Faculty of Engineering
Ain Shams University

MEP212 – Heat transfer

Design of Journal Bearing

Submitted by:

Team 9

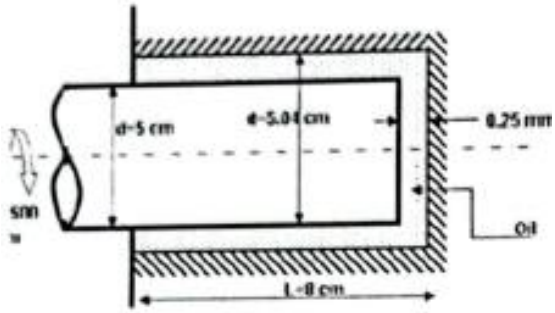
Name	ID
Mohamed Hassan Mohamed Hassan	1807742
Mostafa Mohamed Mostafa	1802669
Mohamed Medhat Mohamed	1806378
Ahmed Khaled Khalaf Mohamed	1700059
Hassan Hossam Hassan	1806831
Mahmoud Hussein Ibrahim	1804719
Mohamed Reda Abdelrahman	1809013
Hassan lofty Abdelhamid	1809861

- Introduction:

The aim of the project is how to use the knowledge we had learned in all the majors of the mechanical field such that control and design to implement a model for an application simulates a small part of the real work.

- Calculations:

We use X-axis as flow direction and y normal direction.



Since this is a parallel flow between two plates therefore, v = 0

So, we can reduce the continuity equation to:

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0 \quad \rightarrow \frac{\partial u}{\partial y}=0$$

$$\therefore u = u(y)$$

Therefore, the x-component of velocity doesn't change in the flow direction.

Since $u = u(y), v = 0, \frac{\partial P}{\partial x} = 0$

So, we can reduce the x-momentum equation to:

$$\rho\left(u\frac{\partial u}{\partial x}+v\frac{\partial v}{\partial y}\right)= \mu\frac{\partial^2 u}{\partial y^2}-\frac{\partial P}{\partial x}$$

$$\therefore \frac{\partial^2 u}{\partial y^2} = 0$$

Now we have a second order differential equation, if we integrate twice, we get:

$$u(y) = C_1 y + C_2$$

The fluid velocities at the plate surfaces must be equal to the velocities of the plates (because of the no slip condition).

When we apply the boundary conditions which are: $u(0) = 0, u(L) = V$, we get:

$$u(y) = \frac{y}{L}V$$

The plates are isothermal and there is no charge in the flow direction therefore, the temperature only depends on y then, the energy equation with dissipation is reduced to:

$$\rho C_\rho \left(u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y}\right) = k\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right) + \mu\varphi$$

$$\varphi = 2\left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2\right] + \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)^2$$

$$\therefore 0 = k\frac{\partial^2 T}{\partial y^2} + \mu\left(\frac{\partial v}{\partial y}\right)^2$$

Since: $\frac{\partial u}{\partial y} = \frac{V}{L}$

$$\therefore k\frac{\partial^2 T}{\partial y^2} = \mu\left(\frac{V}{L}\right)^2$$

Dividing both sides by k and integrating twice, we get:

$$T(y) = \frac{\mu}{2k}\left(\frac{y}{L}V\right)^2 + C_3 y + C_4$$

$T(0) = T_0,\ T(L) = T_0$ Applying to give the temperature distribution:

$$T(y) = T_0 + \frac{\mu V^2}{2k}\left(\frac{y}{L} - \frac{y^2}{L^2}\right)$$

$$\frac{\partial T}{\partial y} = \frac{\mu V^2}{2kL}\left(1 - 2\frac{y}{L}\right)$$

