

MECHATRONICS SYSTEM INTEGRATION

EXPERIMENT 1: DIGITAL LOGIC SYSTEM

GROUP NUMBER : A

PROGRAMME: MECHATRONICS

GROUP MEMBERS	MATRIC NO
AISYAH SOFEA BINTI OTHMAN	2115386
SITI ALIAA BINTI IBRAHIM	2112618
MUHAMMAD ADLI FAHMI BIN TAJUL ARIS	2113095
ETHAR ABDALLA ABDELKARIM OSMAN	2111282
NURAIN AINAA AQILAH BINTI ROSLI	2114560

DATE OF EXPERIMENT:

Wednesday, 18th October 2023

DATE OF SUBMISSION:

Wednesday, 25th October 2023

Abstract

This laboratory experiment focused on showing an ascending number sequence through the application of an Arduino microcontroller interfaced with a 7- segment display. The 7- segment display served as a visual output device capable of displaying figures from 0 to 9. The primary objective of this experiment was to conduct an understanding of basic digital logic, binary- coded decimal, and multiplexing concepts through hands- on engagement with microcontroller- controlled displays. The successful execution of this interface allowed for precise control of each display segment. This facilitated the direct display of numeric values and alphanumeric characters. The findings from this experiment contribute to the broader understanding of microcontroller integration with display technologies and have practical implications for applications in the field of digital electronics.

Table of Contents

Objectives.....	3
Introduction.....	3
Material & Equipment.....	4
Experimental Setup.....	4
Methodology.....	6
Data Collection.....	10
Data Analysis.....	11
Result.....	11
Discussion.....	11
Conclusion.....	12
Recommendation.....	13
References.....	13
Appendices.....	14
Acknowledgments.....	14
Certificate of Originality and Authenticity.....	15

Objectives

1. To demonstrate ascending number sequence using Arduino and 7-segment display.
2. To design and implement the working of a 7-segment display using Arduino.
3. To understand the interface between Arduino and 7-segment display.

Introduction

The purpose of this experiment is to gain practical knowledge in handling 7-segment displays to visually represent the numerical digits (0-9) and encourage problem-solving skills by allowing students to troubleshoot and debug any issues that arise during experimentation, improving their ability to identify and solve electronic problems.

A 7-segment display consists of seven individual LED segments arranged in a specific pattern to form the number 0 to 9 and a few alphabetic characters. The two primary types of seven-segment displays are Common Cathode (CC) and Common Anode (CA). This categorization is dependent on which LED pin is attached to the common pin. Common Anode displays are those in which the anodes of these LEDs are linked to one another. Additionally, they are known as Common Cathode displays if the Cathodes are linked together.

The Arduino is a popular open-source microcontroller platform that allows a professionals to create a wide range of electronic projects. Arduino interface with 7-segment displays using either common cathode or common anode configuration. The common cathode has all the negative terminals connected together, while the common anode has all the positive terminals connected.

