MECHATRONICS SYSTEM INTEGRATION (MCTA 3203)

SEMESTER 1 2025/2026

WEEK 2: DIGITAL LOGIC DESIGN

SECTION 2

GROUP 16

LECTURER: ZULKIFLI BIN ZAINAL ABIDIN & WAHJU SEDIONO

Date of Experiment: Monday, 13 October 2025

Date of Submission: Monday, 20 October 2025

NO.	GROUP MEMBERS	MATRIC NO
1.	MUHAMMAD HAIKAL ZULHILMI BIN FAIROF	2313025
2.	MUHAMMAD FAIZ IZWAN BIN NOR EFFENDI	2313509
3.	AINUL JAARIAH BINTI ALIUDIN	2312664

ABSTRACT

In this project, a 7-segment display is interfaced with an Arduino to create a numerical display that cycles between 0 and 9 and resets to 0. The numbers are cycled by an increment button, and the display is reset to 0 by another button. The Arduino Mega 2650 is used in this project to control the buttons and the numerical display. This straightforward circuit helps students become accustomed to more complex systems by demonstrating fundamental digital logic systems and Arduino programming. The experiment demonstrated the potential uses and applications of systems such as interactive displays that highlight the fundamentals of electrical systems and Arduino programming, as well as digital counters.

TABLES OF CONTENTS

1.0 INTRODUCTION

This report explains how to integrate an Arduino Board with a 7-segment display to produce a cyclable numerical display that changes from 0 to 9 in response to button presses. The goal of this project is to assist students in comprehending the fundamental concepts of digital electronics, microcontroller programming, and other fields. The students will program the Arduino microcontroller, connect the parts to a breadboard, and create an algorithm for the counter using the instructions in the manual. Students can explore microcontrollers and electrical systems in depth and realize their imaginative ideas through this project.

2.0 MATERIALS AND EQUIPMENT

- Arduino Mega 2650 board
- Common cathode 7-segment display
- 220-ohm resistors (7 of them)
- Pushbuttons (2 or more)
- Jumper wires
- Breadboard
- Circuit Setup

3.0 EXPERIMENTAL SETUP

This section describes the hardware configuration and circuit assembly used in the project.

3.1 Circuit Setup

1. Each of the seven segments (a–g) of the display was connected to separate digital pins on the Arduino (e.g., D0–D6).
2. The common cathode of the display was connected to one of the Arduino GND pins.
3. 220-ohm resistors were placed in series with each segment to limit the current.
4. One leg of each pushbutton was connected to separate digital pins (e.g., D9 and D10), while the other leg was connected to GND.
5. 10K-ohm pull-up resistors were used by connecting one end of each resistor to the digital pin and the other to the 5V output.

3.2 Circuit Assembly

The circuit was assembled on a breadboard following the designed connection layout. All components, including the Arduino board, 7-segment display, resistors, and pushbuttons, were carefully connected to ensure proper functionality, as shown in Figure 3.2.1 below.