When the differential of T(y) is equal to 0 we get the maximum temperature

$$\frac{\partial T}{\partial y} = \frac{\mu V^2}{2kL}\left(1 - 2\frac{y}{L}\right) = 0$$

$$\therefore y = \frac{L}{2}$$

Therefore, at the mid plane we get the maximum temperature as both the sides are reserved at the same temperature so by substituting with the location of the maximum temperature we get:

$$T_{max} = T\left(\frac{L}{2}\right) = T_0 + \mu V^2 \left(\frac{\left(\frac{L}{2}\right)}{L} - \frac{\left(\frac{L}{2}\right)^2}{L^2}\right)$$

$$T_{max} = T_0 + \frac{\mu V^2}{8k}$$

In our project we have:

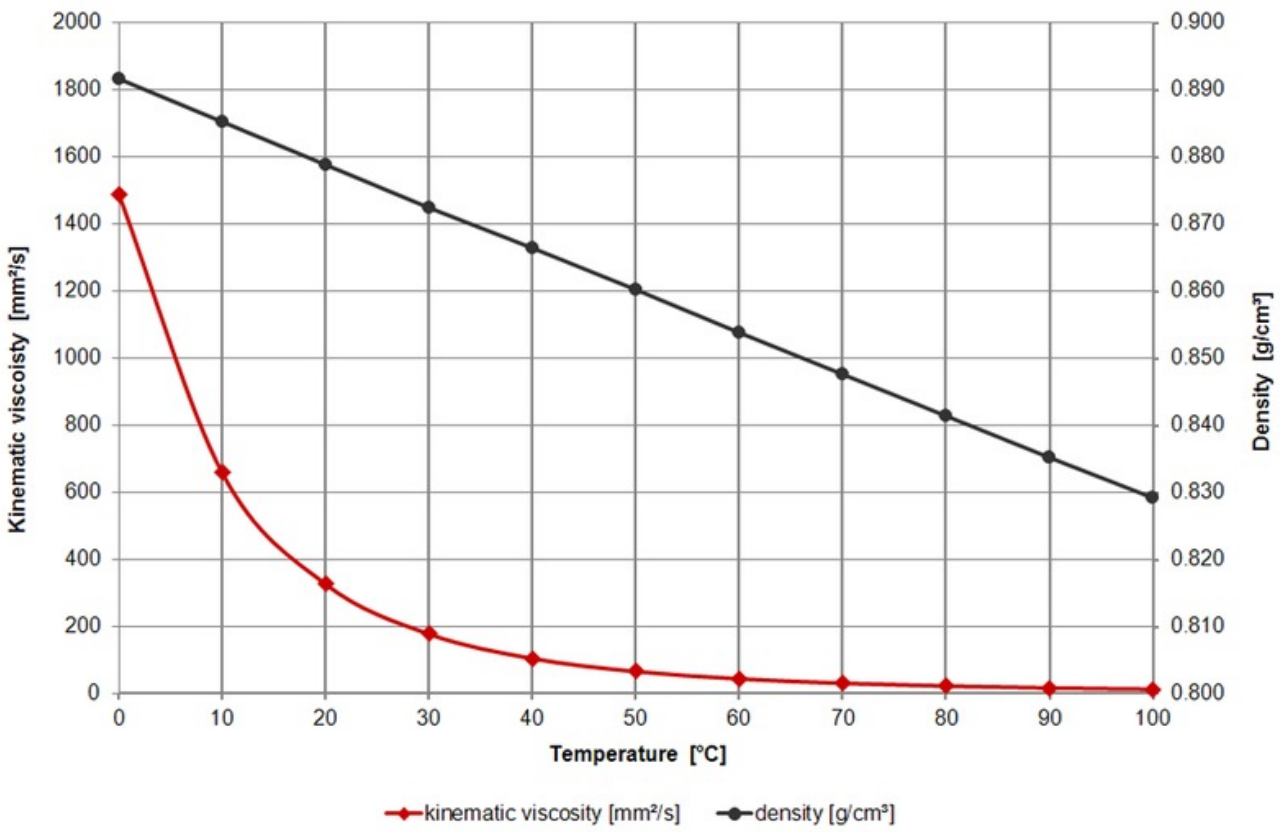
$T_0 = 20\ °\boldsymbol{C}$

$\mu = 2.1\ Pa.s$

$k = 0.136\ W.m^{-1}.K^{-1}$

$$V = \omega * r = \frac{240 * 2\pi}{60} * 12.5 * 10^{-3} = 0.3142\ m.s^{-1}$$

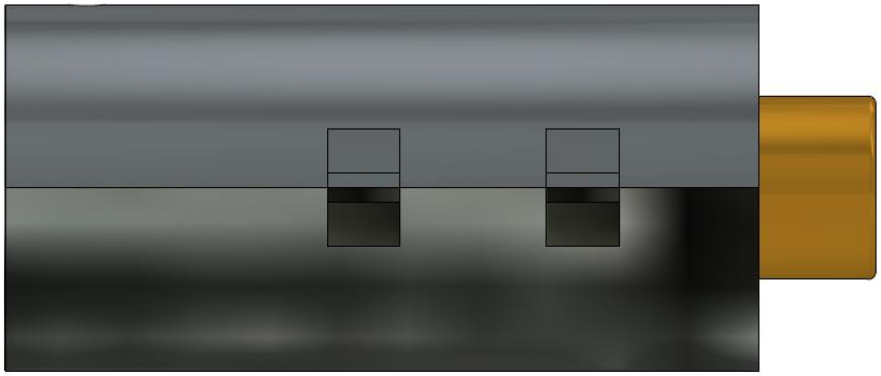
$$T_{max} = 20 + \frac{2.1 * 0.3142^2}{8 * 0.136} = 20.2\ °\boldsymbol{C}$$

