EE555 Automation Lab
Lab 4
15.02.2021

Pratik Kr. Rana
204102507

Automation Lab 4

Objective:

1. To control a DC motor using a Potentiometer and measure its speed with the help of an encoder output.

2. To obtain desired speed of the DC Motor using PID Controller.

Components Used:

- Arduino Uno R3
- DC Encoder Motor
- L239D Motor Driver
- Potentiometer
- 9V Battery
- Breadboard
- Wires

Procedure:

1. The circuit as shown in Fig. 1 was designed in TinkerCAD.
2. For measuring the speed of the Motor, one of the channel (Channel A) was assigned to a pin in Arduino.
3. Channel A gave the count of pulses due to the encoder motor.
4. This was used to calculate the rpm of the motor as shown below:

Rpm of Motor = Frequency of Pulses / Pulse per rotation (PPR)

The encoder count per revolution was obtained after various trials so that the rpm displayed in the monitor is properly calibrated with respect to the potentiometer value.

C++ Program for Arduino:

```
//Motor encoder output pulse per rotation


#define ENC_COUNT_REV 9.65


// motor control pin
const int motorDirPin = 5;  // Input 1
const int motorPWMPin = 6;  // Input 2
const int EnablePin = 9;    // Enable
```

Pratik Kr. Rana
204102507

```
//encoder pin
const int encoderPinA = 2;


//Analog pin for potentiometer
int speedcontrol = A0;


// Pulse count from encoder
volatile long encoderValue = 0;


//Total measures total number of pulses (sum of encoder values)
volatile long total=0;


// 100 milli-seconds interval for measurements
int interval = 100;

// Counters for milliseconds during interval
long previousMillis = 0;
long currentMillis = 0;


// Variable for RPM measuerment
int rpm;
```

Pratik Kr. Rana
204102507

```
// Variable for PWM motor speed output
int motorPwm = 0;

void setup ()
{

Serial.begin (9600); // initialize serial communication



// Set encoder as input with internal pullup
pinMode(encoderPinA, INPUT_PULLUP);

// Set PWM connection as output
pinMode(motorPWMPin,OUTPUT);
pinMode(motorDirPin,OUTPUT);


//Set EnablePin HIGH
digitalWrite(EnablePin,HIGH);

// Attach interrupt
attachInterrupt(digitalPinToInterrupt(encoderPinA), updateEncoder,CHANGE);

// Setup initial values for timer
previousMillis=millis();
}
void loop()
```

Pratik Kr. Rana
204102507

```
{
  // Control motor with potentiometer
  motorPwm = map(analogRead(speedcontrol), 0, 1023, 0, 255);

  // Write PWM to controller
  analogWrite(motorPWMPin , motorPwm);

  // Update RPM value every second
  currentMillis = millis();

  if (currentMillis - previousMillis > interval)
  {
    previousMillis = currentMillis;
    // Calculate RPM
    rpm = (float)(encoderValue*60/ENC_COUNT_REV);

    // Only update display when there is a reading
    if (motorPwm > 0 || rpm > 0) {
      Serial.print("Speed of the motor is: ");;
      Serial.print(rpm);
      Serial.println(" RPM");
      encoderValue = 0; //reset encoder value
    }
  }
}

void updateEncoder()
{
  // Increment value for each pulse from encoder
```

Pratik Kr. Rana
204102507

```
  encoderValue++;


  //Total of encoderValue
  total++;
}
```

