



الجامعة الإسلامية العالمية ماليزيا
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
يُونِيسُيْتِيْ اِسْلَامْ اِنْتَارَا بَغْسِيَا مِلِّيْسِيَا
Garden of Knowledge and Virtue

MCTA 3203

MECHATRONICS SYSTEM INTEGRATION

SEMESTER 1, 25/26

WEEK 02 : DIGITAL LOGIC SYSTEM

SECTION 2

GROUP 19

LECTURER : DR WAHJU SEDIONO

DATE OF EXPERIMENT : 13 OCTOBER 2025

DATE OF SUBMISSION : 20 OCTOBER 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 aims to interface an Arduino Uno with a common cathode 7-segment display using two push buttons which is one for incrementing numerical values and another for resetting the display. The objective is to display numbers from 0 to 9, increasing sequentially with each press of the increment button, while the reset button returns the display to 0. The experiment involves both hardware assembly and software programming, where the Arduino detects button inputs to control the displayed output. The programming logic applies digital input detection and conditional statements to manage the display behavior. The results indicate a successful interface between hardware and software, demonstrating the capability of microcontrollers in digital counting applications. This experiment highlights the importance of efficient coding, systematic circuit design, and effective troubleshooting in embedded systems, which are essential for applications such as digital counters, timers, and automation systems.

TABLE OF CONTENTS

1.0 INTRODUCTION	4
2.0 MATERIALS AND EQUIPMENT	4
3.0 EXPERIMENTAL SETUP	5
4.0 METHODOLOGY	5
5.0 DATA COLLECTION	8
6.0 DATA ANALYSIS	10
7.0 RESULT	10
8.0 DISCUSSION	13
9.0 CONCLUSION	14
10.0 RECOMMENDATIONS	14
11.0 REFERENCES	15
ACKNOWLEDGEMENTS	16
STUDENTS'S DECLARATION	16

1.0 INTRODUCTION

Digital logic systems form the foundation of all modern digital electronics. Logic gates (AND, OR, NOT, NAND, NOR, XOR, XNOR) perform basic Boolean operations that process binary inputs into outputs.

A 7-segment display is a simple electronic display device used to represent decimal numbers. Each digit is composed of seven LEDs (labeled A–G) that can be individually turned on or off to form numeric characters.

In this experiment, an Arduino Uno is used to control the display by setting digital output pins HIGH or LOW to illuminate the corresponding segments. Pushbuttons are used to manually increment or reset the count displayed.

2.0 MATERIALS AND EQUIPMENT

- Arduino Uno board
- Common cathode 7-segment display
- 10k-ohm resistor (2)
- 220-ohm resistor (7)
- Push buttons (2)
- Breadboard
- Jumpers

3.0 EXPERIMENTAL SETUP

1. Connection of the 7-Segment Display:

1. Each of the 7 segments (a, b, c, d, e, f, g) was connected to separate Arduino digital pins (e.g., D0–D6).
2. The common cathode pin of the display was connected to one of the Arduino GND (ground) pins.
3. 220 Ω resistors were used between each segment pin and the corresponding Arduino pin to limit the current.

2. Connection of the Push Buttons:

1. One leg of each push button was connected to separate digital pins (e.g., D9 and D10).
2. The other leg of each push button was connected to GND.
3. 10k Ω pull-up resistors were used for each push button by connecting one end of the resistor to the digital pin and the other end to the Arduino's 5V output.
- 4.

4.0 METHODOLOGY

1. The circuit was built according to the circuit setup instructions.
2. The Arduino code is uploaded into the Arduino Uno.
3. Pressing the increment count button displayed the following number on the 7-segment display.
4. Press again and record the outcome until 9.
5. To see the outcome of reset button was pressed and it turn to 0.

