MCTA 3203
MECHATRONICS SYSTEM INTEGRATION
SEMESTER 1, 25/26

WEEK 05 : CONTROLLING A DC MOTOR USING L298P MOTOR DRIVER SHIELD AND GPIO

SECTION 2
GROUP 19
LECTURER : DR WAHJU SEDIONO
DATE OF EXPERIMENT : 3 NOVEMBER 2025
DATE OF SUBMISSION : 10 NOVEMBER 2025

PREPARED BY :

NAME	MATRIC NUMBER
MUHAMMAD IMRAN BIN VEDDIN	2316409
ALYA KHAIRAH BINTI KHAIRUL JAMIL	2317100
MUHAMMAD IRFAN RUSHDI BIN ASRI	2319011

ABSTRACT

This experiment investigates the control of a DC motor using the L298P Motor Driver Shield with an Arduino UNO. The objective is to understand how motor speed and direction can be manipulated through Pulse Width Modulation (PWM) and GPIO pins. By varying PWM duty cycles, the motor's rotational speed is adjusted, while digital signals on direction pins determine rotational direction. The experiment demonstrates how hardware and software integration can achieve efficient DC motor control. Observations confirm that higher PWM values increase motor speed, validating the theoretical relationship between duty cycle and output voltage. The L298P shield provides a reliable interface for motor control applications in robotics and automation systems.

TABLE OF CONTENTS

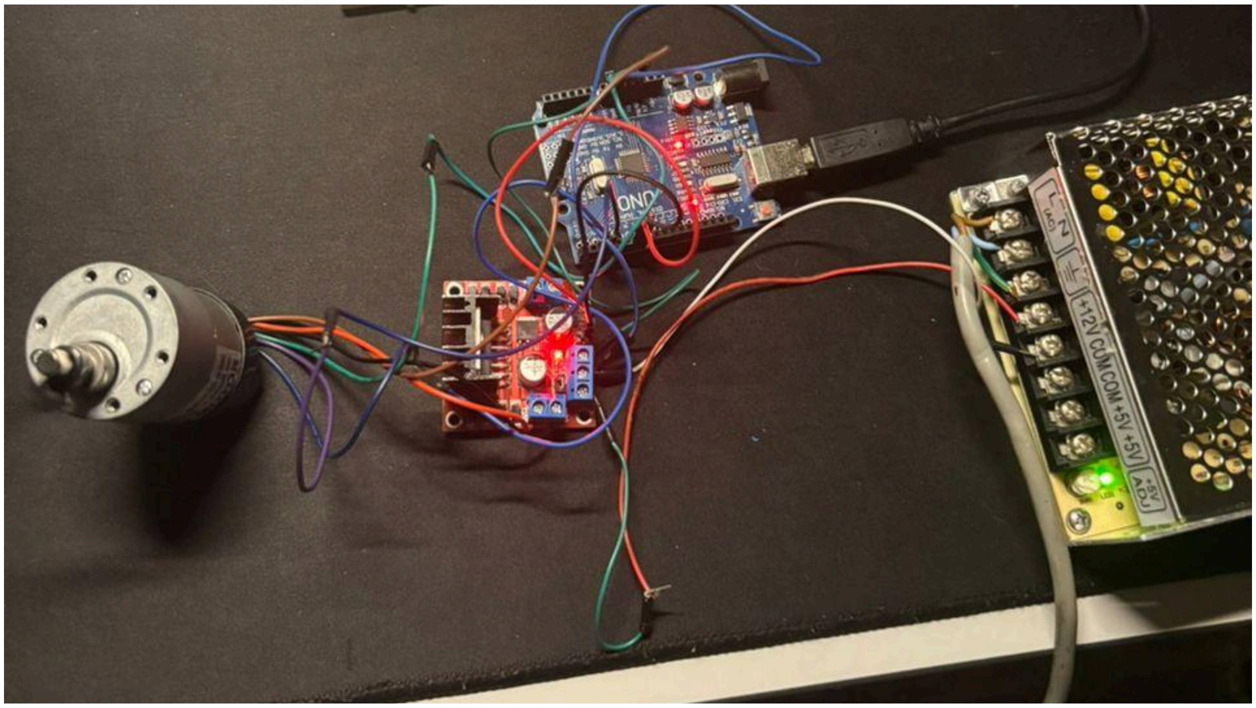
1.0 INTRODUCTION

This experiment introduces the L298P H-bridge motor driver shield and demonstrates how to control a DC motor’s speed and direction using an Arduino's GPIO and PWM pins. L298P shield will mount on an Arduino UNO, connect a 6–12V DC motor with an external power supply). The goals are to understand how the ENA/ENB (PWM) and INx pins work and observe the effect of PWM duty cycle on motor speed. To conclude, the main purpose is to learn how to control the motor’s speed and direction using PWM (Pulse Width Modulation)and digital control signals.

