LABORATORY REPORT

LAB 1: Digital Logic System

GROUP H

PROGRAMME: MECHATRONICS ENGINEERING

GROUP MEMBERS:	MATRIC NO:
1. SYEDA SAMIA	2123536
2. ATHIRAH HAZIQAH BINTI BADERULHISHAM	2210240
3. SITI HEIDI AMIRA BINTI AZRUL	2111970

DATE OF SUBMISSION:
Wednesday, 20th March 2024

TABLE OF CONTENTS

Table of Contents

INTRODUCTION

We will discuss in this report an experiment done using a 7-segment display, pushbuttons, resistors and an Arduino Mega. Firstly, resistors are needed to limit the current and protect the display and other components from overheating. Moreover, Pushbottuns help translate what the user wants to the system so it is able to provide the wanted result. Furthermore, the display is where the result wanted from the system is shown to the user. Furthermore, the 7-segment display could be of two types: anode or cathode. An anode display is connected to the positive voltage whereas the cathode one connects to the ground, so fewer pins are required than if each segment had its own pairs of cathode & anode pins. Last but not least, By using the 7-segment display and applications involving integrated circuit interfacing we delve into the study of logic gates, electronic circuit interfacing and ALUs, and how these concepts help us conquer this subject's field.

PROCEDURE

Materials And Equipment:

- Arduino Uno board
- Arduino USB cable
- Common cathode 7-segment display
- 220-ohm resistors (2 of them)
- Pushbutton
- Jumper wires
- Breadboard

Experimental Setup:

1. Connecting the common cathode 7-segment display to the Arduino Uno as follows:
    a. Connect segments a, b, c, d, e, f, g of the display to digital pins D2, D3, D4, D5, D6, D7 and D8 on the Arduino like on Fig 1.1.
    b. Connect the common cathode pins of the display to the GND (ground) pins on the Arduino.
    c. Use 220-ohm resistors to connect each of the common cathode pins to the Arduino GND pins to limit the current.
2. Connecting the pushbutton to the Arduino:
    a. Connect one leg of the pushbutton to a separate digital pin D9 and connect the other leg of the pushbutton to GND.

Methodology:

1. Using the circuit setup shown in fig 1.1, we built the needed circuit.
2. An Arduino code was uploaded to the Arduino Mega.
3. The pushbutton was pressed to increment the number shown on the 7-segment display from 0 to 9, after the number 9 the display returned to 0 again.