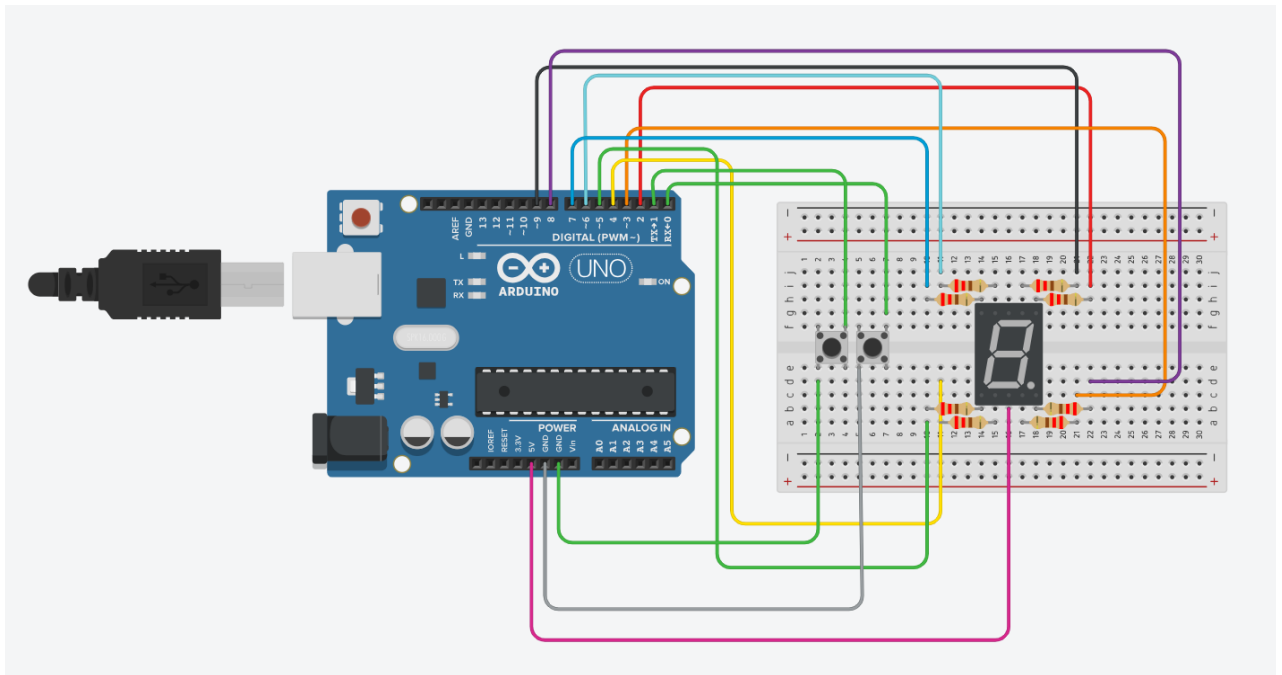
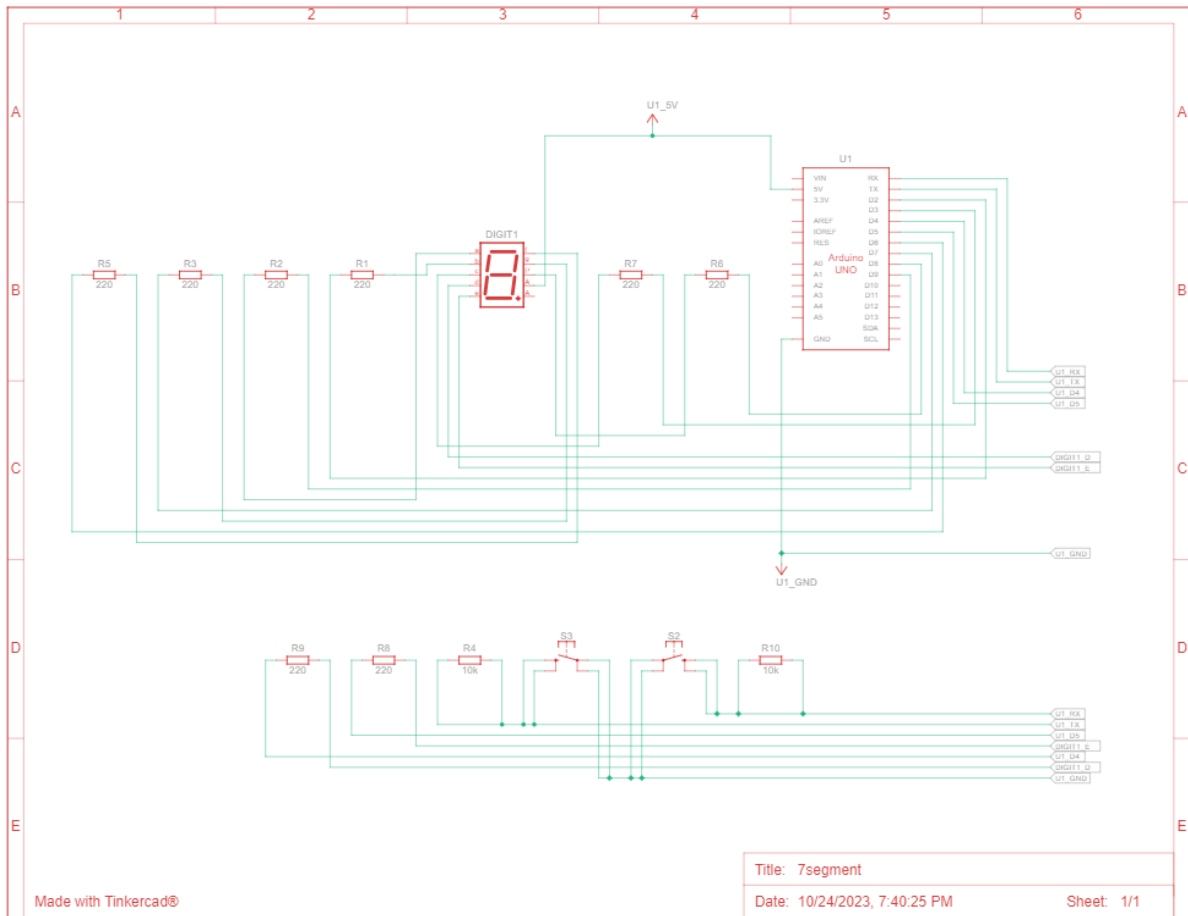
It is expected that by interfacing a 7-segment display with an Arduino, we will successfully control the display to show numeric digits (0-9) and possibly other characters using the appropriate Arduino code. When inserting two push buttons into an experiment, it is expected that they will serve as input devices to trigger certain actions. For instance, one push button might be used to cycle through a displayed number, and another might be used to reset the display or initiate a certain function. The hypothesis assumes that push buttons will provide a means of user interaction with the 7-segment display. Specifically, it is expected that pressing the push button will lead to a change in the displayed number or initiate some other response that can be observed on the display.