2.0 MATERIALS AND EQUIPMENT

Compenent	**Quantity**	**Notes**
Arduino board	1	Arduino Uno
USB cable	1	For connection and power.
Motor Driver Module	1	L298N
DC Gear Motor	1	SPG30-20k
External Power Supply	1	12V
Jumper wires	As needed	
Computer	1	Running Arduino IDE, and required libraries.

3.0 EXPERIMENTAL SETUP



3.1 Circuit assembly :

Power Connections

1. Power Supply (12 V) was connected to L298N VCC.
2. Power Supply GND was connected to L298N GND
3. Arduino is powered via USB (from your computer).

L298N Connections

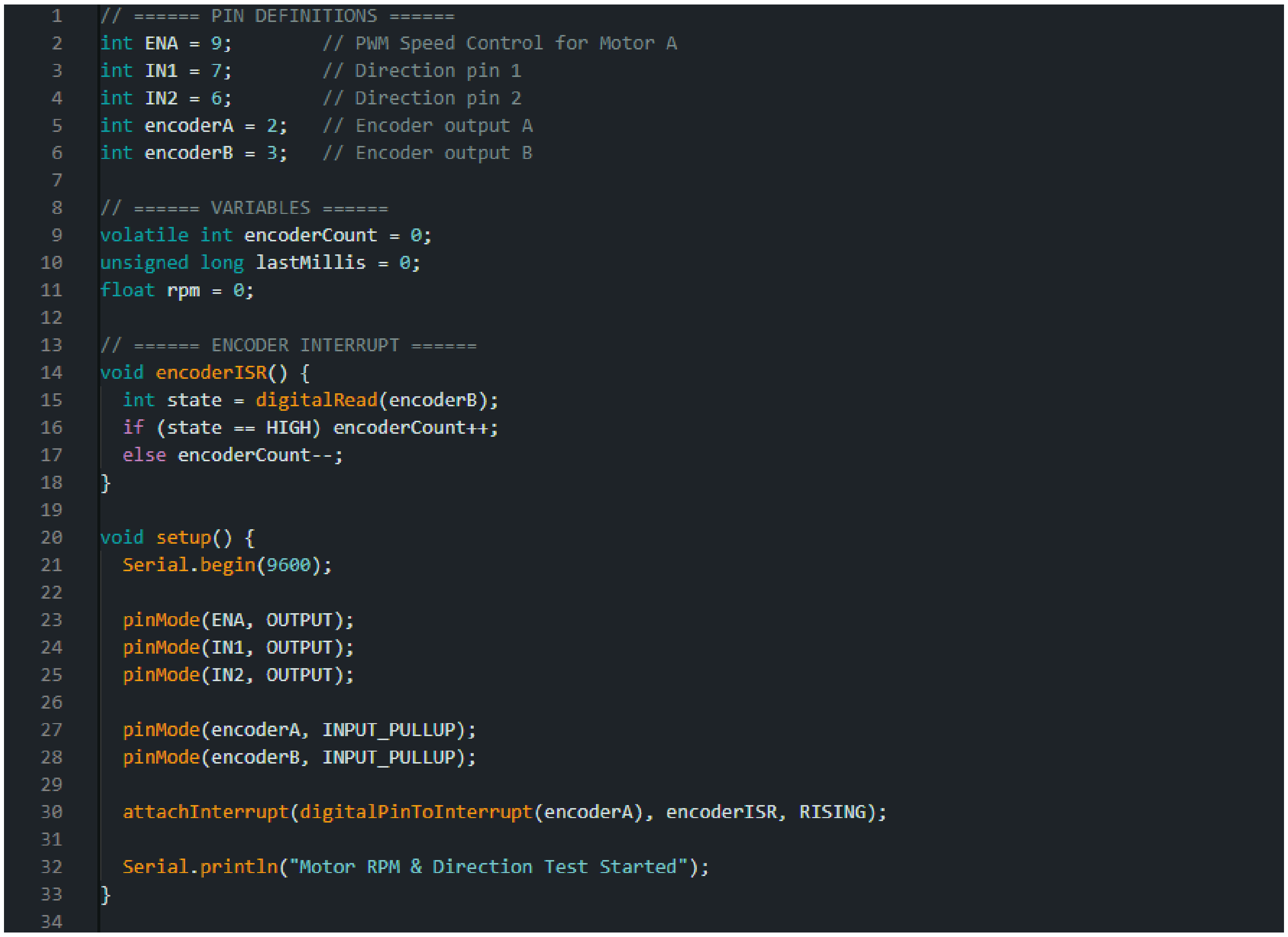
1. L298N IN1 and IN2 was connected to Arduino digital pin (D7 and D6) to control motor direction.
2. L298N OUT1 and OUT2 was connected to DC motor terminals.

