BERZIET UNIVERSITY

Faculty of Information Technology Computer Systems Engineering Department

COMPUTER DESIGN LAB

ENCS4110

DC MOTORS AND PWM

Report NO.1/ Experiment _2

Prepared by:

Nour Naji– 1190270

Partners:

Rasha Dar Abu Zidan -1190547

Supervised by: Dr. Jamal Seyam

Teacher Assistant: Eng. Raha Zabadi

Section: 2

Date: 20 September 2021

Abstract:

1. Aim of the experiment

In this experiment we are going to control the speed of a DC motor using Pulse Width Modulation (PWM). Because DC motors need high current, we can't drive the motor from a digital pin (even if the motor runs on 5 volt). Therefore, we need a driving circuit. Transistors can be used to drive motors. We will use Arduino to generate PWM signal that to drive the DC motor with different speeds.

2. Methods used

We used tinkercad application, which is a free and easy to use application for 3D design, electronics and coding, first we need a Breadboard to mount the pieces to it, Arduino, DC-Motor, BJT, NPN Transistor (2N2222) , Diode (1N4001), Resistors (330 Ohm), H-Bridge L298 and Wires, We installed the two circuits as required by using the above-mentioned pieces, writing the codes and observing the results

Table of Contents

List Of Figures

1. Theory

1.1 DC motor

DC motor is defined as a class of electric motors that convert direct current electrical energy into mechanical energy, then we can conclude that any electric motor that is operated using direct current or direct current is called DC motor, it works by creating a magnetic field in the air gap When the field coil of the DC motor is energized. The generated magnetic field is in the direction of the armature radius. The magnetic field enters the motor from the north pole side of the field coil and the motor "exits" from the south pole side of the field coil.[1]

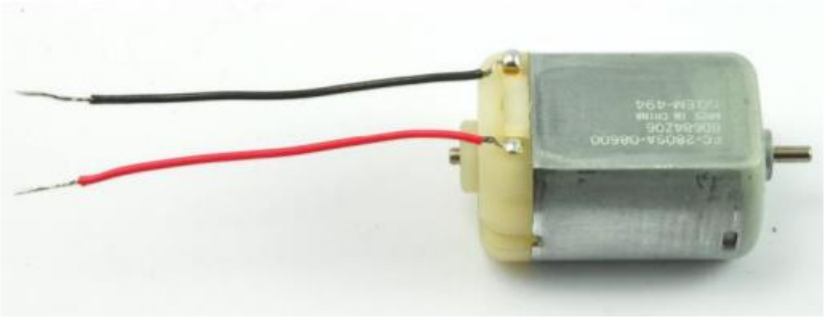


Figure 1-1.1: A small DC motor

1.1.1 Using BJT to drive a motor

We use a BJT transistor as a driver for the DC motor. We need a resistor on the base to limit the current from the digital pin, in order not to fry the transistor. And the diode is needed to protect the transistor

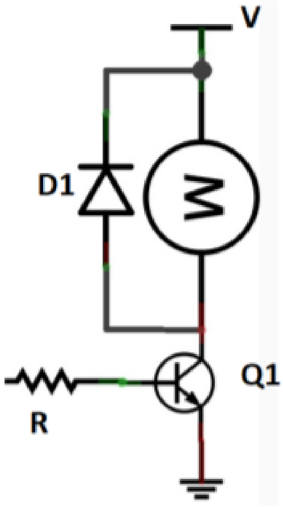


Figure 1.1.1. 1: Using BJT to drive a motor

1.1.2 Using MOSFET to drive a motor

In motor drive systems, a gate drive or "pre-drive" IC is often used in conjunction with N-channel power MOSFETs to provide the high current needed to drive the motors. There are a number of design considerations that must be made when selecting driver ICs, MOSFETs, and in some cases associated passive components. This process is often not well understood, and the applications are less than optimal, that's why MOSFETs are used to power DC motors.[2]

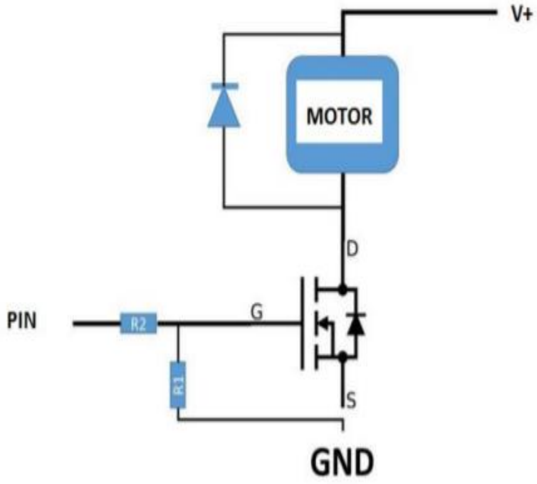


Figure 1.1.2. 1:Using MOSFET to drive a motor

1.2 Arduino

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.[3]

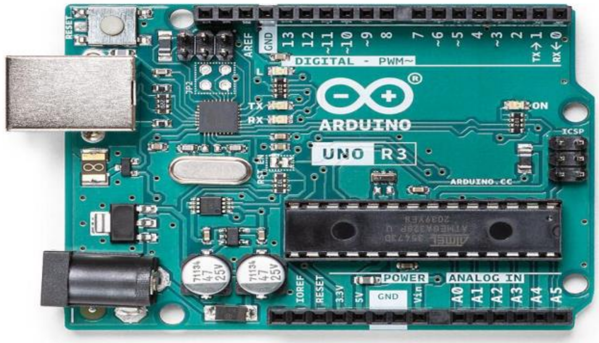


Figure 1-2.1: Arduino

1.3 H-Bridge L298

L298 is a high voltage and high current motor drive chip which receives TTL logic signals. They are mostly used when it is needed to operate different loads like motors and solenoid etc. where an H-Bridge is required, Control unit can only provide TTL outputs. Current control and PWM operable single-chip device are needed. It has two enable inputs to enable or disable the particular device attached at its output independently. Thus, H-Bridge is basically used to control the rotating direction in DC motors.

If Q1 and Q4 are ON while Q2 and Q3 are OFF, the motor can run clockwise, now of Q2 and Q3 are ON, while Q1 and Q4 are OFF, the motor is going to reverse the direction.

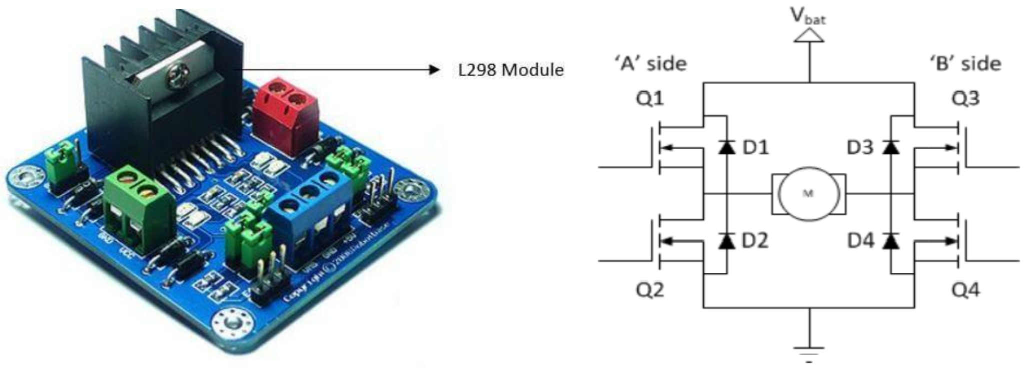


Figure 1-3.1: H-Bridge Motor Driver

1.4 Pulse Width Modulation (PWM)

Pulse Width Modulation (PWM) is a fancy term for describing a type of digital signal. Pulse width modulation is used in a variety of applications including sophisticated control circuitry. A common way we use them here at Spark Fun is to control dimming of RGB LEDs or to control the direction of a servo. We can accomplish a range of results in both applications because pulse width modulation allows us to vary how much time the signal is high in an analog fashion. While the signal can only be high (usually 5V) or low (ground) at any time, we can change the proportion of time the signal is high compared to when it is low over a consistent time interval.

