MIDDLE EAST TECHNICAL UNIVERSITY
DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

EE 463 – Static Power Conversion -1

Hardware Project Final Report

I-Diodes

Mert Sevim
Polat Öztürk
Kaan Özsoy

Table of Contents

INTRODUCTION

In this project, we designed and implemented a controlled power electronic driver to operate a DC motor, aiming to meet specific performance criteria such as adjustable output voltage and current control. The primary objective was to convert the standard AC grid voltage into a regulated DC output capable of driving the motor under various load conditions, including a high-power scenario.

To achieve this, we selected a topology consisting of a 3-Phase Full Bridge Diode Rectifier followed by a Buck Converter. This configuration allows us to rectify the AC input into a high-voltage DC bus and then step it down to the required level using a high-frequency switching scheme. The system is controlled by a microcontroller that executes a closed-loop PI algorithm to regulate the armature current and voltage, ensuring stable operation.

Throughout this report, we present the step-by-step implementation of the hardware, including power component selection, gate driver design, and current sensing strategies. We also detail the software structure used for the control loops. Finally, we provide experimental results from our demo session, verifying the system's performance under resistive loads, motor soft-start conditions, and the high-power "Tea Bonus" test.

Topology Selection: 3 Phase Full Bridge Diode Rectifier + Buck Converter

In this configuration, the three-phase diode rectifier takes a 380 V AC input and creates a stable, high DC voltage (~520 V). This voltage is then taken by the buck converter, providing the adjustable DC output required by the motor. The buck converter's high-frequency switching ensures a ripple frequency of f_ripple > 1 kHz, a critical requirement for the project.

Pros:

- Provides a highly reliable, stable, and high DC voltage.
- Control is simple.
- Easily meets the f_ripple > 1 kHz requirement.
- Cost-effectiveness.

Cons:

- Efficiency could be a problem.

3 Phase Full Bridge Diode Rectifier + Buck Converter topology easily meets switching frequency requirements. It also generates a much more stable and higher DC bus voltage than other single-phase alternatives by utilizing the existing three-phase supply. For these reasons, it was chosen as a reliable and cost-effective solution that best balances high performance requirements with low control complexity.

SIMULATION

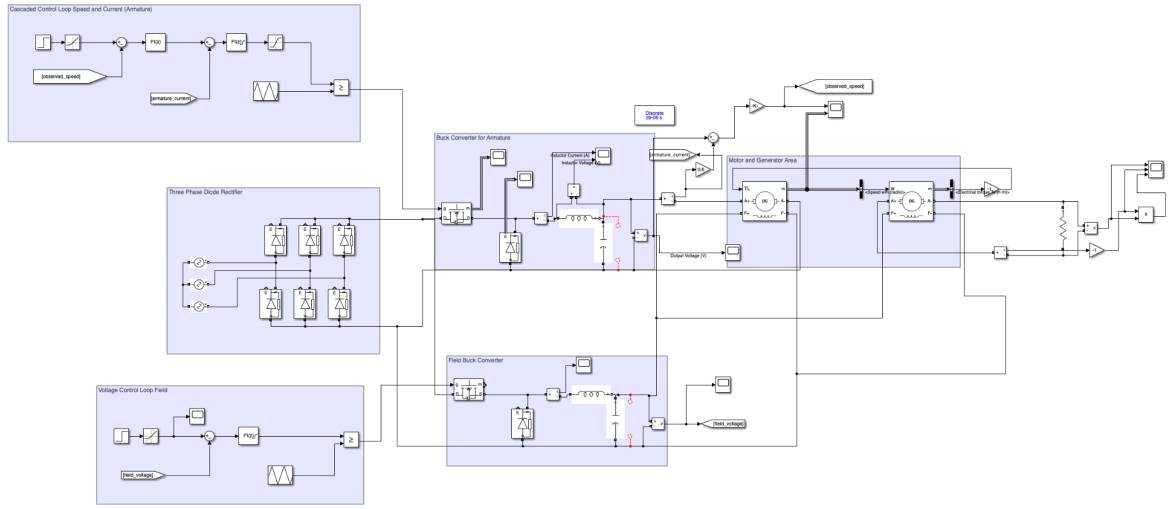

Figure 1. Overall Circuit Schematic for Motor Control

In this simulation we start with creating a motor-generator combination which will simulate the actual system. To do that we connect one motor's speed output to the other motors, which will act as a generator, speed input. Then we connect the generator's torque output to the motor's torque input as a reverse value. Then we connect generator field voltage to our 180 V buck generated field voltage and connect 17 ohm resistive load to the armature of it. We can see this structure in Figure 2.

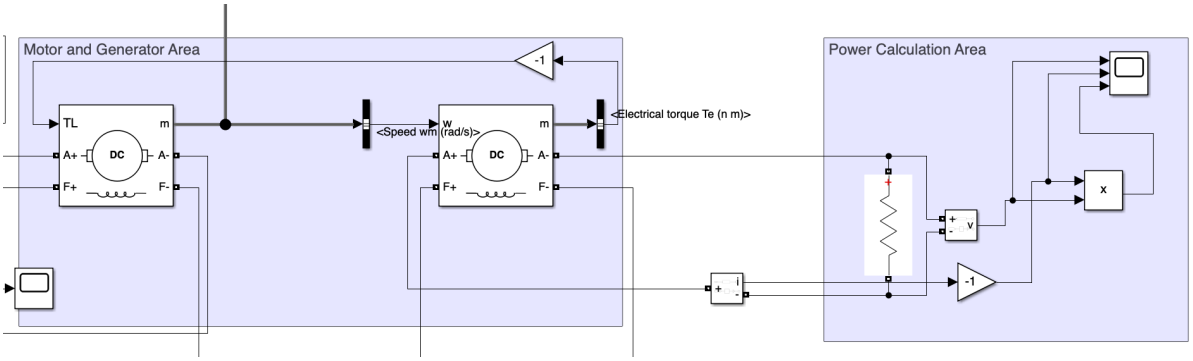

Figure 2. Motor and Generator Area

Then we connect this structure to the previously explained topologies:

- Buck Converter
- Three Phase Diode Rectifier

We connect the field of motor and generator to the same buck converter with 180 V controlled output. We set its output to 180 V using the PI control loop. We measure the output voltage and by subtracting it from 180 we give it to the PI loop as an error term. We can see the field control loop in Figure 3.