4.1 Code used

```
1  const int segmentA = 0;
2  const int segmentB = 1;
3  const int segmentC = 2;
4  const int segmentD = 3;
5  const int segmentE = 4;
6  const int segmentF = 5;
7  const int segmentG = 6;
8  const int Button_Increment = 9; // Button to increment the count
9  const int Button_Reset = 8;    // Button to reset the count
10
11 // Button states
12 int count = 0; // Current count
13 bool Last_Button_increment = LOW;
14 bool Last_Button_State_Reset = LOW;
15
16 void setup()
17 {
18   pinMode (segmentA, OUTPUT);
19   pinMode (segmentB, OUTPUT);
20   pinMode (segmentC, OUTPUT);
21   pinMode (segmentD, OUTPUT);
22   pinMode (segmentE, OUTPUT);
23   pinMode (segmentF, OUTPUT);
24   pinMode (segmentG, OUTPUT);
25   pinMode (Button_Increment, INPUT);
26   pinMode (Button_Reset, INPUT);
27 }
28
29 void loop()
30 {
31   bool Current_Button_Increment = digitalRead (Button_Increment);
32   bool Current_Button_State_Reset = digitalRead (Button_Reset);
33
34   if (Current_Button_Increment == LOW && Last_Button_increment == HIGH) //-----Increment count
35   {
36     count++;
37     if (count > 9)
38     {
39       count = 0; //auto reset if exceed 9
40     }
41     delay (200);
42   }
43
44   if (Current_Button_State_Reset == LOW && Last_Button_State_Reset == HIGH) //-----Reset Button
45   {
46     count = 0; // Reset count to 0
47     delay (200);
48   }
49
50   Last_Button_increment = Current_Button_Increment;
51   Last_Button_State_Reset = Current_Button_State_Reset;
52   displayNumber (count); //Function to display number
53 }
54
55 //-----
56 void displayNumber (int count) //----- Function for count number
57 {
58   digitalWrite (segmentA, HIGH);
59   digitalWrite (segmentB, HIGH);
60   digitalWrite (segmentC, HIGH);
61   digitalWrite (segmentD, HIGH); //----- Reset all segments first
62   digitalWrite (segmentE, HIGH);
63   digitalWrite (segmentF, HIGH);
64   digitalWrite (segmentG, HIGH);
65
66   switch (count)
67   {
68     case 0:
69       digitalWrite (segmentA, LOW);
70       digitalWrite (segmentB, LOW);
71       digitalWrite (segmentC, LOW); //----- Display 0
72       digitalWrite (segmentD, LOW);
73       digitalWrite (segmentE, LOW);
74       digitalWrite (segmentF, LOW);
75     }
```

```

83     break;
84
85     case 1:
86         digitalWrite(segmentB, LOW); //----- Display 1
87         digitalWrite(segmentC, LOW);
88         break;
89
90     case 2:
91         digitalWrite (segmentA, LOW);
92         digitalWrite (segmentB, LOW);
93         digitalWrite (segmentD, LOW); //----- Display 2
94         digitalWrite (segmentE, LOW);
95         digitalWrite (segmentG, LOW);
96         break;
97
98     case 3:
99         digitalWrite (segmentA, LOW);
100        digitalWrite (segmentB, LOW);
101        digitalWrite (segmentC, LOW); //----- Display 3
102        digitalWrite (segmentD, LOW);
103        digitalWrite (segmentG, LOW);
104        break;
105
106    case 4:
107        digitalWrite (segmentB, LOW);
108        digitalWrite (segmentC, LOW); //----- Display 4
109        digitalWrite (segmentF, LOW);
110        digitalWrite (segmentG, LOW);
111        break;
112
113    case 5:
114        digitalWrite (segmentA, LOW);
115        digitalWrite (segmentC, LOW);
116        digitalWrite (segmentD, LOW); //----- Display 5
117        digitalWrite (segmentF, LOW);
118        digitalWrite (segmentG, LOW);
119        break;
120
121    case 6:
122        digitalWrite (segmentA, LOW);
123        digitalWrite (segmentC, LOW);
124        digitalWrite (segmentC, LOW); //----- Display 6
125        digitalWrite (segmentD, LOW);
126        digitalWrite (segmentE, LOW);
127        digitalWrite (segmentF, LOW);
128        digitalWrite (segmentG, LOW);
129        break;
130
131    case 7:
132        digitalWrite (segmentA, LOW);
133        digitalWrite (segmentB, LOW); //----- Display 7
134        digitalWrite (segmentC, LOW);
135        break;
136
137    case 8:
138        digitalWrite (segmentA, LOW);
139        digitalWrite (segmentB, LOW);
140        digitalWrite (segmentC, LOW); //----- Display 8
141        digitalWrite (segmentD, LOW);
142        digitalWrite (segmentE, LOW);
143        digitalWrite (segmentF, LOW);
144        digitalWrite (segmentG, LOW);
145        break;
146
147    case 9:
148        digitalWrite (segmentA, LOW);
149        digitalWrite (segmentB, LOW);
150        digitalWrite (segmentC, LOW); //----- Display 9
151        digitalWrite (segmentD, LOW);
152        digitalWrite (segmentF, LOW);
153        digitalWrite (segmentG, LOW);
154        break;
155    }
156 }