When the signal is high, we call this "on time". To describe the amount of "on time", we use the concept of duty cycle. Duty cycle [duty cycle= (ON period)/ (ON+OFF periods)] is measured in percentage. The percentage duty cycle specifically describes the percentage of time a digital signal is on over an interval or period of time. This period is the inverse of the frequency of the waveform.

If a digital signal spends half of the time on and the other half off, we would say the digital signal has a duty cycle of 50% and resembles an ideal square wave. If the percentage is higher than 50%, the digital signal spends more time in the high state than the low state and vice versa if the duty cycle is less than 50%. Here is a graph that illustrates these three scenarios: [4]

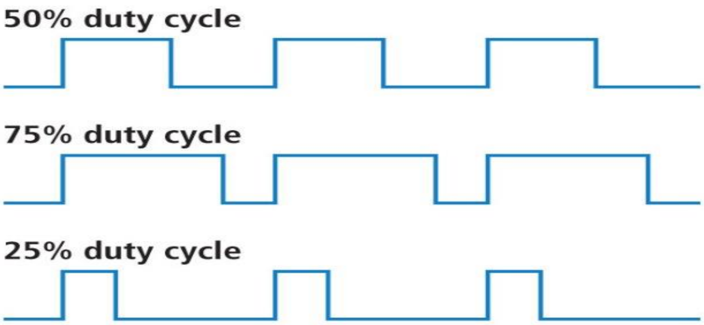
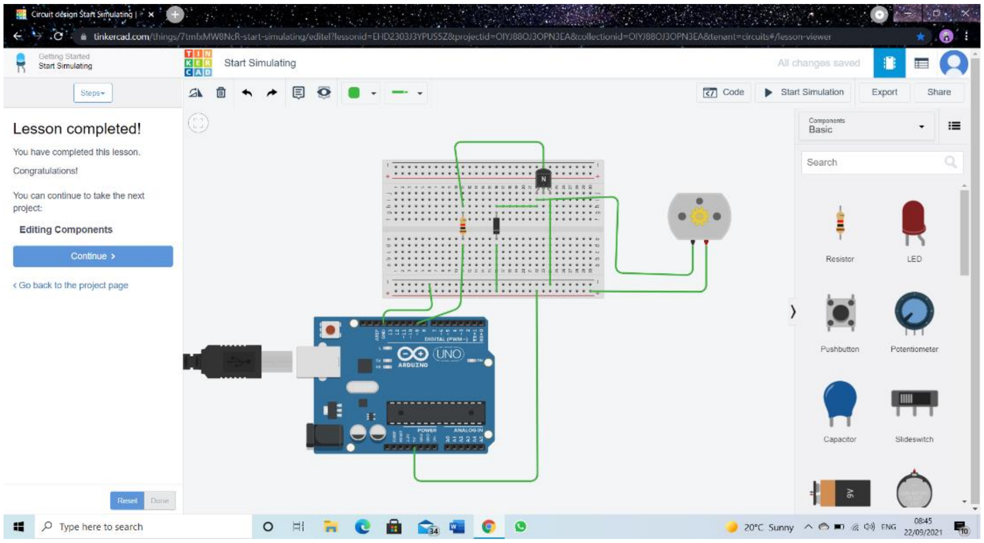


Figure 1.4. 1:50%, 75%, and 25% Duty Cycle Example

2. Procedure and Discussion

2.1 Use a transistor the control the speed of a DC-Motor

We connected the circuit as shown in the figure [2.1.1], We used Arduino, Breadboard, DC-Motor, Wires, BJT, Diode (1N4001) and Resistors (330 Ohm), we connected pin number 9 to the base on the Arduino, and we written the required code as shown in the figure [2.1.2].



Lesson completed!

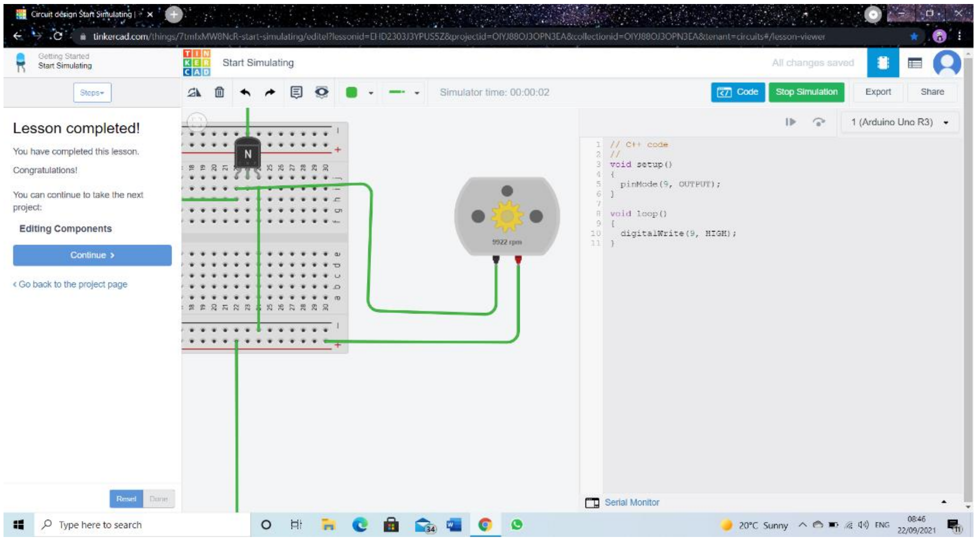
You have completed this lesson.

Congratulations!

You can continue to take the next project:

Editing Components

Figure 2.1. 1:Use a transistor the control the speed of a DC-Motor



Lesson completed!

You have completed this lesson.

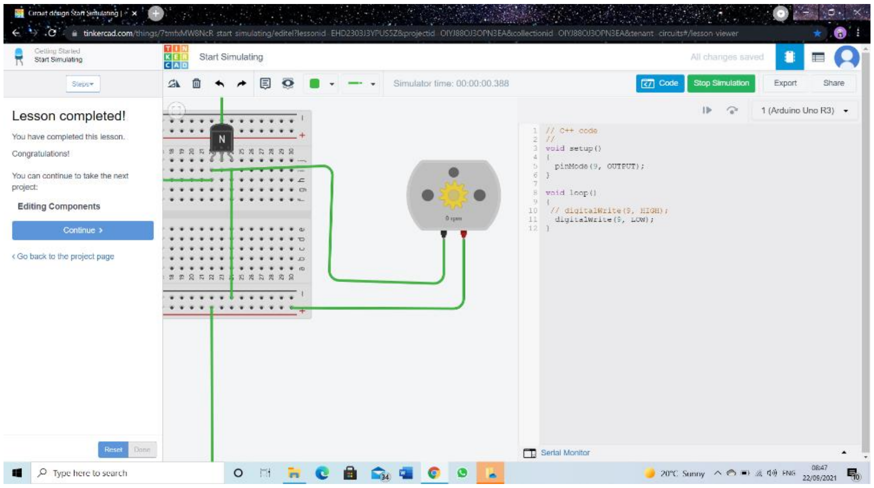
Congratulations!