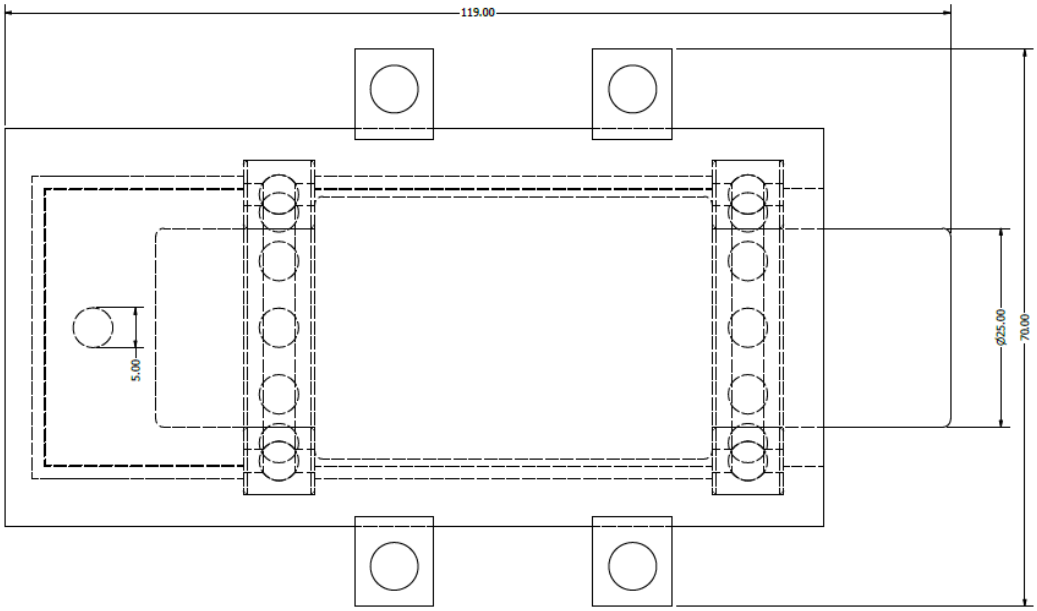


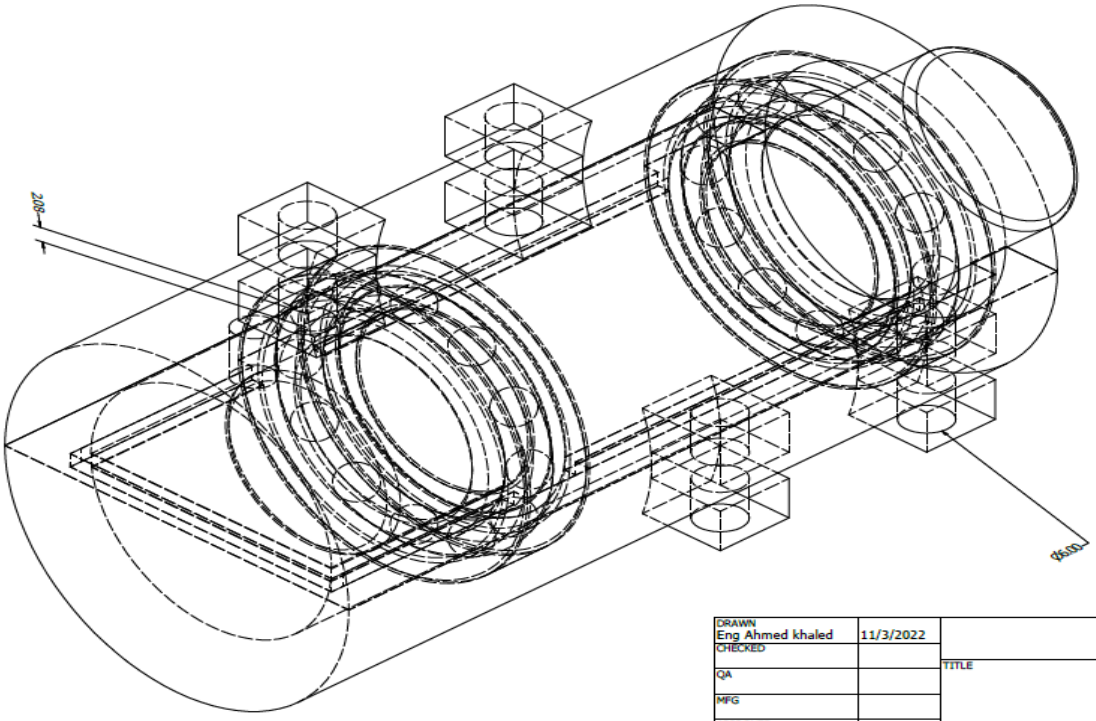
- fluid constrains:

Temp. [°C]	Dyn. Viscosity [mPa.s]	Kin. Viscosity [mm²/s]	Density [g/cm³]
0	753.52	868.78	0.8674
10	378.65	439.85	0.8609
20	206.89	242.10	0.8545
30	121.90	143.70	0.8483
40	76.551	90.903	0.8421
50	50.861	60.849	0.8358
60	35.409	42.685	0.8295
70	25.631	31.135	0.8232
80	19.181	23.478	0.8170
90	14.742	18.185	0.8106
100	11.619	14.443	0.8045

