LAB REPORT :

DIGITAL LOGIC SYSTEM: BASIC LOGIC GATES, ELECTRONIC CIRCUIT INTERFACING, BASIC ALU, 7 SEGMENT DISPLAY, ICS BASED INTERFACING APPLICATION

MECHATRONIC SYSTEM INTEGRATION

(MCTA 3203)

NAME	METRIC NO.
AHMAD FARHAN BIN AHMAD NAHRUDI	2317373
MUHAMMAD AIMAN BIN SAFAWI	2318249
MOHD NAFIZ ZUHAIREE BIN MOHD NEEZAR	2317423

DATE OF SUBMITION :
22/10/2025

ABSTRACT

This experiment examines the fundamentals of digital logic systems and how they are applied using a microcontroller. The focus here is on the design and interface of a 7-segment display on an Arduino Uno board to output numeric values as digital control. This experiment is aimed at making students appreciate how digital inputs, binary logic, and outputs function together in a real-world circuit. A counter program was developed that could run the display with two push buttons — one for counting up and the other for resetting. Through this setup, the students observed how individual LEDs of the 7-segment display respond to particular digital inputs, producing number sequences from 0 to 9. The results confirmed successful hardware and software interfacing, affirming the relationship between logic states, electrical interfacing, and programming. Further, the experiment determines how simple digital devices form the foundation of complex electronic systems such as calculators, clocks, and embedded control systems.

TABLES OF CONTENTS

1.0 INTRODUCTION

This experiment focuses on exploring the use of the CYTRON CT UNO microcontroller to control a common anode 7-segment display with two push-button inputs. One button is designed to increment the displayed value, while the other resets the count back to zero. The primary objective of this study is to design and implement a basic digital counting system capable of displaying numerical values from 0 to 9, increasing by one with each press of the increment button and returning to zero upon activation of the reset button. This setup serves as a practical demonstration of fundamental microcontroller-based display control techniques, which are widely utilized in various digital applications such as electronic counters, digital clocks, and automated monitoring systems.

The 7-segment display serves as the primary output interface in this project. It consists of seven light-emitting diode (LED) segments arranged in a specific pattern to form numeric digits. In this experiment, a common anode configuration is employed, where all the anode terminals of the LED segments are connected to a constant 5V power supply. Each segment illuminates when a LOW signal is transmitted from the Arduino Mega's output pins. The two push buttons act as the main user input components — one is used to increase the displayed value sequentially, while the other resets the display count to zero. This simple yet effective configuration illustrates the core interaction between digital input and output components within an embedded system.

The experiment integrates several key principles of digital electronics and embedded systems programming, including:

- Digital Logic and Signal Processing: The Arduino detects digital input signals from the push buttons and translates them into corresponding output signals to activate the correct segments on the display.

- Microcontroller Programming: Core functions such as `digitalRead()` and `digitalWrite()` are utilized to process button states and control the

illumination of individual LED segments, demonstrating how code governs hardware operations.

- Counting and Logical Control: A counter variable is programmed to store, update, and display the current numeric value, showcasing how software logic can be applied to manage sequential operations in digital systems.

It is anticipated that the CYTRON CT UNO will effectively manage the 7-segment display, ensuring accurate numerical representation and smooth system operation. Each press of the increment button should increase the displayed value in proper sequence, while activation of the reset button should immediately return the display to zero. The system is also expected to perform with high stability and precision, incorporating adequate debouncing measures to prevent false triggers and maintaining consistent communication between the hardware circuitry and the control program. Through this experiment, the fundamental concepts of microcontroller interfacing, signal processing, and real-time digital control are effectively demonstrated.

2.0 MATERIALS AND EQUIPMENT

- Cytron CT UNO (ATmega328P microcontroller)
- Common anode 7-segment display
- Breadboard
- Jumper wires
- 220 Ω resistors (7 units)
- Push buttons (2 units)
- USB Type-A to Micro-USB Cable and computer with Arduino IDE

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 eight segments (a–g) of the display was connected to separate digital pins on the Arduino (e.g., D2–D9).