Material & Equipment

- Arduino Uno board
- Common cathode 7-segment display
- Eight 220-ohm resistors
- Two pushbuttons
- Jumper wires
- Breadboard
- Circuit setup

Experimental Setup

To perform this experiment, we identified the 7-segment display as a common cathode type. The common cathode pin, distinguished by its shorter leg, was connected to one of the Arduino's 5V pins. Each segment (a, b, c, d, e, f, g) of the 7-segment display was connected to individual digital output pins on the Arduino. Each 7-segment pin will be connected to the 220 ohm resistor to limit the current flow. The specific pins were determined by our circuit diagram, and our Arduino code was programmed accordingly to control these pins, illuminating the appropriate segments for displaying specific numbers or characters. Two push buttons were introduced as input devices for user interaction. One terminal of each push button was connected to separate digital input pins on the Arduino. In one configuration, we utilised pull-up resistors, connecting a 10k ohm resistor between the digital input pin and VCC to ensure proper operation. In this experiment, we use two push buttons, one to increase the number sequence manually and another one to reset the number to zero. To facilitate efficient connections, jumper wires were used to interconnect the components based on the circuit diagram provided. A breadboard was employed to neatly organise and simplify the wiring, enhancing the overall setup. After ensuring all connections were in place, we uploaded the Arduino code to the board to implement the functionality we had designed in our experiment. Below, we attached our schematic diagram and circuit view using 'Tinkercad'.



Methodology

This experiment is to learn how to interface and control a 7-segment display using an arduino uno. Hence, it is important to set up the workspace by gathering all the required materials and components. The first step is to connect the 7-segment display to the breadboard and then connect all individual segment pins of it (a,b,c,d,e,f,g and dp) to the corresponding digital pins on the arduino that have been decided prior using jumper. By using a common cathode display, the common cathode pin is connected to 5V and the segments to the digital pins. The codes that will control the 7-segment display were written to achieve the objectives using arduino sketch. The tools menu on arduino sketch works to select appropriate arduino board and COM port to enable the uploading of the codes. After no error detected after compiling the codes, it can be then uploaded to the arduino. Another option used was tinkercad to simulate the circuit using the code prior to observing and analysing the results. The next thing to do is to observe the 7-segment display and do modification to obtain the results desired which is capable of displaying numbers from 0 to 9 in sequence, reset and incremental sequence upon contact with the push button. The observations are recorded. As precautions, the connections are ensured to be secure to avoid short circuit and damaging the components including using resistors in the connection. The power source is disconnected before making any changes to the circuit.

