KULLIYYAH OF ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

Mechatronics System Integration (MCTA3203)

WEEK 2: DIGITAL LOGIC SYSTEM

Section 1

(Group E)

Adam Mikhail Bin Khairul Effendy	2117613
Muhammad Afiq bin Mazhalimi	2110491
Muhammad Fahmi Ikhwan bin Ahmad Suparjo	2111167
Eiman Bin Azzam	2110565
Zulhilmi bin Abdul Rahman	2117679

Date of submission: 25/10/2023

Prepared for: Dr. Zulkifli bin Zainal Abidin

Abstract

The seven-segment display is a basic gadget that utilizes seven segments of lights to display numbers. The technology to display lights in sequences is expanded to various uses such as in alarm clocks, traffic lights and gas station price signs. To light up a seven segment display, a digital input HIGH or LOW has to be sent to the specified segments. To interface the display, a microcontroller such as an Arduino UNO is utilized to send digital inputs to the display to create a counter that increments from 0 to 9. Pushbuttons and pull-down resistors are used to control the increment and reset the counter, by sending a digital input to the microcontroller. By coding the Arduino UNO using the Arduino IDE, sequences of numbers can be created.

Table of Contents

1.1 Introduction

The experiment is done to help us understand how a 7-segment display interfaces with an Arduino Uno and how to control the display manually using buttons. The Arduino Uno will light up each segment of the display by sending digital signals HIGH or LOW and sequences can be created through coding to create a counter, with a pushbutton as a trigger that is read by the Arduino to increment the counter.

Objectives:

1) To understand how a 7-segment display works.
2) To know how to interface a 7-segment display with an Arduino Uno.

1.2 Materials & Equipments

1) Arduino Uno board, 1
2) Common cathode 7-segment display, 1
3) 220-ohm resistors, 7
4) Pushbuttons, 2 or more
5) Jumper wires
6) Breadboard, 1

1.3 Experimental Setup

A. The common cathode 7-segment display was connected to the Arduino Uno (Appendix A) as follows:

· Each of the 7 segments (a, b, c, d, e, f, g) of the display was connected to separate digital pins on the Arduino (e.g., D0 to D6).

· The common cathode pin of the display was connected to one of the GND (ground) pins on the Arduino.

· 220-ohm resistors were used to connect each of the segment pins to the Arduino pins to limit the current.

B. The pushbutton was connected to the Arduino Uno (Appendix A) as follows:

· One leg of each pushbutton was connected to a separate digital pin (e.g., D9 and D10) and the other leg of each pushbutton was connected to GND.

· 10K-ohm pull-down resistors were used for each pushbutton by connecting one end of each resistor to the digital pin and the other end to the 5V output of the Arduino.

1.4 Methodology

1. The circuit was constructed according to the circuit setup instructions. (Appendix A)
2. The provided Arduino code was uploaded and modified to our Arduino Mega. (Appendix B)
3. The Serial Monitor was opened using the Arduino IDE.
4. By pressing the increment button, the count is increased. The 7-segment display showed the numbers from 0 to 9 sequentially.
5. By pressing the reset button, 1. the count resets to 0.

1.5 Data Collection

- No data collection

1.6 Data Analysis

- No data collection

1.7 Results

- The experiment resulted in a functional counter display from 0 to 9, with buttons that allowed the display to be adjusted to increase in steps of 1 and reset to 0.
- Video link for results: Refer to Appendix C below or at GitHub