```

5.0 DATA COLLECTION

In this experiment, the 7-segment display was programmed to show all digits 0–9 sequentially using two types of circuits setup. The 1st circuit automatically counts and displays the number 0 to 9 and resets back to 0, 2nd circuit uses 1 pushbuttons to count and display number 0 to 9 in increment order and 1 reset button returns it to zero.

1st circuit

Step	Expected display	Actual Display	Segments ON (A - G)	Observation
1	0	0	A,B,C,D,E,F	Initial condition
2	1	1	B,C	Works correctly
3	2	2	A,B,G,E,D	Works correctly
4	3	3	A,B,C,D,G	Works correctly
5	4	4	F,G,B,C	Works correctly
6	5	5	A,F,G,C,D	Works correctly
7	6	6	A,F,G,E,C,D	Works correctly
8	7	7	A,B,C	Works correctly
9	8	8	A,B,C,D,E,F,G	Works correctly
10	9	9	A,B,C,D,F,G	Works correctly and loops back to step 1

2nd circuit

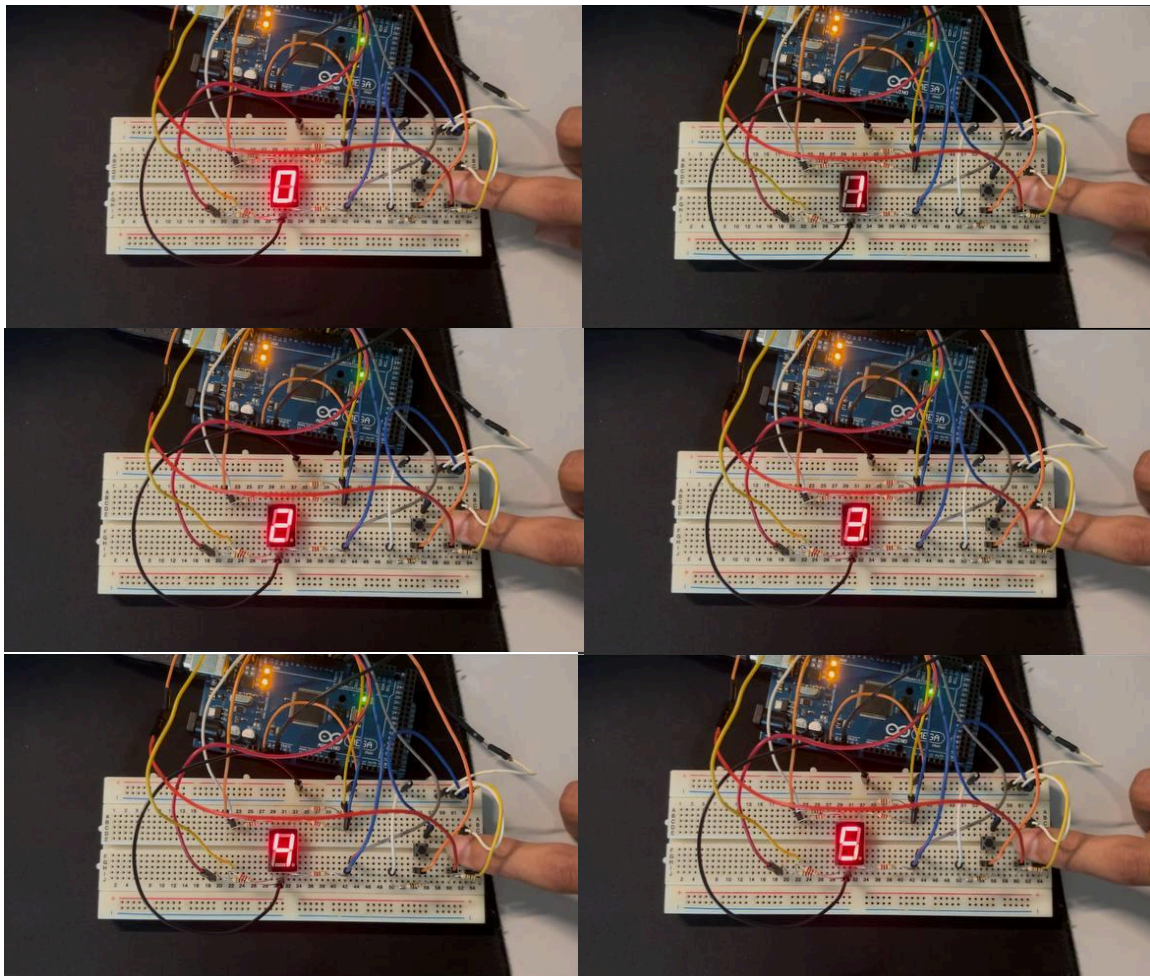
Step	Button Pressed	Expected display	Actual Display	Segments ON (A - G)	Observation
1	None	0	0	A,B,C,D,E,F	Initial condition
2	Increment (1x)	1	1	B,C	Works correctly
3	Increment (2x)	2	2	A,B,G,E,D	Works correctly
4	Increment (3x)	3	3	A,B,C,D,G	Works correctly
5	Increment (4x)	4	4	F,G,B,C	Works correctly
6	Increment (5x)	5	5	A,F,G,C,D	Works correctly
7	Increment (6x)	6	6	A,F,G,E,C,D	Works correctly
8	Increment (7x)	7	7	A,B,C	Works correctly
9	Increment (8x)	8	8	A,B,C,D,E,F,G	Works correctly
10	Increment (9x)	9	9	A,B,C,D,F,G	Works correctly
11	Reset	0	0	A,B,C,D,E,F	Works correctly