4.0 METHODOLOGY

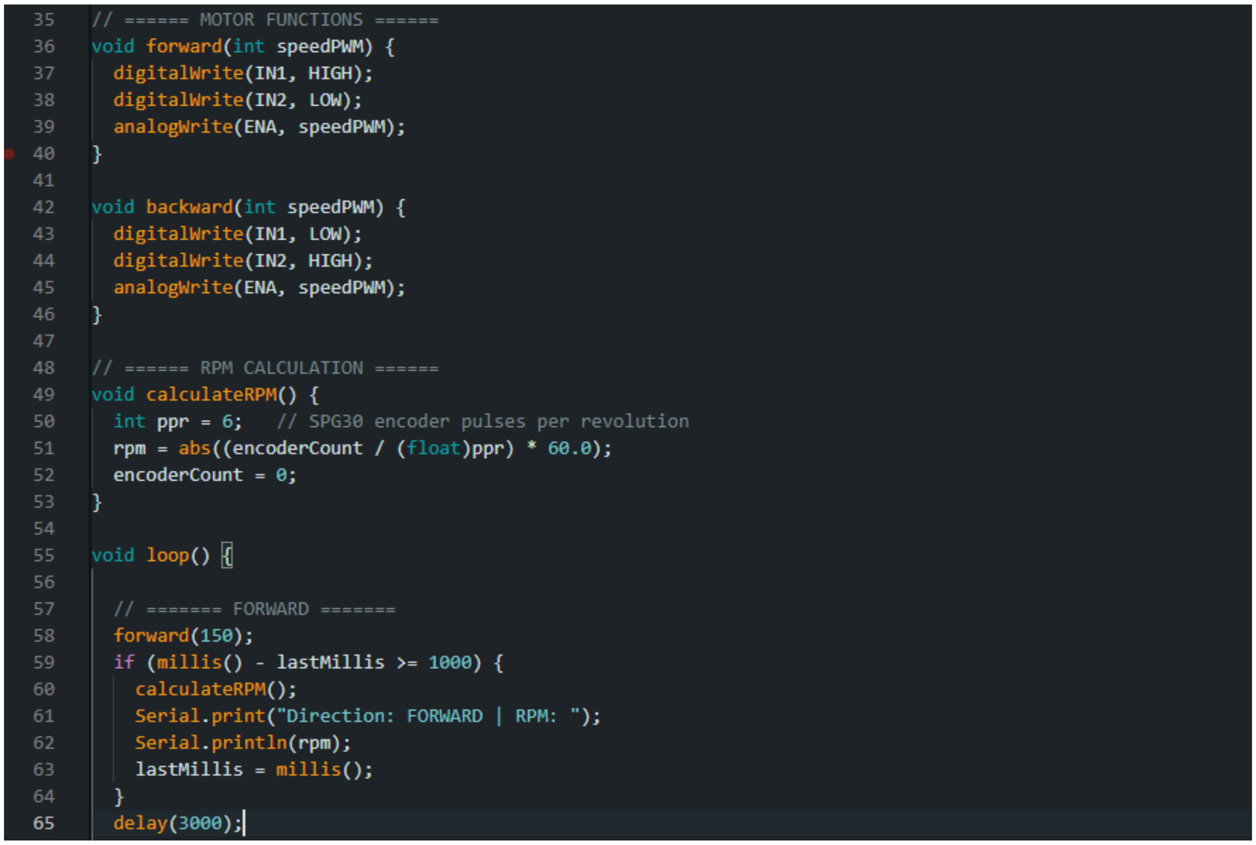
1. The circuit was built according to the circuit setup instructions.
2. The Arduino code was uploaded into the Arduino Uno.
3. DC gear motor rotate clockwise and counter clockwise based on the coding from arduino.