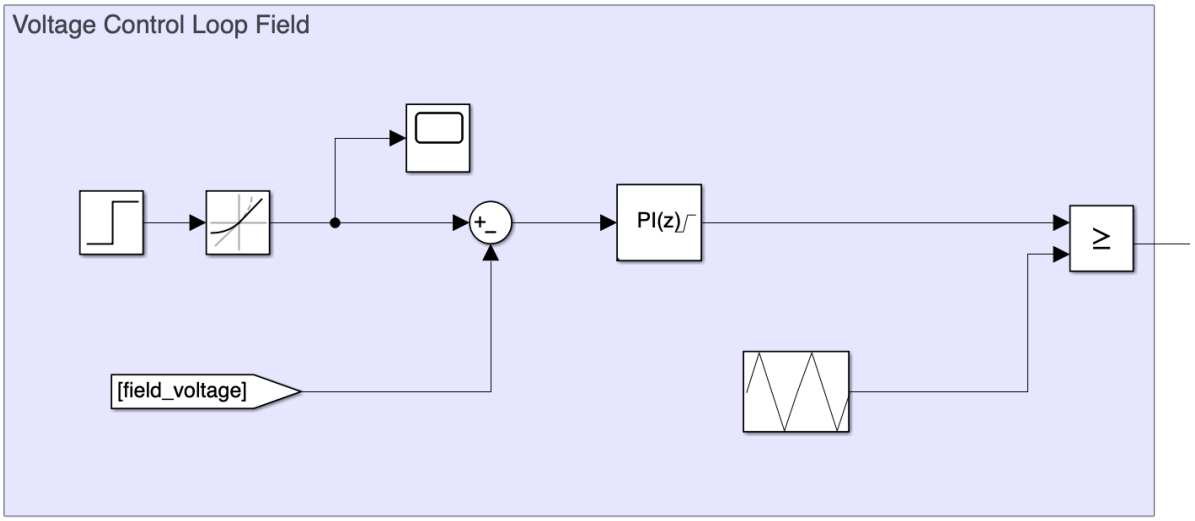

Figure 3. Field Control Loop

To control speed of the motor we control the armature current by using cascaded PI loop which consists of first speed loop and then current loop. We give a difference between reference speed and the observed speed to the outer PI loop then by subtracting output of it from the armature current we create the error term for the inner PI loop. After that we use this output as a duty in our PWM generation. We can see this control loop in Figure 4.

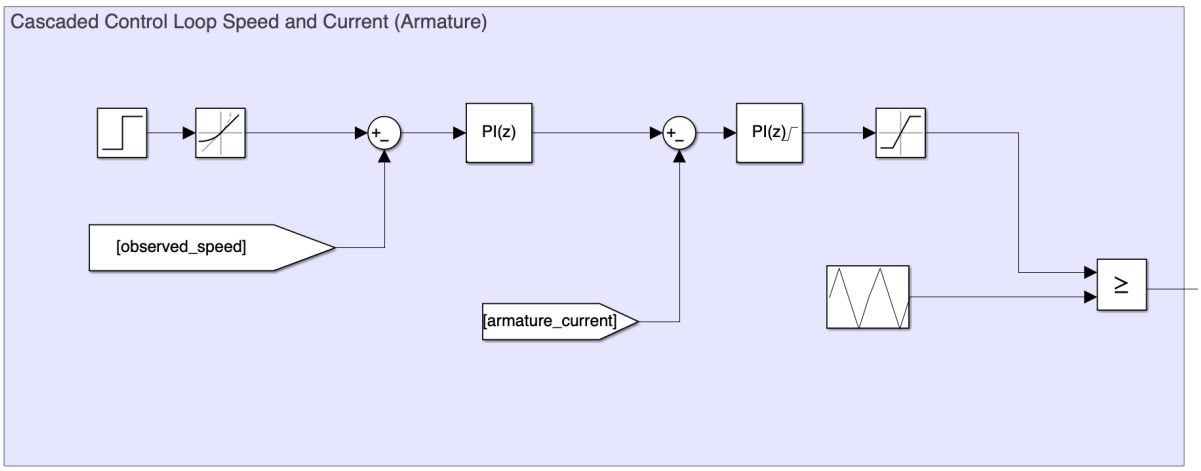

Figure 4. Armature Control Loop

To avoid sudden current jumps we first start with the increasing field voltage to the 180 V by giving its reference as a ramp. Then we start to control armature current again giving its speed reference as a ramp. We can see the measurements of the motor in Figure 5.