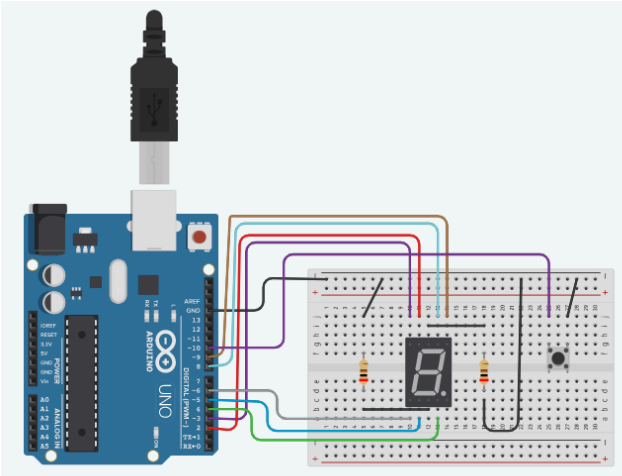

Fig 1.1

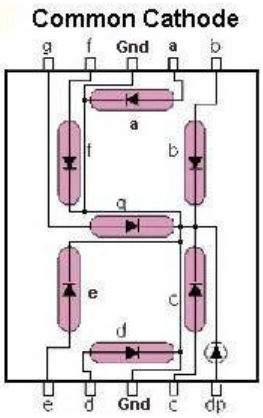

Fig 1.2

Arduino Code:

```
const int a = 2;
const int b = 3;
const int c = 4;
const int d = 5;
const int e = 6;
const int f = 7;
const int g = 8;

bool bPress = false;     //to track if the button is pressed
const int buttonPin = 9;

//Variables will change
int buttonPushCounter = 0;
int buttonState = 0;
int lastButtonState = 0;

void setup()
{
  pinMode(a,OUTPUT);
  pinMode(b,OUTPUT);
  pinMode(c,OUTPUT);
  pinMode(d,OUTPUT);
  pinMode(e,OUTPUT);
  pinMode(f,OUTPUT);
  pinMode(g,OUTPUT);
```

```
  pinMode(buttonPin,INPUT_PULLUP);
  Serial.begin(9600);
  displayDigit(buttonPushCounter);
}

void loop()
{
  buttonState = digitalRead(buttonPin);

  //compare the buttonState to its previous state
  if(buttonState != lastButtonState)
  {
    //if the state has changed,increment the counter
    if(buttonState == LOW)
    {
      //if the current state is HIGH then the button went from off to on
      bPress = true;
      buttonPushCounter++;
      if(buttonPushCounter>9) buttonPushCounter = 0;
      Serial.println("on");
    }
    else{
      //if the current state is LOW then the button went from on to off
      Serial.println("off");
    }
    //Delay a little bit to avoid bouncing
    delay(50);
  }
  //save the current state as the last state,for next time through the loop
  lastButtonState = buttonState;

  if(bPress ){
    turnoff();
    displayDigit(buttonPushCounter);
  }
}

void displayDigit(int digit)
{
  //condition for displaying segment a
  if(digit != 1 && digit !=4)
  digitalWrite(a,HIGH);

  //condition for displaying segment b
  if(digit != 5  && digit != 6)
  digitalWrite(b,HIGH);

  //condition for displaying segment c
  if(digit != 2)
  digitalWrite(c,HIGH);

  //condition for displaying segment d
  if(digit != 1 && digit !=4 && digit != 7)
  digitalWrite(d,HIGH);

  //condition for displaying segment e
```

```
  if(digit == 2 || digit ==6 || digit ==8 || digit == 0)
  digitalWrite(e,HIGH);

  //condition for displaying segment f
  if(digit != 1 && digit !=2 && digit != 3 && digit != 7)
  digitalWrite(f,HIGH);

  //condition for displaying segment g
  if(digit != 0 && digit !=1 && digit!= 7)
  digitalWrite(g,HIGH);
}

void turnoff()
{
  digitalWrite(a,LOW);
  digitalWrite(b,LOW);
  digitalWrite(c,LOW);
  digitalWrite(d,LOW);
  digitalWrite(e,LOW);
  digitalWrite(f,LOW);
  digitalWrite(g,LOW);
}
```

RESULTS

After a while of trying to figure out whether the 7-segment display we were using was an anode or cathode type, and trying multiple different configurations for the placement of the resistors, then tweaking the Arduino code to fit our configuration. We were able to finally make the display work and let it exhibit the numbers from 0 to 9 by pressing the pushbutton.

DISCUSSION

To start, ‘bPress’ is a boolean variable tracking the button press state. The digital pin that is connected to the pushbutton is called ‘buttonPin’ in the code while ‘buttonPushCounter’ stores the count of button presses. And ‘buttonState’ and ‘lastButtonState’ monitor the current and previous states of the button, respectively. Secondly, in the setup function, Pin modes are set for the segments and push button, serial communication is initialized and initial count (0) is displayed on the 7-segment display. Next, the code reads the current state of the push button and compares it with the previous state to detect button press changes. When a change is detected, the counter increments and then the appropriate number shows up on the display, according to the conditions stated for each number, and also “on” is printed. When the button is released “off” is printed on the serial monitor, the delay is added to avoid bouncing. The present state of the button is saved for the next operation.

To interface an I2C LCD with Arduino:

1. Connect SDA and SCL pins from the LCD to the corresponding pins on the Arduino.

2. Install the LiquidCrystal_I2C library.

3. Initialize the LCD object with the correct address.

4. Use functions like lcd.begin(), lcd.print(), and lcd.setCursor() to interact with the LCD.

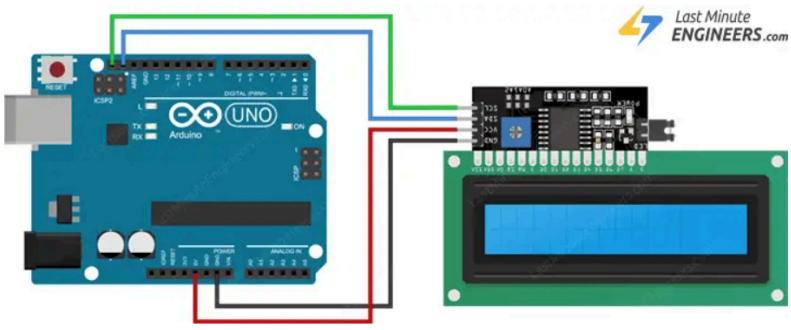

Fig 2.1

Comparing with 7-segment displays and matrix LED:

- I2C LCDs use libraries for easy interfacing, while 7-segment displays and matrix LEDs require manual control of individual segments or LEDs.