4. The RPM of the DC gear motor was measured and sent to the Arduino to be displayed on the Serial Monitor.

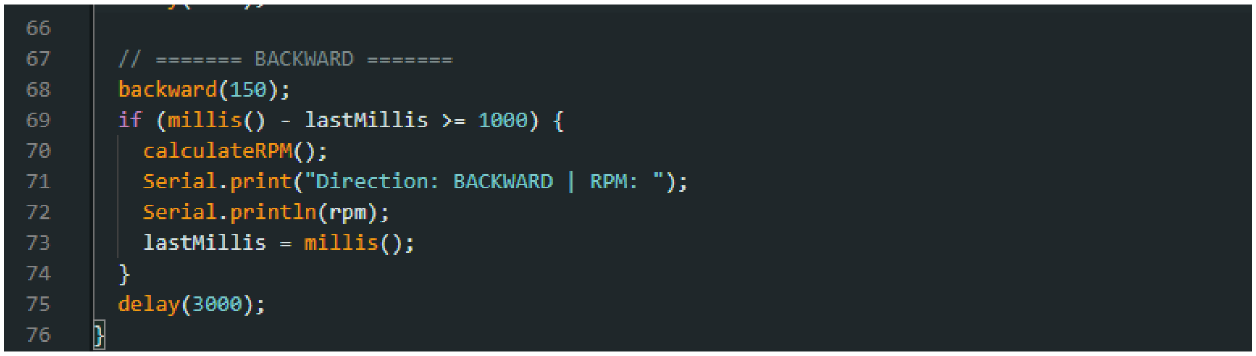
4.1 Code used :



```
// ====== PIN DEFINITIONS ======
int ENA = 9;        // PWM Speed Control for Motor A
int IN1 = 7;        // Direction pin 1
int IN2 = 6;        // Direction pin 2
int encoderA = 2;   // Encoder output A
int encoderB = 3;   // Encoder output B

// ====== VARIABLES ======
volatile int encoderCount = 0;
unsigned long lastMillis = 0;
float rpm = 0;

// ====== ENCODER INTERRUPT ======
void encoderISR() {
  int state = digitalRead(encoderB);
  if (state == HIGH) encoderCount++;
  else encoderCount--;
}

void setup() {
  Serial.begin(9600);

  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);

  pinMode(encoderA, INPUT_PULLUP);
  pinMode(encoderB, INPUT_PULLUP);

  attachInterrupt(digitalPinToInterrupt(encoderA), encoderISR, RISING);

  Serial.println("Motor RPM & Direction Test Started");
}

```



```
// ====== MOTOR FUNCTIONS ======
void forward(int speedPWM) {
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  analogWrite(ENA, speedPWM);
}

void backward(int speedPWM) {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  analogWrite(ENA, speedPWM);
}

// ====== RPM CALCULATION ======
void calculateRPM() {
  int ppr = 6;   // SPG30 encoder pulses per revolution
  rpm = abs((encoderCount / (float)ppr) * 60.0);
  encoderCount = 0;
}

void loop() {

  // ======= FORWARD =======
  forward(150);
  if (millis() - lastMillis >= 1000) {
    calculateRPM();
    Serial.print("Direction: FORWARD | RPM: ");
    Serial.println(rpm);
    lastMillis = millis();
  }
  delay(3000);
```



```

  // ======= BACKWARD =======
  backward(150);
  if (millis() - lastMillis >= 1000) {
    calculateRPM();
    Serial.print("Direction: BACKWARD | RPM: ");
    Serial.println(rpm);
    lastMillis = millis();
  }
  delay(3000);
}
```

5.0 DATA COLLECTION

During the experiment, data were collected by running the DC motor at various PWM duty cycles using the L298N motor driver shield connected to the Arduino UNO. The motor's speed and direction were controlled through PWM pins, while the output speed was monitored using an encoder. The encoder generated pulses acording to the motor's rotation, and these pulses were counted by the Arduino to calculate the Revolutions Per Minute (RPM).