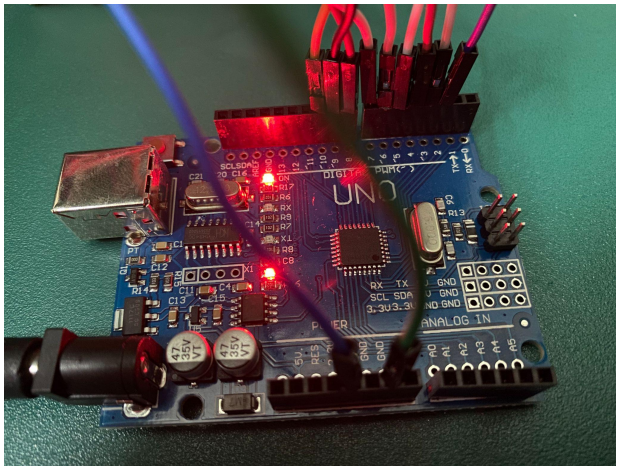


Figure 1: Wire connections on arduino uno

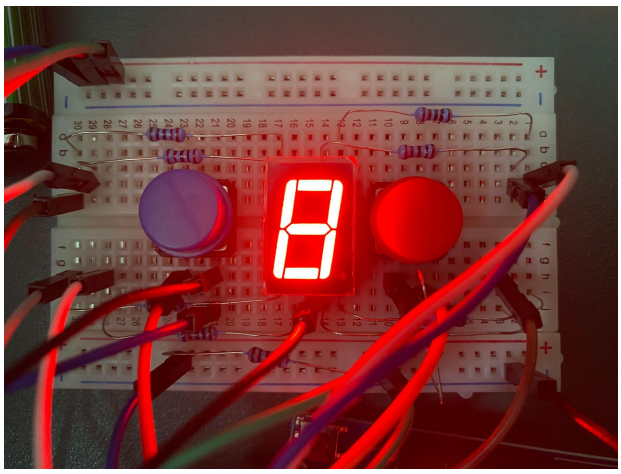


Figure 2: Wire connections on 7-segment and push button

1.8 Discussion

How to interface an I2C LCD with Arduino? Explain the coding principle behind it compared with 7 segments display and matrix LED.

To interface the I2C, we have to connect 4 pins. Which is the VCC, GND and the SDA data line) and SCL (clock line). For the Arduino Mega , the SCL and SDA pins are digital pins 21 and 20 respectively. We also require the LiquidCrystal_I2C library to be installed in the Arduino IDE.

Firstly, determine the address of your I2C, run the I2C scanner sketch from the Wire.h library. After that , pass the I2C address and the display dimensions of the LCD through the constructor. For example, an address of 0x3F and a 16x2 display should be entered as :

“ LiquidCrystal_I2C lcd(0x3F, 16, 2); “

Once initialized, you can use various lcd functions such as lcd.clear() to clear the screen. To print numbers or text on the LCD, use the lcd.print function.

An I2C LCD is more complex compared to a 7 segments display. While a 7 segment display can only control up to 7 bits of HIGH and LOW, an I2C display uses a 5x8 matrix, an array consisting of 8 bytes to represent the characters. Coding wise, the I2C LCD is easier to code as it calls built in functions from a library to create complex characters compared to 7 segments or matrix LEDs that require you to call HIGH or LOW for each bit.

1.9 Conclusion

In conclusion, the experiment involving the integration of a seven-segment display with two push buttons for count up and reset functions has provided valuable insights into the practical application of digital displays and user interface design. The implementation of these buttons allowed for user interaction, making it possible to increment the displayed count and reset it to zero, demonstrating the versatility and utility of the seven-segment display in various real-world scenarios.

Through this experiment, we have experienced firsthand how a simple arrangement of seven segments can be harnessed to communicate numerical information effectively. The count-up button showcases the display's capacity to convey increasing values, while the reset button highlights its responsiveness to user input for control and interaction.

This experiment underscores the relevance and importance of understanding the foundational components of digital technology. The seven-segment display, with its minimalistic design and ease of use, serves as a gateway to comprehending the broader principles of digital communication. The knowledge gained from this experiment is not only instructive but also

illuminates the way in which digital displays are interwoven into our daily lives, from household appliances to public information systems, contributing to our enhanced technological literacy and appreciation of the digital world.

1.10 Recommendations

To make this experiment, some recommendations are required to help us get started with this project. Here are some of the recommendations:

1) Gather the required components

Gathering the required components is the initial step in any electronics or Arduino project, and it serves several important purposes. Collecting the necessary components in advance ensures that everything needed to complete the project without interruptions or delays. As gathering components, it can visualize how everything will be connected and identify any potential issues with compatibility, power requirements, or physical space constraints. This step provides an opportunity to learn about the components used. Other than that, we can understand voltage and current requirements, and gain a deeper understanding of how each component works. In summary, gathering the required components is an essential preparatory step in any electronics project. It helps ensure a smoother, more efficient, and successful project execution while promoting safety and learning along the way.

