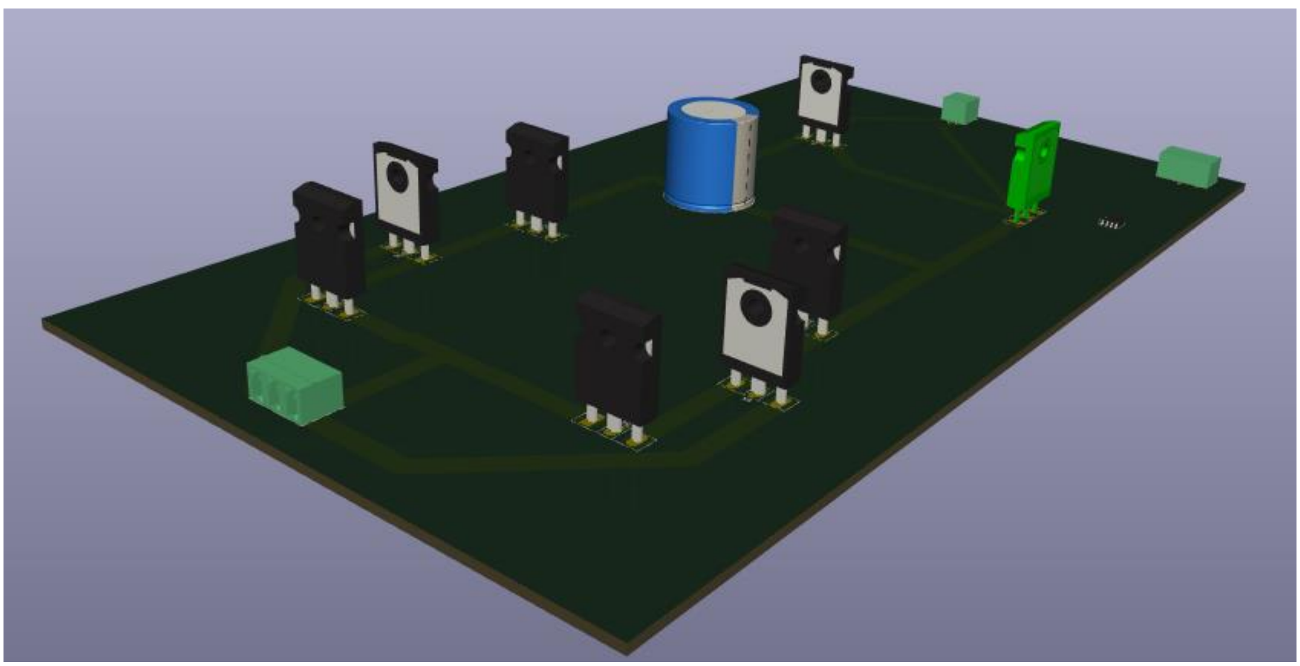

Figure 34: view of the circuit at another angle.

7. Thermal Analysis

To make thermal analysis, system builded within PLECS enviroment

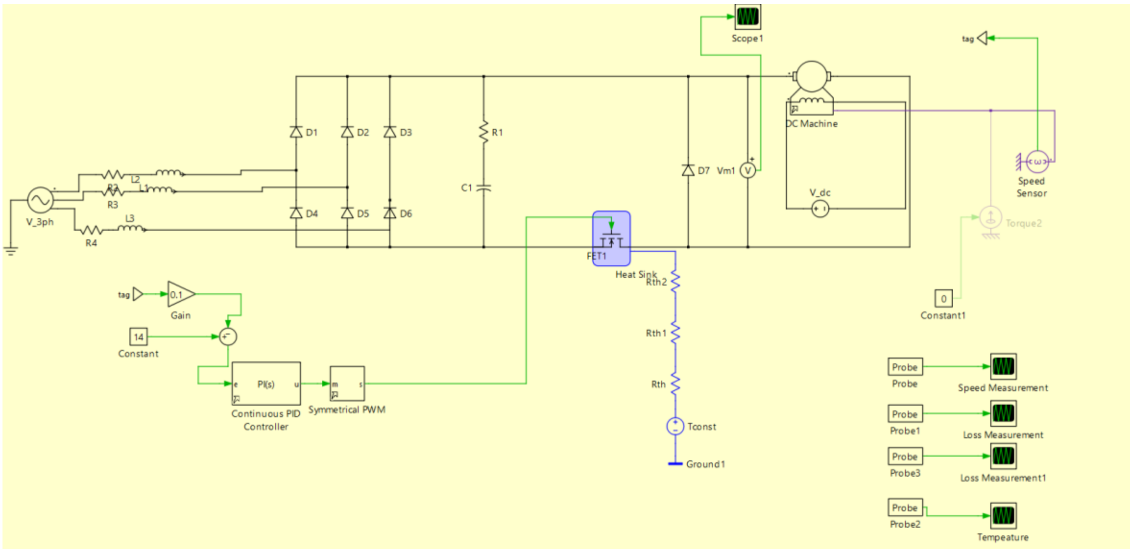

Figure 35: PLECS Shematics of the system with thermal analysis

In our DC motor drive design operating at a 20 kHz switching frequency, the selected MOSFET (IRF300P227) is expected to dissipate **approximately 20 W** (combined conduction and switching losses). To keep the junction temperature below **100 °C** in a **20 °C** ambient environment, we need a total junction-to-ambient thermal resistance ($\boldsymbol{R_{\theta JA}}$) of about **4 °C/W**.