You can continue to take the next project:

Editing Components

```
// C++ code
//
void setup()
{
  pinMode(9, OUTPUT);
}

void loop()
{
  digitalWrite(9, HIGH);
}
```

Figure 2.1. 2: code-1



Lesson completed!

You have completed this lesson.

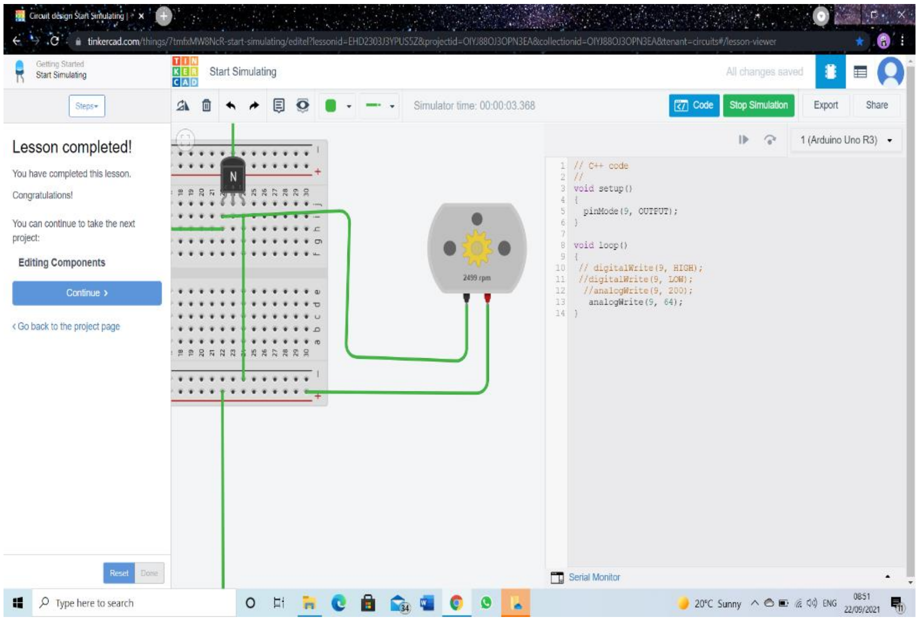
Congratulations!

You can continue to take the next project:

Editing Components

```
// C++ code
//
void setup()
{
  pinMode(9, OUTPUT);
}

void loop()
{
 // digitalWrite(9, HIGH);
  digitalWrite(9, LOW);
}
```

Figure 2.1. 3:code-2



Lesson completed!

You have completed this lesson.

Congratulations!

You can continue to take the next project:

Editing Components

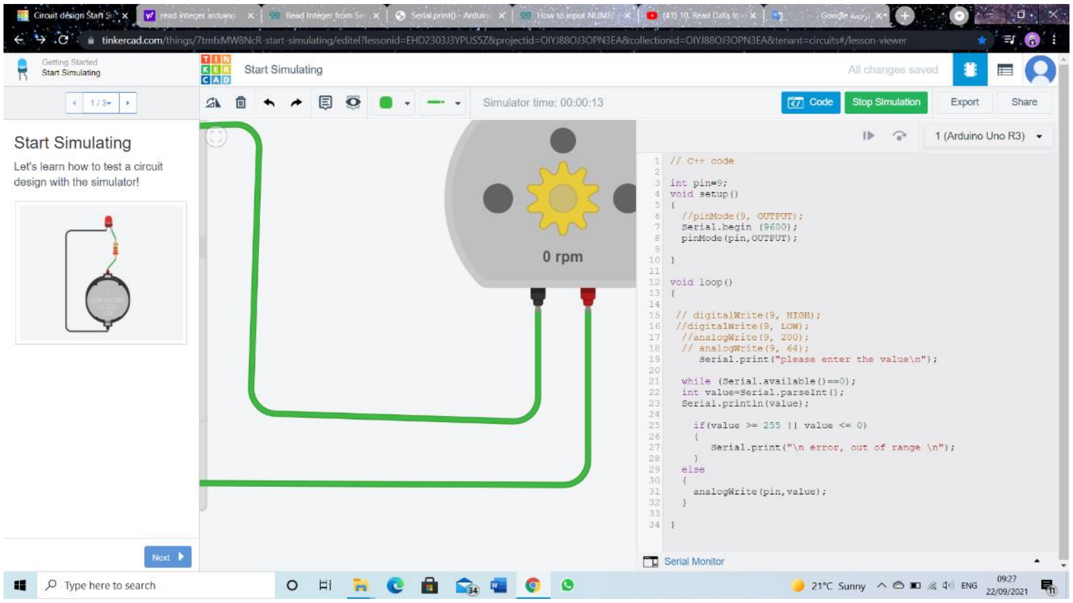
```
// C++ code
//
void setup()
{
  pinMode(9, OUTPUT);
}

void loop()
{
 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
   analogWrite(9, 64);
}
```

Figure 2.1. 4:code-3

- **digitalWrite(9,HIGH)** ➔We note that the motor reaches the highest speed [9922 rpm].
- **digitalWrite(9,LOW)** ➔ We notice that the motor speed decreases and eventually stop [0 revolution].
- **analogWrite (9,200)** ➔ We notice that the motor speed increases, but does not reach the maximum value [~ 7640 rpm] , Where the highest speed is at **analogWrite (9,255)** .
- **analogWrite(9,64)** ➔ We note that the motor speed decreases, but does not stop [~ 2499 rpm].

