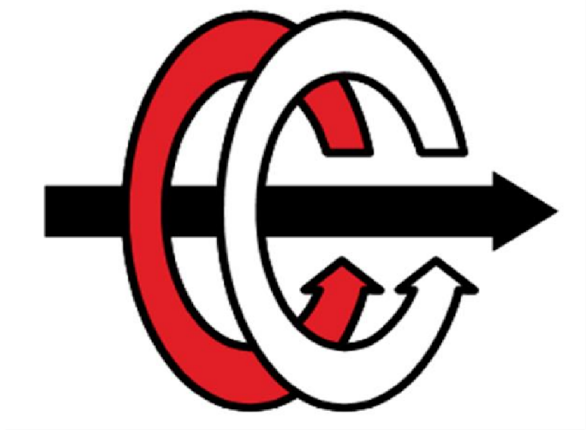
MIDDLE EAST TECHNICAL UNIVERSITY



Department of Electrical and Electronics Engineering

EE463 Hardware Project Report

AC to DC Motor Drive

Mehmet Elen
Mert Elmas
Yusuf Selim Karataş

Table of Contents

1) Introduction 3
2) Project Definition 3
3) The Solution 5
    a) Simulink Design and Test Results 5
    b) Theoretical Analysis 7
    c) Component Selection 9
    d) Demonstration Results 9
    e) Obstacles & Shortcomings 15
        i) Problems with the first mosfet 15
            - Heating 15
            - Limitations in the frequency of the PWM output of Arduino 15
        ii) Shortcomings 15
            - 4 quadrant operation 16
            - Smoothness of the design 16
4) Conclusion 17
5) References 17
6) Appendices 17
    a) Arduino Code 17
    b) Link to the video 19

1. Introduction

This report describes the implementation of an AC to DC Motor drive. It includes simulink design for our solution, the theoretical analysis for the design, component selection and the results obtained during the demonstration process. As source, 400Vl-l, 50 Hz AC source is used whose amplitude is then adjusted by the variac to obtain the desired ac voltage which is 220 V_{rms}. Later this AC voltage is rectified and its amplitude is changed to a desired value through a Buck converter via changing the duty cycle of the switching.By doing so we obtain a DC motor with controllable speed. In the following parts this process is narrated with more detail.

2. Project Definition

In this project, we were required to make a controlled rectifier that will be used to drive a DC Motor with the below mentioned specifications.

- Power Input: 3 Phase, or 1 Phase AC Grid (Adjustable with variac)
- Output: Adjustable DC Output (Vmax> 180 Vdc)

Possible Topologies given as example:

- 3-Phase Thyristor Rectifier
- 1-Phase Thyristor Rectifier
- Diode Rectifier + Buck Converter

Motor Specs:

You are required to drive the following motor(on the left). The motor will be loaded with the generator(on the right) coupled to the generator:

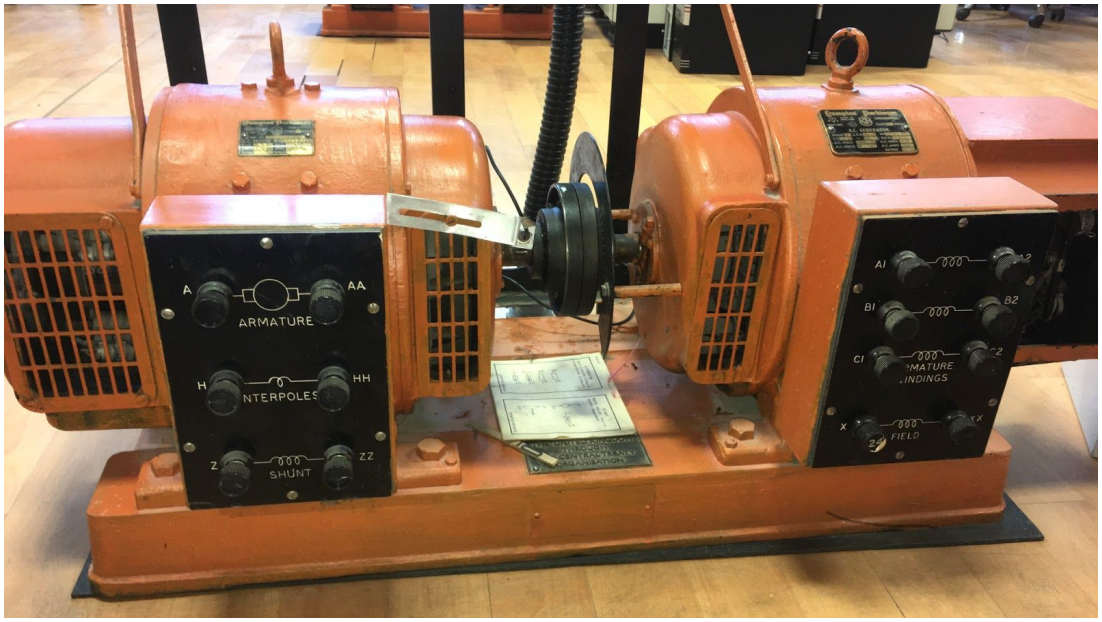


Figure 1 : Motor to be driven and its coupled generator that will be the power supply for the load, the kettle.

Figure 2: Specification plaque for the shunt motor

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

A separate external DC source to feed the field winding (i.e. separately excited DC machine) will be provided in the project . However, it is possible to make other types of connections such as shunt, series or compound.

The variac can be used to gradually apply AC voltage to the drive, and left to be at any value, but can not be used to control voltage during operation.

3. The Solution

Firstly, we converted the AC voltage with a single phase diode bridge rectifier to a DC one so that it would be possible to adjust its amplitude without adjusting the knob of the variac but with just a manual control of the potentiometer which adjusts the pwm of the buck converter that we have designed by an arduino. Later we feed this adjusted dc like voltage to the dc machine which is coupled with a generator which then eventually supplies power to 2kW load , the kettle. The reason for calling it a “dc-like” to this voltage is because of the ripples which could not be smoothed out by the capacitor and the inductor in the buck converter. These ripples are also can be diminished by increasing the frequency of the switching.

a. Simulink Design and Test Results

We were concerned about the controllability of the thyristor rectifiers therefore the diode rectifier-Buck converter topology is used as the design solution. In figure 3 , the simulink schematic can be seen.