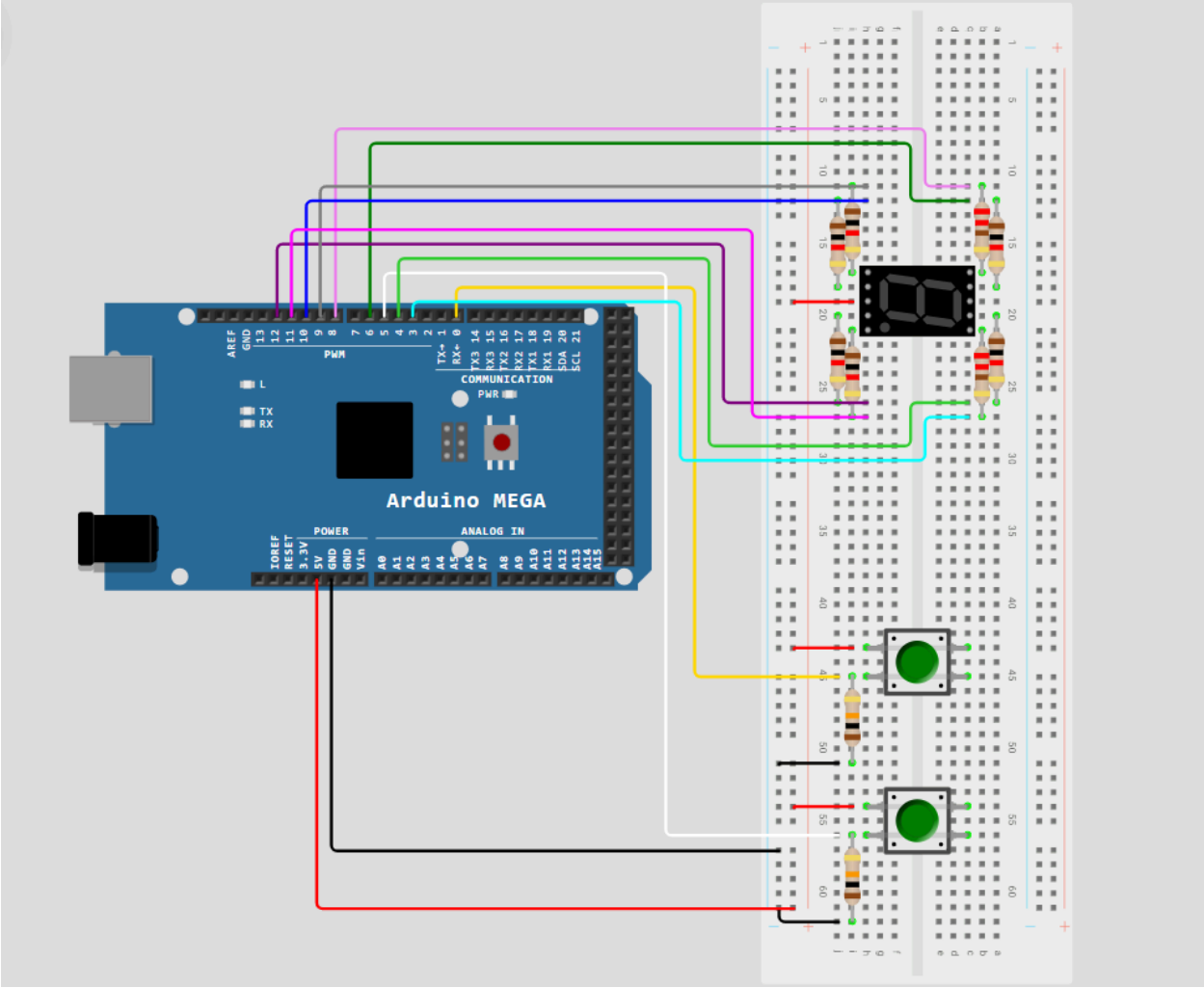

Figure 3.2.1: *Circuit assembly of the 7-segment display counter system.*

4.0 METHODOLOGY

The experiment was carried out through a series of steps to build, program, and test the 7-segment display counter system.

1. Build the circuit according to the circuit setup instructions.
2. Upload the Arduino code to your Arduino Mega 2650.
3. Open the Serial Monitor in the Arduino IDE.
4. Press the increment button to increase the count. The 7-segment display should show the numbers from 0 to 9 sequentially.
5. Press the reset button to reset the count to 0.

4.1 System Preparation

The Arduino Mega 2560 board was connected to a computer via USB. The assembled circuit from the previous setup was verified to ensure all connections were correct and components were securely placed on the breadboard. The Arduino IDE was launched to begin the programming process.