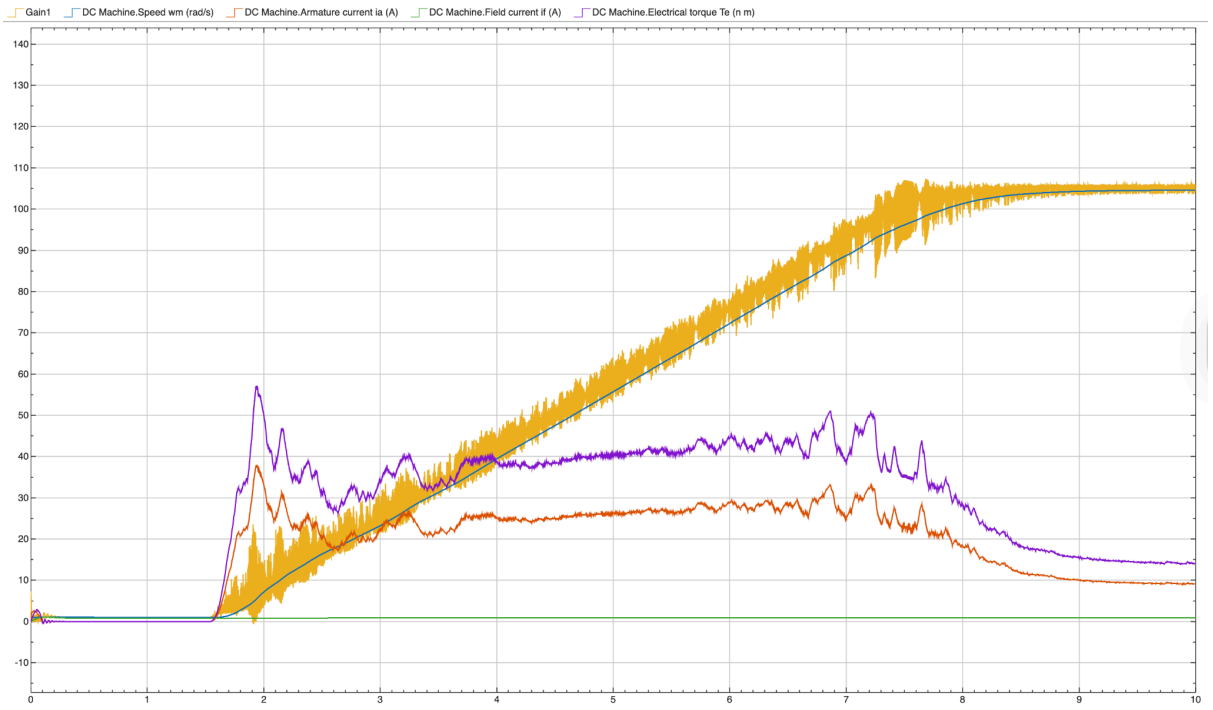

Figure 5. Motor Measurements

In this figure we can see that after speed gets its steady state armature current and torque decrease. Also we can see that our observed speed tracks the actual speed well.
Finally if we look at the generator's output voltage, current and power at steady state we achieve 1.4kW. By optimizing some parameters we are planning to achieve 1.6kW power generation. We can see it in Figure 6.

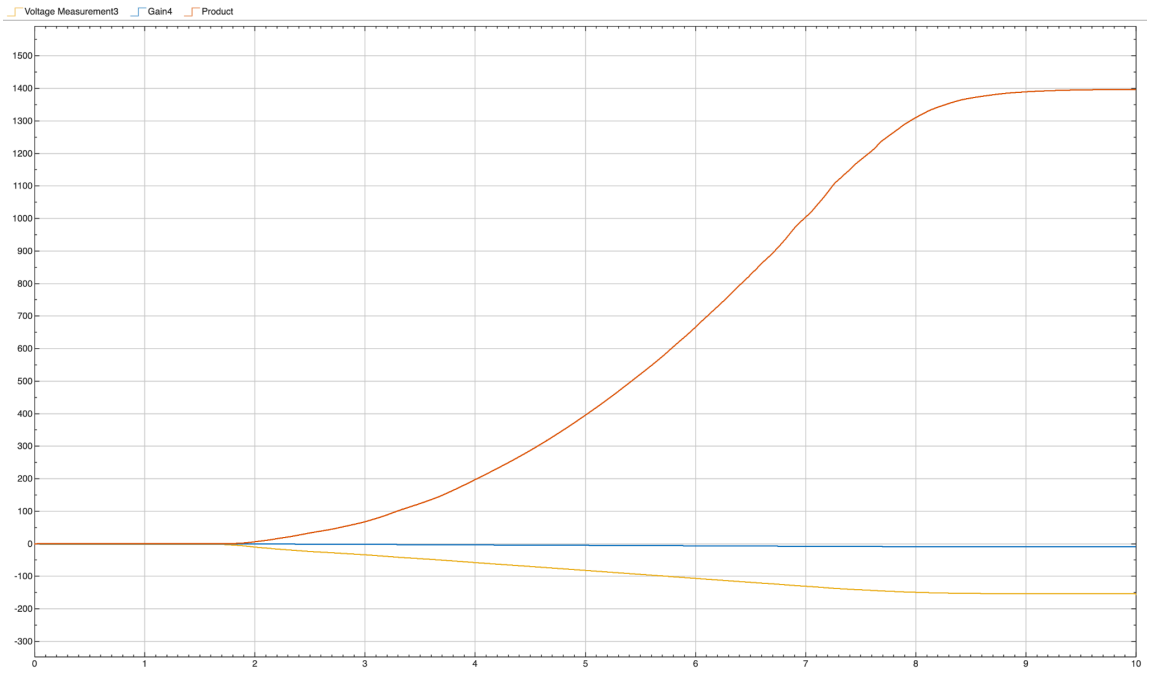

Figure 6. Figure For Generator Measurements (Red: Power, Blue: Current, Yellow: Voltage)

IMPLEMENTATION

Power Unit

The power unit is the core of our motor driver project. Based on our simulation results and design requirements, we implemented a topology consisting of a 3-Phase Full Bridge Diode Rectifier followed by a Buck Converter. This structure allows us to convert the standard grid voltage into a controlled DC output for the motor.