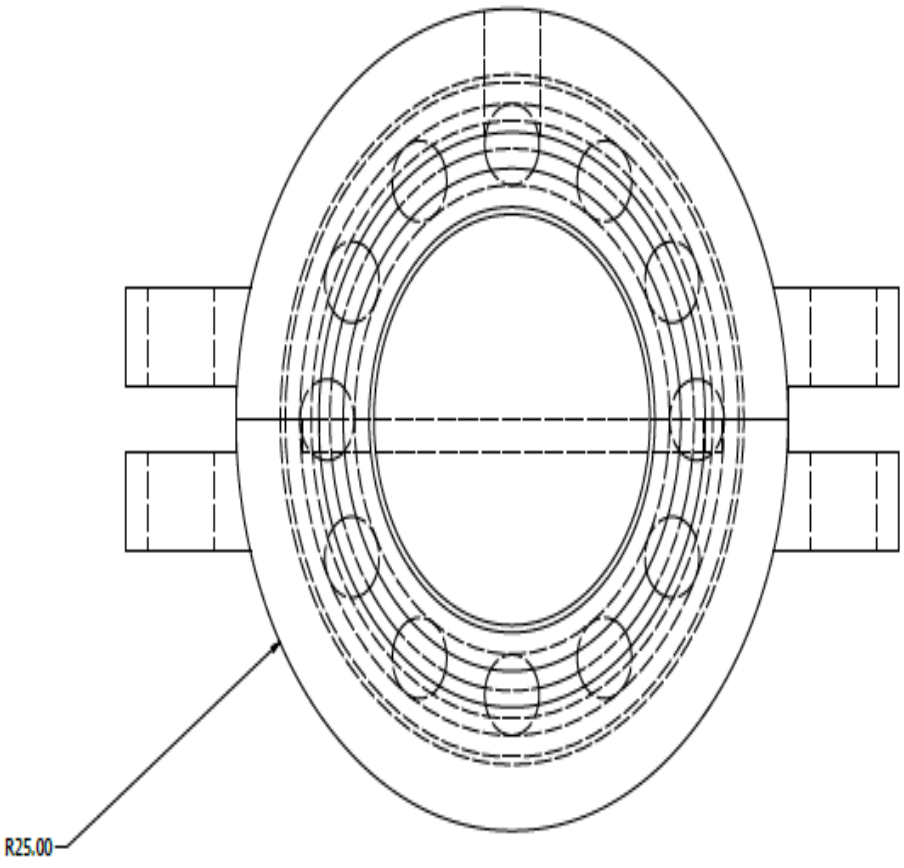
- Mechanical design process:



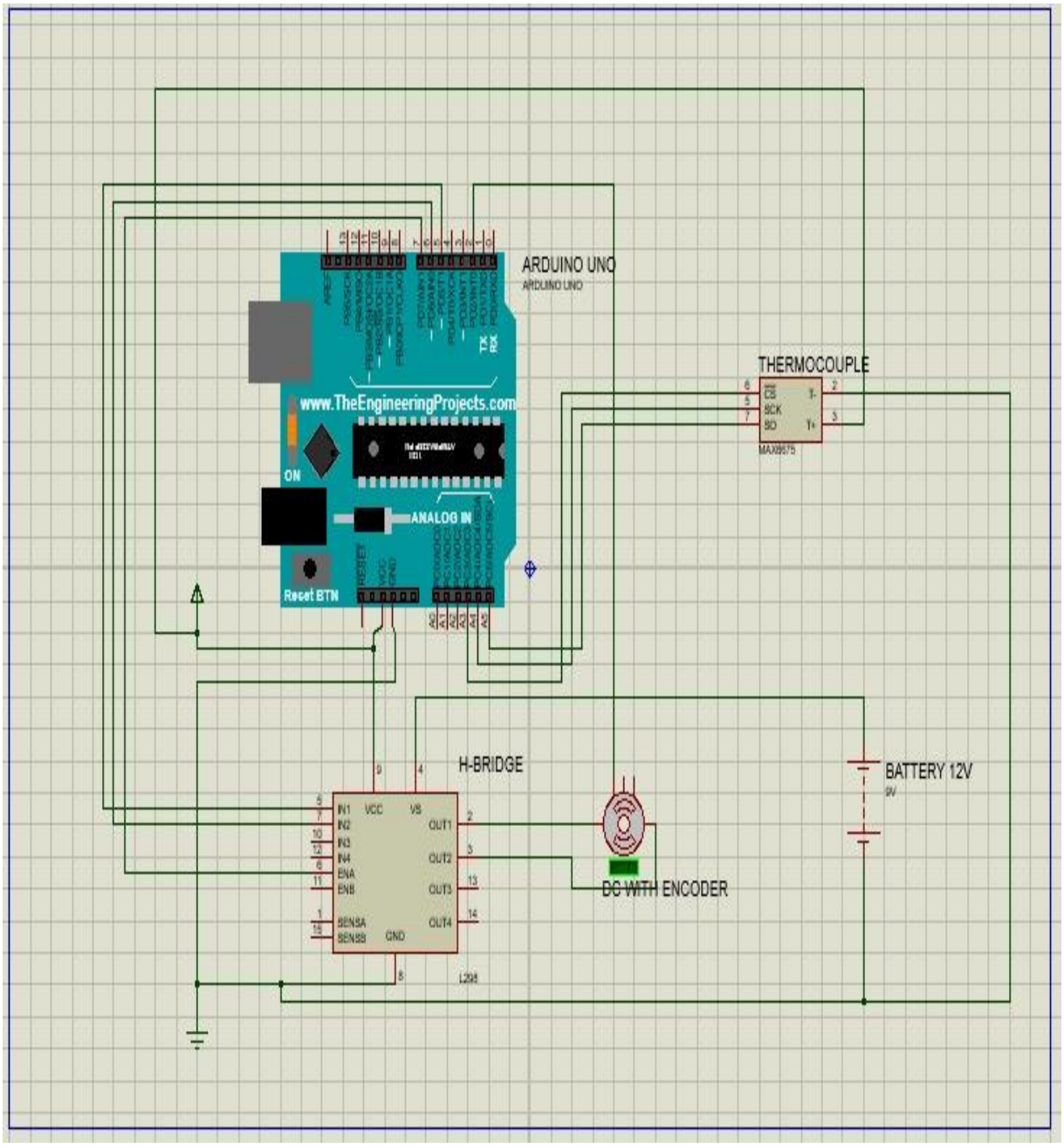




DRAWN Eng Ahmed khaled	11/3/2022	
CHECKED		
QA		TITLE
MFG		
APPROVED		
		SIZE / DWG NO



- Electrical design:



- Arduino code:

Journal_Bearing | Arduino 1.8.20 Hourly Build 2022/04/25 09:33

File Edit Sketch Tools Help

Journal_Bearing

```
/*
  Smooth MAX6675 Thermocouple

  Reads a temperature from the MAX6675 thermocouple,
  smooth and displays it in the default Serial.

  https://github.com/YuriiSalimov/MAX6675_Thermocouple

  Created by Yurii Salimov, May, 2019.
  Released into the public domain.
*/
#include <Thermocouple.h>
#include <MAX6675_Thermocouple.h>
#include <SmoothThermocouple.h>

/*Encoder Pins*/
#define CH1_PIN 2
#define CH2_PIN 3

/*Thermocouple Pins*/
#define CS_PIN  4
#define SO_PIN  5
#define SCK_PIN 6

/*Motor Control*/
#define PWM_PIN 9
#define IN1_PIN 10
#define IN2_PIN 11

/*************************************/
/**

/**
  Thermocouple Initialization
  Smoothing factor of a temperature value.
*/
#define SMOOTHING_FACTOR 2
float T_max = 0;
float T_node = 23;
int T_mapped;

Thermocouple* thermocouple = NULL;
/*************************************/

/**
 * Oil Specs:
 * Mu = 2.1 Pascal.Sec
 * K = 0.136 W/m.K
 */

 float Mu = 2.1;
 float K = 0.136;

/**
  Motor & Encoder Initialization
*/
byte Encoder0PinALast;
unsigned long int  Num_Of_Pulses;
boolean DIR = true;
double Speed=0;
double Time = 0;
float V_Linear;
float Omega;
```

```
// the setup function runs once when you press reset or power the board
void setup() {
  Serial.begin(9600);

  Encoder_Init();

  pinMode(IN1_PIN, OUTPUT);
  pinMode(IN2_PIN, OUTPUT);

  digitalWrite(IN1_PIN, LOW);
  digitalWrite(IN2_PIN, HIGH);
  analogWrite(PWM_PIN, 255);

  Thermocouple* originThermocouple = new MAX6675_Thermocouple(SCK_PIN, CS_PIN, SO_PIN);
  thermocouple = new SmoothThermocouple(originThermocouple, SMOOTHING_FACTOR);

  /* OR
    thermocouple = new SmoothThermocouple(
      new MAX6675_Thermocouple(SCK_PIN, CS_PIN, SO_PIN),
      SMOOTHING_FACTOR
    );
  */
}

// the loop function runs over and over again forever
void loop() {
  Thermo_Readings();
  //Serial.println("LOOOOOOOOOOOOOOOOOOP");

  Omega = RPM_Read();
```

```
  Serial.println(T_mapped);
  Serial.println(Omega);

  V_Linear = Omega * ((2*3.14)/60) * 0.0125;

  delay(100);
  T_max = (T_node + ((Mu*(V_Linear*V_Linear))/8*K));

  delay(100);
 // T_mapped = map(Temp(), 20, 20.2, 0, 100);
  int Temp_mapped = (float)Temp()*100;
  delay(100);

  if(Temp_mapped >=  2305 && Temp_mapped <= 2310)