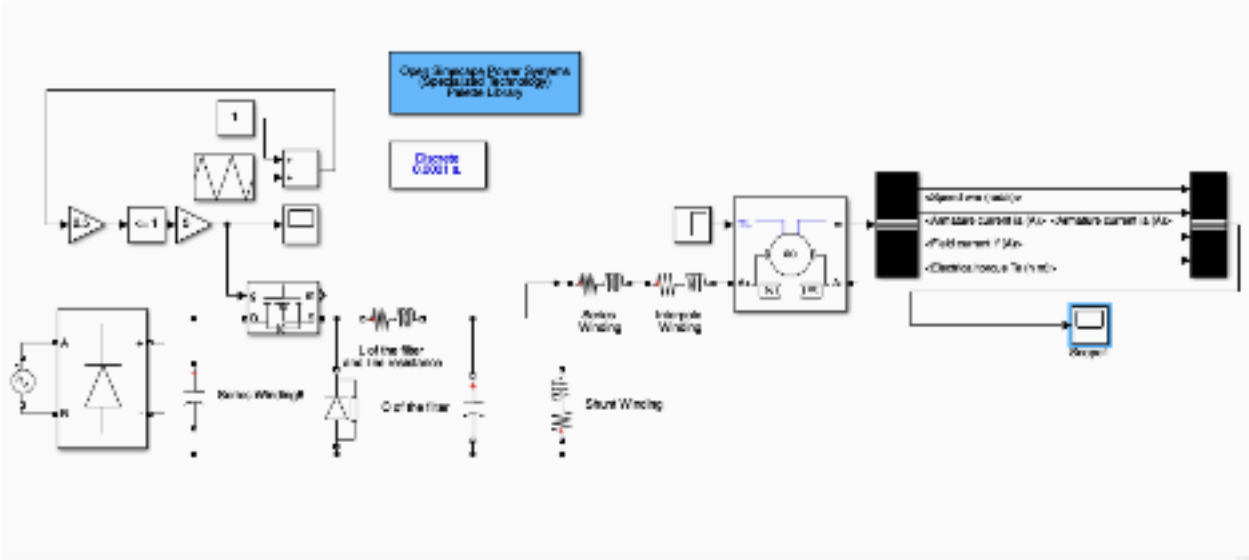


Figure 3:

In figure 3, one can observe the AC to DC converter accomplished by diode rectifier, the switch by mosfet, filter part of the buck converter and the dc machine. PWM are created as follows. First an offset of magnitude 1 is added to a triangular wave which was previously -1 to 1 and then multiplied by 2.5 to obtain a 0 to 5 triangular wave. Later this voltage is compared to 1 and the output of this comparison is multiplied with a gain between 0 to 5 which perfectly models the voltage division done by the potentiometer. In accordance with this value divided by 5 the desired duty cycle is obtained

The graphical results of this topology is as follows:

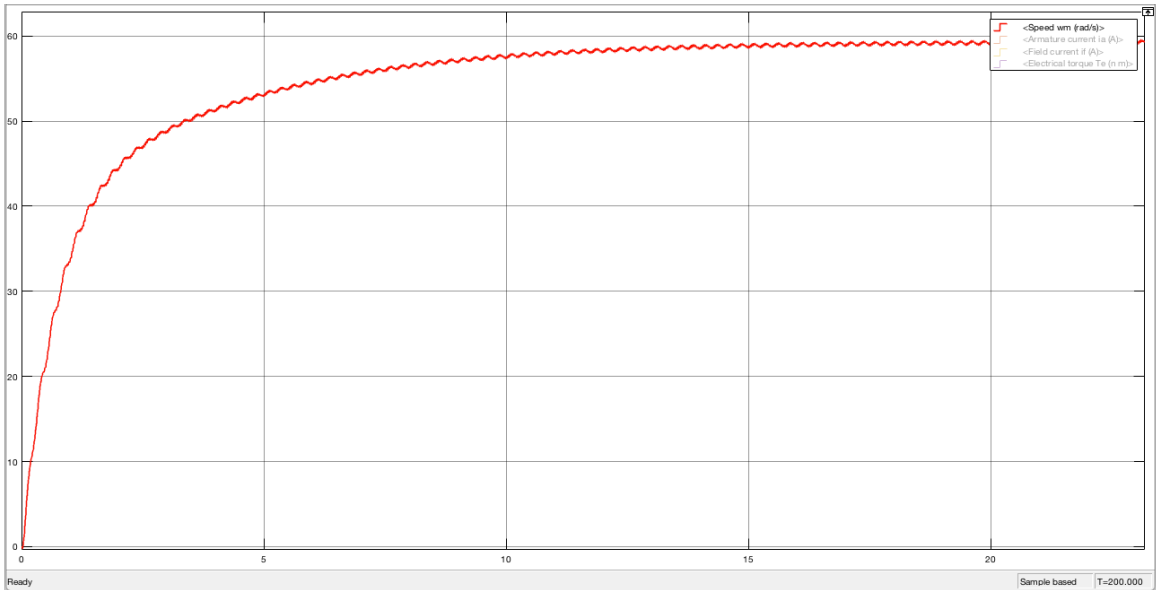


Figure 4: Graphical analysis of speed of the motor in rad/s

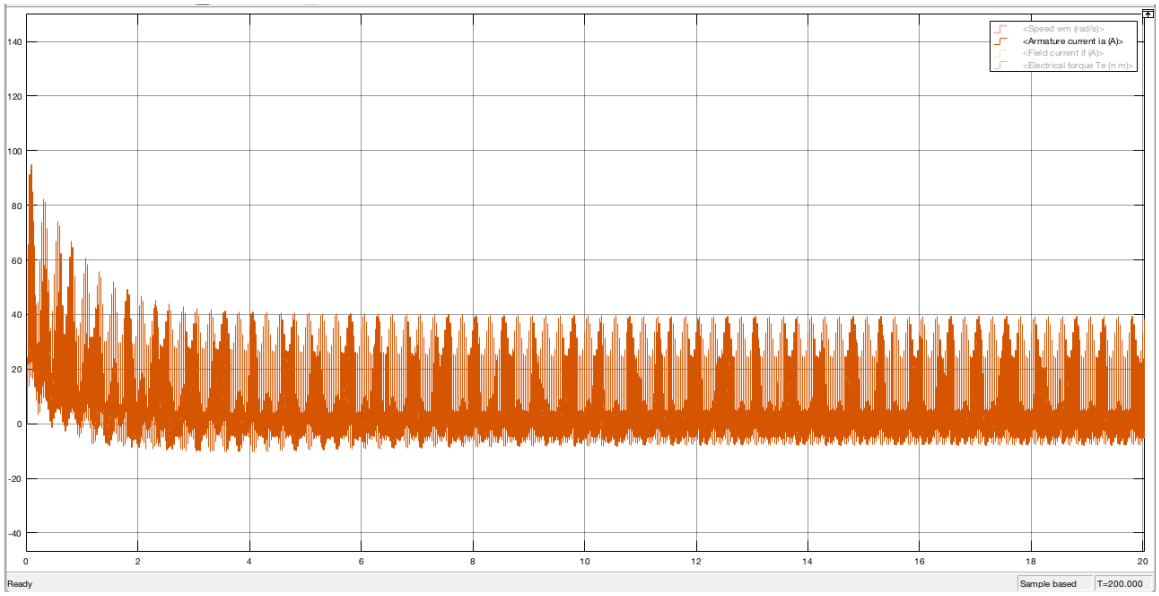


Figure 5: Graphical analysis of the armature current of the motor in Amps

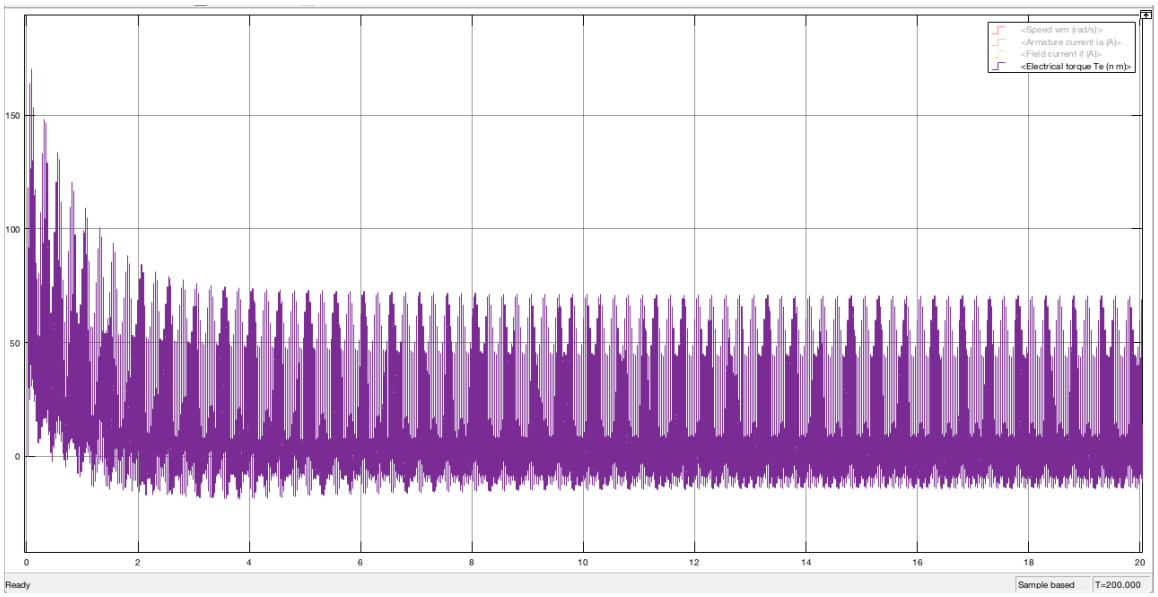


Figure 6: Graphical analysis of electrical torque of the motor in N.m

The relationship between the torque and the armature current which is $T = I_a \, x \, k_t$ can be verified by comparing figure 5 and figure 6 . As it is seen, they have a strong resemblance with each other. Also it is noticeable that the speed also has ripples

b. Theoretical Analysis

The main idea behind this AC to DC motor drive project lies on the DC to DC converter which in our case it is the Buck converter. In figure … , the general schematic for buck converter with passive Low-pass filter can be seen.

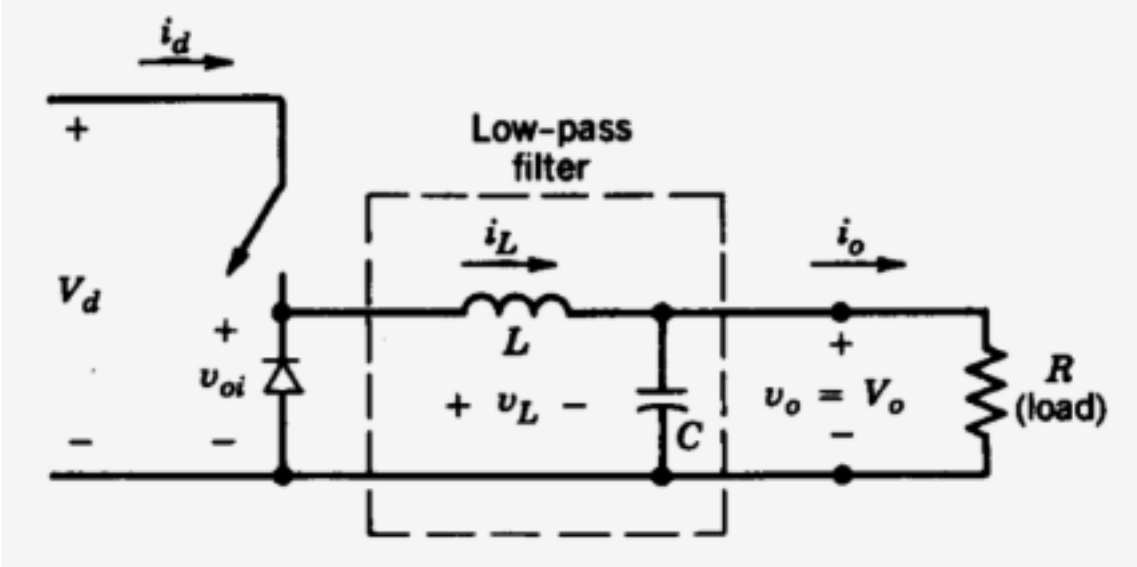


Figure 7:

It is pretty straight forward from this point on. As the switch opens and closes the capacitor is charged but not as high as the source voltage, Vd .Therefore the relationship between the output voltage V_0 and V_d is obtained as follows: $V_0 = D*V_d$ where D is the duty cycle of the pulse generated by the switch and hence we obtain a controllable DC output.

The input for the Buck converter is generated by rectifying the AC grid voltage to a DC voltage with a single phase diode rectifier. The general schematic and the expected output voltage with an inductive load for a single phase diode rectifier can be seen in figure… .