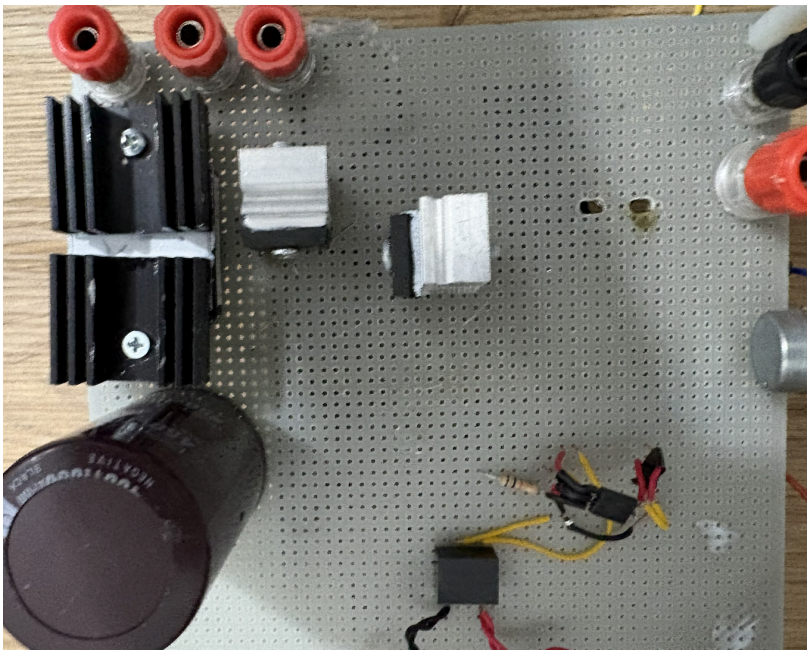

Figure 7: Circuit of The Power Unit

Rectifier Stage

For the input stage, we used a VUO98-16NO7 3-phase full bridge rectifier module.

- **Input Voltage Adjustment:** Although the standard grid voltage is 380 V, we calculated that an input voltage of approximately 260 V AC is sufficient to obtain the required 180 V DC output for the motor. Therefore, we used a variac to adjust the input voltage to this level.
- **Voltage Rating:** The line-to-line input voltage is up to 380 V AC, which results in a rectified DC bus voltage of approximately 520 V at maximum input. The selected module has a repetitive peak reverse voltage (VRRM) of 1600 V. This provides a very large safety margin against voltage spikes from the grid.
- **Current Rating:** Our nominal load current is around 12 A at maximum. The module is rated for 105 A, which is significantly higher than our requirement. This high rating ensures the rectifier can easily handle the inrush current of the motor without overheating or failing.

Figure 8: VUO98-16NO7 3-phase full bridge rectifier

DC Link Capacitor

To stabilize the DC bus voltage, we placed a 1000uF 450V capacitor between the output of the rectifier and the input of the buck converter.

- Function: This capacitor acts as a DC link filter. It smooths out the voltage ripples generated by the rectification process and provides a stiff DC voltage source for the buck converter. This ensures that the switching element receives a stable input voltage, which is essential for consistent control.

Buck Converter Stage

The buck converter steps down the rectified high DC voltage to the required DC voltage for the DC motor field and armature control. We selected the components to minimize switching losses and ensure safety.

Power MOSFET (Switching Element) We selected the IXFH80N65X2 N-Channel Power MOSFET.

- **Voltage Margin:** With a DC bus voltage of 520 V, the MOSFET's 650 V breakdown voltage provides a safety buffer of over 100 V. This protects the switch against inductive spikes during the turn-off process.
- **Efficiency:** The MOSFET has a continuous drain current rating of 80 A, which is much higher than our peak currents. This over-rating, combined with the component's ultra-low on-resistance, helps us reduce conduction losses and keeps the thermal stress on the heatsink low.

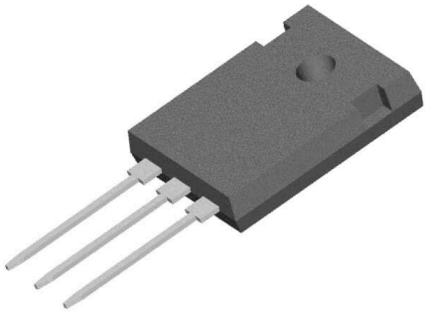

Figure 9: IXFH80N65X2 N-Channel Power MOSFET

Freewheeling Diode For the freewheeling path, we used the DSEI30-06A fast recovery diode.

- **Fast Recovery:** Since we are using a 1 kHz switching frequency, diode reverse recovery time is critical. The ultra-fast recovery characteristic of this diode minimizes the switching losses.
- **Current Handling:** We calculated the worst-case average diode current to be approximately 5.78 A. The DSEI30-06A is rated for 30 A, which gives us a comfortable safety margin.
- **Voltage Rating:** Its 600 V reverse voltage rating is sufficient to block the DC bus voltage.

Output Filter Strategy In the final implementation, we decided not to use an external LC filter. We determined that the DC motor's internal armature inductance and capacitance were sufficient to perform the necessary filtering. Therefore, we relied on the motor's inherent parameters to smooth out the current ripple and ensure stable operation, removing the need for extra passive components at the output.

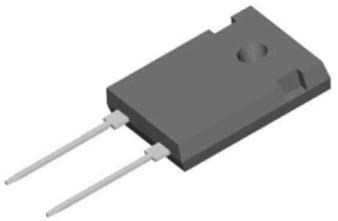

Figure 10: DSEI30-06A freewheeling diode

Output Filter Strategy In the final implementation, we decided not to use an external LC filter. We determined that the DC motor's internal armature inductance and capacitance were sufficient to perform the necessary filtering. Therefore, we relied on the motor's inherent parameters to smooth out the current ripple and ensure stable operation, removing the need for extra passive components at the output.

Gate Driving Circuit

To be able to drive the MOSFET with a microcontroller, we designed a gate driving circuit with an optocoupled gate driver IC, TLP250, but in our tests, we get problem with it, which results in high output all the time, so we changed it to the HCPL-3120. It is an optocoupled IC with the same pinout except for the output voltage pin. We can see the diagram of the circuit we built in the figure below.