4.2 Arduino Programming

```
//Define segment
const int segmentA = 4;
const int segmentB = 3;
const int segmentC = 12;
const int segmentD = 10;
const int segmentE = 9;
const int segmentF = 6;
const int segmentG = 8;

//Define button
const int buttonIncrement = 0;    // Button to increment the count
```

```
const int buttonReset = 5;        // Button to reset the count
int count = 0;                     // Current count

// Button states
bool lastButtonStateIncrement = LOW;
bool lastButtonStateReset = LOW;

void setup() {
// Initialize the digital pins as OUTPUTS and INPUTS
pinMode (segmentA, OUTPUT);
pinMode (segmentB, OUTPUT);
pinMode (segmentC, OUTPUT);
pinMode (segmentD, OUTPUT);
pinMode (segmentE, OUTPUT);
pinMode (segmentF, OUTPUT);
pinMode (segmentG, OUTPUT);
pinMode (buttonIncrement, INPUT);
pinMode (buttonReset, INPUT);
}

void loop() {
bool currentButtonStateIncrement = digitalRead (buttonIncrement);
bool currentButtonStateReset = digitalRead (buttonReset);
// Detect button press (LOW means pressed due to pull-up)

if (currentButtonStateIncrement == LOW && lastButtonStateIncrement ==
HIGH)  //--------------------------Increment count
{
count++;
```

```
if (count > 9)
{
count = 0;// Wrap around if count exceeds 9
}
delay (200);
}

if (currentButtonStateReset == LOW && lastButtonStateReset == HIGH)
//---------------------------Reset Button
{
count = 0; // Reset count to 0
delay (200);
}

// Update last button state
lastButtonStateIncrement = currentButtonStateIncrement;
lastButtonStateReset = currentButtonStateReset;
// Display the current count on the 7-segment display

displayNumber (count); //Function to display number
}

void displayNumber (int count)
{

digitalWrite (segmentA, HIGH);
digitalWrite (segmentB, HIGH);
digitalWrite (segmentC, HIGH);
```

```
digitalWrite (segmentD, HIGH);  //--------------------------- Reset all
segments first

digitalWrite (segmentE, HIGH);

digitalWrite (segmentF, HIGH);

digitalWrite (segmentG, HIGH);


switch (count)


{

case 0:

digitalWrite (segmentA, LOW);

digitalWrite (segmentB, LOW);

digitalWrite (segmentC, LOW);  //--------------------------- Display 0

digitalWrite (segmentD, LOW);

digitalWrite (segmentE, LOW);

digitalWrite (segmentF, LOW);

break;


case 1:

digitalWrite(segmentB, LOW);   //--------------------------- Display 1

digitalWrite(segmentC, LOW);

break;


case 2:

digitalWrite (segmentA, LOW);

digitalWrite (segmentB, LOW);

digitalWrite (segmentD, LOW);   //--------------------------- Display 2

digitalWrite (segmentE, LOW);

digitalWrite (segmentG, LOW);
```

```
break;

case 3:
digitalWrite (segmentA, LOW);
digitalWrite (segmentB, LOW);
digitalWrite (segmentC, LOW);    //---------------------------- Display 3
digitalWrite (segmentD, LOW);
digitalWrite (segmentG, LOW);
break;

case 4:
digitalWrite (segmentB, LOW);
digitalWrite (segmentC, LOW);    //---------------------------- Display 4
digitalWrite (segmentF, LOW);
digitalWrite (segmentG, LOW);
break;

case 5:
digitalWrite (segmentA, LOW);
digitalWrite (segmentC, LOW);
digitalWrite (segmentD, LOW);    //---------------------------- Display 5
digitalWrite (segmentF, LOW);
digitalWrite (segmentG, LOW);
break;

case 6:
digitalWrite (segmentA, LOW);
digitalWrite (segmentC, LOW),
```

```
digitalWrite (segmentD, LOW);    //---------------------------- Display 6
digitalWrite (segmentE, LOW);
digitalWrite (segmentF, LOW);
digitalWrite (segmentG, LOW);
break;

case 7:
digitalWrite (segmentA, LOW);
digitalWrite (segmentB, LOW);    //---------------------------- Display 7
digitalWrite (segmentC, LOW);
break;

case 8:
digitalWrite (segmentA, LOW);
digitalWrite (segmentB, LOW);
digitalWrite (segmentC, LOW);
digitalWrite (segmentD, LOW);    //---------------------------- Display 8
digitalWrite (segmentE, LOW);
digitalWrite (segmentF, LOW);
digitalWrite (segmentG, LOW);
break;

case 9:
digitalWrite (segmentA, LOW);
digitalWrite (segmentB, LOW);
digitalWrite (segmentC, LOW);    //---------------------------- Display 9
digitalWrite (segmentD, LOW);
digitalWrite (segmentF, LOW);
```

```
digitalWrite (segmentG, LOW);

break;

}}
```

4.3 Testing Procedure

After uploading the code, the circuit was tested by pressing the increment button to increase the count from 0 to 9 and using the reset button to return the display to 0. The operation was repeated several times to ensure stability and reliability.

4.4 Observation and Data Recording

The displayed numbers were observed and recorded to confirm that each button press produced the correct numerical output. The timing and responsiveness of the display were also noted.