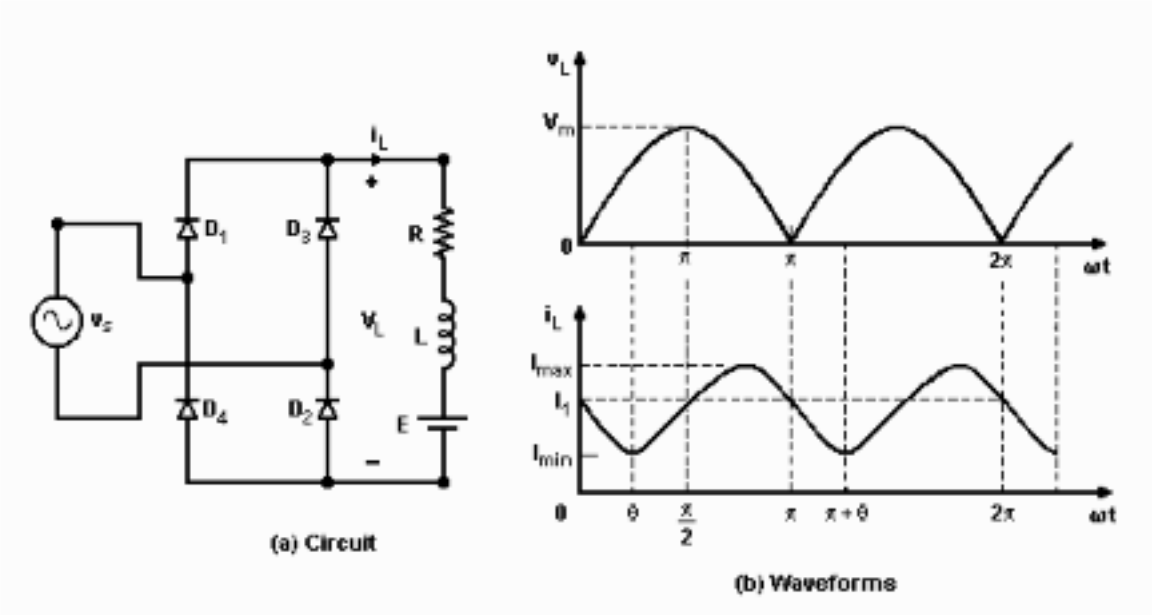


Figure 8: Schematics for single phase diode bridge rectifier and its output voltage and output current graphs.

From figure 8 it can be seen that this topology would not yield a pure dc output but rather a waveform with ripples. These ripples are further smoothed out by the filter of the Buck converter.

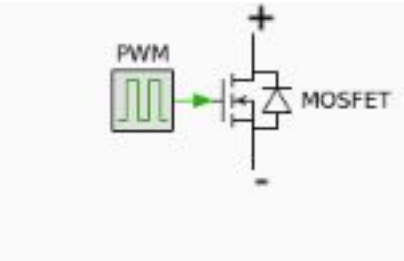


Figure 9: Simplified model of electrical switch done by an N-MOS.

Figure 9 shows a simple model for NMOS switching. As the PWM is high the mofet will allow current to pass and this will in turn feed the Buck circuit. The average output voltage depends on the duty cycle of this PWM.

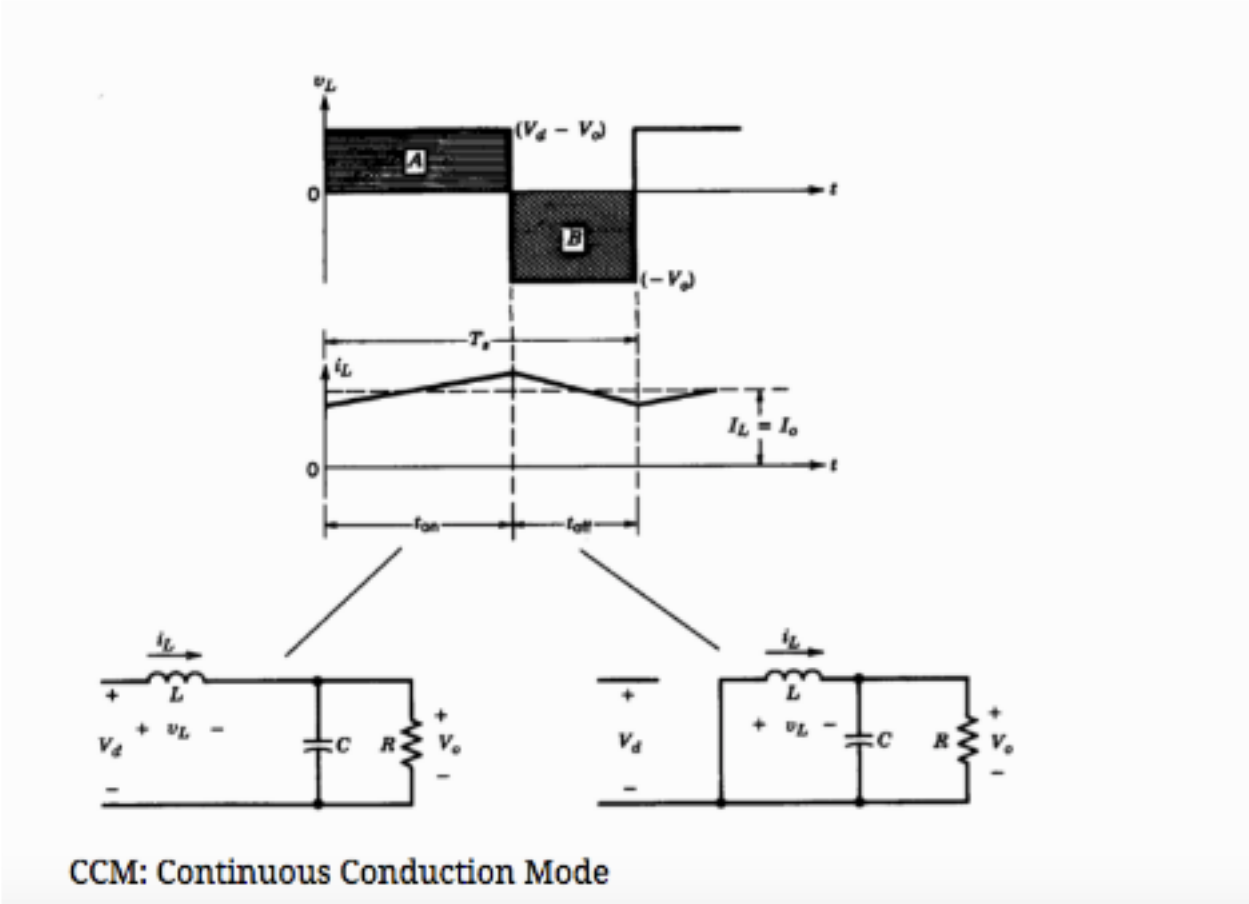


Figure 10: Operation modes of the buck converter for continuous conduction mode.

c. Component Selection

The determining factors in component selection were ripple reducing capability of the RLC Low-pass filter, high current durability of the switching component, MOSFET and isolation between low voltage and high voltage sides which is accomplished with the optocoupler.

-450V, 330uF capacitor
-400V, 400uF capacitor
-10uF capacitor
-IXTQ44N50P MOSFET
-HER3006PT diode
-10mH inductor
-HCPL 3120 optocoupler
-ARDUINO UNO
-15 V LED DRIVER
-Various resistors

d. Demonstration Results

At the first step of the driver circuit, Ac voltage is half-rectified with a bridge rectifier. Bridge rectifier reverses negative cycle of the AC input voltage. Connected to the grid input with a variac, input and output voltage waveforms are given in the **Figure XX.** There is not a significant voltage drop over diodes in bridge rectifier.