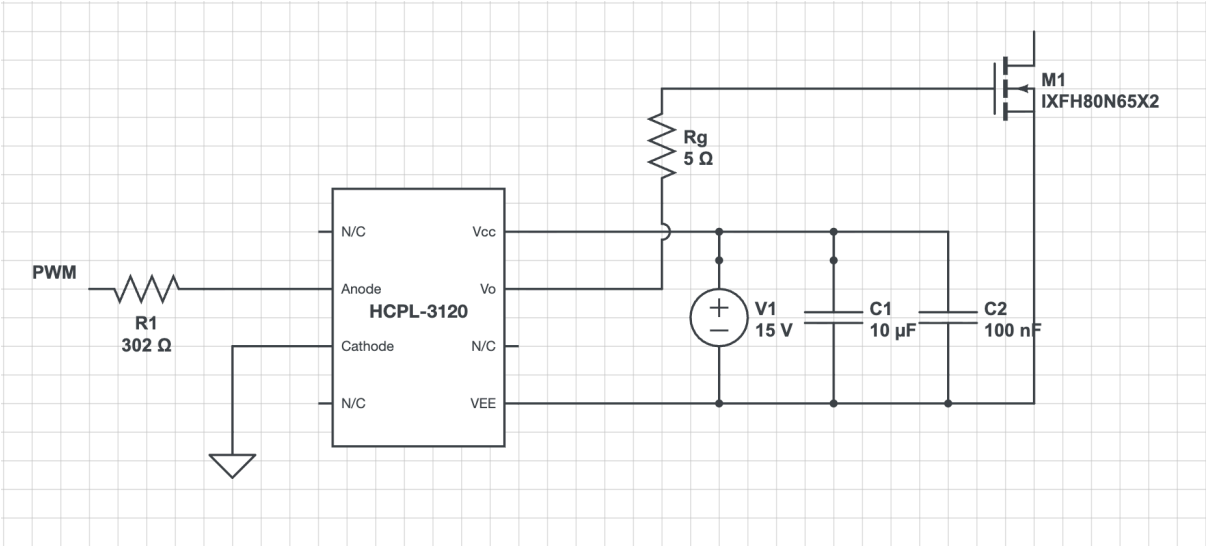

Figure 11: Gate Driver Circuit Diagram

In this circuit, we add an input resistor to the PWM to limit the current. From the datasheet, we know that the input forward voltage is 1.5V and Arduino's output voltage is 5V. Also, we know that

the recommended input current is 7mA with a maximum of 16mA. So by choosing a 302 Ohm resistor, we limit current to this:

$$\frac{5-1.5}{302}=11.58\text{mA}$$

Thus, it protects our microcontroller and gate driving IC by setting the current limit. In output side, we add a 100nF capacitor, which is a supply for the instantaneous current spikes. It prevents Vcc drop and ringing. And we add a 10 microF bulk capacitor for energy storage, which supplies each edge while the voltage supply supplies this capacitor. So it stabilize 15V source. We connect our output to the gate of the MOSFET using a 5 Ohm resistor to limit the peak gate current. And since we have a MOSFET on the high side of the buck converter, we use a floating 15v supply since the source of the mosfet actually not actually ground. We can see our implementation of it in the figure below.

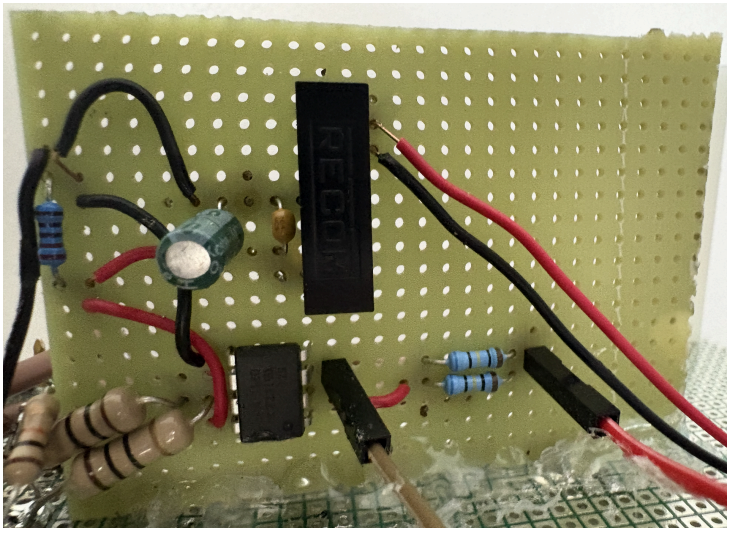

Figure 12: Gate Driving Implementation

Current Sensing

For the current control loop, at first we decided to use ACS758LCB-100U, but this sensor is capable of measuring up to 100A, so in our rating, it has poor resolution, so we switched to the ACS712TELC-30A. This sensor measures up to 30A bi-directional. And at 0 A, it gave Vcc/2 since we supply it with the 5V, we expect to see 2.5V. But when we test it, we measure 2.58-2.61V in 0A, and since each ampere corresponds to 66mV, it actually affects the output so much that we arrange our code according to this offset. Since it is a Hall effect sensor, it is already isolated from the measured circuit, so we directly connect the same voltage supply with the microcontroller. And measure the voltage at the armature from high side output of the buck converter. We can see the practical implementation in our circuit in the figure below.