2) Understand 7-Segment Display

Understanding the 7-segment display is a crucial step in interfacing it with an Arduino. A 7-segment display is a common output device used to display numeric and sometimes alphabetic characters. It consists of seven individual LED segments arranged in a specific pattern. The pattern of segments that are illuminated creates the appearance of numbers and letters on the display. Before connecting the 7-segment display to the Arduino, it is necessary to identify the common pin (anode or cathode). This is essential for proper wiring and coding. Always double-check the connections and documentation to ensure the correct type and pins for specific display.

3) Safety Precautions

Safety precautions are vital when working with electronics and microcontrollers like the Arduino to ensure its well-being and the protection of components. Always turn off the power supply to Arduino and any external components before making any physical connections or disconnections. This prevents accidental short circuits or electric shocks. When interfacing with LEDs, including those on 7-segment displays, always use current-limiting resistors to prevent excessive current flow, which can damage the LEDs. Calculate the resistor values based on the LED's forward voltage and desired current. Insulate and secure all wires and components properly to prevent short circuits. Ensure that exposed wires or solder joints do not come into contact with each other, as this can cause sparks or damage.Lastly, ensure that all connections are secure and properly seated.

1.11 References

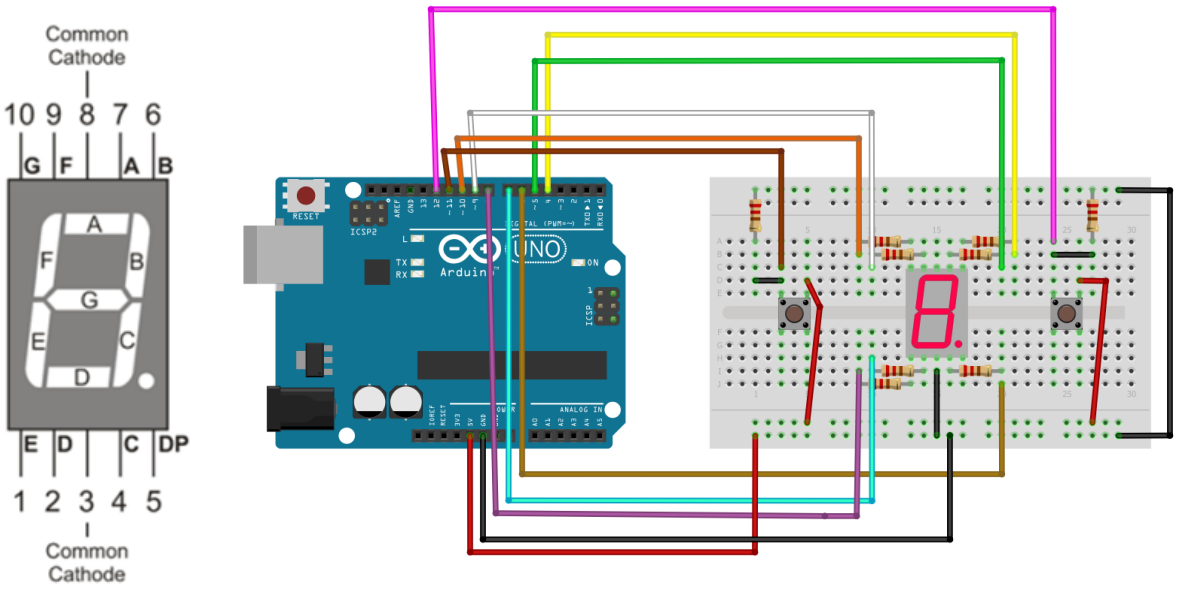
Interfacing the I2C LCD monitor:

https://lastminuteengineers.com/i2c-lcd-arduino-tutorial/

Interfacing an 8x8 matrix LED:

https://docs.arduino.cc/built-in-examples/display/RowColumnScanning

1.12 Appendices



Appendix A: Connection of Arduino Uno to 7-segment display

```
const int segmentA = 2; // D0
const int segmentB = 3; // D1
const int segmentC = 4; // D2
const int segmentD = 5; // D3
const int segmentE = 6; // D4
const int segmentF = 7; // D5
const int segmentG = 8; // D6

const int button1 = 9;
const int button2 = 10;

int i=0;

void setup() {
// Initialize the digital pins as OUTPUTs
pinMode(segmentA, OUTPUT);
pinMode(segmentB, OUTPUT);
pinMode(segmentC, OUTPUT);
pinMode(segmentD, OUTPUT);
pinMode(segmentE, OUTPUT);
pinMode(segmentF, OUTPUT);
pinMode(segmentG, OUTPUT);

pinMode(button1,INPUT);
pinMode(button2,INPUT);

}
```

```
void loop() {

int x=digitalRead(button1);
int y=digitalRead(button2);

switch(i){
  case 0:call0();
  break;
  case 1:call1();
  break;
  case 2:call2();
  break;
  case 3:call3();
  break;
  case 4:call4();
  break;
  case 5:call5();
  break;
  case 6:call6();
  break;
  case 7:call7();
  break;
  case 8:call8();
  break;
  case 9:call9();
  break;
  case 10:i=0;
  }
if(x==1)
  {
  i++;
  delay(500);
    }

if(y==1)
{
  i=0;
}

}

void call0(){

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

```
}

void call1(){

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

}
void call2(){

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

}
void call3(){

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

}
void call4(){

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

}
void call5(){

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

```
digitalWrite(segmentE, LOW);
digitalWrite(segmentF, HIGH);
digitalWrite(segmentG,HIGH);

}
void call6(){

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

}
void call7(){

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

}
void call8(){

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

}
void call9(){

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

}
void reset(){
digitalWrite(segmentA, HIGH);
digitalWrite(segmentB, HIGH);
```

```
digitalWrite(segmentC, HIGH);
digitalWrite(segmentD, HIGH);
digitalWrite(segmentE, HIGH);
digitalWrite(segmentF, HIGH);
digitalWrite(segmentG,LOW);
}
void anime(){
  for(int i=0;i<3;i++){
digitalWrite(segmentA, HIGH);
delay(10);
digitalWrite(segmentB, HIGH);
delay(10);
digitalWrite(segmentC, HIGH);
delay(10);
digitalWrite(segmentD, HIGH);
delay(10);
digitalWrite(segmentE, HIGH);
delay(10);
digitalWrite(segmentF, HIGH);
delay(10);
digitalWrite(segmentG,HIGH);
delay(10);
digitalWrite(segmentA, LOW);
delay(10);
digitalWrite(segmentB, LOW);
delay(10);
digitalWrite(segmentC, LOW);
delay(10);
digitalWrite(segmentD, LOW);
delay(10);
digitalWrite(segmentE, LOW);
delay(10);
digitalWrite(segmentF, LOW);
delay(10);
digitalWrite(segmentG,LOW);
delay(10);
}
}
```

Appendix B : Arduino Coding (Refer to GitHub for .ino file)

https://github.com/EimanAzzam/Group-E-Mecha-Integration/blob/main/Week2/Appendix-C.mp4

Appendix C: Video demonstration of System(download raw at GitHub/link above)

1.13 Acknowledgments

We would like to acknowledge and give our thanks to our instructor, Dr Zulkifli, for helping us understand the topic of digital I/O and basic interfacing while also giving us room to explore and experiment freely. May we meet again in more advanced topics, surely his advice will be immensely helpful in our future projects.

Also, we would like to thank Allah S.W.T for giving us guidance throughout our lives and helping us overcome the difficulties faced in making this experiment.

1.14 Student's Declaration

Declaration:

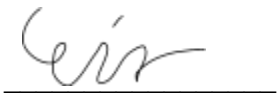
We certify that this project/assignment is entirely our own work, except where we have given fully documented references to the work of others, and that the material in this assignment has not previously been submitted for assessment in any formal course of study.

_adam_____ Date: 24/10/2023

Name: Adam Mikhail Bin Khairul Effendy

fahmi_____

Name: Muhammad Fahmi Ikhwan Bin Ahmad Suparjo



Name: Eiman Bin Azzam

afiq

Name: Muhammad Afiq bin Mazhalimi

_Zul_____

Name: Zulhilmi bin Abdul Rahman