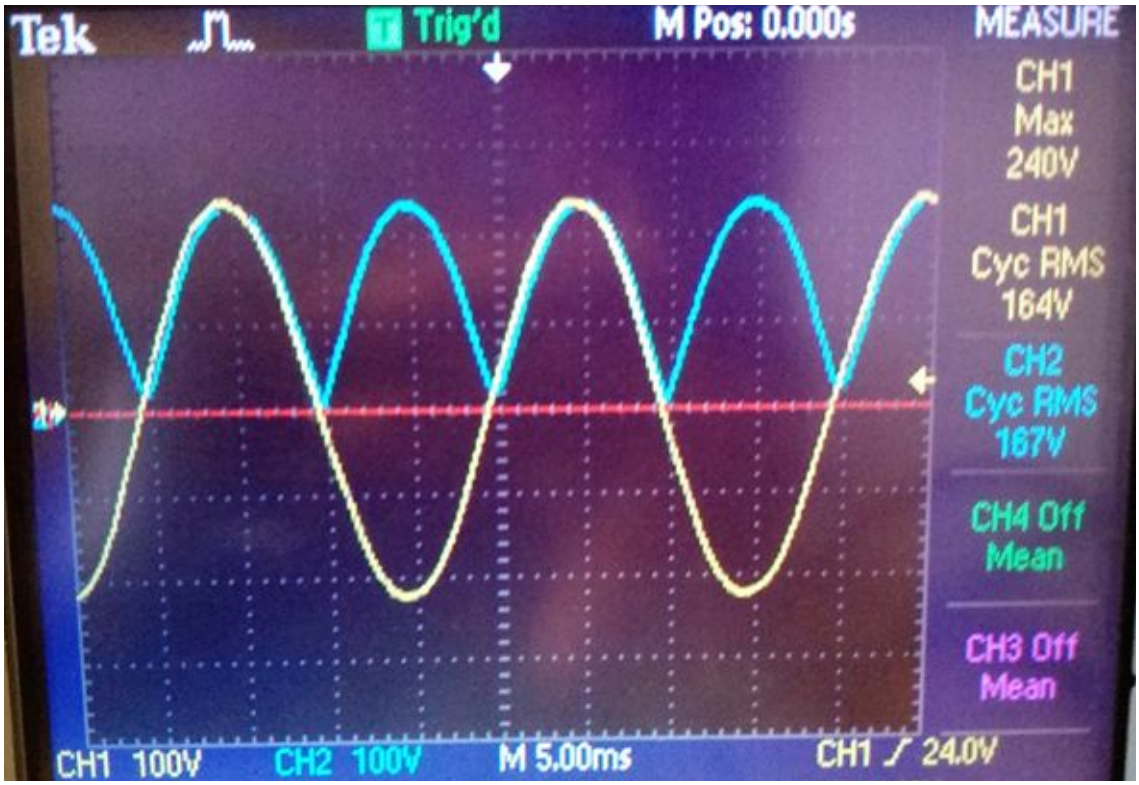


Figure 1. Input and Output Voltage Waveforms of Bridge Rectifier

PWM is produced to drive MOSFET of the buck convertor. To obtain DC Voltage value at the output as desired, a potentiometer is used to change PWM. Change in PWM effects the Duty cycle of the buck convertor and changes input voltage to output voltage ratio, which is already Duty cycle.

PWM is obtained by sensing voltage across the potentiometer with Arduino Uno. According to sensed voltage, Duty cycle of PWM is decided by microcontroller and produced in corresponding Pin.

Arduino produces PWM around 500 Hz which is not a good frequency for switching since it may cause ripple in the current. While designing LC Filter it was planned to be around 15 kHz, but it was discovered that Arduino Uno can deliver PWM just only at several frequencies. Frequency of switching is changed to 1, 4, 8, 32, 64 kHz and optimum value is observed as 32 kHz.

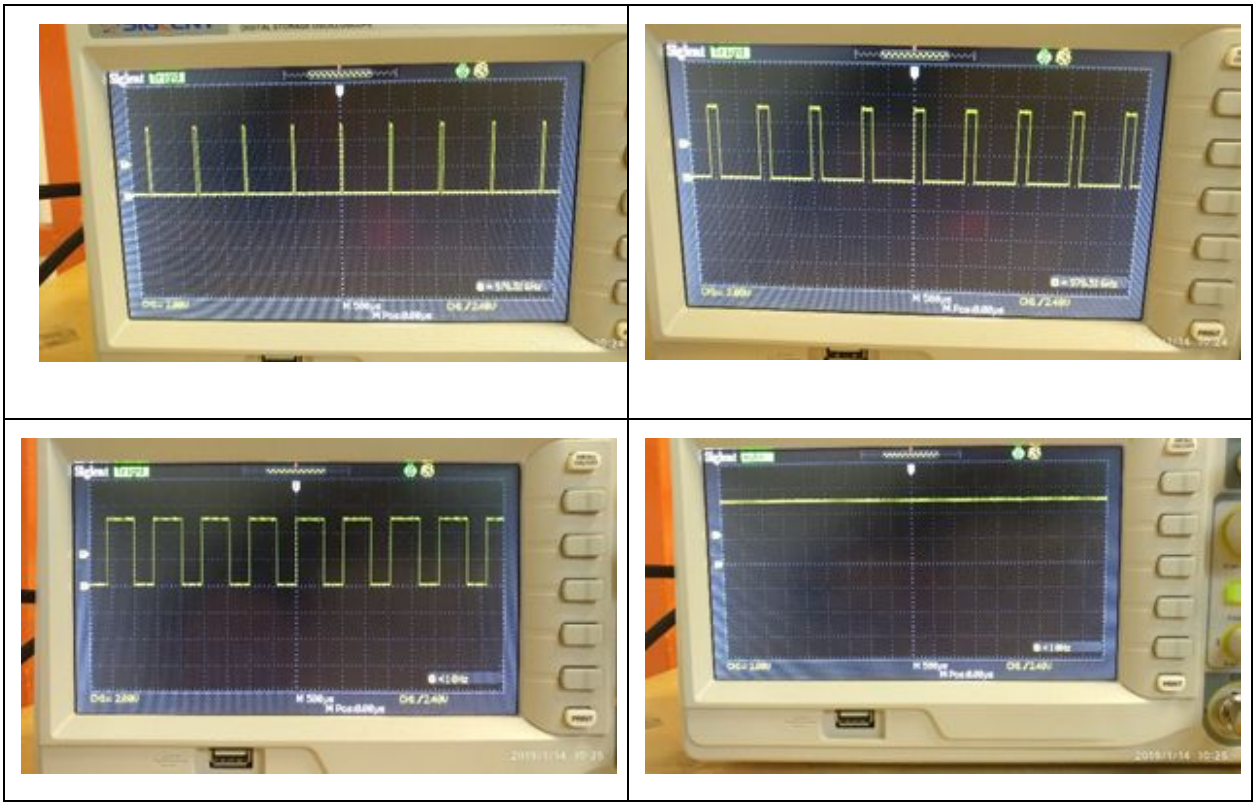


Figure 2. Observed PWM waveforms at oscilloscope at 1 kHz for several Duty Cycle

Duty Cycle = 0	**Duty Cycle = 0.2**
Duty Cycle = 0.7	**Duty Cycle = 1**

Table 1. Corresponding Duty Cycles of the Figure XX

At 1 kHz frequency, at several Duty cycles PWM are obtained with Arduino and observed at oscilloscope. Observed waveforms are given in **Figure XX:** When duty cycle is zero, obtained voltage is 0 DCV; when duty cycle is 1, obtained voltage is 5 VDC.