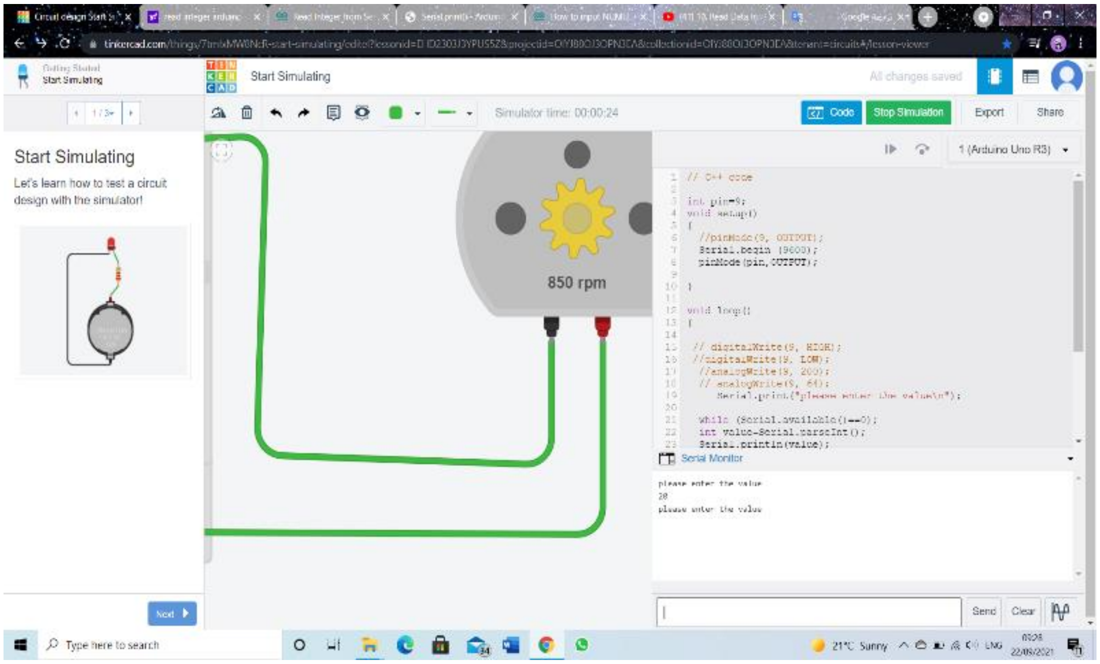
2.1.1 Task 1

Using Serial monitor, write a code to read a number from keyboard and change the speed of the motor. You have to check if the number between 0 and 255. Print an error message if the number outside this range. Show the result to the instructor.



```
// C++ code

int pin=9;
void setup()
{
  //pinMode(9, OUTPUT);
  Serial.begin (9600);
  pinMode(pin,OUTPUT);

}

void loop()
{

 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 64);
     Serial.print("please enter the value\n");

  while (Serial.available()==0);
  int value=Serial.parseInt();
  Serial.println(value);

    if(value >= 255 || value <= 0)
    {
      Serial.print("\n error, out of range \n");
    }
  else
  {
    analogWrite(pin,value);
  }

}
```

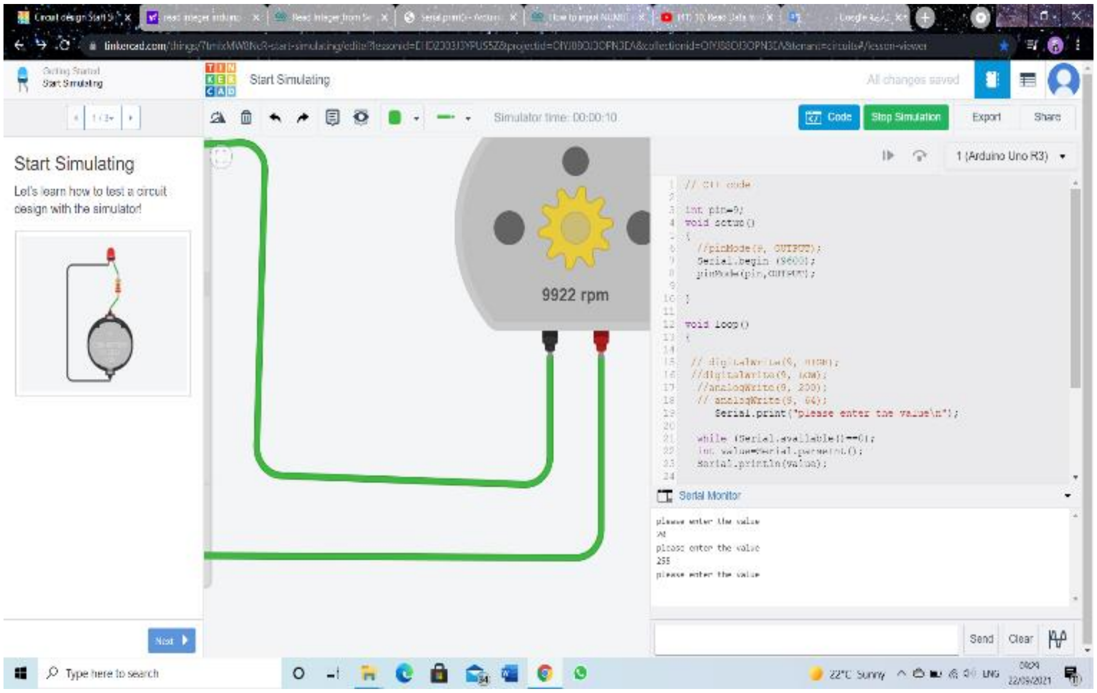
Figure 2.1.1. 1:Task 1-code1



```
// C++ code

int pin=9;
void setup()
{
  //pinMode(9, OUTPUT);
  Serial.begin (9600);
  pinMode(pin,OUTPUT);

}

void loop()
{

 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 64);
     Serial.print("please enter the value\n");

  while (Serial.available()==0);
  int value=Serial.parseInt();
  Serial.println(value);
```

```
please enter the value
20
please enter the value
```

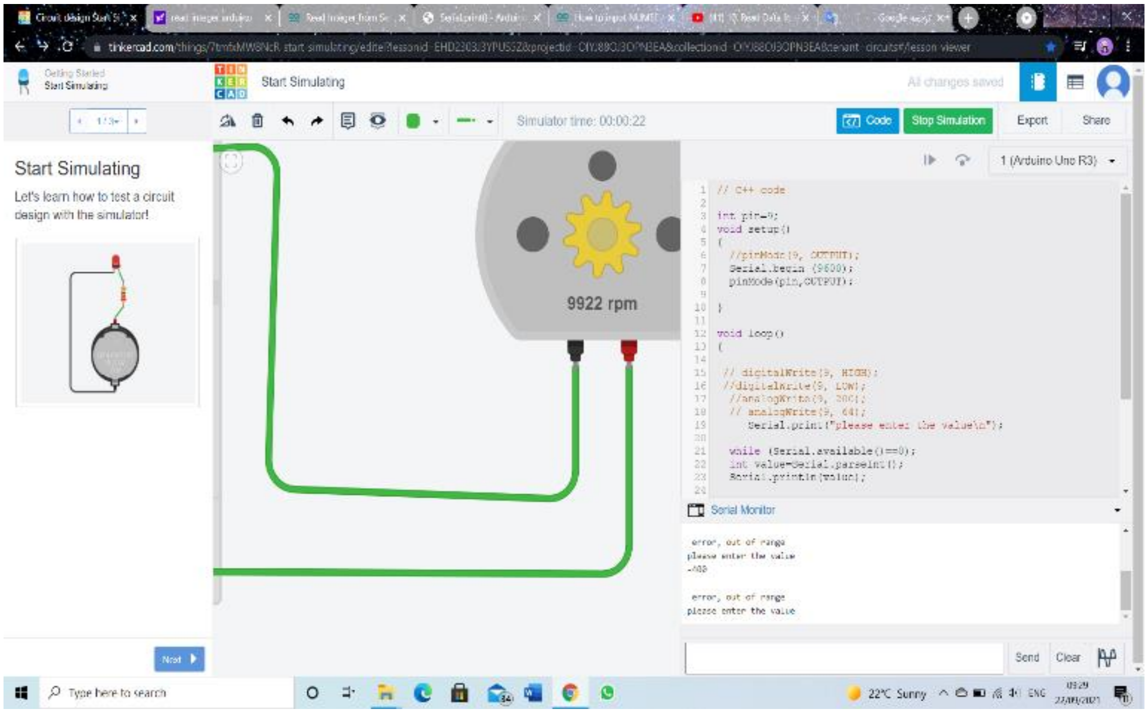
Figure 2.1.1. 2::Task 1-2



```
// C++ code

int pin=9;
void setup()
{
  //pinMode(9, OUTPUT);
  Serial.begin (9600);
  pinMode(pin,OUTPUT);

}

void loop()
{

 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 64);
     Serial.print("please enter the value\n");

  while (Serial.available()==0);
  int value=Serial.parseInt();
  Serial.println(value);
```

```
please enter the value
30
please enter the value
255
please enter the value
```