The programming codes used includes: *ACTIVE LOW*

Tinkercad	Arduino IDE
<pre>// Define the pins for each segment (D0 to D6) const int segmentA = 9; // D0 const int segmentB = 2; // D1 const int segmentC = 3; // D2 const int segmentD = 4; // D3 const int segmentE = 5; // D4 const int segmentF = 6; // D5 const int segmentG = 7; // D6 const int segmentDP = 8; // D7 const int pushbutton_reset = 0; const int pushbutton_inc = 1; int count=0; void setup() { // Initialize the digital pins as OUTPUTs</pre>	<pre>// Define the pins for each segment (D0 to D6) const int segmentA = 9; // D0 const int segmentB = 2; // D1 const int segmentC = 3; // D2 const int segmentD = 4; // D3 const int segmentE = 5; // D4 const int segmentF = 6; // D5 const int segmentG = 7; // D6 const int segmentDP = 8; // D7 void setup() { // Initialize the digital pins as OUTPUTs pinMode(segmentA, OUTPUT); pinMode(segmentB, OUTPUT);</pre>

```

pinMode(segmentA, OUTPUT);
pinMode(segmentB, OUTPUT);
pinMode(segmentC, OUTPUT);
pinMode(segmentD, OUTPUT);
pinMode(segmentE, OUTPUT);
pinMode(segmentF, OUTPUT);
pinMode(segmentG, OUTPUT);
pinMode(pushbutton_reset, INPUT);
pinMode(pushbutton_inc,
INPUT_PULLUP);

}

void loop() {
    // Turn on each segment one by one
    int state1 = digitalRead(pushbutton_reset);
    int state2 = digitalRead(pushbutton_inc);
    Serial.println(state1);

    if (state1==LOW)
    {

count = (count + 1) % 10; // Cycle through
numbers 0 to 9

//Display the current count on the display
displayNumber(count);
delay(1000);

}
if(state2==LOW){
count=0;
displayNumber(0);

}

}

void displayNumber(int num) {
    //no0
    switch(num){
        case 0:
            digitalWrite(segmentA, LOW);
            digitalWrite(segmentB, LOW);
            digitalWrite(segmentC, LOW);

```

```

            pinMode(segmentC, OUTPUT);
            pinMode(segmentD, OUTPUT);
            pinMode(segmentE, OUTPUT);
            pinMode(segmentF, OUTPUT);
            pinMode(segmentG, OUTPUT);
        }
        void loop() {
            // Turn on each segment one by one

            // digitalWrite(segmentA, LOW);
            // digitalWrite(segmentB, LOW);
            // digitalWrite(segmentC, LOW);
            // digitalWrite(segmentD, LOW);
            // digitalWrite(segmentE, LOW);
            // digitalWrite(segmentF, LOW);
            // digitalWrite(segmentG, LOW);

            delay(5000);
            //no0
            digitalWrite(segmentA, LOW);
            digitalWrite(segmentB, LOW);
            digitalWrite(segmentC, LOW);
            digitalWrite(segmentD, LOW);
            digitalWrite(segmentE, LOW);
            digitalWrite(segmentF, LOW);
            digitalWrite(segmentG, HIGH);
            //digitalWrite(segmentDP, LOW);

            delay(2000);
            //no1
            digitalWrite(segmentA, HIGH);
            digitalWrite(segmentB, LOW);
            digitalWrite(segmentC, LOW);
            digitalWrite(segmentD, HIGH);
            digitalWrite(segmentE, HIGH);
            digitalWrite(segmentF, HIGH);
            digitalWrite(segmentG, HIGH);
            //digitalWrite(segmentDP, LOW);

            delay(2000);
            //no2
            digitalWrite(segmentA, LOW);
            digitalWrite(segmentB, LOW);
            digitalWrite(segmentC, HIGH);
            digitalWrite(segmentD, LOW);
            digitalWrite(segmentE, LOW);
            digitalWrite(segmentF, HIGH);
            digitalWrite(segmentG, LOW);
            //digitalWrite(segmentDP, LOW);

```

<pre> digitalWrite(segmentD, LOW); digitalWrite(segmentE, LOW); digitalWrite(segmentF, LOW); digitalWrite(segmentG, HIGH); //digitalWrite(segmentDP, LOW); break; //no1 case 1: digitalWrite(segmentA, HIGH); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); digitalWrite(segmentD, HIGH); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, HIGH); digitalWrite(segmentG, HIGH); //digitalWrite(segmentDP, LOW); break; //no2 case 2: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW); digitalWrite(segmentC, HIGH); digitalWrite(segmentD, LOW); digitalWrite(segmentE, LOW); digitalWrite(segmentF, HIGH); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); break; //no3 case 3: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, HIGH); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); break; //no4 case 4: digitalWrite(segmentA, HIGH); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); </pre>	<pre> delay(2000); //no3 digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, HIGH); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); delay(2000); //no4 digitalWrite(segmentA, HIGH); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); digitalWrite(segmentD, HIGH); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); delay(2000); //no5 digitalWrite(segmentA, LOW); digitalWrite(segmentB, HIGH); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); delay(2000); //no6 digitalWrite(segmentA, LOW); digitalWrite(segmentB, HIGH); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); digitalWrite(segmentE, LOW); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); delay(2000); //no7 digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW); </pre>
---	---