4.5 Troubleshooting

If the system failed to function correctly, the wiring connections and code logic were reviewed and corrected. Adjustments were made to ensure consistent performance of the display counter system.

5.0 DATA COLLECTION

Observations:

Trial	Initial Digit	Button Presses	Final Digit	Notes
1	0	Inc	1	Normal inc
2	1	Inc	2	Normal inc
3	2	Inc	3	Normal inc
4	3	Inc	4	Normal inc
5	4	Inc	5	Normal inc

6	5	Inc	6	Normal inc
7	6	Inc	7	Normal inc
8	7	Inc	8	Normal inc
9	8	Inc	9	Normal inc
10	9	reset	0	Reset worked
11	0	Inc x10	0	Wrapped correctly

6.0 DATA ANALYSIS

Data collection shows that the 7-segment display counter operated as planned during the test. The display progressed steadily from 0 to 9 with each increment button click, and the reset button consistently reset the count to 0. Furthermore, the correct implementation of the counting mechanism was confirmed when the counter appropriately wrapped around back to 0 after ten presses of the increment button from 0. All things considered, the data show that the Arduino Mega and program consistently produced the desired outputs and controlled the 7-segment display.

7.0 RESULTS

The results of the study showed that the seven-segment display counter system operated as planned. Every time a button was pressed, the display displayed the appropriate number between 0 and 9, but the reset button reset the count back to 0. The visual outputs seen during testing are represented in the figures below, which show each number on the seven-segment display.