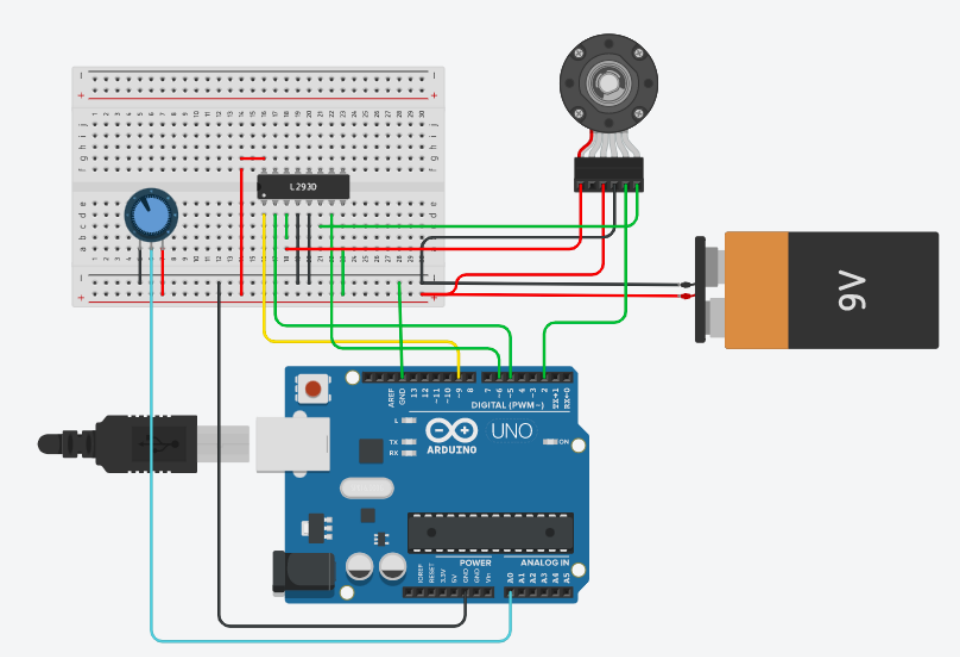


Fig. 1: Simulation Circuit Model

Pratik Kr. Rana
204102507

Output:

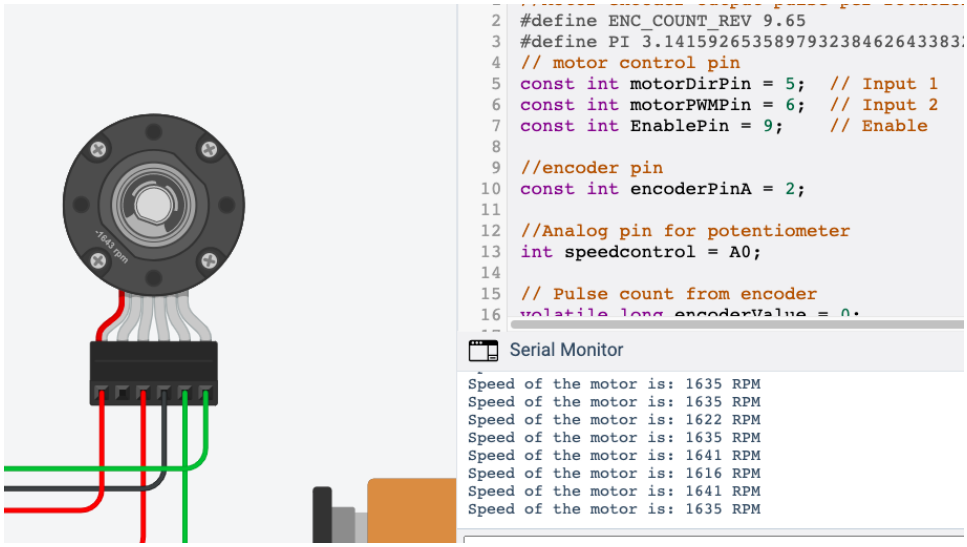


Fig.2: Motor rotating at around 1640rpm (Matches with Activity Monitor)

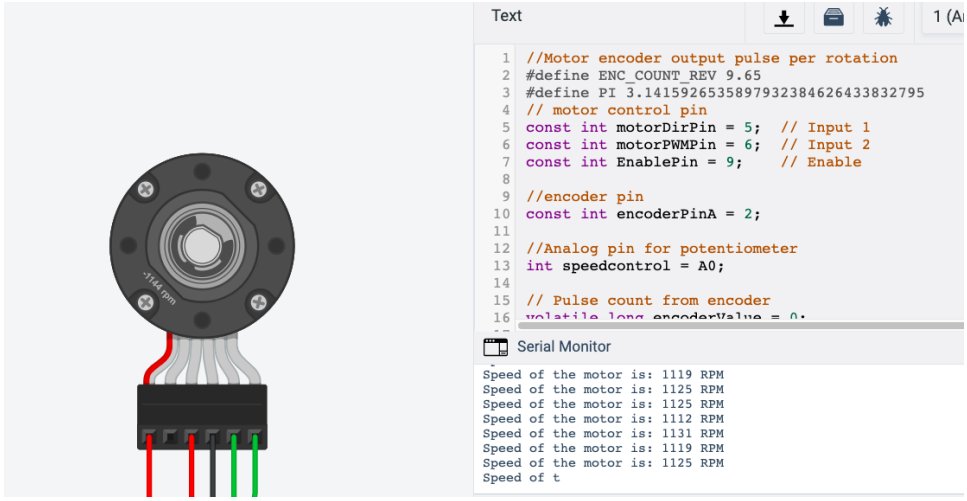


Fig.3: Motor rotating at around 1144rpm (Matches with Activity Monitor)

Pratik Kr. Rana
204102507

PID Control of the DC Motor:

The desired value was set to 170(a constant value).