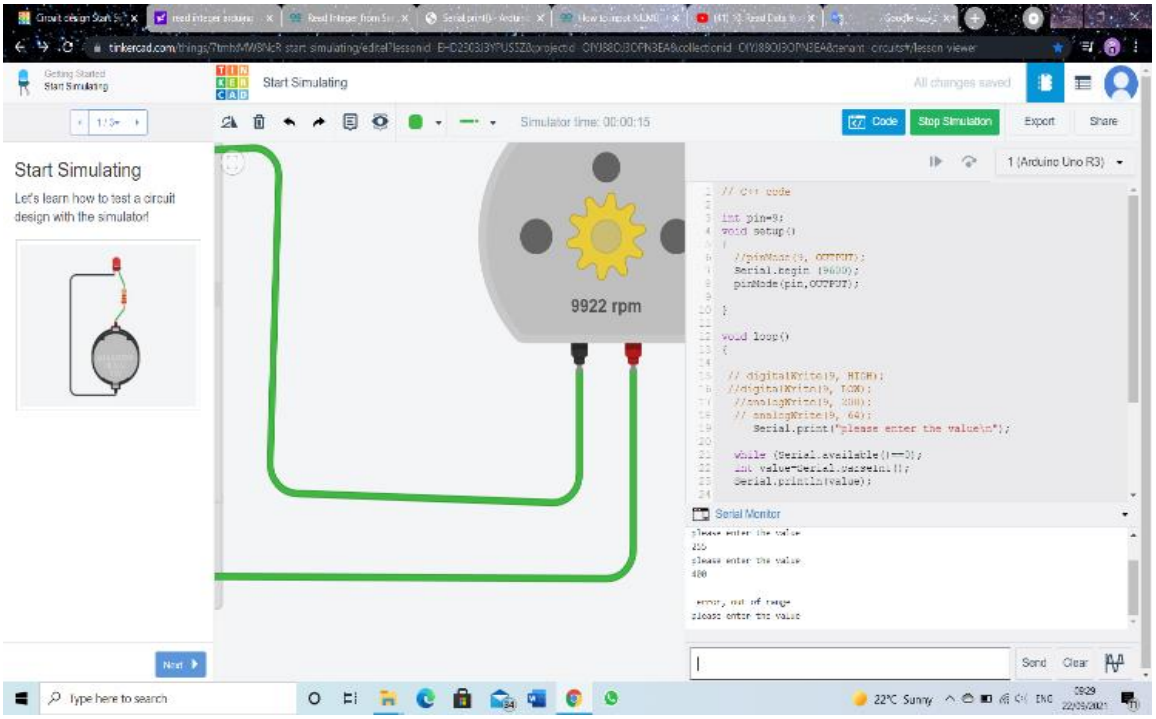
Figure 2.1.1. 3:Task 1-3



```
// C++ code

int pin=9;
void setup()
{
  //pinMode(9, OUTPUT);
  Serial.begin (9600);
  pinMode(pin,OUTPUT);

}

void loop()
{

 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 64);
     Serial.print("please enter the value\n");

  while (Serial.available()==0);
  int value=Serial.parseInt();
  Serial.println(value);
```

```
 error, out of range
please enter the value
-400

 error, out of range
please enter the value
```

Figure 2.1.1. 6:Task 1-4



```
// C++ code

int pin=9;
void setup()
{
  //pinMode(9, OUTPUT);
  Serial.begin (9600);
  pinMode(pin,OUTPUT);

}

void loop()
{

 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 64);
     Serial.print("please enter the value\n");

  while (Serial.available()==0);
  int value=Serial.parseInt();
  Serial.println(value);
```

```
please enter the value
255
please enter the value
400

 error, out of range
please enter the value
```

Figure 2.1.1. 5:Task 1-5

2.2 Use H-Bridge to control the speed and direction of a DC-Motor

In this part, we used L298 Motor Driver Module to control the speed and direction of a DC-Motor, we connected the circuit as shown in the figure [2.2.1], We used Arduino, Breadboard, DC-Motor, Wires, H-Bridge L298 and Resistors (330 Ohm), we have written the required code as shown in the figure [2.2.2].

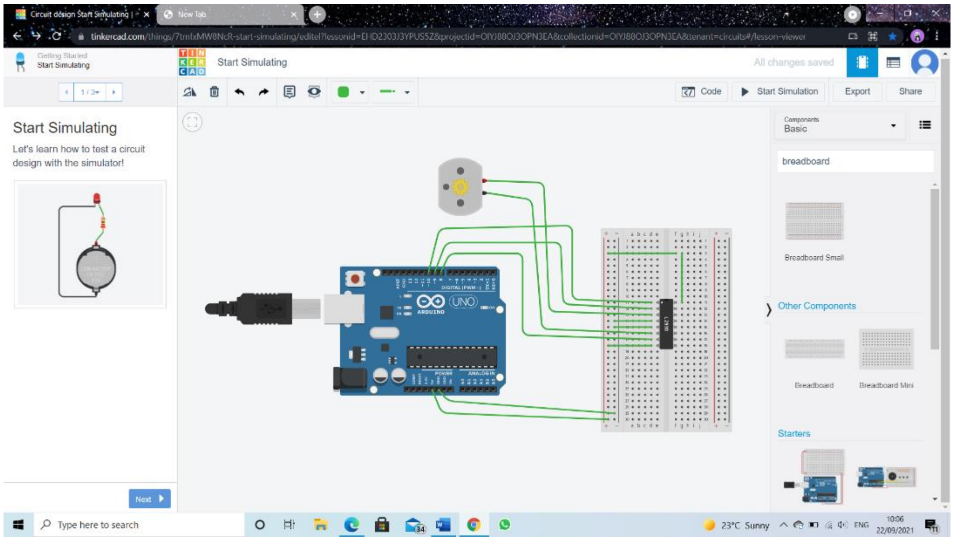
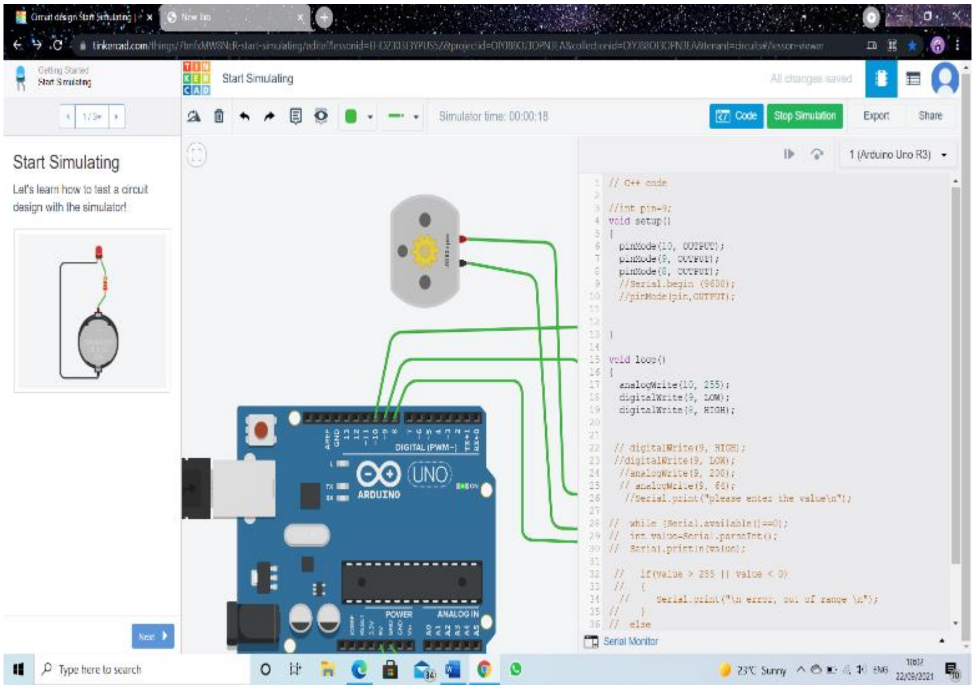
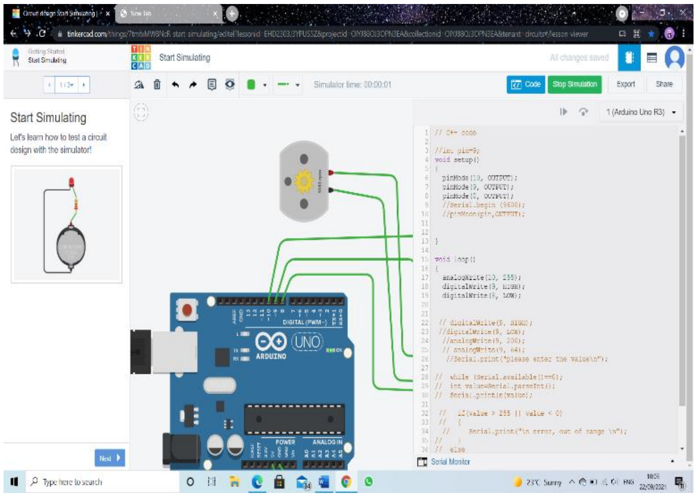