2. The common cathode of the display was connected to one of the Arduino GND pins.

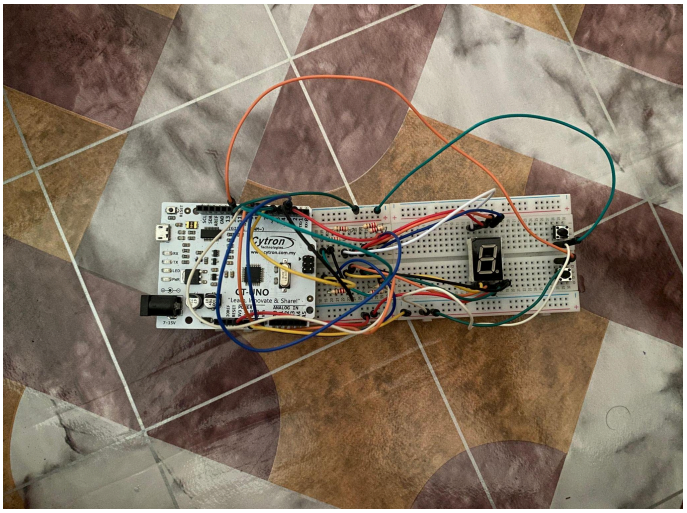
3. 220-ohm resistors were placed in series with each segment to limit the current.

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 CT UNO board, 7-segment display, resistors, and pushbuttons, were carefully connected to ensure proper functionality, as shown in figure below.



4.0 METHODOLOGY

1. Circuit Construction: The display and buttons were connected according to the provided schematic diagram on a breadboard.
2. Programming: The Arduino sketch was uploaded, defining each segment pin as OUTPUT and programming logic for digits 0–9.
3. Operation: Pressing the increment button increased the displayed number sequentially, while pressing the reset button returned the display to zero.
4. Verification: The output was verified visually on the 7-segment display and through Serial Monitor readings in the Arduino IDE.

4.1 System Preparation

The Cytron CT UNO was linked to the computer using a USB cable. The previously assembled circuit was checked to confirm that all wiring was accurate and every component was firmly positioned on the breadboard. The Arduino IDE software was then opened to start the programming procedure.

4.2 Arduino Programming

```
//DEFINE SEGMENT PIN

const int segmentA = 3;  // D0

const int segmentB = 2;  // D1

const int segmentC = 8;  // D2

const int segmentD = 7;  // D3

const int segmentE = 6;  // D4

const int segmentF = 4;  // D5

const int segmentG = 5;  // D6

const int segmentDP = 9;  // DP

//DEFINE BUTTON

const int inc = 10; //INCBUTTON
```

```
const int reset = 11;  //REBUTTON



int num = 0;   // CURRENT NUMBER

int lastIncState = HIGH;

int lastResetState = HIGH;



int segments[] = {segmentA, segmentB, segmentC, segmentD, segmentE,
segmentF, segmentG};



//SET ALL CLEAR SEGMENT FOR EACH NUMBER

void clear() {

  for (int i = 0; i < 7; i++) {

    digitalWrite(segments[i], HIGH); // HIGH = off

  }

}

void displayDigit(int num) {

  clear();



  switch (num) {

    case 0: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW);
digitalWrite(segmentC, LOW);

            digitalWrite(segmentD, LOW); digitalWrite(segmentE, LOW);
digitalWrite(segmentF, LOW); break;

    case 1: digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW);
break;

    case 2: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW);
digitalWrite(segmentD, LOW);
```

```
            digitalWrite(segmentE, LOW); digitalWrite(segmentG, LOW);
break;

    case 3: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW);
digitalWrite(segmentC, LOW);

            digitalWrite(segmentD, LOW); digitalWrite(segmentG, LOW);
break;

    case 4: digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW);

            digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW);
break;

    case 5: digitalWrite(segmentA, LOW); digitalWrite(segmentC, LOW);
digitalWrite(segmentD, LOW);

            digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW);
break;

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

            digitalWrite(segmentE, LOW); digitalWrite(segmentF, LOW);
digitalWrite(segmentG, LOW); break;

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

    case 8: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW);
digitalWrite(segmentC, LOW);

            digitalWrite(segmentD, LOW); digitalWrite(segmentE, LOW);
digitalWrite(segmentF, LOW);

            digitalWrite(segmentG, LOW);break;

    case 9: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW);
digitalWrite(segmentC, LOW);

            digitalWrite(segmentD, LOW); digitalWrite(segmentF, LOW);
digitalWrite(segmentG, LOW); break;

  }

}
```

```
void setup() {

  pinMode(segmentA, OUTPUT);
  pinMode(segmentB, OUTPUT);
  pinMode(segmentC, OUTPUT);
  pinMode(segmentD, OUTPUT);
  pinMode(segmentE, OUTPUT);
  pinMode(segmentF, OUTPUT);
  pinMode(segmentG, OUTPUT);
  pinMode(segmentDP, OUTPUT);

  pinMode(inc, INPUT_PULLUP);
  pinMode(reset, INPUT_PULLUP);

  displayDigit(num);
}

void loop() {
int incStateNow = digitalRead(inc);
int resetStateNow = digitalRead(reset);

 if (incStateNow == LOW && lastIncState == HIGH) {
    num++;
```

```
    if (num > 9) num = 0;
    displayDigit(num);
    delay(250);
  }
 if (resetStateNow == LOW && lastResetState == HIGH) {
    num = 0;
    displayDigit(num);
    delay(250);
  }
lastIncState = incStateNow;
lastResetState = resetStateNow;
}
}}
```