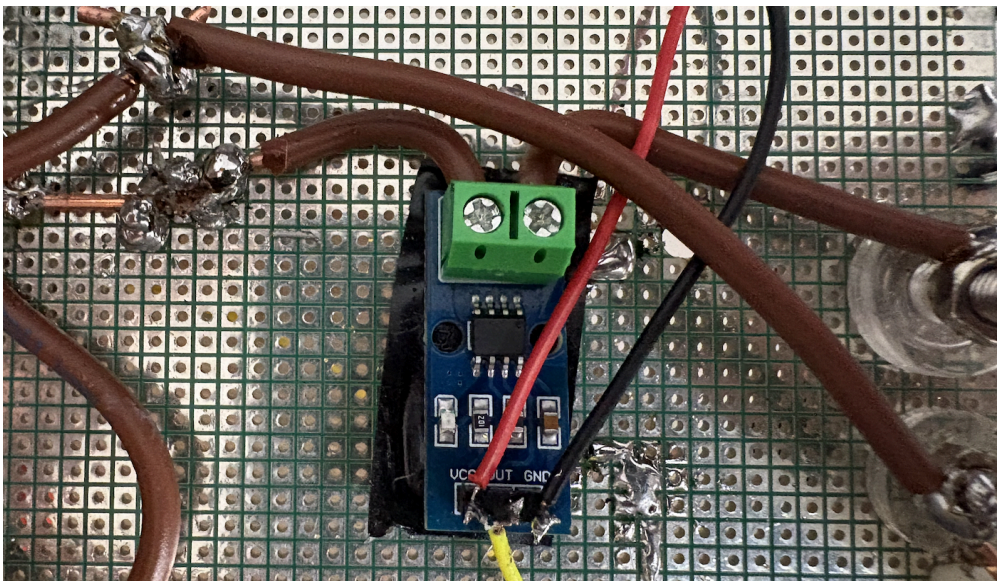

Figure 13: Current Sensing Sensor

Software

Open Loop Software

For regular testing and debugging, we implement an open-loop PWM generator software that generates 1kHz PWM and prints the current sensor readings to the serial monitor. We start by resetting Timer1 and counter 1 to 0. And then we set Timer1 on fast PWM mode, mode 14, with top value equal to the ICR1 register.

```
16    TCCR1A = 0;
17    TCCR1B = 0;
18    TCNT1  = 0;
19
20
21    TCCR1A |= (1 << WGM11);
22    TCCR1B |= (1 << WGM12) | (1 << WGM13);
23
```

Figure 14: Timer Register Code

Then we set the PWM output in pin 9 with non-inverting mode, which means we get high until the counter reaches the OCR1A register. And also we set the precaler of Timer1 as 8, so its frequency becomes 16MHz/8, which is 2MHz.

```
25    TCCR1A |= (1 << COM1A1);
26
27    TCCR1B |= (1 << CS11);
28    ICR1 = 1999;
29    OCR1A = 0000;
```

Figure 15: Timer Pin Selection Code

By setting the top of the counter as 1999, we get pwm frequency of 2MHz/(1+ICR1). Which makes 1KHz. Finally, we set the ADC to a prescale of 64. Thus, we get 250kHz sampling frequency.

```
32      ADCSRA = (ADCSRA & 0b11111000) | 0b110;
33    }
```

Figure 16: ADC Register Code

Then we set the duty with resolution in terms of thousand with this function.

```
35    void setDuty(uint16_t duty) {
36      if (duty > 1000) duty = 1000;
37
38      OCR1A = (uint32_t)duty * (ICR1 + 1) / 1000;
39    }
```

Figure 17: Duty Function

And for soft-start, we increment our duty using this function at the desired time to the desired duty.

```
55    uint16_t softStartDuty() {
56      uint32_t elapsed = millis() - t_start_ms;
57
58      if (elapsed >= SOFTSTART_MS) return DUTY_TARGET_PERMILLE;
59
60      return (uint32_t)DUTY_TARGET_PERMILLE * elapsed / SOFTSTART_MS;
61    }
```

Figure 18: Soft Start Function

So our loop becomes like this for soft-starting a motor at open loop and monitoring the current.

```
63    void loop() {
64      uint16_t duty = softStartDutyPermille();
65      setDutyPermille(duty);
66
67      float adc0 = (2.61f - (((analogRead(ADC_CH0)) * 5.0f) / 1023.0f)) / 0.066f;
68      uint16_t adc1 = analogRead(ADC_CH1);
69
70      static uint32_t lastPrintMs = 0;
71      if (millis() - lastPrintMs >= 20) {
72        lastPrintMs = millis();
73        Serial.print("Duty=");
74        Serial.print(duty);
75        Serial.print("  Current=");
76        Serial.print(adc0);
77
78      }
79    }
```

Figure 19: Open Loop Code

Closed Loop Software

In closed-loop software, we measure the current, and then by using the PI loop, we control its value by changing the duty of the buck converter. We create a PI loop using the C struct. In this PI controller, we also add saturation. It limits the output voltage.

```
47  struct PIController {
48    float Kp = 0;
49    float Ki = 0;
50    float integrator = 0;
51    float out_min = 0;
52    float out_max = 0;
53
54    float update(float err, float Ts) {
55      float u_p = Kp * err;
56      float integ_new = integrator + (Ki * Ts) * err;
57      float u = u_p + integ_new;
58
59      if (u > out_max) {
60        u = out_max;
61        if (err < 0) integrator = integ_new;
62      } else if (u < out_min) {
63        u = out_min;
64        if (err > 0) integrator = integ_new;
65      } else {
66        integrator = integ_new;
67      }
68      return u;
69    }
70  };
71  PIController piI;
```

Figure 20: PI Control Struct

This time to reduce the noise, we took n samples from the ADC and calculated the average of them, then put an IIR lowpass filter on the averaged samples.

```
85  static inline float readAdcVoltageAveraged(uint8_t pin, uint8_t samples) {
86    uint32_t sum = 0;
87    for (uint8_t k = 0; k < samples; k++) {
88      sum += (uint16_t)analogRead(pin);
89    }
90    float adc_avg = (float)sum / (float)samples;
91    return adc_avg * ADC_LSB;
92  }

111 float readArmatureCurrent() {
112   float v_out = readAdcVoltageAveraged(ADC_I_PIN, I_ADC_SAMPLES);
113
114   float i_raw = ((I_OFFSET_V - v_out) / ACS712_SENS_V_PER_A);
115
116   i_filt = (I_FILT_ALPHA * i_filt) + ((1.0f - I_FILT_ALPHA) * i_raw);
117
118   return i_filt;
119 }
```