Figure 2.2. 1:control the speed and direction of a DC-Motor



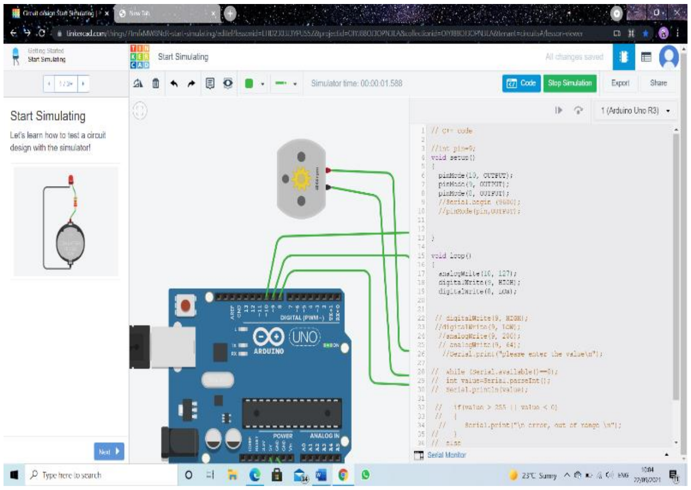
```
// C++ code

//int pin=9;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  //Serial.begin (9600);
  //pinMode(pin,OUTPUT);



}

void loop()
{
  analogWrite(10, 255);
  digitalWrite(9, LOW);
  digitalWrite(8, HIGH);


 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 60);
   //Serial.print("please enter the value\n");

//  while (Serial.available()==0);
//  int value=Serial.parseInt();
//  Serial.println(value);

 //   if(value > 255 || value < 0)
 //   {
  //      Serial.print("\n error, out of range \n");
//    }
//  else
```

Figure 2.2. 2:code-2



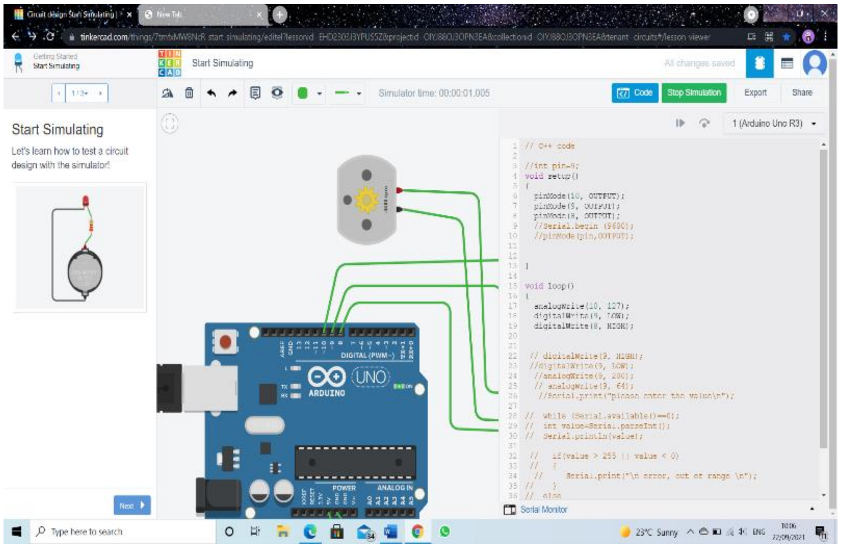
```
// C++ code

//int pin=9;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  //Serial.begin (9600);
  //pinMode(pin,OUTPUT);


}

void loop()
{
  analogWrite(10, 255);
  digitalWrite(9, HIGH);
  digitalWrite(8, LOW);


 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 64);
   //Serial.print("please enter the value\n");

//  while (Serial.available()==0);
//  int value=Serial.parseInt();
//  Serial.println(value);

 //   if(value > 255 || value < 0)
 //   {
  //     Serial.print("\n error, out of range \n");
//    }
//  else
```

Figure 2.2. 4:code-3



```
// C++ code

//int pin=9;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  //Serial.begin (9600);
  //pinMode(pin,OUTPUT);


}

void loop()
{
  analogWrite(10, 127);
  digitalWrite(9, HIGH);
  digitalWrite(8, LOW);


 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 64);
   //Serial.print("please enter the value\n");

//  while (Serial.available()==0);
//  int value=Serial.parseInt();
//  Serial.println(value);

 //   if(value > 255 || value < 0)
 //   {
  //     Serial.print("\n error, out of range \n");
//    }
//  else
```