4.3 Testing Procedure

Once the program was successfully uploaded, the circuit was evaluated by pressing the increment button to increase the count sequentially from 0 to 9, and the reset button was used to return the display to 0. This process was repeated multiple times to verify the system's stability and reliability.

4.4 Observation and Data Recording

The numerical outputs displayed were carefully monitored and documented to ensure that each button press generated the correct value. Additionally, the response time and overall performance of the display were observed.

4.5 Troubleshooting

In cases where the system did not operate as expected, both the wiring connections and the program logic were inspected and revised. Necessary adjustments were implemented to achieve consistent and accurate operation of the display counter system.

5.0 DATA COLLECTION

BUTTON PRESS	DISPLAYED NUMBER	NOTES
Reset (Initial)	0	Normal Increment
Increment	1	Normal Increment
Increment	2	Normal Increment
Increment	3	Normal Increment
Increment	4	Normal Increment
Increment	5	Normal Increment
Increment	6	Normal Increment
Increment	7	Normal Increment
Increment	8	Normal Increment
Increment	9	Normal Increment
Increment	0	Wraps Around Correctly
Reset	0	Reset Worked

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 Cytron CT UNO and program consistently produced the desired outputs and controlled the 7-segment display

7.0 RESULTS

The result of the study shows that the 7-segment display successfully counted from 0 to 9 using the push-button inputs. The increment button advanced the count, and the reset button cleared it. The display remained stable at each digit and responded accurately to button presses, indicating proper wiring and code execution. No flickering or false triggering was observed, confirming good electrical contact and proper resistor usage. The visual of each output of seven segments-observed during the testing are represented in figures below.