Figure 21: ADC Sampling and Filtering Functions

In each control step, we get an averaged filtered current measurement, then we compare it with a reference and give an error to the PI function, and it gives us a voltage command, then we give it to the buck as a duty ratio. While doing that, we both give reference to the ramp and also check the voltage command; follow this order for safety. We also print out the necessary values in the serial monitor.

```
155  void controlStep() {
156    float i_meas = readArmatureCurrent();
157    float v_sensor = readAdcVoltageAveraged(ADC_I_PIN, I_ADC_SAMPLES);
158
159    float i_ref_cmd = softStartRamp(i_ref_target);
160    i_ref = i_ref_cmd;
161
162    float i_ref_limited = i_ref_cmd;
163    if (i_ref_limited < 0) i_ref_limited = 0;
164    if (i_ref_limited > I_MAX) i_ref_limited = I_MAX;
165
166    float e_i = i_ref_limited - i_meas;
167    float v_cmd = piI.update(e_i, TS);
168
169    float duty = v_cmd / VBUS_NOM;
170
171    float duty_cap = 0.60f * softStartRamp(1.0f);
172    if (duty > duty_cap) duty = duty_cap;
173
174    pwmSetDuty(duty);
175
176    static uint32_t lastPrintMs = 0;
177    if (millis() - lastPrintMs >= 20) {
178      lastPrintMs = millis();
179      Serial.print("Voff=");   Serial.print(I_OFFSET_V, 3);
180      Serial.print(" Iref=");  Serial.print(i_ref_limited, 2);
181      Serial.print(" I=");     Serial.print(i_meas, 3);
182      Serial.print(" Vcmd=");  Serial.print(v_cmd, 1);
183      Serial.print(" duty=");  Serial.print((float)OCR1A / (ICR1 + 1), 3);
184      Serial.print(" Vsens="); Serial.println(v_sensor, 4);
185    }
186  }
```

Figure 22: Control Function

Demo Results

We performed various tests to verify the operation of our motor driver. The tests included a resistive load (R-Load), motor operation with soft start and rated speed, and the high-power "Tea Bonus" test. The recorded data from our demo session is presented in Table 1.

Table 1: Demo experiment data.

Parameter	**Test Case 1: R-Load**	**Test Case 2: Motor (Soft Start)**	**Test Case 3: Motor (Rated)**	**Test Case 4: Tea Bonus**
Notes	R_{Load} all parallel, 60% duty cycle	Motor with soft start at low voltage	Motor with soft start at rated voltage	Vo = 180V, 5 min test for tea bonus
$V_{in,rms}$ (V)	46.95	68.8	185	196
$P_{in,rms}$ (W)	133	103	447	1450
$V_{out,avg}$ (V)	37	55.26	177	180
$P_{out,avg}$ (W)	77	92	391	1380
Efficiency (%)	57.89	89.32	87.47	95.17

Test Analysis

- **R-Load Test:** We first tested the topology with a resistive load (all resistors in parallel) at a constant 60% duty cycle. The system worked stably, and we measured an output voltage of 37 V.
- **Motor Tests:** We tested the DC motor under soft-start and rated conditions. In the rated test (Test Case 3), we supplied 447 W of input power and obtained 391 W of output power. This shows that our buck converter is transferring power effectively to the motor with reasonable efficiency.
- **Tea Bonus (High Power Test):** For the bonus requirement, we adjusted the input voltage to approximately 196 V AC using a variac. The controller successfully regulated the output voltage to the target 180 V. During this test, we achieved a continuous output power of 1380 W. The system ran for 5 minutes without thermal failure. Since the losses do not increase too much compared to the output power, increasing output power causes better efficiency compared to the other cases.

LOSS CALCULATIONS

In this section, power losses of critical components were calculated to verify the thermal stability of the designed power electronics circuit and to analyze the adequacy of the passive cooling system. The "Tea Bonus" test conditions (1380 W output power, 180 V output voltage), where the system operates continuously at its highest power and therefore experiences maximum thermal stress, were used as a reference for the analysis.

1. 3-Phase Full Bridge Rectifier (VUO98-16NO7) Loss and Thermal Calculations:

The input current drawn by the system (approximately 5 A) is quite low compared to the nominal current capacity of the rectifier module used (105 A). Therefore, it is predicted that the conduction and switching losses on the rectifier will be negligible compared to the total power. In line with this 'low loss' assumption, the IDC current was calculated assuming that the input and output powers are approximately equal.

$$Vdc = 1.35\,x\,196\,V = 265\,V \qquad Idc = \frac{Pin}{Vdc} = 5.48\,A$$

$$Iavg = \frac{Idc}{3} = 1.83\,A \qquad Irms = \frac{Idc}{\sqrt{3}} = 3.16\,A$$

$$Pdiode = (Vt0\ x\,Iavg) + (rt\,x\,I^2rms) = 1.54\,W$$

$$Prect,total = 6\,x\,Pdiode = 9.24\,W$$

$$Tj = Ta + Ploss(RthJC + R\Theta C + R\Theta A) = 25 + 9.24\,W\,x(0.7 + 0.5 + 5)\,K/W = 82.28°\,C$$

The rectifier is working at safe conditions even without any fan.

2. MOSFET Loss and Thermal Calculations:

The IXFH80N65X2 N-Channel Power MOSFET was selected as the main switching element of the Buck converter stage due to its high current capacity and thermal stability. The power loss analysis performed in this section is based on the worst-case loading scenario where the system draws 1450 W of power from the grid. To increase the reliability of the analysis, the coefficient of increase of the MOSFET's conduction resistance R_DS(on) at high temperatures (125°C) was taken into account in the calculations.

$$D = \frac{Vout}{Vdc} = 0.68 \qquad Irms,mos = \frac{Idc}{\sqrt{D}} = 6.65\,A$$

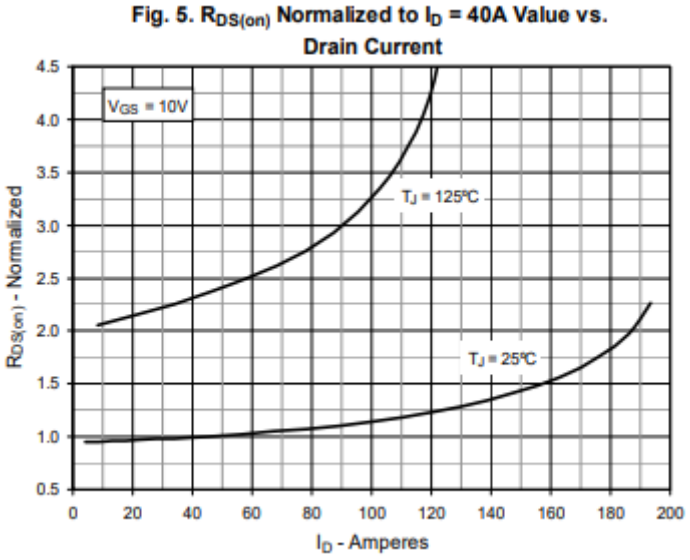

Figure 23: Rds(on) vs ID and Tj graph.