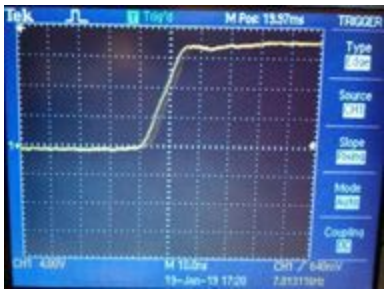


Figure 3. Increasing edge of the PWM Cycle

MOSFET which is used in this project for switching purposes, is the medium which power and small signal data crosses. To prevent any undesired result, optocoupler is used which is one of the galvanic isolation methods. Obtained PWM waveforms are given in **Figure XX** for several Duty Cycle.

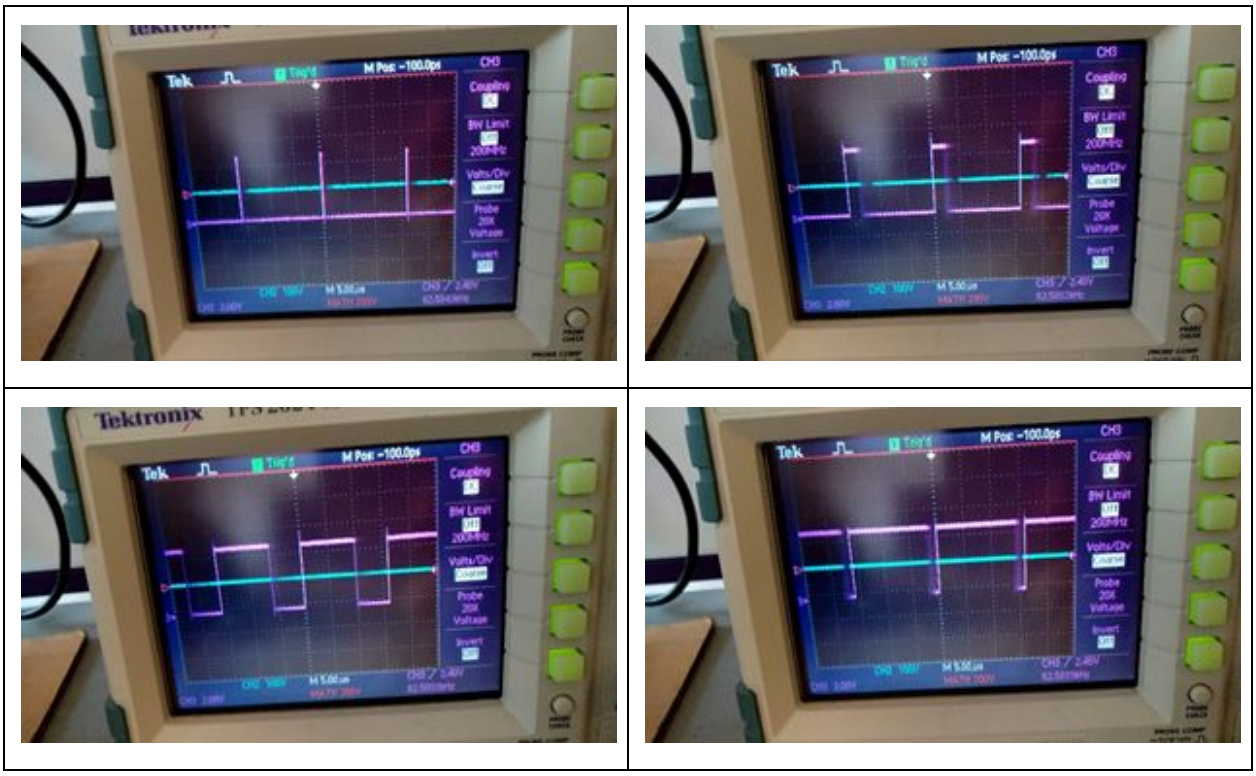


Figure 4. PWM waveform at the output of the optocoupler for several Duty cycle values

Bridge Rectifier and Buck Converter has been combined and AC/DC conversion has been done. Since in this project, ripple in voltage and current is not considered a lot, buck convertor components are selected depending on the other factors. Output voltage of buck convertor is given in **Figure XX** as yellow waveform. As seen in the figure, output voltage reaches to the peak at approximately 235 Volts and makes minimum at 200 Volts when duty cycle is 1. Voltage ripple is about 35 Volts which is a good value for this project. Current ripple at load is about 2 Amperes. Since motor have a high inertia, voltage ripple will be diminished.