Because the MOSFET itself has a **junction-to-case thermal resistance** of 0.48 °C/W ($R_{\theta JC}$) and a **case-to-sink interface** contributes an additional 0.2 °C/W ($R_{\theta CS}$), these internal resistances sum to 0.68 °C/W. Subtracting 0.68 °C/W from the 4 °C/W overall target leaves approximately **3.32 °C/W** of allowable sink-to-ambient (R_θSA). Therefore, a heatsink with an R_θSA of **3.3 °C/W or lower** ensures the MOSFET junction remains at or below 100 °C under continuous 20 W dissipation.

After examining commercially available solutions, we chose an **Aavid Thermalloy 530002B02500G** heatsink with an $\boldsymbol{R_{\theta SA}}$ around **2.7 °C/W** under natural convection. This design offers a comfortable safety margin—at 20 W, the heatsink will rise about 54 °C above ambient, and the MOSFET/device interface adds a further 13–14 °C, totaling roughly 88 °C. This value stays within our 100 °C junction temperature target, even if ambient conditions vary slightly.

We selected this particular model for several reasons. First, its **extruded aluminum construction** provides an effective balance between weight, size, and thermal performance. Second, its **mounting profile** accommodates typical power packages such as TO-220 or TO-247, ensuring straightforward integration with the IRF300P227 MOSFET. Finally, the **2.7 °C/W** rating offers a comfortable **margin** for transient events or small deviations in switching/conduction losses at 20 kHz. Should the application's load conditions or ambient temperature rise further, a lower-rated heatsink or forced-air cooling could be employed to maintain safe junction temperatures.

In summary, selecting a heatsink with $R_{\theta SA}$ **≈ 2.7 °C/W** ensures our 20 kHz, 20 W dissipation requirement keeps the MOSFET junction below 100 °C. This choice prevents thermal runaway, extends device lifetime, and provides robust operation during load transients in our high-frequency DC motor drive.

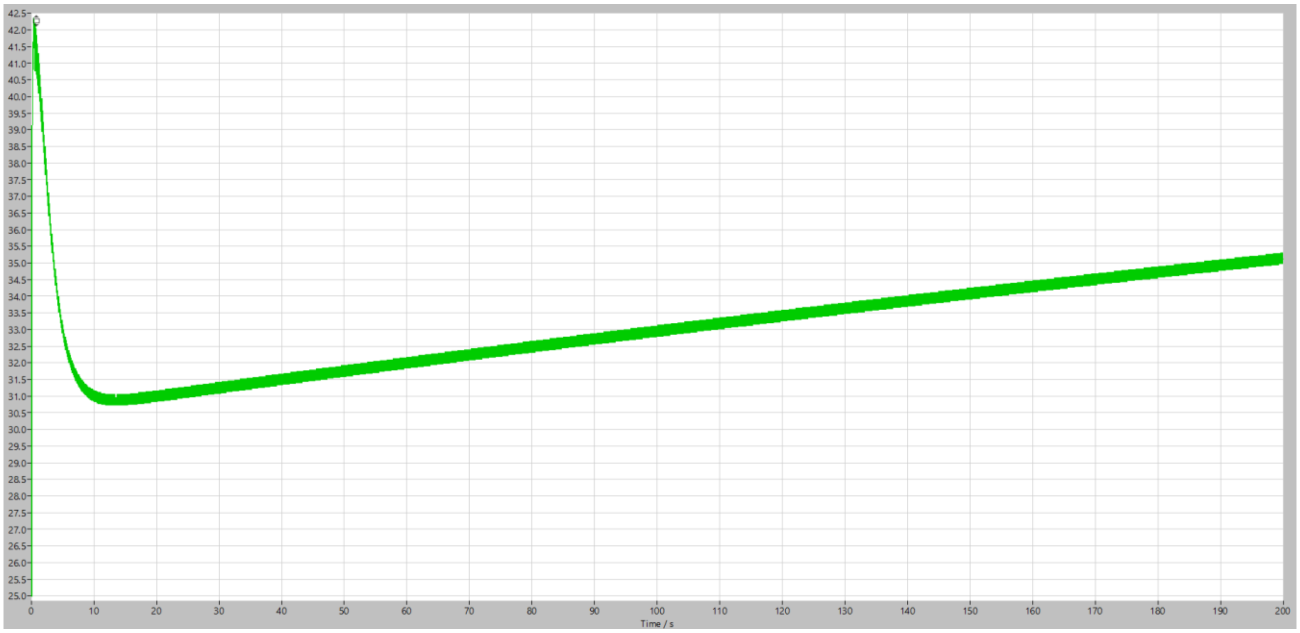

Figure 36: Thermal Analysis of the System After Long Term Run

It has been observed that with the heatsink added to the system, the temperature does not increase to problematic values and operates within safe ranges. Since the conduction losses of the system decreased after the system adjusted itself, temperature problems could be solved more easily.