<pre> digitalWrite(segmentD, HIGH); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); break; //no5 case 5: digitalWrite(segmentA, LOW); digitalWrite(segmentB, HIGH); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); break; //no6 case 6: digitalWrite(segmentA, LOW); digitalWrite(segmentB, HIGH); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); digitalWrite(segmentE, LOW); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); break; //no7 case 7: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); digitalWrite(segmentD, HIGH); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, HIGH); digitalWrite(segmentG, HIGH); //digitalWrite(segmentDP, LOW); break; //no8 case 8: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); </pre>	<pre> digitalWrite(segmentC, LOW); digitalWrite(segmentD, HIGH); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, HIGH); digitalWrite(segmentG, HIGH); //digitalWrite(segmentDP, LOW); delay(2000); //no8 digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); digitalWrite(segmentE, LOW); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); delay(2000); //no9 digitalWrite(segmentA, HIGH); digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); } </pre>
--	--

<pre> digitalWrite(segmentE, LOW); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); break; //no9 case 9: digitalWrite(segmentA, LOW); digitalWrite(segmentB, LOW); digitalWrite(segmentC, LOW); digitalWrite(segmentD, LOW); digitalWrite(segmentE, HIGH); digitalWrite(segmentF, LOW); digitalWrite(segmentG, LOW); //digitalWrite(segmentDP, LOW); break; } } </pre>	
--	--

Data Collection

Number displayed	0	1	2	3	4	5	6	7	8	9
7-Segment (based on light 'ON')	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON

Table 1.1 : 7-segment display numbers in sequence of 0 to 9 automatically without push button in Arduino IDE and Tinkercad

	Activation mode	Pushbutton Function	
		Reset	Incremental Sequence
Arduino IDE	ACTIVE LOW	Not working	Not working
Tinkercad	ACTIVE LOW	Working	Working

Table 1.2 : 7-segment display numbers in sequence of 0 to 9 with push button

Data Analysis

This 7-segment display is controlled by using two push buttons, one for incrementing the count and another for resetting the count to 0. Referring to the code used in the tinkercad and the circuit diagram; 'pushbutton_reset'(connected to pin 0) is used to reset the count to 0 and 'pushbutton_inc'(connected to pin 1) is used to increment the count.

Whenever the 'pushbutton_reset' is pressed, the count is set to 0 and displayed on the 7-segment display. However, when 'pushbutton_inc' is pressed, the count is incremented by 1 to 10 each time and the updated count is displayed using the 'displayNumber' function.

Result

Overall, the experiment was successful in achieving its theoretical objectives. Utilising the Tinkercad platform for simulating the circuit proved to be instrumental in demonstrating the desired outcome. We successfully showcased an ascending number sequence and implemented a reset function to return the display to zero using an Arduino microcontroller and a 7-segment display. This allowed us to gain a comprehensive understanding of the interfacing between Arduino and the 7-segment display.

However, it's essential to acknowledge that certain objectives could not be fully realized when applied in the practical circuit. This experience highlights the importance of further refinement and optimization for future experiments, possibly involving adjustments to the circuit design or exploring additional variables. Overall, the experiment served as a valuable learning opportunity, bridging the gap between theory and practical implementation.

Discussion

The code defines the pins corresponding to each segment (D0 to D6) of the 7-segment display. It also identifies the pins associated with two push buttons, which are essential to controlling the display. To manipulate the display segments and monitor the push button inputs, several functions are utilised in the code. These functions are as follows:

- setup(): In this function, digital pins are initialised as OUTPUTs, allowing for control over the segments and reading input from the push buttons.

- `loop()`: This function operates to verify the state of the push buttons (`state1` and `state2`) to check whether they are pressed (LOW) or not. Upon detecting a button press, the code updates the count, showing a sequential display of numbers from 0 to 9. The code employs a `delay(1000)` to ensure each number is displayed for a second before transitioning to the next one. The second button, when pressed, sets the count to 0.
- `displayNumber(int num)`: This function is used in controlling the individual segments of the 7-segment display based on the number to be displayed. Each case within the switch statement corresponds to a specific number (0-9) and dictates the state of each segment, allowing for the accurate representation of the number on the display.