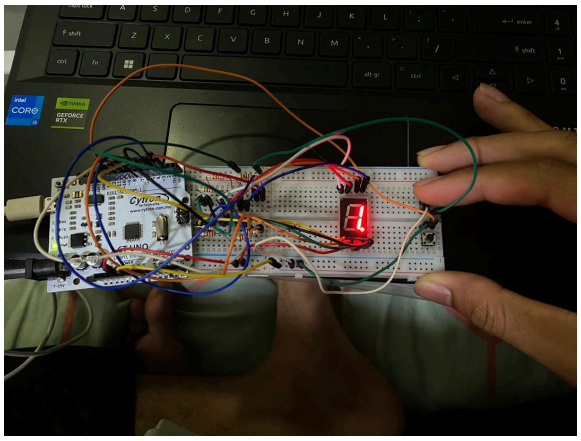


Figure 7.1 : *The seven-segments display no. 1*

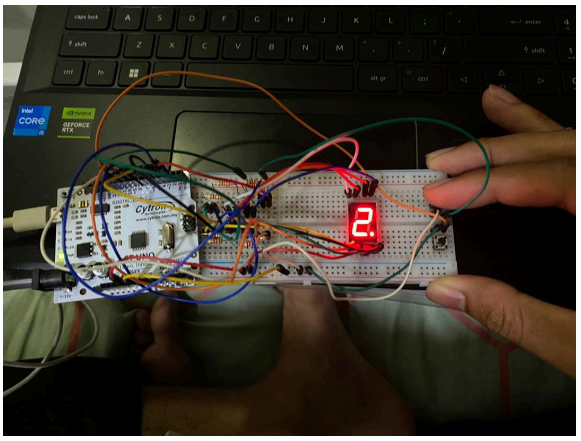


Figure 7.2 : *The seven-segments display no. 2*

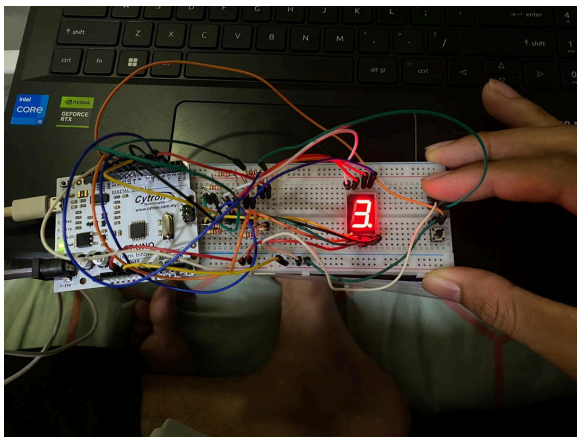


Figure 7.3 : *The seven-segments display no. 3*

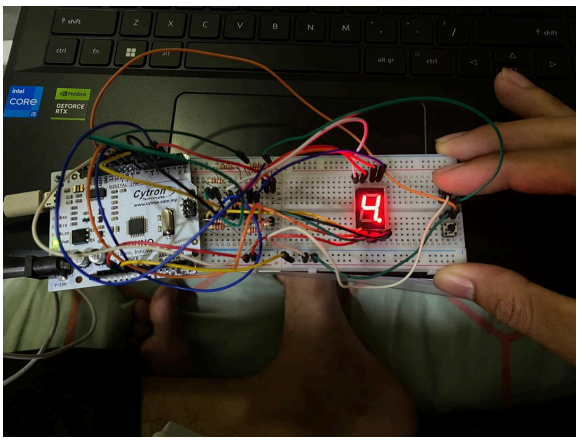


Figure 7.4 : *The seven-segments display no. 4*

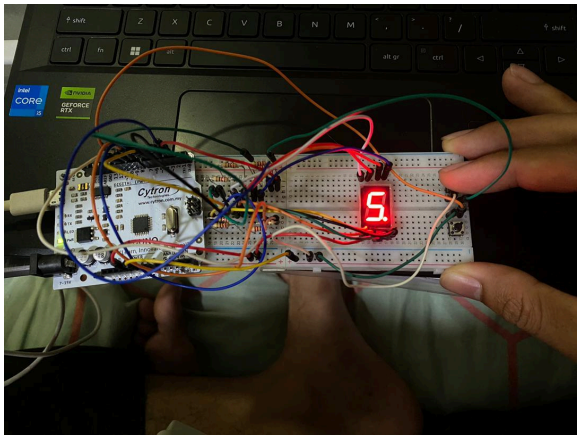


Figure 7.5 : *The seven-segments display no. 5*

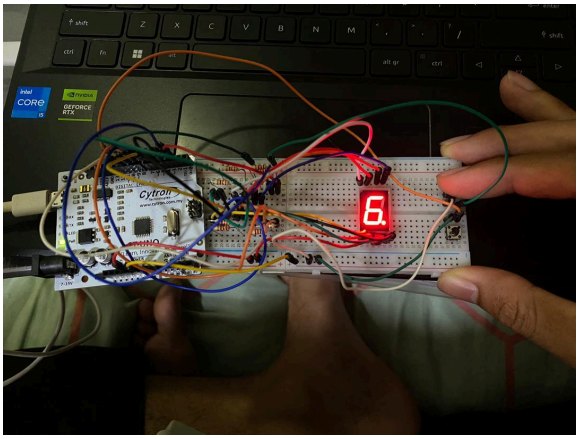


Figure 7.6 : *The seven-segments display no. 6*

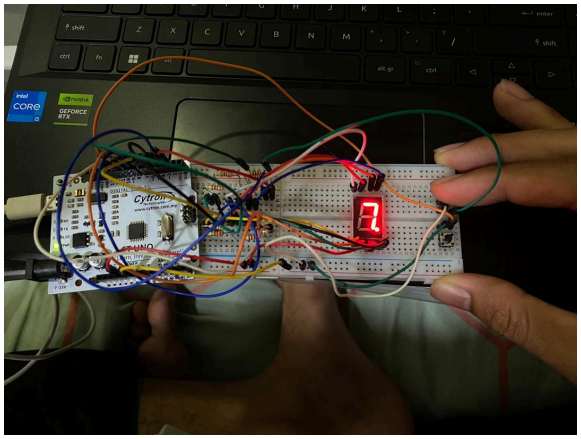


Figure 7.7 : *The seven-segments display no. 7*

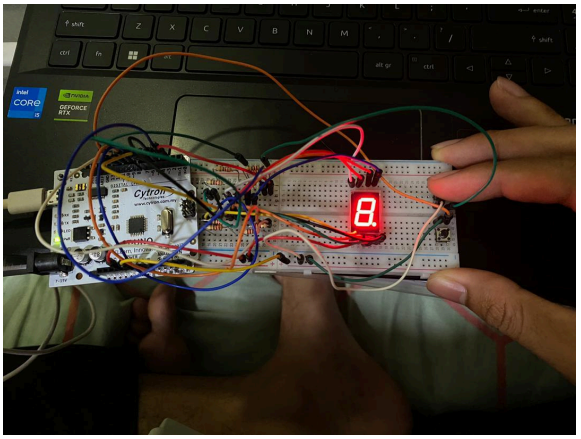


Figure 7.8: *The seven-segments display no. 8*

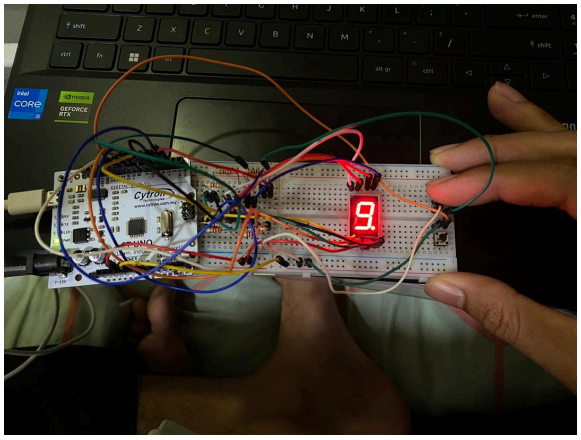