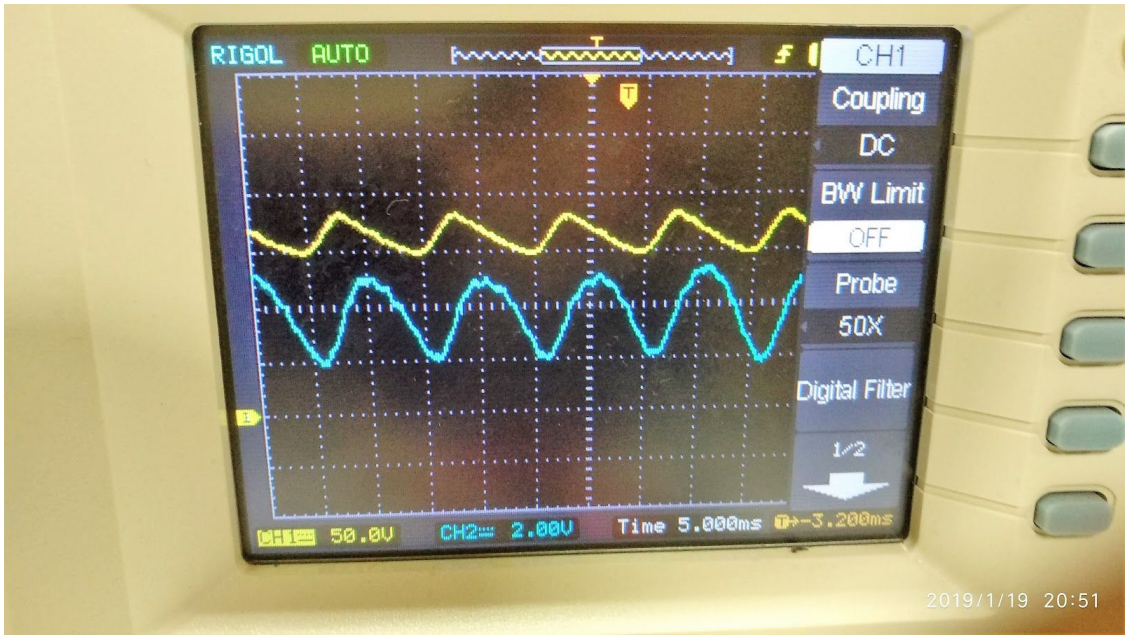


Figure 5. Voltage and current ripple at the output of the convertor

As final stage of the project, we have plugged in a Kettle to a generator which is coupled to the motor we have driven and we have managed to boil water with kettle. When we have plugged the kettle, output voltage of the convertor decreased and current increased significantly and reached to 10 Amperes. Thus, we have proven that our convertor can operate for 2 kW systems. Ripples in waveforms are given in **Figure XX.**

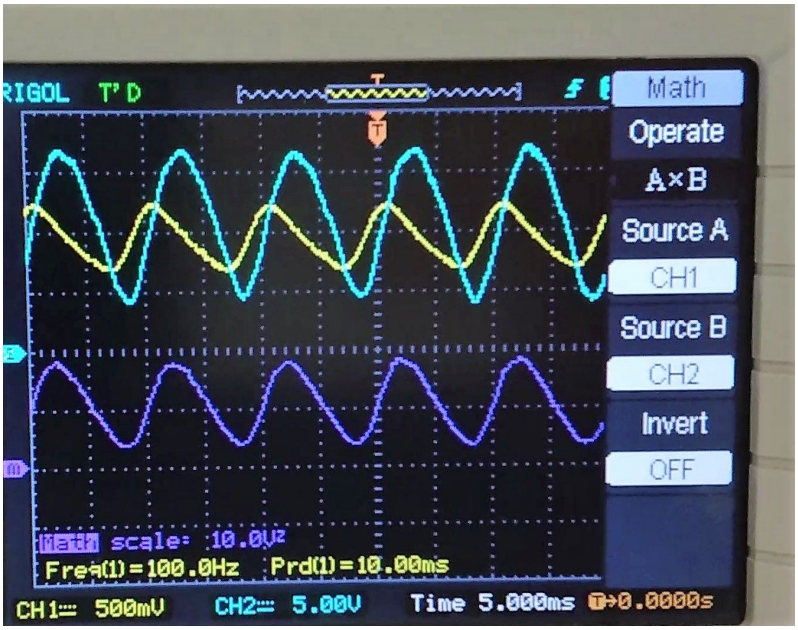


Figure 6.Voltage and current ripple at the output of the convertor while supplying 2 kW

Figure 7.Heat Dissipation on the MOSFET

During boiling operation, current has increased significantly and thus, temperature of the convertor changed. Temperature of the convertor is observed with a thermal camera and obtained data is given in the Figure 7. It is seen that temperature of the MOSFET reaches to 50 C° and temperature of the bridge rectifier reaches to 80 °C.

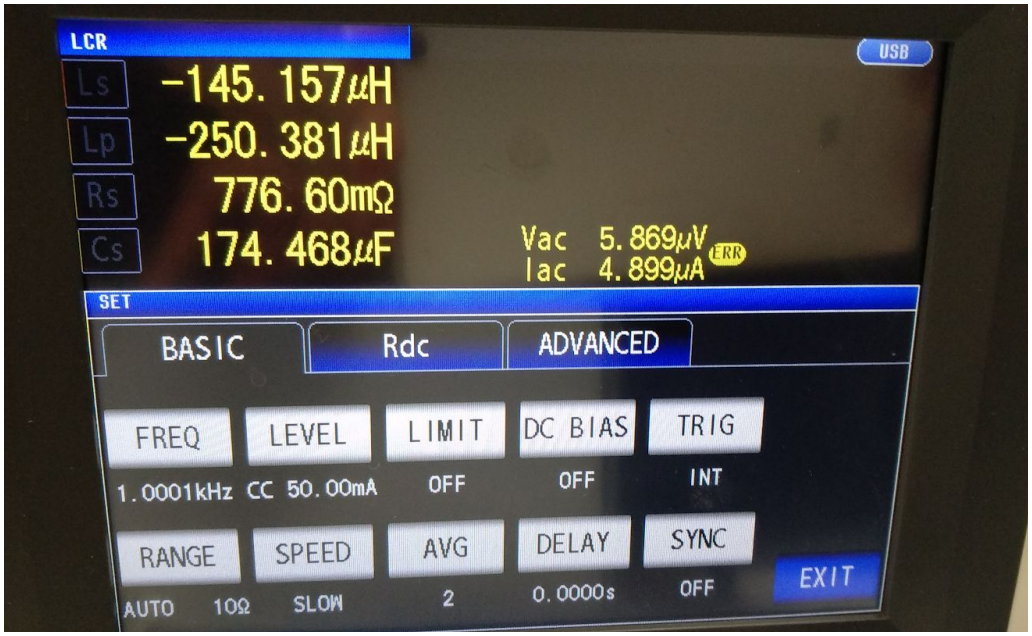


Figure 8.Inductor measurements

e. Obstacles & Shortcomings

i. Problems with the first mosfet

- Heating

Our first trials with the … mosfet model, created excessive heating problem and hence we needed powerful fans to cool down the devices temperature before it blows up considering the thermal camera would only show more than couple degrees below the internal temperature of the MOSFET. Therefore we replace this model with … since it has a higher current carrying capability.

- Limitations in the frequency of the PWM output of Arduino

Inductor in the Buck converter designed for switching frequencies between 10 to 15 kHz range. However, the only arduino code that we were able to find was only capable of generating PWM of 0.5, 1, 4, 8, 32, 64 khz adjustable by a potentiometer. Therefore it was down to choosing between 8kHz and 32 kHz and 32kHz option was chosen rather than 8kHz. The reason for this was at higher frequencies

ii. Shortcomings

- 4 quadrant operation

We were really hoping that we could operate the drive on four quadrants. However, due to our inability to figure out how to overcome conceptual problem with our H-bridge design, we failed to meet this requirement. In figue ... the design for the H-bridge can be seen. It is obvious that ..... .

- Smoothness of the design

Due to the inefficient planning and negligence, we forgot to buy a container box and our design was rather messy. We implemented our buck converter and incomplete H-bridge on a pcb* board but arduino, cooling fans, and the driver hung around and barely mounted on the table with black tape as can be seen in figure 9 .

