KULLIYAH OF ENGINEERING

MCTA 3203

MECHATRONICS SYSTEM INTEGRATION

SEMESTER 1 2025/2026

SECTION 1

GROUP 10

DIGITAL LOGIC SYSTEM LAB REPORT

NAME	MATRIC NO.
Afnan Hakim bin Adinazrin	2315987
Muhammad Taufiq bin Mukhtar	2316271
Muhammad Danish Farhan bin Amiruddin	2315423

INSTRUCTED BY:

DR ZULKIFLI BIN ZAINAL ABIDIN

TABLE OF CONTENTS

ABSTRACT

This experiment focuses on understanding digital logic systems through the interfacing of a 7-segment display with an Arduino Uno. The circuit uses push buttons to manually control counting and resetting, showing how digital inputs and outputs work together. By programming the Arduino and connecting each segment through current-limiting resistors, the display sequentially shows numbers from 0 to 9. The activity demonstrates key concepts of logic control, circuit interfacing, and microcontroller programming in simple digital applications.

INTRODUCTION

Digital logic systems form the foundation of modern electronic and computing devices. They operate using binary signals, representing logical states of 0 and 1, to perform decision-making and control functions. Understanding these systems is essential for designing and implementing digital circuits that process information accurately.

In this experiment, the focus is on exploring basic digital logic principles through the interfacing of a 7-segment display with an Arduino Uno microcontroller. The project involves programming the Arduino to control each segment of the display and using push buttons to increase or reset numerical output. This hands-on activity provides practical experience in digital input and output control, circuit interfacing, and coding logic—skills fundamental to microcontroller-based system design and mechatronic applications.

MATERIALS AND EQUIPMENT

1. Arduino Uno board
2. Common cathode 7-segment display

3. 220-ohm resistors (7 of them)
4. PushbuΣons (2 or more)
5. Jumper wires
6. Breadboard

EXPERIMENTAL SETUP

1. Connect the common cathode 7-segment display to the Arduino Uno as follows:

    i. Connect each of the 7 segments (a, b, c, d, e, f, g) of the display to separate digital pins on the Arduino (e.g., D2 to D8).

    ii. Connect the common cathode pin of the display to one of the GND (ground) pins on the Arduino.

    iii. Use 220-ohm resistors to connect each of the segment pins to the Arduino pins to limit the current.

2. Connect the pushbuttons to the Arduino:

    i. Connect one leg of each pushbutton to a separate digital pin (e.g., D10 and D11) and connect the other leg of each pushbutton to GND.

    ii. Use 10K-ohm pull-up resistors for each push button by connecting one end of each resistor to the digital pin and the other end to the 5V output of the Arduino.

Methodology

The experiment was carried out to understand the principles of digital logic systems and to implement a basic 7-segment display interfaced with an Arduino Uno. The methodology involved several key stages: circuit design**,** component setup**,** program development**,** and system testing**.**

1. Circuit Design

The circuit was designed to display numeric outputs on a common cathode 7-segment display. Each of the seven segments (labeled A to G) was connected to a dedicated digital output pin on the Arduino (D2–D8).

- Each connection was made through a 220-ohm resistor to limit the current and prevent damage to the display.
- The common cathode pin of the display was connected to the GND of the Arduino board.

Two push buttons were also included:

- **Increment button:** Connected to digital pin D10.
- **Reset button:** Connected to digital pin D11.

2. Hardware Setup

All components — including the Arduino Uno, 7-segment display, resistors, push buttons, jumper wires, and breadboard — were assembled according to the provided schematic. The connections were verified for continuity and correctness before powering the circuit.

3. Program Development

The Arduino IDE was used to develop and upload the control program.

- Each segment (A–G) was declared as an output pin using the `pinMode()` function.
- The code defined the correct segment combinations for digits 0 to 9.
- The program utilized digitalWrite() commands to activate or deactivate segments to display the desired number.
- The increment button increased the counter value sequentially, while the reset button reset the display to zero.

· Appropriate delay functions were used to ensure proper visibility between transitions.