6.0 DATA ANALYSIS

The collected data were analyzed to determine the effect of PWM duty cycle on the DC motor's rotational speed. As the PWM duty cycle increased, the motor received a higher average voltage, resulting in a proportional increase in speed. This confirmed the theoretical relationship between PWM control and motor performance. Minor variations in RPM readings were observed, likely due to mechanical friction and small differences in motor load. Overall, the analysis demonstrated that PWM provides an efficient and precise method for controlling DC motor speed and direction.

7.0 RESULT

Task 1: Explain the function of the ENA and ENB pins.

The ENA and ENB pins on the L298P motor driver are used to enable and control the speed of Motor A and Motor B respectively. They are connected to PWM-capable pins on the Arduino. When a PWM signal is applied to these pins, the motor speed can be varied by changing the duty cycle of the PWM. Setting the ENA or ENB pin HIGH enables the corresponding motor, while setting it LOW disables it.

Task 2: Describe the reason PWM is used for speed control.

PWM (Pulse Width Modulation) is used for speed control because it allows the adjustment of the motor's effective voltage by varying the ON and OFF time of the signal. This method is efficient because it minimizes power loss and provides smooth control over motor speed. A higher duty cycle means the motor receives more average voltage, increasing speed, while a lower duty cycle reduces speed.

Task 3: Describe the outcome when both IN1 and IN2 are set HIGH.

When both IN1 and IN2 inputs are set to HIGH, both sides of the motor terminal receive the same voltage level. This causes no potential difference across the motor terminals, which effectively stops the motor immediately. This condition is known as active braking, as the motor halts due to short-circuiting both terminals together through the driver.

Task 4: Explain how braking can be implemented using the L298P.

Braking is implemented in the L298P by setting both direction pins (IN1 and IN2 for Motor A, or IN3 and IN4 for Motor B) to HIGH. This configuration short-circuits the motor terminals, producing an immediate stop due to the back electromotive force (EMF) being dissipated internally. This technique is useful for quickly stopping or changing direction without coasting.

Task 5: Modify the code to control two DC motors simultaneously.

To control two motors simultaneously, both sets of enable and direction pins (ENA, IN1, IN2 for Motor A; ENB, IN3, IN4 for Motor B) must be defined and controlled in the Arduino code. An example modification is as follows:

```
// ====== PIN DEFINITIONS ======
// Motor A
int ENA = 9;        // PWM Speed Control for Motor A
int IN1 = 7;        // Direction pin 1 for Motor A
int IN2 = 6;        // Direction pin 2 for Motor A
int encoderA = 2;   // Encoder output A (Motor A)
int encoderB = 3;   // Encoder output B (Motor A)

// Motor B
int ENB = 10;       // PWM Speed Control for Motor B
int IN3 = 12;       // Direction pin 1 for Motor B
int IN4 = 13;       // Direction pin 2 for Motor B

// ====== VARIABLES ======
volatile int encoderCount = 0;
unsigned long lastMillis = 0;
float rpm = 0;

// ====== ENCODER INTERRUPT ======
void encoderISR() {
  int state = digitalRead(encoderB);
  if (state == HIGH) encoderCount++;
  else encoderCount--;
}

void setup() {
  Serial.begin(9600);

  // Motor A
  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(encoderA, INPUT_PULLUP);
  pinMode(encoderB, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(encoderA), encoderISR, RISING);

  // Motor B
  pinMode(ENB, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);

  Serial.println("Dual Motor RPM & Direction Test Started");
}

// ====== MOTOR FUNCTIONS ======
void forward(int speedPWM) {
  // Motor A
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  analogWrite(ENA, speedPWM);

  // Motor B
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  analogWrite(ENB, speedPWM);
}

void backward(int speedPWM) {
  // Motor A
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  analogWrite(ENA, speedPWM);

  // Motor B
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
  analogWrite(ENB, speedPWM);
}

// ====== RPM CALCULATION ======
void calculateRPM() {
  int ppr = 6;   // Encoder pulses per revolution (SPG30)
  rpm = abs((encoderCount / (float)ppr) * 60.0);
  encoderCount = 0;
}

// ====== MAIN LOOP ======
void loop() {
  // ======= FORWARD =======
  forward(150);
  if (millis() - lastMillis >= 1000) {
    calculateRPM();
    Serial.print("Direction: FORWARD | RPM (Motor A): ");
    Serial.println(rpm);
    lastMillis = millis();
  }
  delay(3000);

  // ======= BACKWARD =======
  backward(150);
  if (millis() - lastMillis >= 1000) {
    calculateRPM();
    Serial.print("Direction: BACKWARD | RPM (Motor A): ");
    Serial.println(rpm);
    lastMillis = millis();
  }
  delay(3000);
}
```