Figure 2.2. 3:code-3



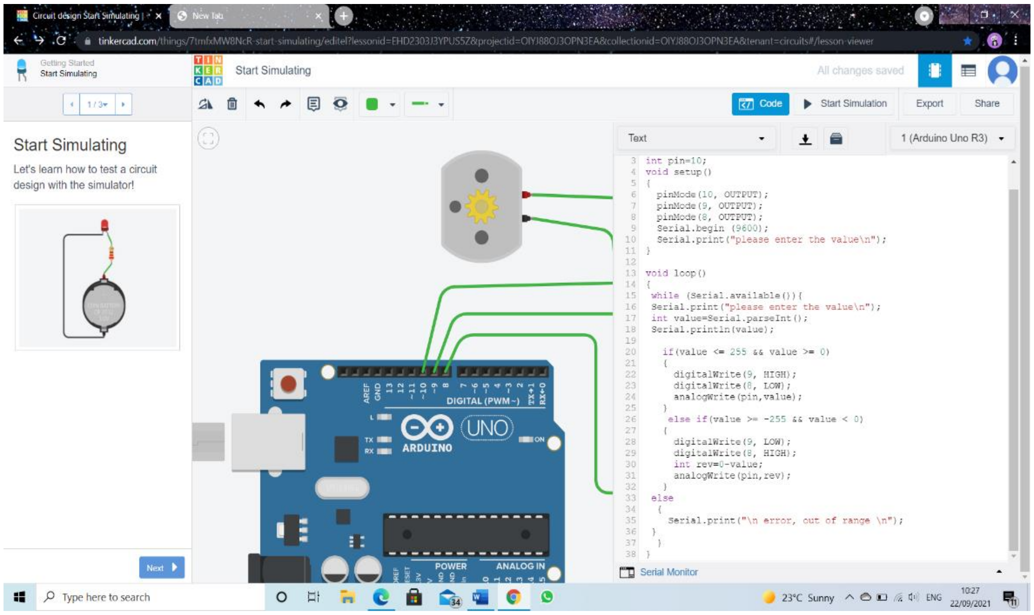
```
// C++ code

//int pin=9;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  //Serial.begin (9600);
  //pinMode(pin,OUTPUT);


}

void loop()
{
  analogWrite(10, 127);
  digitalWrite(9, LOW);
  digitalWrite(8, HIGH);


 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 64);
   //Serial.print("please enter the value\n");

//  while (Serial.available()==0);
//  int value=Serial.parseInt();
//  Serial.println(value);

 //   if(value > 255 || value < 0)
 //   {
  //     Serial.print("\n error, out of range \n");
//    }
//  else
```

Figure 2.2. 5:code-5

- We noticed that pen 8 and 9 control the direction, while pen 10 controls the speed.
- We noticed that when pen 8 is 1, pen 9 is 0 and we generated a PWM signal at pin 10 with a duty cycle of 100% [This means that the motor will run at a speed of 255], That the speed was counter clockwise by 9393 rpm.
- And when we changed pen 8 to 0 , pen 9 to 1, The speed was with the clock by 9393 rpm.
- When we change the duty cycle at pin#10 to 50% [This means that the motor will run at a speed of 127], with 8 is 1, pen 9 is 0, That the speed was counter clockwise by 4580 rpm.
- And when we changed pen 8 to 0 , pen 9 to 1, The speed was with the clock by 4580 rpm.

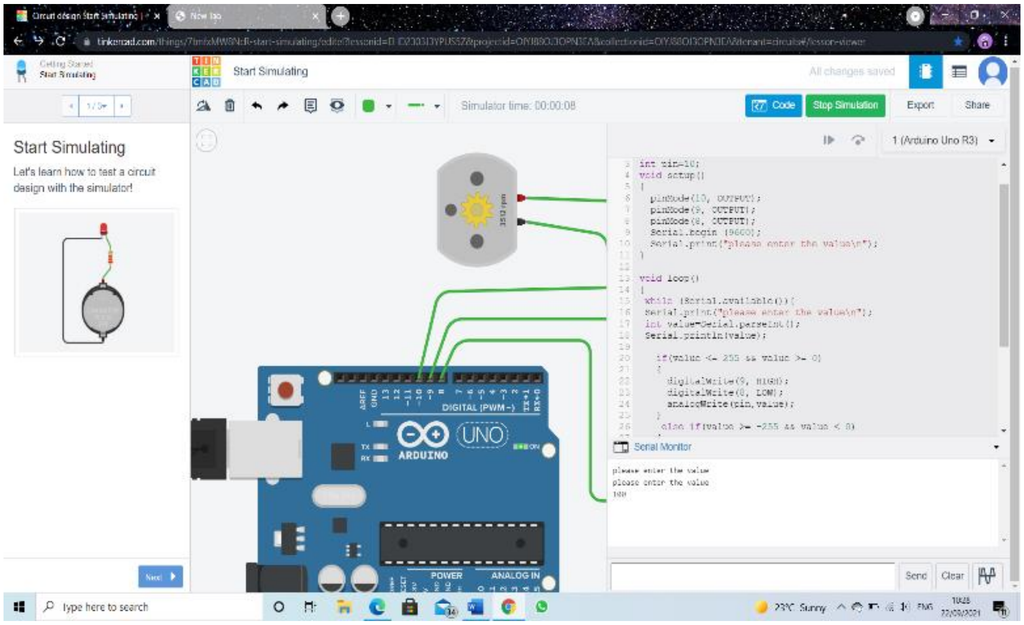
2.2.1 Task 2

Write code such that you can control the direction and speed of a DC-Motor from the keyboard. Show the result to the instructor.



```
int pin=10;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  Serial.begin (9600);
  Serial.print("please enter the value\n");
}

void loop()
{
 while (Serial.available()){
 Serial.print("please enter the value\n");
 int value=Serial.parseInt();
 Serial.println(value);

   if(value <= 255 && value >= 0)
   {
     digitalWrite(9, HIGH);
     digitalWrite(8, LOW);
     analogWrite(pin,value);
   }
    else if(value >= -255 && value < 0)
   {
     digitalWrite(9, LOW);
     digitalWrite(8, HIGH);
     int rev=0-value;
     analogWrite(pin,rev);
   }
 else
  {
    Serial.print("\n error, out of range \n");
 }
  }
}
```

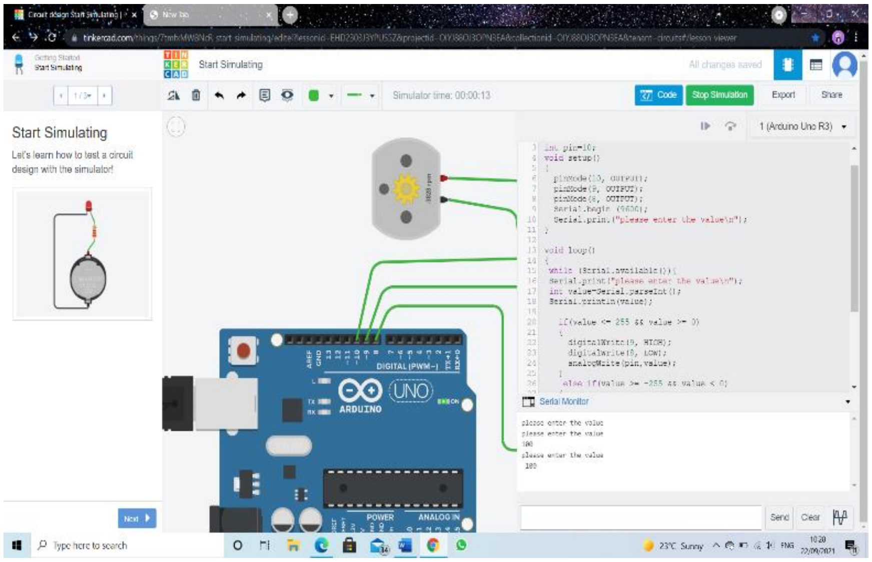
Figure 2.2.1. 1: Task2-code1



```
int pin=10;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  Serial.begin (9600);
  Serial.print("please enter the value\n");
}

void loop()
{
 while (Serial.available()){
 Serial.print("please enter the value\n");
 int value=Serial.parseInt();
 Serial.println(value);

   if(value <= 255 && value >= 0)
   {
     digitalWrite(9, HIGH);
     digitalWrite(8, LOW);
     analogWrite(pin,value);
   }
    else if(value >= -255 && value < 0)
```

```
please enter the value
please enter the value
100
```