Interfacing I2C LCD with Arduino

The key difference in coding between connecting an I2C LCD to an Arduino and connecting a 7-segment display is that the I2C LCD is easier to program with since it utilises a basic serial communication protocol that makes displaying text and simple images much simpler. Because of specialised libraries that abstract the technical details of I2C communication, I2C LCDs are easier to code. The coding process is streamlined because the user may focus on high-level functions rather than low-level details. The 7-Segment Display, on the other hand, involves additional control over the individual segments, that includes digital logic and binary-coded decimal calculations. Individual LEDs in matrix LED displays must be controlled using more advanced multiplexing techniques that require precise timing and low level control.

Conclusion

In conclusion, this experiment successfully achieved its objectives of demonstrating the 7-segment display to visually represent numerical digits in sequence, and it was also designed, constructed, and tested throughout the experiment. Before starting this experiment, we aimed to control and modify the 7-segment to display numerical digits; therefore, we connected an Arduino microcontroller to a 7-segment display and added push buttons as input devices. With the use of Tinkercad, we were able to visualize and validate the desired outcomes before constructing a physical circuit connection using Arduino. We have come to the realization that we were right to assume the push button would be a vital tool for interacting with the 7-segment display. Therefore, our hypothesis was indeed realized.

However, throughout this experiment, numerous possible problems occurred and compromised the system's performance and functionality. We would have hands-on experience troubleshooting and debugging electronic circuits, and it would improve our

insight regarding the theoretical theories. With this experience, we are capable of implementing the interface between Arduino and the 7-segment display in our innovations in the future. It is important to enhance our skills and knowledge about this matter so that we are able to integrate components into our next experiment or project.

Recommendation

An important area for improvement in future experiments lies in the emphasis on error handling in the code. The experiment should instruct students on how to anticipate and effectively manage unexpected user inputs. For instance, students should be guided on how to handle situations where non-numeric characters are entered on a 7-segment display designed for numeric values. To achieve this the following recommendations should be consider.

1. Future experiments should focus on error handling at their core. This entails teaching the importance of mistake detection and repair in the context of electronic projects to the students.
2. Encouragement should be given to students so they can anticipate and spot probable mistake sources in their work. They should pay special attention to the possibility of running into invalid or improper input values.
3. Students should follow the experiment's instructions when implementing validation checks in their code. These checks are essential for ensuring that the system can react politely to invalid values or input.

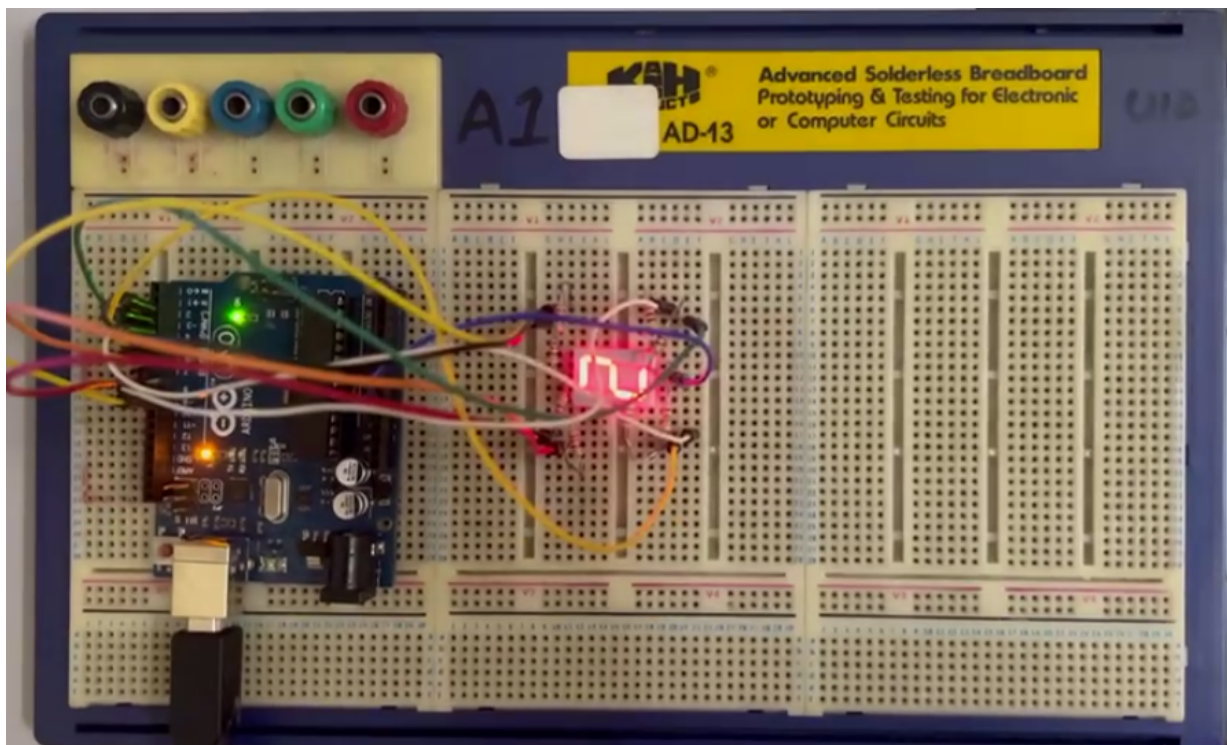
Activities that incorporate error handling as a fundamental component help students develop both practical skills and a deeper understanding of error prevention. These abilities are invaluable and can be used immediately in their next engineering and electrical projects.