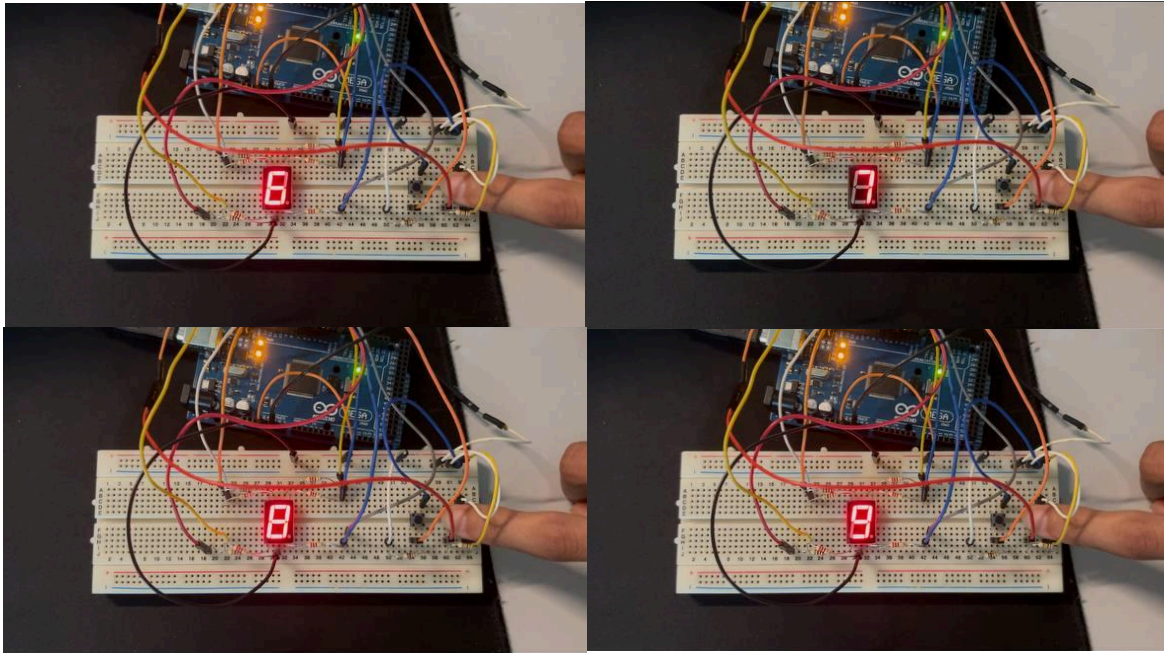
6.0 DATA ANALYSIS

1. All digits 0–9 were displayed correctly on the 7-segment display.
2. Each segment combination matched the standard truth table for 7-segment displays.
3. The increment button reliably increased the displayed value by one each time.
4. The reset button cleared the display back to “0,” confirming correct digital input handling.
5. No major errors or flickering were observed; minor debounce could occur from rapid button presses.
6. Overall performance was stable — display brightness consistent and switching delay negligible.

7.0 RESULT

The system successfully displayed from 0 to 9 with button control. Increment and resetting button operated correctly.





Question

How can you interface an I2C LCD with Arduino?

Interfacing an I2C LCD display with an Arduino is much simpler compared to using a 7-segment display. It only requires two input and output pins, SDA (Serial Data) and SCL (Serial Clock).

Steps to interface an I2C LCD with arduino

1. On Arduino IDE, install the library LiquidCrystal_I2C. This library is to help control the LCD.
2. Connect VCC to pin 5V.
3. Connect SDA and SCL into analog pin
4. Connect GND to GND pin.
5. Construct the Arduino.

Explain the coding principle behind it compared to a 7-segment display and a matrix LED.

<u>Featured</u>	<u>I2C LCD</u>	<u>7-segment</u>	<u>LED Matrix</u>
<u>Pins nedded</u>	<u>2 (SDA, SCL)</u>	<u>7-14 (more if multiple digit)</u>	<u>Many (or fewer if using driver IC)</u>
<u>Communication</u>	<u>Serial (I2C)</u>	<u>Parallel (direct pin control)</u>	<u>Multiplexed (row/column scanning)</u>
<u>Library support</u>	<u>Easy (LiquidCyrstal_I2C)</u>	<u>Some(manual or SevSeg)</u>	<u>Advanced (LedControl, MD_MAX72xx)</u>
<u>Coding Complexity</u>	<u>Low</u>	<u>Medium</u>	<u>High</u>
<u>Best For</u>	<u>Text display</u>	<u>Numeric display</u>	<u>Visual patters / animations</u>