- I2C LCDs are efficient for displaying text and basic graphics, while 7-segment displays are suitable for numeric displays, and matrix LEDs offer flexibility for more complex graphics and animations.

CONCLUSION

The process of interfacing a 7-segment display with Arduino can involve challenges such as determining whether it's an anode or cathode type, experimenting with resistor configurations, and adjusting code accordingly. However, with perseverance and troubleshooting, it's possible to successfully display numbers by pressing a pushbutton.

RECOMMENDATIONS

Before starting any experiments in the future, the components needed should be tested a few days before so to find out earlier if any of them are faulty and to figure out appropriate actions like buying or borrowing the items if they seemed to not work properly. Always have more than the required amount of jumper cables since they are weak structurally and could easily break, extra cables will come in handy whenever facing trouble.

REFERENCES

MAHALE, A. (2020, July 12). 7 SEGMENT DISPLAY USING PUSH BUTTON. Medium.

https://medium.com/@aditimahale200/7-segment-display-using-push-button-be805114aaa9

Last Minute Engineers. (2020, November 8). Interface an I2C LCD with Arduino. Last Minute Engineers; Last Minute Engineers. https://lastminuteengineers.com/i2c-lcd-arduino-tutorial/

APPENDICES

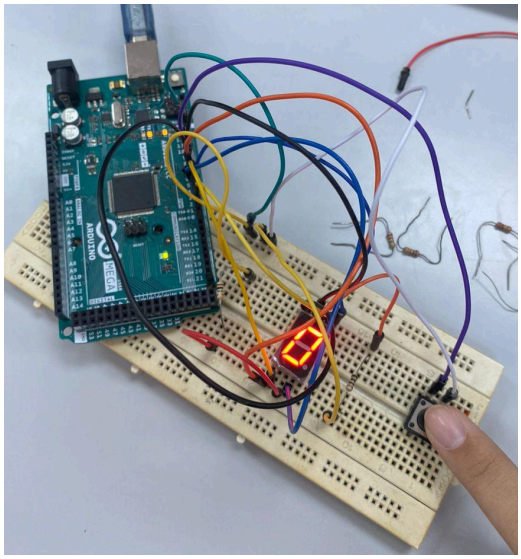

Fig 3.1

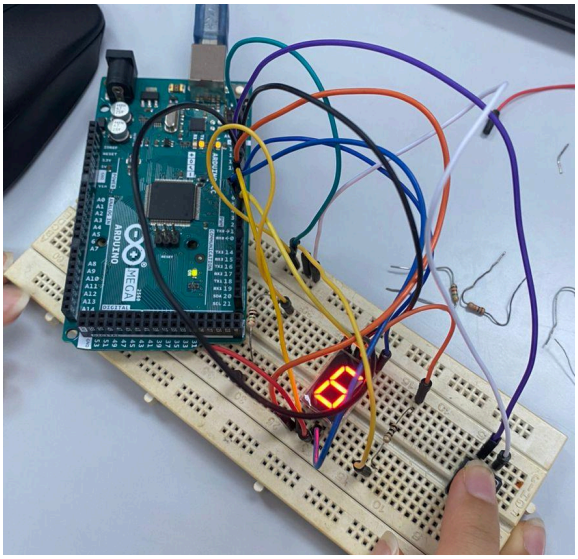

Fig 3.2

ACKNOWLEDGEMENT

We would like to extend our gratitude to the team for their collective efforts in overcoming the challenges encountered during the interfacing of the 7-segment display with Arduino. Special thanks to [mention specific team members if applicable] for their dedication in troubleshooting and adapting the configuration and code to achieve the desired functionality. This experience underscores the significance of teamwork, perseverance, and problem-solving skills in achieving success in electronics projects.

STUDENT'S DECLARATION

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.

Name: Syeda Samia Matric Number: 2123536 Contribution: Introduction, Procedure, Results, Recommendations.	Read	✓
	Understand	✓
	Agree	✓
Name: Athirah Haziqah bt Baderulhisham Matric Number: 2210240 Contribution:	Read	✓
	Understand	✓
	Agree	✓
Name: Siti Heidi Amira Binti Azrul Matric Number: 2111970 Contribution:	Read	✓
	Understand	✓
	Agree	✓