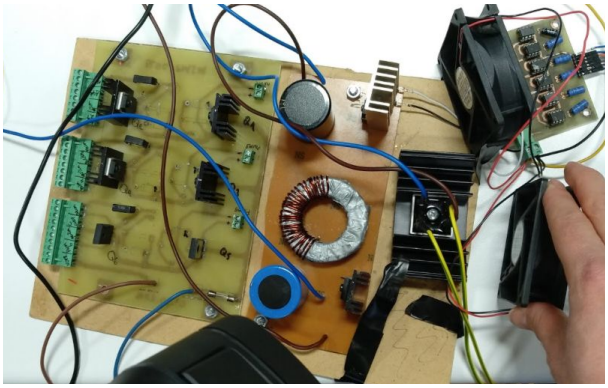


Figure 9: Picture of the Buck converter, the switching mosfet, the optocoupler, the incomplete H-bridge and the fans.

4. Conclusion

In conclusion, this report illustrates the procedure done throughout the project. It details the analysis of the problem, the appropriate solution to counter that problem and implementation of it. It also includes the obstacles that we faced such as heating, current incapability, limitations in Arduino,

The overall gain that we have obtained from this project is that we learned how to make an industrially suitable and a robust design and applied it in our project. We had fun in this project and formed formidable friendships. It provided us with important knowledge to be used in days to come of our lives as engineers such as using the right components and the importance of the neatness. Throughout this project process, we have observed, unlike our team, many of our friends struggled with multiple aspects of their designs whether it was choosing the right components our finding out their right operation conditions. So, not only from our mistakes but also from other's mistakes helped us great deal with the understanding of the motor drives, power electronics mosfets, igbts and thyristors. We thank them and thank you dearly.

5. References

http://keysan.me/ee463/
https://www.plexim.com/academy/power-electronics/buck-conv
http://www.eng.uwi.tt/depts/elec/staff/rdefour/ee33d/s3_fwrc2.html

6. Appendices

a. Arduino Code

```
#include <PID_v1.h>
//----------------------------------------PIN NUMBER ASSIGNMENT----------------------------------------//
const byte BUTON1=2;  //REFERENCE SPEED BUTTON
const byte BUTON2=3;  //ROTATION CHANGE BUTTON

const int EMF=A5;    //VOLTAGE SENSE PIN

const int PWM0=5;    //BUCK CONVERTOR PWM
const int PWM1=6;    //H-BRIDGE PWM-1
const int PWM2=9;    //H-BRIDGE PWM-2
const int PWM3=10;   //H-BRIDGE PWM-3
const int PWM4=11;   //H-BRIDGE PWM-4

//----------------------------------------VARIABLE ASSIGNMENT----------------------------------------//
long sayac=0;

int rot_sayac=0;
bool clockwise_rotation= true;
const int rot_speed[1][7]={{500,600,750,1000,1200,1400,1500}};
double ref_speed=500;
double measured_speed=0;
double measured_value=0;
double measured_voltage=0;
double volt_ratio=0.008;         //UPDATE
double volt_rpm_ratio=50;        //UPDATE
double real_voltage;
double DUTY;
```

```
double PID_OUT;
double Error;
double ref;
double mea;

double Kp=100,Ki=0.1,Kd=0;
PID pid(&Error,&PID_OUT,&ref,Kp,Ki,Kd,DIRECT);

//-------------------------------------------SETUP-----------------------------------------------------//
void setup() {
  // put your setup code here, to run once:

  pinMode(BUTON1,INPUT_PULLUP);
  pinMode(BUTON2,INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(BUTON1),change_ref_speed,RISING);
  attachInterrupt(digitalPinToInterrupt(BUTON2),change_rotation,RISING);

  pinMode(EMF,INPUT);
  pinMode(PWM0,OUTPUT);
  //62 kHz   https://playground.arduino.cc/Main/TimerPWMCheatsheet
  TCCR0A = _BV(COM0A1) | _BV(COM0B1) | _BV(WGM01) | _BV(WGM00);
  TCCR0B = _BV(CS00);
  pinMode(PWM1,OUTPUT);
  pinMode(PWM2,OUTPUT);
  pinMode(PWM3,OUTPUT);
  pinMode(PWM4,OUTPUT);

  pid.SetMode(AUTOMATIC);
  pid.SetTunings(Kp,Ki,Kd);
  pid.SetOutputLimits(0,255);


  Serial.begin(9600);

}

void change_ref_speed() //int *rotsayac,int *rot_speed[1][7]
{
  rot_sayac=rot_sayac%6;
  rot_sayac=rot_sayac+1;
  ref_speed=rot_speed[0][rot_sayac];

}
void change_rotation()
{
  ref_speed=0;    //STOP ROTATION
  while(real_voltage>10)
  {
    Serial.println("HIZIN AZALMASI BEKLENIYOR...");
    delay(100);
  }
  clockwise_rotation= true or clockwise_rotation;  //TOGGLE
  ref_speed=rot_speed[0][rot_sayac];

}

double meas_speed()
{
   measured_value=analogRead(EMF);
   measured_voltage=(measured_value/1024)*5;
   real_voltage=measured_voltage*volt_ratio;
   measured_speed=real_voltage*volt_rpm_ratio;
   return measured_speed;

}
void measure_frequency()
```

```
{
 Serial.println("BASLADI");
 sayac=0;
  while(sayac<1000000)
  {
  digitalWrite(13,100);
  delayMicroseconds(0.01);

  sayac=sayac+1;

  }
 Serial.println("1000 oldu");
  sayac=0;
}
//-----------------------------------------LOOP-----------------------------------------------------//
void loop() {

  if(clockwise_rotation)
  {
    digitalWrite(PWM1,HIGH);
    digitalWrite(PWM4,HIGH);
    digitalWrite(PWM2,LOW);
    digitalWrite(PWM3,LOW);
  }
  else if(!clockwise_rotation)
  {
    digitalWrite(PWM1,LOW);
    digitalWrite(PWM4,LOW);
    digitalWrite(PWM2,HIGH);
    digitalWrite(PWM3,HIGH);
  }

  else
  {
    Serial.println("Boolean ERROR");
  }
ref=3;

  mea=analogRead(A5);
  mea=(mea/1024)*5;
  Error=ref-mea;
  Serial.print("DUTY:");
  Serial.print(DUTY);
  Serial.print("/MEA");
  Serial.print(mea);
  Serial.print("/PID_OUT");
  Serial.print(PID_OUT);
  Serial.print("/ERROR");
  Serial.print(Error);
  Serial.println();
  delay(1000);

  pid.Compute();
  DUTY=255-PID_OUT;
  analogWrite(PWM0,DUTY);

  Serial.println(".....");
  Serial.println(ref_speed);
}
```

b. Link to the Video

https://youtu.be/oLoTbArwjGo