  {
    Serial.println("WARNNING !");
  }
  else if(Temp_mapped >= 2310)
  {
    analogWrite(PWM_PIN, 0);
    Serial.println("Motor Stopped !!");
  }


  // Reads temperature
  //Temp();
  //Serial.print("Pulse: ");
  //Serial.println(Num_Of_Pulses);
 // Num_Of_Pulses = 0;
  delay(100);
```

```
double Temp()
{
  const double celsius = thermocouple->readCelsius();
  const double kelvin = thermocouple->readKelvin();
  const double fahrenheit = thermocouple->readFahrenheit();

  // Output of information
/*  Serial.print("Temperature: ");
  Serial.print(celsius);
  Serial.print(" C, ");
  Serial.print(kelvin);
  Serial.print(" K, ");
  Serial.print(fahrenheit);
  Serial.println(" F");
*/
  return (celsius);
}

void Thermo_Readings()
{
  const double celsius_1 = thermocouple->readCelsius();
  const double kelvin_1 = thermocouple->readKelvin();
  const double fahrenheit_1 = thermocouple->readFahrenheit();

  // Output of information
  Serial.print("Temperature: ");
  Serial.print(celsius_1);
  Serial.print(" C, ");
  Serial.print(kelvin_1);
  Serial.print(" K, ");
  Serial.print(fahrenheit 1);
```

```
 void Encoder_Init()
 {
   Num_Of_Pulses = 0;
   DIR = true;
   pinMode(CH2_PIN, INPUT);
   attachInterrupt(digitalPinToInterrupt(2), Wheel_Speed, CHANGE);
 }

 void Wheel_Speed()
 {
   Num_Of_Pulses++;
   /*int Lstate = digitalRead(CH1_PIN);
   if((Encoder0PinALast == LOW) && (Lstate == HIGH))
   {
     int Val = digitalRead(CH2_PIN);
     if(Val == LOW && DIR)
     {
       DIR = false;
     }
     else if(Val == HIGH && !DIR)
     {
       DIR = true;
     }
   }
   Encoder0PinALast = Lstate;

   if(!DIR) Num_Of_Pulses++;
   else Num_Of_Pulses--;*/
 }

 float RPM_Read()
```

```
  /*int Lstate = digitalRead(CH1_PIN);
  if((Encoder0PinALast == LOW) && (Lstate == HIGH))
  {
    int Val = digitalRead(CH2_PIN);
    if(Val == LOW && DIR)
    {
      DIR = false;
    }
    else if(Val == HIGH && !DIR)
    {
      DIR = true;
    }
  }
  Encoder0PinALast = Lstate;

  if(!DIR) Num_Of_Pulses++;
  else Num_Of_Pulses--;*/
}

float RPM_Read()
{
  if (millis() >= Time+2000)
  {
     Speed=(float) ( (Num_Of_Pulses/979) / 2 )*60; // pulses / 950 = revs , over 2 second >> speed unit rev/s
     Num_Of_Pulses=0;
     Time=millis();
     Serial.print(Speed);
     Serial.println(" RPM");
  }
  return Speed;
}
```