The insights and lesson in this experiment was can encourage students to face problem solving. Electronics projects often involve debugging and problem solving, which are valuable skills in engineering. In the same time, this experiment also helps students to build a strong foundation in breadboarding and wiring.

References

1. Zulkifli. (2014). *Mechatronics Interfacing Lab Manual*, (Rev. ed.). Unpublished Class Materials.
2. Interfacing Seven Segement Display with Arduino (Joseph, 2022)
<https://circuitdigest.com/microcontroller-projects/interfacing-seven-segment-display-with-Arduino>

Appendices



- The final version of the circuit connection without a push button, this circuit only shows the sequence number of our 7-segment display.

Acknowledgments

We would like to express our sincere gratitude to the individuals who provided invaluable assistance, guidance, and support during this experiment. First and foremost, we extend our appreciation to Assoc Prof Dr. Zulkifli Bin Zainal Abidin for their comprehensive instruction and mentorship throughout the experiment. Their insights, feedback, and enthusiasm played a important role in our understanding of Arduino programming.

Our fellow group members also deserve special acknowledgment for their collaboration and support. Our discussions, knowledge sharing, and problem-solving sessions greatly enriched our understanding of this experiment's concepts and enhanced the overall learning experience. The collective contributions of our group members have not only enriched our learning experience but have also significantly contributed to the successful completion of this project.

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 undertaken 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 **read** and **understand** the content of the report and no further improvement on the report is needed from any of the individual's contributions 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	<i>sofea</i>	
Name	AISYAH SOFEA OTHMAN	Read <input checked="" type="checkbox"/>
Matric Number	2115386	Understand <input checked="" type="checkbox"/>
Contribution	<ul style="list-style-type: none">- Introduction- Experimental Setup- Recommendations- Reference- Appendices- Acknowledgments	Agree <input checked="" type="checkbox"/>


Signature	<i>aliaa</i>	
Name	SITI ALIAA BINTI IBRAHIM	Read <input checked="" type="checkbox"/>
Matric Number	2112618	Understand <input checked="" type="checkbox"/>
Contribution	<ul style="list-style-type: none">- Material & Equipment	Agree <input checked="" type="checkbox"/>

	<ul style="list-style-type: none"> - Conclusion - Reference - Appendices 	
--	---	--

Signature	<i>adli</i>	
Name	ADLI FAHMI	Read ✓
Matric Number	2113095	Understand ✓
Contribution	<ul style="list-style-type: none"> - Data Analysis - Result - Reference - Appendices 	Agree ✓

Signature	<i>ethar</i>	
Name	ETHAR OSMAN	Read ✓
Matric Number	2111282	Understand ✓
Contribution	<ul style="list-style-type: none"> - Abstract - Discussion - Reference - Appendices 	Agree ✓

Signature	<i>ainaa</i>	
Name	NURAIN AINAA AQILAH BINTI ROSLI	Read ✓
Matric Number	2114560	Understand ✓

Contribution	<ul style="list-style-type: none"> - Methodology - Data Collection - Reference - Appendices 	Agree 
---------------------	---	--