$$Rds, on(at\ 125\ C\ and\ 6.65\ A)\ =\ 2\ x\ Rds, on\ =\ 0.076\ ohm$$

$$Pcond\ =\ I^2 rms, mos\ x\ 0.076\ =\ 3.36\ W \qquad Ipeak\ =\ Idc/D\ =\ 8.06\ A$$

$$Psw\ =\ 0.5\ x\ Vdc\ x\ Ipeak\ x\ (ton\ +\ toff)\ x\ fsw\ =\ 0.5\ x\ 265\ x\ 8.06\ x\ (32\ +\ 70)\ x\ 10^{-9} x 10^3\ =\ 0.11\ W$$

$$Ptotal\ =\ 3.47\ W$$

$$Tj\ =\ Ta\ +\ Ptotal(RthJC\ +\ R\Theta C\ +\ R\Theta A)\ =\ 25\ +\ 3.47(0.14\ +\ 0.5\ +\ 10)\ =\ 61.9\ ^\circ C$$

The primary reason for the calculated low temperature increase is the minimization of switching losses by following a low switching frequency strategy. The rectifier is working at safe conditions even without any fan.

3. Buck Converter Diode Loss and Thermal Calculations (DSEI30-06A):

In the buck converter circuit, a DSEI30-06A Fast Recovery Epitaxial Diode was used as the freewheeling diode, ensuring the continuity of the coil current when the MOSFET goes off. In the analysis, the diode conduction and switching losses were calculated for "Worst-Case" conditions, based on a 1450 W input power drawn from the grid and a 1 kHz switching frequency.

$$Iavg, diode\ =\ Ipeak\ x\ (1\ -\ D)\ =\ 2.58\ A$$

$$Pcond\ =\ Iavg, diode\ x\ Vf\ =\ 3.23\ W$$

$$Psw\ =\ 0.5\ x\ Vdc\ x\ Ipeak\ x\ trr\ x\ fsw\ =\ 0.04\ W$$

$$Ptotal\ =\ 3.23\ +\ 0.04\ =\ 3.27\ W$$

$$Tj = Ta + Ptotal(R\Theta JC + R\Theta C + R\Theta A) = 25 + 3.27(1 + 0.5 + 10) = 62.6\,^{\circ}C$$

The calculated junction temperature is well below the diode's maximum operating temperature of 150°C. This result confirms that the passive cooling strategy is also extremely safe for the diode.

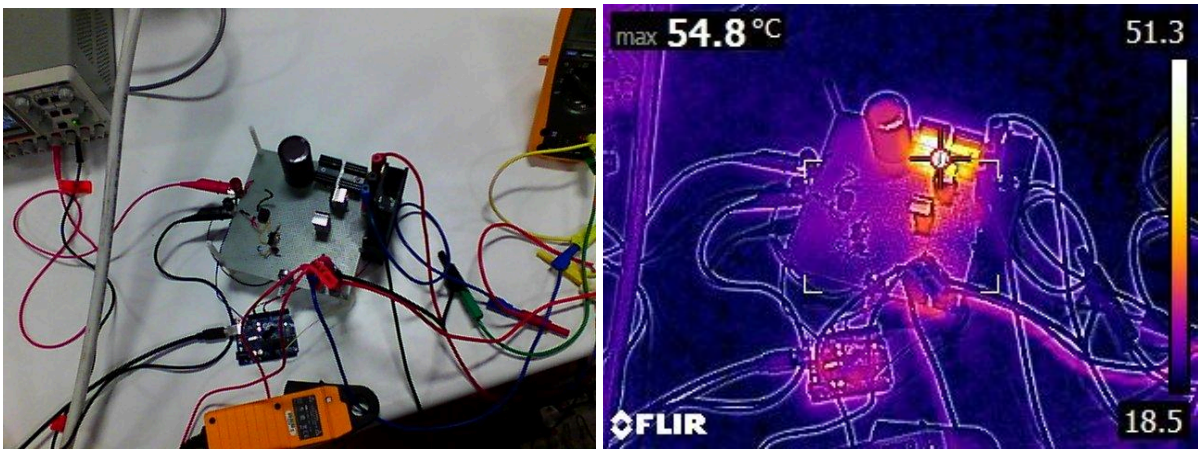

Figure 24: Thermal Results of Tea Bonus part of the project.

Theoretical analyses performed under the worst-case loading conditions (1450 W input power) predicted that the highest thermal stress in the system would occur in the Rectifier module, and that passive cooling would raise the temperature to 82.3°C. Thermal camera measurements performed with the addition of a fan confirmed this prediction; it was observed that the Rectifier was still the hottest component of the system as expected, but thanks to active cooling, the temperature was limited to 54.8°C. This result, approximately 27°C lower than the calculated theoretical value, is well below the component's 150°C safety limit. Thus, it has been proven that the implemented cooling strategy (heatsink + fan) keeps the system in a safe operating zone even under maximum load.

CONCLUSION

In conclusion, we successfully implemented and tested a DC motor driver using a 3-Phase Rectifier and Buck Converter topology. Throughout the process, we addressed practical hardware challenges by replacing the initial gate driver with the HCPL-3120 for stability and switching to the ACS712-30A current sensor to improve measurement resolution. We also validated our design choice to rely on the motor’s internal inductance for filtering, eliminating the need for an external LC circuit. The final experimental results confirmed effective closed-loop control, and the system successfully achieved the "Tea Bonus" by delivering 1380 W at a regulated 180 V for five minutes without any thermal issues