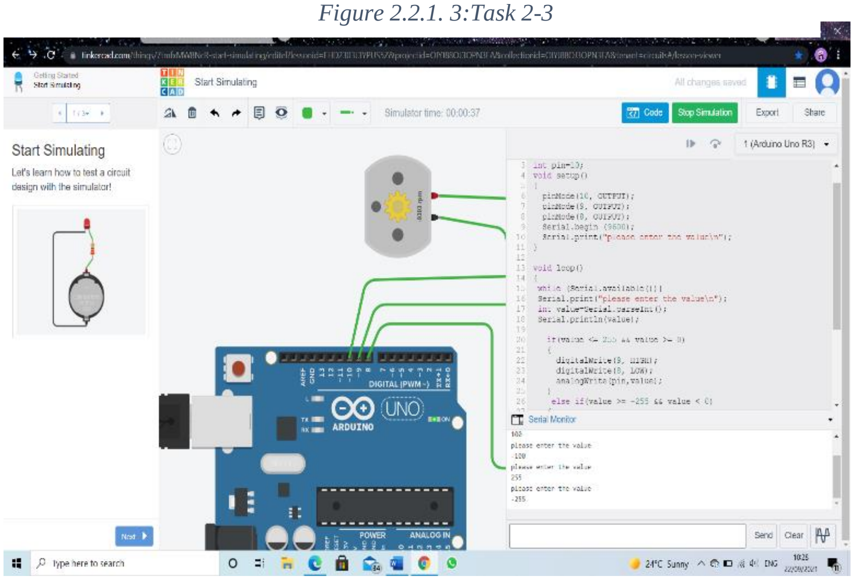
Figure 2.2.1. 2:Task 2-2



```
int pin=10;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  Serial.begin (9600);
  Serial.print("please enter the value\n");
}

void loop()
{
 while (Serial.available()){
 Serial.print("please enter the value\n");
 int value=Serial.parseInt();
 Serial.println(value);

   if(value <= 255 && value >= 0)
   {
     digitalWrite(9, HIGH);
     digitalWrite(8, LOW);
     analogWrite(pin,value);
   }
    else if(value >= -255 && value < 0)
```

```
please enter the value
please enter the value
100
please enter the value
 100
```

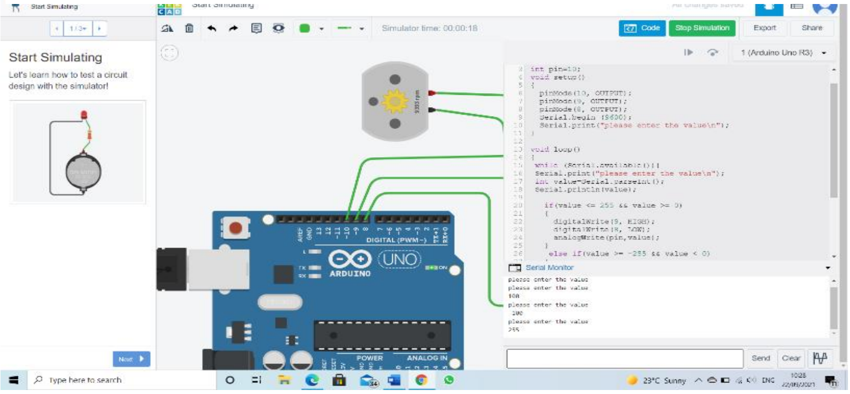
Figure 2.2.1. 3:Task 2-3



```
int pin=10;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  Serial.begin (9600);
  Serial.print("please enter the value\n");
}

void loop()
{
 while (Serial.available()){
 Serial.print("please enter the value\n");
 int value=Serial.parseInt();
 Serial.println(value);

   if(value <= 255 && value >= 0)
   {
     digitalWrite(9, HIGH);
     digitalWrite(8, LOW);
     analogWrite(pin,value);
   }
    else if(value >= -255 && value < 0)
```

```
100
please enter the value
-100
please enter the value
255
please enter the value
-255
```

Figure 2.2.1. 4:Task 2-4



```
int pin=10;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  Serial.begin (9600);
  Serial.print("please enter the value\n");
}

void loop()
{
 while (Serial.available()){
 Serial.print("please enter the value\n");
 int value=Serial.parseInt();
 Serial.println(value);

   if(value <= 255 && value >= 0)
   {
     digitalWrite(9, HIGH);
     digitalWrite(8, LOW);
     analogWrite(pin,value);
   }
    else if(value >= -255 && value < 0)
```

```
please enter the value
please enter the value
100
please enter the value
 100
please enter the value
255
```

Figure 2.2.1. 5:Task 2-5

3. Conclusion

We learned about Tinkercad application and how to install circuits on it, and also learned how to control the speed and direction of a DC motor using Pulse Width Modulation (PWM). And we used the Arduino to generate a PWM signal to drive the DC motor at different speeds.

4. References

[1]. https://byjus.com/physics/dc-motor/

[2]. https://www.monolithicpower.com/en/using-mosfet-drivers-for-motor-drives

[3] https://www.arduino.cc/en/guide/introduction

[4] https://learn.sparkfun.com/tutorials/pulse-width-modulation/all

5. Appendices

5.1. Appendix A

```
// C++ code

int pin=9;
void setup()
{
  //pinMode(9, OUTPUT);
 Serial.begin (9600);
  pinMode(pin,OUTPUT);

}

void loop()
{

 // digitalWrite(9, HIGH);
 //digitalWrite(9, LOW);
  //analogWrite(9, 200);
  // analogWrite(9, 64);
   Serial.print("please enter the value\n");

  while (Serial.available()==0);
  int value=Serial.parseInt();
  Serial.println(value);

if(value > 255 || value < 0)
{
      Serial.print("\n error, out of range \n");
    }
  else
  {
    analogWrite(pin,value);
  }
}
```

5.2. Appendix B

```
// C++ code

int pin=10;
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  Serial.begin (9600);
  Serial.print("please enter the value\n");
}

void loop()
{
 while (Serial.available()){
 Serial.print("please enter the value\n");
 int value=Serial.parseInt();
 Serial.println(value);

   if(value <= 255 && value >= 0)
   {
     digitalWrite(9, HIGH);
```

```
      digitalWrite(8, LOW);
      analogWrite(pin,value);
    }
     else if(value >= -255 && value < 0)
    {
      digitalWrite(9, LOW);
      digitalWrite(8, HIGH);
      int rev=0-value;
      analogWrite(pin,rev);
    }
 else
  {
    Serial.print("\n error, out of range \n");
 }
  }
}
```