Figure 7.9 : *The seven-segments display no. 9*

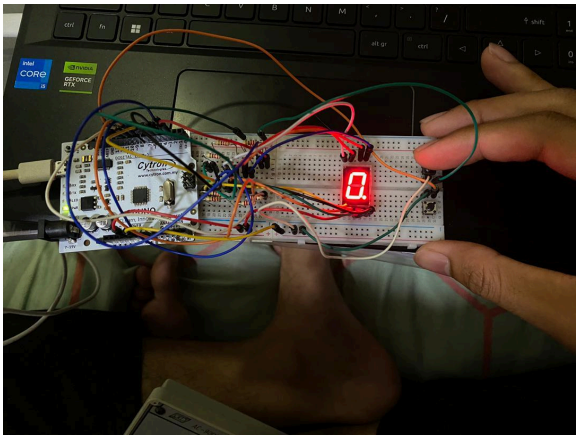


Figure 7.10 : *The seven-segments display no. 0*

8.0 DISCUSSIONS

In this experiment, a seven-segment display counter system was successfully designed and implemented using an Cytron CT UNO. When the circuit was connected according to the design, button inputs controlled the numerical output on the display ranging from 0 to 9. The increment button increased the displayed number by one, while the reset button returned the count to zero. Since each button press produced the expected output, the results confirmed that the system operated correctly. This demonstrated the proper coordination between the Arduino program and the hardware circuit in managing logical control.