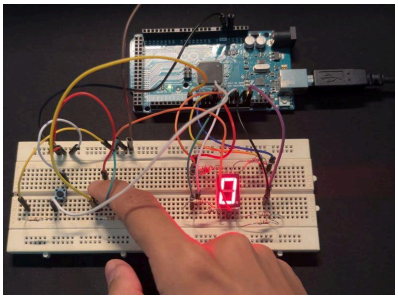

Figure 7.1: *The 7-Segment displayed no.0*

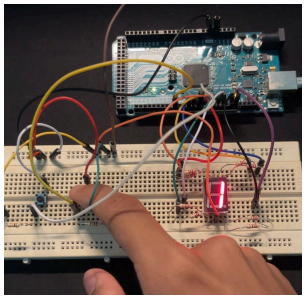

Figure 7.2: *The 7-Segment displayed no.1*

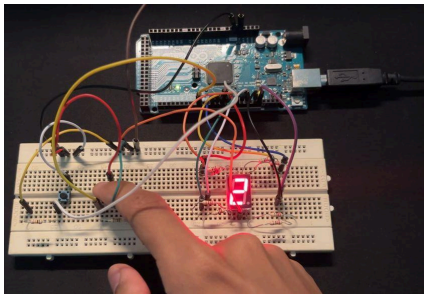

Figure 7.3: *The 7-Segment displayed no.2*

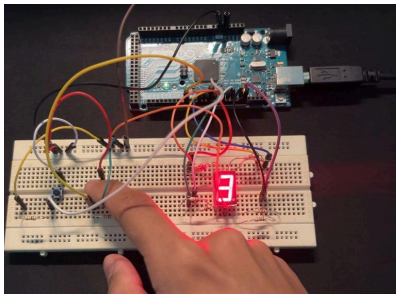

Figure 7.4: *The 7-Segment displayed no.3*

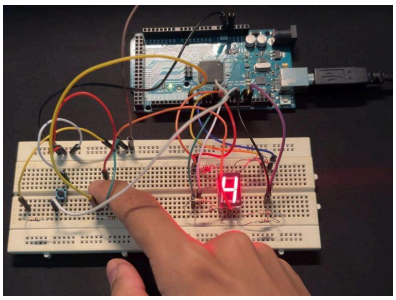

Figure 7.5: *The 7-Segment displayed no.4*

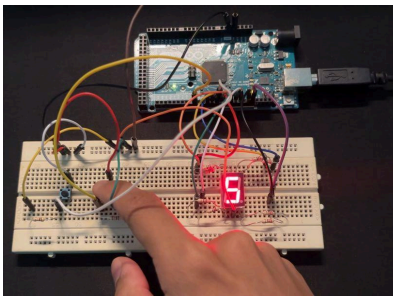

Figure 7.6: *The 7-Segment displayed no.5*

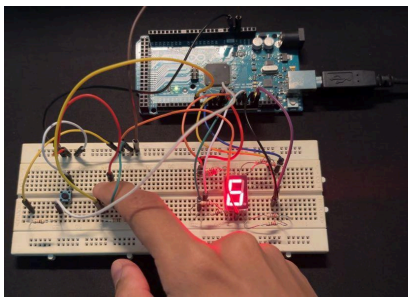

Figure 7.7: *The 7-Segment displayed no.6*

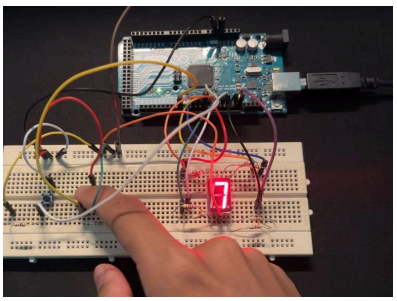

Figure 7.8: *The 7-Segment displayed no.7*

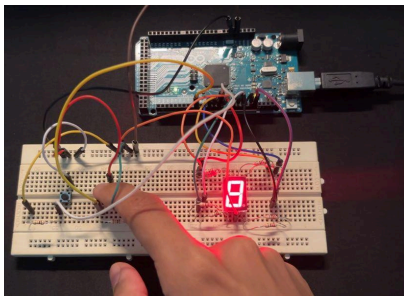

Figure 7.9: *The 7-Segment displayed no.8*

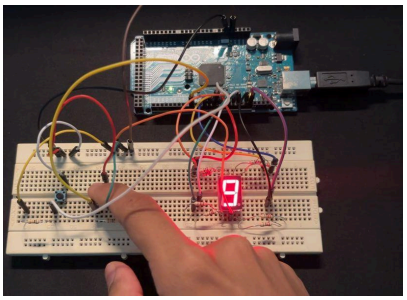

Figure 7.10: *The 7-Segment displayed no.9*

8.0 DISCUSSION

In this experiment, a seven-segment display counter system was successfully designed and implemented using an Arduino Mega 2560. Button inputs caused the display to show numerical outputs ranging from 0 to 9 when the circuit was connected according to the design. The increment button increased the displayed number by one, while the reset button returned the value to zero. Because every button press resulted in the anticipated output on the screen, the results verified that the system was functioning properly. This illustrated how the Arduino program and the hardware circuit could properly control logic.

To represent numerical characters, the 7-segment display illuminates particular segments (a–g). Each segment in this configuration was linked to an Arduino digital output pin. For each number, the program controlled which segments turned on by sending out a series of HIGH and LOW signals. Although it required more pins, this direct control method made it possible to represent each digit precisely. All things considered, the experiment effectively illustrated how microcontrollers can use digital output logic to control basic display systems.

Question: How can you interface an I2C LCD with Arduino? Explain the coding principle behind it compared to a 7-segment display and a matrix LED.

Answer: An I2C LCD interfaces with an Arduino using only SDA (data) and SCL (clock), along with VCC and GND. The display is controlled via the I2C protocol, where data and commands are sent serially to the LCD's driver IC, and programming is simplified using the LiquidCrystal_I2C library (`lcd.init()`, `lcd.backlight()`, `lcd.print()`). On the other hand, a 7-segment display requires multiple digital pins, each controlling a segment individually, and an LED matrix uses row-column multiplexing to light LEDs in patterns. Because the I2C LCD uses serial communication and a driver library, it requires fewer pins and less manual coding, making it more efficient and easier to program than 7-segment or matrix LED displays.

9.0 CONCLUSION

In conclusion, this project successfully demonstrated how to construct a simple numerical counter system using an Arduino Mega 2650 and a 7-segment display. The system functioned effectively, with each button press raising the displayed value from 0 to 9 and the reset button precisely returning the display to 0.

Individuals obtained hands-on experience with digital electronics, microcontroller programming, and fundamental circuit interface through this project. Their comprehension of

how hardware and software work together to carry out digital logic operations was improved by the project. Additionally, it gave valuable exposure to troubleshooting common issues like button debouncing and wiring errors.

Overall, the study achieved its objectives by demonstrating the fundamental concepts of digital counting systems and their potential applications in display systems, counters, and embedded system designs.