8.0 DISCUSSION

In this experiment, a DC motor was controlled using the L298N motor driver shield connected to an Arduino UNO. The objective was to control the motor's speed and direction and to measure its rotational speed (RPM) using an encoder. The L298N driver acts as an interface between the Arduino and the motor, allowing the microcontroller to control higher current required by the motor safely.

The provided Arduino code uses the ENA pin (PWM) to adjust motor speed and the IN1–IN2 pins to control direction. When IN1 is set HIGH and IN2 is LOW, the motor rotates forward; when IN1 is LOW and IN2 is HIGH, it rotates backward. By applying a PWM signal to the ENA pin, the Arduino can vary the motor's input voltage effectively, changing its speed.

An encoder attached to the motor shaft provides feedback for speed measurement. The encoder produces pulses as the motor rotates, and each pulse represents a fraction of one full revolution. The code uses interrupts on the encoder output (encoderA) to count these pulses accurately in real time, even when the motor spins quickly. Every second, the program calculates the motor's RPM (Revolutions Per Minute) using the number of pulses counted and the encoder's pulse-per-revolution (PPR) value.

During the experiment, the serial monitor displayed the motor's rotation direction and its measured RPM for both forward and reverse motion. The motor speed increased proportionally with higher PWM values. The readings also showed small variations in RPM due to mechanical friction and minor differences in motor load. The encoder helped verify the accuracy of speed control, demonstrating how feedback sensors are important in mechatronic systems.

9.0 CONCLUSION

In conclusion, the experiment successfully achieved its objectives by demonstrating the use of the L298P Motor Driver Shield for controlling a DC motor's speed and direction. The integration of PWM control and GPIO switching proved effective for efficient and precise motor control. Students gained practical understanding of motor driver operation, PWM principles, and Arduino interfacing.