To display numerical characters, the seven-segment display illuminated specific segments (a–g). Each segment in this setup was connected to a digital output pin of the Arduino. The program determined which segments were activated for each number by sending corresponding HIGH and LOW signals. Although this direct control approach required more pins, it allowed each digit to be displayed accurately. Overall, the experiment effectively demonstrated how a microcontroller can use digital output logic to operate simple 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.

An I2C LCD can be interfaced with an Arduino using only two communication lines, SDA (Serial Data) and SCL (Serial Clock), along with power connections (VCC and GND). This setup uses the **LiquidCrystal_I2C** library, where the LCD is initialized with its I2C address (such as 0x27) and display size, and functions like *lcd.begin()*, *lcd.setCursor()* and *lcd.print()* are used to display text. The Arduino sends data serially, and the I2C module converts it into parallel signals to control the display. Compared to a 7-segment display that requires multiple pins and direct control of each segment (a–g) using HIGH and LOW outputs, the I2C LCD is more efficient and uses

fewer pins. Meanwhile, a matrix LED display works through multiplexing, where the Arduino rapidly switches between rows and columns to create images or numbers. Overall, the I2C LCD simplifies wiring and coding by using serial communication instead of manual or multiplexed control.

9.0 CONCLUSION

In conclusion, this project successfully demonstrated the development of a simple numerical counter system using a Cytron CT UNO and a 7-segment display. The system performed efficiently, where each button press incremented the displayed value from 0 to 9, and the reset button accurately returned the count to 0.

Through this project, participants gained practical experience in digital electronics, microcontroller programming, and basic circuit interfacing. It enhanced their understanding of the interaction between hardware and software in executing digital logic operations. Furthermore, it provided valuable experience in troubleshooting common issues such as button debouncing and wiring errors.

Overall, the project effectively met its objectives by showcasing the core principles of digital counting systems and their potential applications in display technologies, counters and embedded system designs.

10.0 RECOMMENDATION

1. Implement code-based debouncing to improve button accuracy.
2. Extend the project to use dual 7-segment displays for two-digit numbers.
3. Experiment with I2C LCD or LED matrix to compare data transmission methods.
4. Explore additional logic gate simulations to expand understanding of combinational logic circuits.

There are numerous potential areas of enhancement that can be brought in to improve both functionality and performance of this project in later projects. One major enhancement is to debounce the button input system to lower signal interference and provide higher accuracy. In the current design, button presses may lead to intermittent or unstable signals due to mechanical bounce effects. To solve this issue, hardware and software debouncing techniques can be implemented. Hardware debouncing, for instance, in the form of RC filter or Schmitt trigger application, would remove sharp voltage spikes, while software debouncing would be made easier by taking time delays or filtering through the program code logically. This use of techniques would result in more stable input readings and less counting error.

Further, the performance of the system can be significantly improved by changing the continuous polling technique to interrupt-based control. Interrupts allow the microcontroller to respond instantaneously when a button is pressed, rather than continuously checking the input states through a loop. Not only does this make the program more responsive, but it also conserves CPU from unnecessary consumption, allowing the system to operate more efficiently and execute other tasks simultaneously.

To increase the system's functionality, the project could be extended by implementing a multi-digit counter, which can be achieved through multiple 7-segment displays connected in series. This addition would enable the system to process numbers greater than single digits and produce larger number outputs. To properly deal with numerous screens without exhausting input/output pins accessible on the Arduino Mega 2560, a

shift register or a common anode driver such as the 74HC595 can be incorporated into the design. This would reduce wiring complexity, improve scalability, and make the overall design more compact and organized.

Additionally, future versions of the system can include more advanced output and interaction components. For example, adding an LCD display would enable not just the display of numerical information but also system messages, thereby providing more user-friendly feedback. Another real-time monitor or feedback can be implemented that enables users to check operational performance or alert errors when running the system. With such an improvement, the project would not only be more precise and stable but also more functional, efficient, and user-friendly in nature for the sake of application and learning.