8. Conclusion

This project successfully demonstrates the end-to-end design of a DC motor drive based on a three-phase rectifier and a buck converter, capable of delivering up to 180 V DC output. By combining careful selection of power semiconductor devices (VS-60APH03-N3 rectifier diodes, STTH6003 freewheeling diode, and IRF300P227 MOSFET) with a well-tailored PI controller, the system achieves stable and efficient motor operation under varying load conditions.

The three-phase rectifier ensures lower output ripple and higher efficiency compared to single-phase alternatives, while the buck converter allows fine regulation of the DC bus voltage. In the simulations, each stage—from the rectifier output to the closed-loop control—was verified under realistic conditions (including non-idealities and parasitic elements). Aided by the PI speed controller, the motor maintains its target speed, with the control loop automatically adjusting the duty cycle in response to load torque changes or supply fluctuations.

Thermal management emerged as a key consideration at 20 kHz switching and roughly 20 W of power loss on the MOSFET. By choosing an Aavid Thermalloy 530002B02500G heatsink ($R_{\theta SA}$ ≈ 2.7 °C/W), the MOSFET's junction temperature remains below 100 °C, preserving device reliability. PCB design in KiCad further integrates these components into a compact layout, accounting for appropriate trace widths and minimizing electromagnetic interference.

Overall, the project achieves its primary goal: delivering a robust, scalable DC motor drive platform with efficient power conversion, effective speed control, and safe thermal performance. The chosen topology and components offer a solid foundation for future refinements—such as forced-air cooling, improved gate-driver design, or higher switching frequencies—while retaining flexibility for different industrial or academic applications.

Arduino Code for PI Controller and PWM Generation for Speed Control

Constants

const int pwmPin = 9;   // PWM output pin for MOSFET driver
const int hallSensorPin = 2;   // Hall-effect sensor pin
const int pwmFrequency = 10000; // PWM frequency (10 kHz)
const float samplingInterval = 0.1; // Sampling interval for PI control (100 ms)
const int vref = 157;   // Reference speed in RPM (adjustable)

Constants

```
const int pwmPin = 9;           // PWM output pin for MOSFET driver
const int hallSensorPin = 2;    // Hall-effect sensor pin
const int pwmFrequency = 10000; // PWM frequency (10 kHz)
const float samplingInterval = 0.1; // Sampling interval for PI control (100 ms)
const int vref = 157;           // Reference speed in RPM (adjustable)
```

PI Controller Constants

```
const float kp = 0.01;           // Proportional gain
const float ki = 0.02;           // Integral gain
```

Variables

```
volatile unsigned int pulseCount = 0; // Pulse count from Hall sensor
float motorSpeed = 0;           // Measured speed in RPM
float error = 0;                // Speed error
float integral = 0;             // Integral term for PI controller
float controlOutput = 0;        // Output of PI controller
unsigned long lastTime = 0;     // Last control loop time
```

Setup Function

```
void setup() {
    // Initialize pins
    pinMode(pwmPin, OUTPUT);
    pinMode(hallSensorPin, INPUT_PULLUP);
```

```
    // Attach interrupt for Hall sensor
    attachInterrupt(digitalPinToInterrupt(hallSensorPin), countPulse, RISING);

    // Set PWM frequency
    analogWriteFrequency(pwmPin, pwmFrequency);

    // Start serial communication for debugging
    Serial.begin(9600);
}
```

Main Loop

```
void loop() {
    unsigned long currentTime = millis();

    // Run PI control loop at defined sampling interval
    if ((currentTime - lastTime) >= (samplingInterval * 1000)) {
        lastTime = currentTime;

        // Calculate motor speed in RPM
        noInterrupts();
        unsigned int pulses = pulseCount;
        pulseCount = 0;
        interrupts();
        motorSpeed = (pulses * 60.0) / (1.0 / samplingInterval); // RPM

        // Calculate PI controller output
        error = vref - motorSpeed;
        integral += error * samplingInterval;
        controlOutput = (kp * error) + (ki * integral);

        // Constrain control output to valid PWM range (0-255)
        controlOutput = constrain(controlOutput, 0, 255);

        // Output PWM signal
        analogWrite(pwmPin, (int)controlOutput);

        // Debugging output
        Serial.print("Speed (RPM): ");
        Serial.print(motorSpeed);
        Serial.print(" | Error: ");
        Serial.print(error);
        Serial.print(" | PWM: ");
        Serial.println((int)controlOutput);
    }
}
```

Interrupt Service Routine

```
void countPulse() {
    pulseCount++;
```

```
}
```