8.0 DISCUSSION

Hardware

The setup for this project involves connecting a **7-segment display** to an **Arduino board**. The display serves as the main component and is connected to the Arduino through several digital pins and a ground connection. Each segment is linked with a **current-limiting resistor** to protect the LEDs from excessive current. There's 2 **pushbutton** included for increment the counter and function as a reset button. This simple project can be easily assembled on a **breadboard**, allowing students to **modify and troubleshoot** the circuit conveniently.

Software

The **Arduino IDE** is used to develop and upload the code for this project, providing a reliable environment for programming the Arduino. The program is organized with a clear structure defining the pins for each segment in the setup section, while the main loop manages the logic for incrementing the displayed numbers. The code is built to be easily scalable and modifiable for future improvements.

Source of error

There were some wiring issues in this setup. Incorrect connections of the segment pins caused the display to show the wrong digits. Additionally, the short length of the wires led to frequent disconnections during operation.

9.0 CONCLUSION

This experiment successfully demonstrated the practical implementation of digital logic systems using a 7-segment display and an Arduino. It helped us understand how digital signals interact with electronic components while applying concepts such as binary counting, circuit connections, and microcontroller programming. The use of pull-up resistors was found essential to ensure the push buttons worked correctly without errors. Through this experiment, we also gained valuable hands-on experience in troubleshooting and circuit wiring. Overall, it strengthened both our theoretical knowledge and practical skills, making it a meaningful learning experience.