PID values were chosen as follows:

Arduino C++ code:

```
// Declaring Variables:
int timer_counter1 = 0;
int DesiredMotorSpeed = 0;
int encoder_pulses = 0;
int PID = 0;

// Declaring Error Variables:
int Ierror = 0;
int Derror = 0;
int pasterror = 0;
int error = 0;

// Delcaring Constant Variables:
double deltaT = 0.1;
int PWM_Value = 50;
int PPR = 9.65;
```

Pratik Kr. Rana
204102507

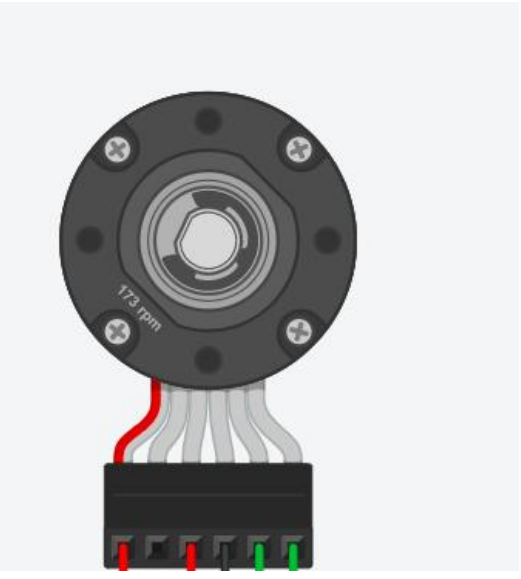
```
// PID Values:
double Kp = 0.18;
double Ki = 0.15;
double Kd = 0.0375;



void setup()
{
  // Setting Output Pins:
  pinMode(6, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(9, OUTPUT);

  // Setting Input Pins:
  pinMode(2, INPUT);

  // Setting Pin 2 on the arduino to the interrupt pin:
  attachInterrupt(digitalPinToInterrupt(2), funct_ch, RISING);

  //interrupts();

  Serial.begin(9600); // Initalizing the serial monitor
}
double speed = 0;
void loop()
{
```

```
 analogWrite(9, PWM_Value); // Set pin 9 to output the PWM_Value found with the PID eqn


 // Setting the motor direction:
 digitalWrite(5, HIGH);
 digitalWrite(6, LOW);


 // Sets the desired motor speed
 DesiredMotorSpeed = 170;
 speed = (60*encoder_pulses)/(9.65); // Calculate the speed
 encoder_pulses = 0; // Resets the encoder pulses to zero for the next cycle


 Ierror = Ierror + error;            // Finds the sum of the error for the integral
 Derror = error - pasterror;         // Finds the difference of the error for the derivative


 double error = DesiredMotorSpeed - speed; // Calculates the error


 // The PID equation written in Arduino code form:
 PID = (Kp*error)+(Ki*(Ierror*deltaT))+(Kd*((error-pasterror)/deltaT))+PWM_Value;
 pasterror = error;


 // Ensures that the PWM_Value is positive:
 if(PID >= 0){
   PWM_Value = PID;
 }
else if(PID < 0){
   PWM_Value = -PID;
 }
```

Pratik Kr. Rana
204102507

```
  // Prints [speed | DesiredMotorSpeed | error] to the serial monitor in that order:
  Serial.print(speed);
  Serial.print(" | ");
  Serial.print(DesiredMotorSpeed);
  Serial.print(" | ");
  Serial.print(error);
  Serial.print(" | ");
  Serial.print(PWM_Value);
  Serial.print('\n');
}
void funct_ch(){
  encoder_pulses = encoder_pulses + 1; // Increments the encoder_pulses by one on RISING value
}
```

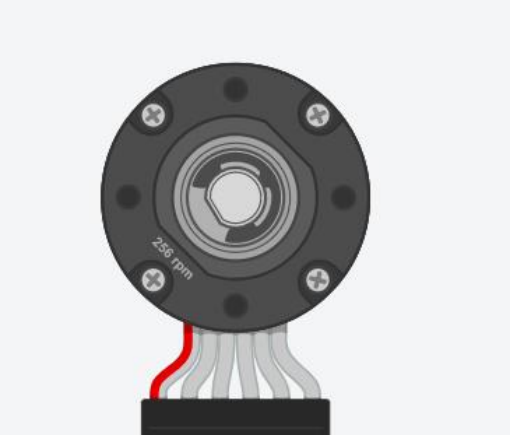
Output:




```
// Sets the desired motor speed

  DesiredMotorSpeed = 170;
  speed = (60*encoder_pulses)/(9.65);
  encoder_pulses = 0; // Resets the en
  Ierror = Ierror + error;      // Fin
  Derror = error - pasterror;   // Fin
```

Fig.: As can be seen, req_rpm is set at 170 and the motor is also rotating about 170rpm

Pratik Kr. Rana
204102507




```
// Sets the desired motor speed

  DesiredMotorSpeed = 260;
  speed = (60*encoder_pulses)/(9.65); // Ca
  encoder_pulses = 0; // Resets the encoder
  Ierror = Ierror + error;       // Finds th
  Derror = error - pasterror;    // Finds th
```

Fig.: As can be seen, req_rpm is set at 260 and the motor is also rotating about 260rpm