Programming code that is used:

```
// --- Pin setup for 7-segment display ---

const int segmentA = 3;

const int segmentB = 8;

const int segmentC = 2;

const int segmentD = 7;

const int segmentE = 6;

const int segmentF = 4;

const int segmentG = 5;



// --- Push button pins ---

const int buttonIncrement = 10;

const int buttonReset = 11;



// --- Variables ---

int count = 0;

int lastButtonStateInc = HIGH;

int lastButtonStateReset = HIGH;



// Lookup table for digits 0-9 (Common Cathode)

const byte numbers[10][7] = {

  {0,0,0,0,0,0,1}, // 0

  {1,0,0,1,1,1,1}, // 1
  {0,0,1,0,0,1,0}, // 2

  {0,0,0,0,1,1,0}, // 3

  {1,0,0,1,1,0,0}, // 4

  {0,1,0,0,1,0,0}, // 5

  {0,1,0,0,0,0,0}, // 6

  {0,0,0,1,1,1,1}, // 7

  {0,0,0,0,0,0,0}, // 8

  {0,0,0,0,1,0,0}  // 9

};


void setup() {

  // Set all segment pins as OUTPUT

  for (int i = 2; i <= 8; i++) {

    pinMode(i, OUTPUT);

  }


  // Set button pins as INPUT_PULLUP

  pinMode(buttonIncrement, INPUT_PULLUP);

  pinMode(buttonReset, INPUT_PULLUP);


  // Start serial monitor (optional)

  Serial.begin(9600);

  displayDigit(count);

}
```

```
void loop() {

  int currentInc = digitalRead(buttonIncrement);

  int currentReset = digitalRead(buttonReset);



  // Increment button logic (active LOW)

  if (currentInc == HIGH && lastButtonStateInc == LOW) {

    count++;

    if (count > 9) count = 0;

    displayDigit(count);

    Serial.print("Count: "); Serial.println(count);

    delay(300); // debounce delay

  }



  // Reset button logic (active LOW)

  if (currentReset == HIGH && lastButtonStateReset == LOW) {

    count = 0;

    displayDigit(count);

    Serial.println("Reset!");

    delay(300); // debounce delay

  }



  lastButtonStateInc = currentInc;

  lastButtonStateReset = currentReset;

}
```

```
// --- Function to display a digit on 7-segment ---

void displayDigit(int num) {

  digitalWrite(segmentA, numbers[num][0]);

  digitalWrite(segmentB, numbers[num][1]);

  digitalWrite(segmentC, numbers[num][2]);

  digitalWrite(segmentD, numbers[num][3]);

  digitalWrite(segmentE, numbers[num][4]);

  digitalWrite(segmentF, numbers[num][5]);

  digitalWrite(segmentG, numbers[num][6]);

}
```

4. Testing and Validation

After uploading the code, the system was powered via the Arduino USB connection.

- The functionality of each segment was tested by manually triggering the program loop.
- Button responses were verified to ensure correct incrementing and resetting actions.
- Observations were made on the Serial Monitor to confirm that software logic matched hardware behavior.
  Any connection or display issues were debugged and corrected by checking wiring and pin assignments.

5. Safety and Precautions

- The experiment was performed on a breadboard to allow easy modification and reduce soldering risks.
- Resistors were used to protect LEDs in the display from excessive current.
- The circuit was powered through the Arduino USB port to maintain low voltage (5V) operation.

Data Collection

The data collection process for this experiment involved recording the output behavior of the 7-segment display in response to the **increment** and **reset** button inputs. The purpose was to verify correct digital logic operation, pin functionality, and sequential display output.

1. Observation Setup

After completing the wiring and uploading the Arduino program, the system was powered via USB connection. The Serial Monitor in the Arduino IDE was opened to display printed outputs for each button press, providing digital confirmation of the counter's state.

2. Data Recording Procedure

- The increment button (D10) was pressed repeatedly to increase the displayed number from 0 to 9.
- Each change on the 7-segment display was observed and recorded alongside the corresponding serial output message (e.g., "Count: 1", "Count: 2", etc.).
- After reaching the maximum count (9), the display was expected to roll over to 0.
- The reset button (D11) was then pressed to reset the count to zero, and the "Reset!" message was verified on the Serial Monitor.

3. Collected Data

Button Action	Expected Display	Observed Display	Serial Output	Remarks
Initial state	0	0	“Count: 0”	System initialized correctly
Press Increment once	1	1	“Count: 1”	Normal operation
Press Increment twice	2	2	“Count: 2”	Normal operation
...	...	...	...	...
Press Increment 9 times	9	9	“Count: 9”	Correct final count
Press Increment again	0	0	“Count: 0”	Counter rolled over
Press Reset	0	0	“Reset!”	System reset verified

4. Summary

The collected data confirmed that:

- The system successfully displayed numeric sequences from 0 to 9.
- The increment and reset functions operated correctly according to program logic.
- The serial output data matched the physical display, validating both hardware and software performance

DATA ANALYSIS

In this experiment, the Arduino was interfaced with a common cathode 7-segment display to show numbers from 0 to 9 using digital pins D2 until D8. Two push buttons were used as inputs. One for increment the number and another for resetting it to zero.