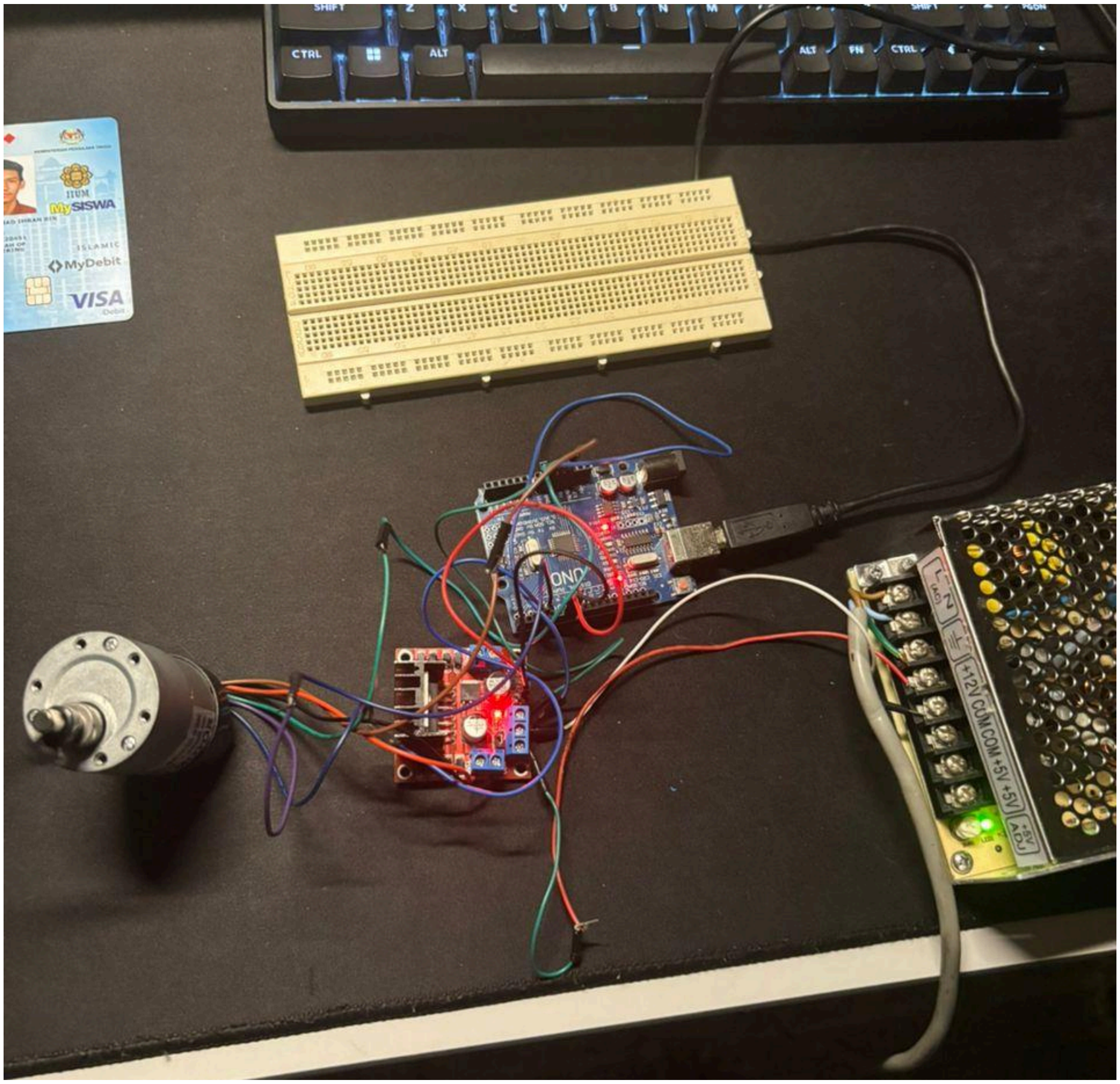
10.0 RECOMMENDATIONS

1. Always check the wiring before turning on the power to avoid damage.
2. Use an external power supply (9–12V) for the motor instead of the Arduino 5V pin.
3. Start testing the motor at a low speed (low PWM value) before increasing to full speed.
4. Record motor speed carefully for each PWM value and take the average of several readings.

11.0 References

[1] https://www.instructables.com/Tutorial-for-L298-2Amp-Motor-Driver-Shield-forArd/
Tutorial for L298 2Amp Motor Driver Shield for Arduino
[2] https://github.com/CytronTechnologies/L298P_MotorDriverShield
L298P_MotorDriverShield
[3] https://my.cytron.io/p-shield-l298p-motor-driver-with-gpio?r=1
Shield L298P Motor Driver with GPIO
[4]
https://docs.cirkitdesigner.com/component/a25f6e70-294e-42fdb77c-9e9349b76b9a/l298p-drive-shield
How to Use L298P drive shield: Examples, Pinouts, and Specs

APPENDICES



ACKNOWLEDGEMENTS

Special thanks to DR WAHJU SEDIONO for their guidance and support during this experiment .

STUDENTS'S DECLARATION

CERTIFICATE OF ORIGINALITY AND AUTHENTICITY

This is to certify that we are **responsible** for the work submitted in this report, that **the original work** is our own except as specific in references and acknowledgement, and that the original work contained herein have not been untaken or done by unspecified sources or person.

We hereby certify that this report has **not been done by only one individual** and **all of us have contributed to the report.** The length of contribution to the reports by each individual is noted within this certificate.

We also hereby certify that we have **read** and **understand** the content of the total report and no further improvement on the reports is needed from any of individual's contributor to the report.

We, therefore, agreed unanimously that this report shall be submitted for **marking** and this **final printed report** have been **verified by us.**

Signature:	[illegible]	Read	/
Name:	MUHAMMAD IMRAN BIN VEDDIN	Understand	/
Matric Number:	2316409	Agree	/
Contribution:	Materials and equipment, experimental setup, methodology, data collection and data analysis		

Signature:	[illegible]	Read	/

Name:	ALYA KHAIRAH BINTI KHAIRUL JAMIL	Understand	/
Matric Number:	2317100	Agree	/
Contribution:	Introduction, Discussion and Recommendations		

Signature:	[illegible]	Read	/
Name:	MUHAMMAD IRFAN RUSHDI BIN ASRI	Understand	/
Matric Number:	2319011	Agree	/
Contribution:	Abstract , Results, conclusions		