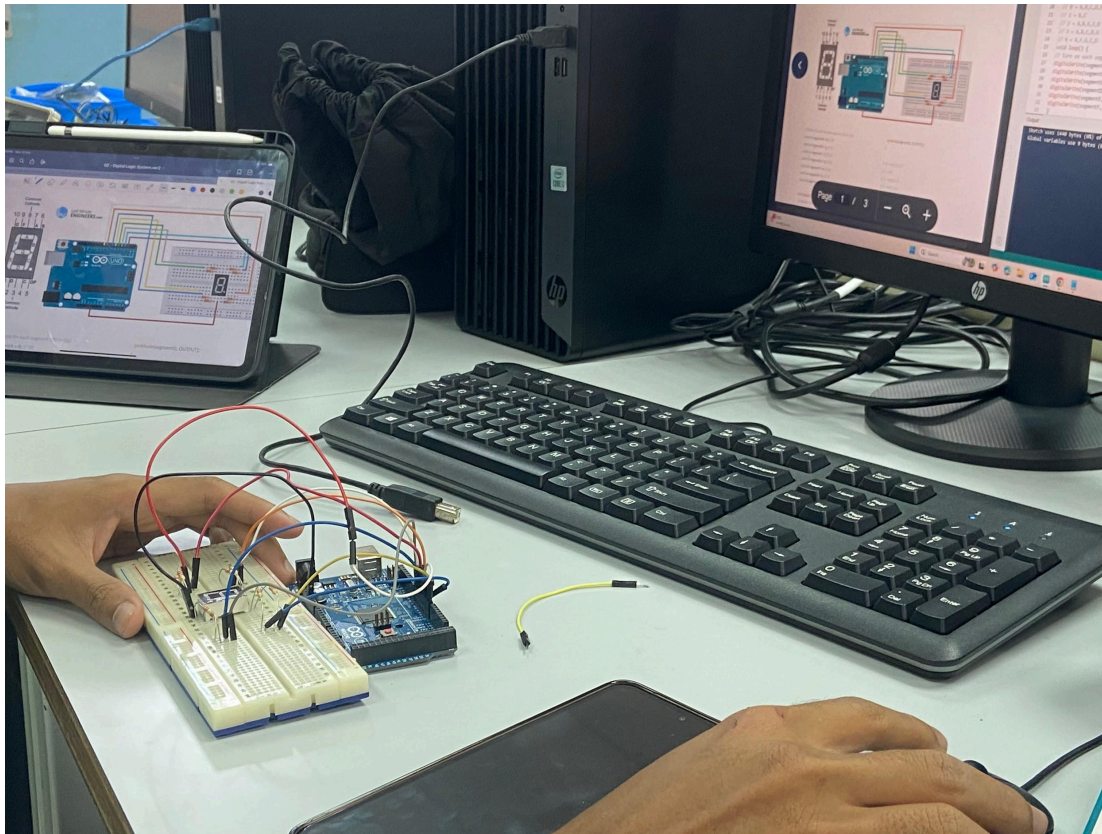
10.0 RECOMMENDATIONS

1. Implement software debouncing in the code to eliminate unintended multiple counts from a single button press.
2. Use a shift register (e.g., 74HC595) to reduce the number of Arduino pins used for controlling the display.
3. Extend the circuit to dual 7-segment displays for counting beyond 9 (00–99).
4. Incorporate display brightness control (PWM) for better visual performance.
5. Use a custom PCB layout to make the wiring neater and reduce connection errors on the breadboard

11.0 REFERENCES

1. Wikipedia. (n.d.). *Multiplexed Display*. Retrieved from https://en.wikipedia.org/wiki/Multiplexed_display
2. **Arduino Forum.** (2020). *Displaying Multiple Digits on a 7-Segment Display Simultaneously*. Retrieved from <https://forum.arduino.cc/t/displaying-multiple-digits-on-7-segment-display-simultaneously/642679>
3. **Makerguides.** (n.d.). *How to control a character I2C LCD with Arduino*. Retrieved from <https://www.makerguides.com/character-i2c-lcd-arduino-tutorial>

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:		Read	/
Name:	MUHAMMAD IMRAN BIN VEDDIN	Understand	/
Matric Number:	2316409	Agree	/
Contribution:	Result, Discussion		

Signature:		Read	/
Name:	ALYA KHAIRAH BINTI KHAIRUL JAMIL	Understand	/
Matric Number:	2317100	Agree	/
Contribution:	Abstract , Materials and Equipment , Experiment Setup , Methodology , Conclusion		

Signature:		Read	/
Name:	MUHAMMAD IRFAN RUSHDI BIN ASRI	Understand	/
Matric Number:	2319011	Agree	/
Contribution:	Introduction, data collection, data analysis, recommendation		