Each segment of the 7-segment display corresponds to a specific combination of HIGH and LOW logic signals from the Arduino. The logical mapping between the binary outputs (1 = ON, 0 = OFF) and the displayed numbers is shown below:

Digit	Segment Combination (a–g)	Binary Logic Pattern
0	a, b, c, d, e, f	1111110
1	b, c	0110000
2	a, b, g, e, d	1101101
3	a, b, c, d, g	1111001
4	a, f, g, c, d	0110011
5	a, f, g, c, d	1011011
6	a, f, e, d, c, g	1011111
7	a, b, c	1110000
8	a, b, c, d, e, f, g	1111111
9	a, b, c, d, f, g	1111011

The output data were verified by observing the display transition through each number. Logical high voltage (5V) from the Arduino successfully activated the intended LED segments.

RESULTS

The experiment successfully demonstrated the manual control of a 7-segment display using push buttons connected to an Arduino Uno.
Key observations include:

- Pressing the increment button increased the displayed value from 0 to 9 sequentially.
- Pressing the reset button set the display back to 0.
- Each segment illuminated accurately according to the binary logic combinations.
- The system functioned with consistent response time and without flickering or lag.

The experiment achieved its main objective which is to understand and implement digital logic control of output devices using Arduino.

Discussion

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 is interfaced with Arduino using only two communication lines — SDA (Serial Data) and SCL (Serial Clock) — instead of connecting multiple digital pins like a 7-segment display. The I2C protocol allows multiple devices to share the same two lines by using unique addresses, making the wiring much simpler and more efficient.

Hardware Connection:

- SDA → Arduino A4
- SCL → Arduino A5
- VCC → 5V

- GND → Ground

Coding Principle:

To use an I2C LCD, we include the library:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);  // LCD address, columns, and rows
```

Then initialize the LCD:

```
lcd.init();
lcd.backlight();
lcd.print("Hello World");
```

This library automatically handles communication through the SDA and SCL lines, sending data in serial form to display characters.

Comparison: I2C LCD vs. 7-Segment Display vs. Matrix LED

Feature	**I2C LCD**	**7-Segment Display**	**LED Matrix**
Connection Type	Serial (2 wires)	Parallel (7+ wires)	Parallel/Multiplexed
Data Control	Controlled via I2C protocol using address	Each segment controlled individually	Rows and columns activated rapidly
Output Type	Alphanumeric (letters + numbers)	Numeric only	Patterns, symbols, or animations
Coding Complexity	Low (library-based)	Moderate (manual logic combinations)	High (requires multiplexing logic)

Power & Pins Used	Fewer pins, lower complexity	Many digital pins	Moderate–high power use

Interpretation

The I2C LCD is more efficient and flexible compared to the 7-segment display and LED matrix, as it allows more information to be shown using fewer pins and simpler code. However, the 7-segment display remains ideal for basic numeric applications and logic demonstrations, which aligns with the learning objective of this experiment — understanding binary output control and manual interfacing.

The successful display of numbers from 0 to 9 confirms that digital logic operations can be applied effectively to control physical electronic components. Each LED segment represents a binary output, and their combinations form recognizable numeric digits.

Expected vs. Observed Outcomes:

- Expected**:** Each button press would cause a smooth transition of numbers with accurate illumination.
- Observed**:** The actual performance matched expectations. There is no significant failure were observed.

Possible Sources of Error:

- Loose jumper connections or insufficient grounding could cause segment flickering.
- Incorrect resistor values might affect brightness or damage LEDs.
- Button debounce effects could cause unintentional multiple increments.
- Using different digital pins than the coding.
- Power supply is not enough or the USB wire use is broken.

Limitations:

- The system is limited to one 7-segment display, thus displaying only single-digit numbers.

Manual push-button control limits automation and scalability.

CONCLUSION

The experiment successfully demonstrated the basic principles of digital logic and circuit interface using an Arduino Uno and a 7-segment display. The counting and reset functions were effectively implemented through push-button inputs, allowing a clearer understanding of digital input–output operations and control logic. This activity provided valuable hands-on experience in programming, wiring, and testing digital components. Overall, the experiment strengthened understanding of how digital systems function and served as a foundation for more advanced projects in microcontroller and automation design. Through this experiment, it was also learned that careful wiring, proper coding, and systematic troubleshooting are essential for achieving accurate and reliable circuit operation.

Recommendations

- Hardware Enhancement**:** Add a debounce circuit or software delay to stabilize button input readings.
- Software Improvement**:** Integrate a counter function with automatic increment timing for continuous counting.
- System Extension**:** Expand to a dual 7-segment display for two-digit outputs using multiplexing.
- Alternative Display**:** Try interfacing with an I2C LCD, which simplifies wiring and allows alphanumeric output.
- Learning Continuation**:** Explore binary-to-decimal decoders (like IC 7447) for hardware-based segment control.

REFERENCES

7 Segment Display. (n.d.). Retrieved from https://components101.com/displays/7-segment-display-pinout-working-datasheet

Experiment Notes: *Week 2 – Digital Logic System: Basic Logic Gates, Electronic Circuit Interfacing, Basic ALU, 7-Segment Display, IC-Based Interfacing Applications (ver.2).*

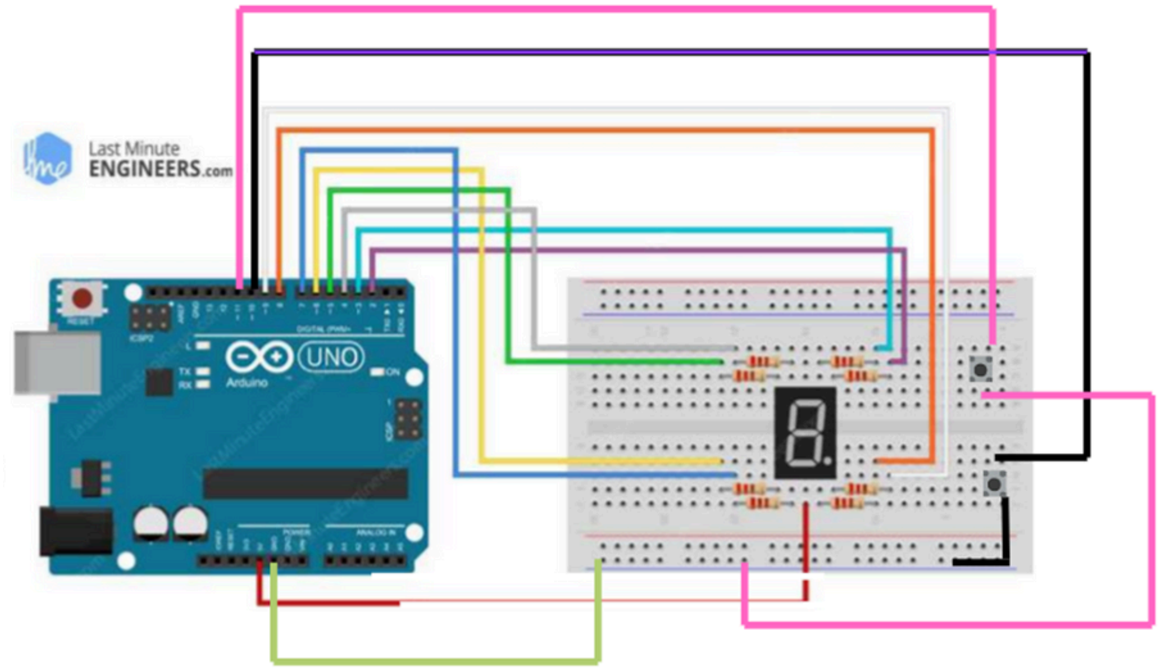
Mazidi, M. A. (2015). AVR Microcontroller and Embedded Systems: Using Assembly and C: Pearson New International Edition. Retrieved from https://books.google.com/books/about/AVR_Microcontroller_and_Embedded_Systems.html?id=gF1TDwAAQBAJ

(N.d.). Retrieved from https://www.displaysino.com/newDetails/The-Differences-and-Applications-of-Segment-Displays-and-Dot-Matrix-Displays.html

Scherz, P. (1970). Practical Electronics for Inventors, Fourth Edition. Retrieved from https://www.abebooks.com/9781259587542/Practical-Electronics-Inventors-Fourth-Edition-1259587541/plp

APPENDICES

Circuit diagram:



Code snippet:

```
// --- Pin setup for 7-segment display ---
const int segmentA = 3;
const int segmentB = 8;
const int segmentC = 2;
const int segmentD = 7;
const int segmentE = 6;
const int segmentF = 4;
const int segmentG = 5;

// --- Push button pins ---
const int buttonIncrement = 10;
const int buttonReset = 11;

// --- Variables ---
int count = 0;
int lastButtonStateInc = HIGH;
int lastButtonStateReset = HIGH;
```

```
// Lookup table for digits 0–9 (Common Cathode)
const byte numbers[10][7] = {
  {0,0,0,0,0,0,1}, // 0
  {1,0,0,1,1,1,1}, // 1
  {0,0,1,0,0,1,0}, // 2
  {0,0,0,0,1,1,0}, // 3
  {1,0,0,1,1,0,0}, // 4
  {0,1,0,0,1,0,0}, // 5
  {0,1,0,0,0,0,0}, // 6
  {0,0,0,1,1,1,1}, // 7
  {0,0,0,0,0,0,0}, // 8
  {0,0,0,0,1,0,0}  // 9
};

void setup() {
  // Set all segment pins as OUTPUT
  for (int i = 2; i <= 8; i++) {
    pinMode(i, OUTPUT);
  }

  // Set button pins as INPUT_PULLUP
  pinMode(buttonIncrement, INPUT_PULLUP);
  pinMode(buttonReset, INPUT_PULLUP);

  // Start serial monitor (optional)
  Serial.begin(9600);
  displayDigit(count);
}

void loop() {
  int currentInc = digitalRead(buttonIncrement);
  int currentReset = digitalRead(buttonReset);

  // Increment button logic
  if (currentInc == HIGH && lastButtonStateInc == LOW) {
    count++;
    if (count > 9) count = 0;
    displayDigit(count);
    Serial.print("Count: "); Serial.println(count);
    delay(300); // debounce delay
  }

  // Reset button logic
  if (currentReset == HIGH && lastButtonStateReset == LOW) {
    count = 0;
    displayDigit(count);
    Serial.println("Reset!");
    delay(300); // debounce delay
  }

  lastButtonStateInc = currentInc;
  lastButtonStateReset = currentReset;
}

// --- Function to display a digit on 7-segment ---
void displayDigit(int num) {
  digitalWrite(segmentA, numbers[num][0]);
  digitalWrite(segmentB, numbers[num][1]);
  digitalWrite(segmentC, numbers[num][2]);
  digitalWrite(segmentD, numbers[num][3]);
  digitalWrite(segmentE, numbers[num][4]);
  digitalWrite(segmentF, numbers[num][5]);
  digitalWrite(segmentG, numbers[num][6]);
}
```

This program controls the 7-segment display using digital pins 2–8 on the Arduino Uno. The increment and reset functions are handled using push buttons connected to pins 10 and 11 respectively. Each number pattern (0–9) is stored in a lookup table for quick display. The system uses internal pull-up resistors and debounce delays for stable input reading.

Acknowledgement

We would like to express our deepest gratitude to Sir Zulkifli bin Zainal Abidin , our course instructor for *Mechatronics System Integration (MCTA3203)*, for his invaluable guidance, encouragement, and continuous support throughout the completion of this Digital Logic System lab project.

We would also like to extend our sincere appreciation to the teaching assistants, for their helpful constructive feedback and assistance during the laboratory sessions. Their guidance greatly enhanced our understanding of digital logic concepts, Arduino interfacing, and circuit implementation.

Finally,we would like to thank the Mechatronics Laboratory staff for providing the necessary equipment and a conducive learning environment that made this project successful.

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 acknowledgement, and that the original work contained herein have 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's 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.

Read	/
Understand	/
Agree	/

Signature:
Name: Afnan Hakim bin Adinazrin
Matric Number: 2315987
Contribution: Abstract,Introduction,Materials & Equipment,
Experimental Setup,Conclusion

Read	/
Understand	/
Agree	/

Signature:
Name: Muhammad Taufiq bin Mukhtar
Matric Number: 2316271
Contribution: Data Analysis,Results,Discussion,

Recommendations,References

Signature:
Name: Muhammad Danish Farhan bin Amiruddin
Matric Number: 2315423
Contribution: Methodology,Data Collection,Appendices,
Acknowledgements

Read	/
Understand	/
Agree	/