10.0 RECOMMENDATION

There are several ways to improve this project's performance and functionality in subsequent implementations. To reduce signal noise and increase accuracy, the button input system could be improved with hardware or software debouncing. The program would operate more responsively and efficiently if interrupts were used in place of continuous polling.

A multi-digit counter can be created by joining several 7-segment displays to further expand the system. A shift register or common anode display (such as the 74HC595) could be used in the circuit's redesign to cut down on the number of I/O pins needed. More sophisticated counting and user interaction features might be included in future projects by integrating an LCD display or a real-time feedback system.

11.0 REFERENCES

1. "02 – Digital Logic System ver2.pdf", Google Drive file, accessed via https://drive.google.com/file/d/1JGdUitPFZEFM8F9qtV-U-ZJ1lsq5YzQYE/view?usp=drive_link
2. Tutorials Point. (n.d.). *Arduino – Blinking LED*. Retrieved from https://www.tutorialspoint.com/arduino/arduino_blinking_led.htm TutorialsPoint
3. A. Rohansen-Roy. (2020, September 27). *Blinking LED* [Tutorial]. Arduino Project Hub. Retrieved from https://projecthub.arduino.cc/arohansenroy/blinking-led-77a79f

12.0 APPENDICES

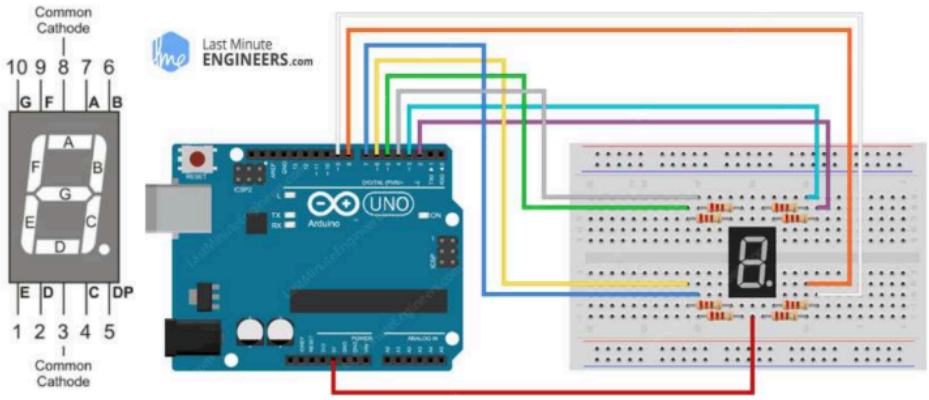

Figure 12.1: *Circuit connection of the 7-segment display with Arduino Uno for reference.*

13.0 ACKNOWLEDGEMENT

We would like to sincerely thank the instructors, Assoc. Prof. Eur. Ing. Ir. Ts. Gs. Inv. Dr. Zulkifli Bin Zainal Abidin & Dr. Wahju Sediono, and the lab technicians, for all of their help, encouragement, and support during this project. Their knowledge and perceptions have greatly influenced the course of this work. Additionally, we would like to express our gratitude to our peers for their support and cooperation, both of which were crucial to the accomplishment of this project.

14.0 STUDENTS DECLARATION

We hereby declare that, except for the places where it is acknowledged, all of the work presented in this report is entirely ours. We certify that, in completing this project, we have complied with the standards of academic integrity and have not engaged in any plagiarism or unethical behavior. Every information and support source used in this work has been appropriately referenced and acknowledged.

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 specified in the references and acknowledgment, and that the original work contained herein has not been untaken or done by unspecified sources or persons. 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 the individual contributors to the report. We therefore agreed unanimously that this report shall be submitted for **marking,** and this **final printed report** has been **verified by us.**

Signature: *haikalzulhilmi*

Name: Muhammad Haikal Zulhilmi Bin Fairof
Matric Number: 2313025

Read [/]
Understand [/]
Agree [/]

Signature: *faizizwan*

Name: Muhammd Faiz Izwan Bin Nor Effendi
Matric Number: 2313509

Read [/]
Understand [/]
Agree [/]

Signature:

Name: Ainul Jaariah Binti Aliudin
Matric Number: 2312664

Read [/]
Understand [/]
Agree [/]