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

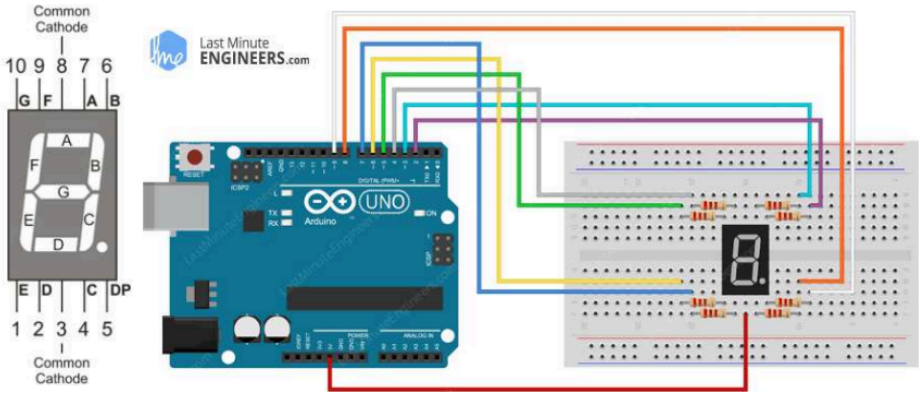


Figure 12.1 : *Circuit Connection of the 7-segment Display for Arduino Uno for References*

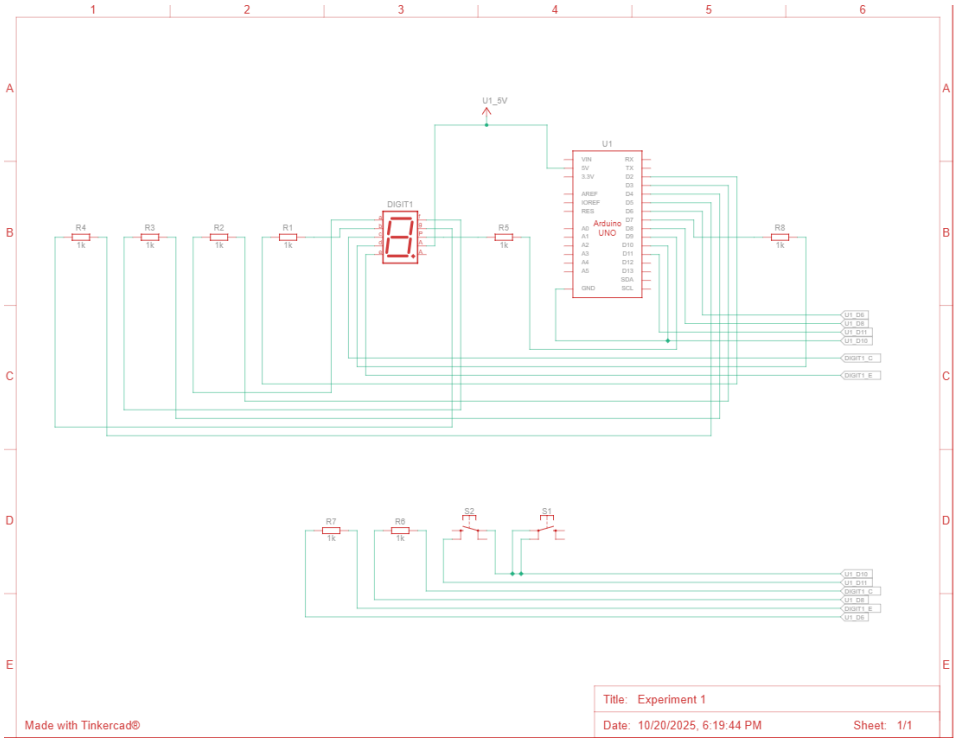


Figure 12.2 *: Circuit Diagram*

13.0 ACKNOWLEDGEMENT

We would like to sincerely thank the instructors, Assoc. Prof. Eur. Ing. Ir. Ts. Gs. Inv. Dr. Zulkifli Bin Zainal Abidin 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: ***ahmadfarhan***

Name: Ahmad Farhan bin Ahmad Nahrudi
Matric Card: 2317373

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

Signature: ***aimansafawi***

Name: Muhammad Aiman bin Safawi
Matric Card: 2318249

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

Signature: ***nafizzuhairee***

Name: Mohd Nafiz Zuhairee bin Mohd Neezar
Matric Card: 2317423

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