ISTANBUL TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

BLG 351E
MICROCOMPUTER LABORATORY EXPERIMENT REPORT

EXPERIMENT NO	:	4
EXPERIMENT DATE	:	11.12.2020
LAB SESSION	:	FRIDAY - 15.30
GROUP NO	:	G11

GROUP MEMBERS:

150170063	:	BURAK ŞEN
150180080	:	BAŞAR DEMİR
150190719	:	OBEN ÖZGÜR

FALL 2020

Contents

1 INTRODUCTION

In this week's experiment, we have experienced fundamentals of Arduino analog circuitry and its read/write applications. During the experiment, we have implemented potentiometer controlled RGB LED, circuit that reads analog value and displays last-digit and also circuit that read analog value and writes value into 4 seven-segment display. We have also tested our circuits.

2 MATERIALS AND METHODS

- Tinkercad
- Arduino Uno R3
- 8 Resistors
- 8 LEDs
- 2 Buttons
- 2 Switches

2.1 PART 1

In the first part of the experiment, we are expected to implement a LED that can light with different colors and brightness with using potentiometers.

At first we tried to understand how do we get the potantiometers values' we used pinMode function to initialize if potentiometers connection pins are input pins or output pins. We come to know that we can use A0, A1 syntax with analog pins and we used them to get inputs from potentiometers, we found out with Serial.print function our potentiometers values' differs between 0 and 1023 which mean 1024 different values. We did try to give this value right to the LED's legs but it did not quite work, after some thinking we remembered LED's RGB values are differs between 0 and 255 therefore we try to scale our potentiometers values' to RGB intervals. We did divide 1024 to 255 and found out that if we divide potantiometers values' by 4 we could differ our values between 0 and 255. After this we could easily give each potentiometer's value to each one of the legs of the LED and we clearly saw the color changes for each potentiometer value change. We found out about PWM and how the LED's color mixed, when we give an input between 0 and 255 analogWrite function creates a voltage pulse according to value if it is 255 it generates a voltage without a pulse if it is 0 it does not generate any pulse but if it is between 0 and 255 it generates pulses. And if we mix these pulses we can create RGB colors with one LED.

Our implementation is given below.

```
int potentiometer0 = A0; // potentiometer red
int potentiometer1 = A1; // potentiometer blue
int potentiometer2 = A2; // potentiometer green

void setup() {
  pinMode(potentiometer0, INPUT);
  // assign potentiometer 0 as input
  pinMode(potentiometer1, INPUT);
  // assign potentiometer 1 as input
  pinMode(potentiometer2, INPUT);
  // assign potentiometer 2 as input

  pinMode(3,OUTPUT);     // assign digital pin 3 as output
  pinMode(5,OUTPUT);     // assign digital pin 5 as output
  pinMode(6,OUTPUT);     // assign digital pin 6 as output
}

void loop() {

  int po0 = analogRead(potentiometer0); // read from potentiometer 0
  int po1 = analogRead(potentiometer1); // read from potentiometer 1
  int po2 = analogRead(potentiometer2); // read from potentiometer 2

  analogWrite(3,(int)(po2/4));
  // write po2 value in range 0-255 to digital pin 3
  analogWrite(5,(int)(po1/4));
  // write po1 value in range 0-255 to digital pin 5
  analogWrite(6,(int)(po0/4));
  // write po0 value in range 0-255 to digital pin 6
}
```

2.2 PART 2

In the second part of the experiment, we are expected to implement a 7 segment display which shows us from a potentiometer's least significant digit.

We use the knowledge that we earned from the first part of the experiment and easily take the potentiometer's value. At first we tried to analyze the circuit, we did know some information about the BJT topic to know that we have an pnp transistor and if we connect ground to the base we could activate the transistor and we could easily manuplate our 7 segment display (Our 7 segment display an Anode 7 segment displpay if we had to activate it we had to give its base a voltage, light pins are connected to ground). Since we need to take the last digit of the 7 segment display, we had to modify the potentiometers value and get the last digit, we did achieve this by using modula operator "%" we take the potentiometer's value's modula by 10 and we found out the last digit of the value. When we want to display we had to know which pins has to be high voltage to display our specific numbers. We did find some binary values for each of the numbers 0 to 9 and we stored them in an array which sorted way to simplify the proccess for us. After this we just had to give the value we modified from the potentiometer to arrays index to get specific binary code to display on the 7 segment display. When we tried it we the display was broken we tried some other things and we found out if we close the transistor, assign the values and open the transistor we could make the 7 segment display work as we wanted. And when the announcment about the resistors came we tried our code in the changed design and it worked again we did not change anything from our code.

```
int pot = A0; //potentiometer pin

byte sevensegment[10] = { // 7 segment number binary codes
  0b0000001,    //0
  0b1001111,    //1
  0b0010010,    //2
  0b0000110,    //3
  0b1001100,    //4
  0b0100100,    //5
  0b0100000,    //6
  0b0001111,    //7
  0b0000000,    //8
  0b0001100     //9
};

void setup()
{
        DDRD = 0b11111111;  //we set ddrd to output
        DDRB = 0b1; // we set ddrb to input
        pinMode(pot, INPUT);
        // we set potantiometer's analog pin as an input
        PORTB = 0b000000000; //we activate the transistor
}

void loop()
{
        int value = analogRead(pot);
        //we take the value from potentiometer
        value = value % 10;
        // we find the least significant digit
        PORTD = sevensegment[value];
        //we pass the value to 7 segment display
        //according to modified value
        //(we use this value as an index for the array)
}
```

2.3 PART 3

In this part, we are expected to implement a system that reads potentiometer value from analog input and display exact value of the potentiometer in the 4-seven segment displays.

Firstly, we have defined our variables and made operations for port manipulations as we always do. We have already found required binary configurations of number that are in interval 0-9 in the previous parts. Therefore, we also used same array for this question. In the setup phase, we have defined digital pins 0 to 7 and digital pins 8 to 11 as output. To read the value that is given from potentiometer, we have used pinMode function. Thanks to this function, we can determine the modes of analog pins. To increase readability of the code, we defined a variable called potentiometer and assign address of analog 0 port.

Most crucial point of this part was understanding the working mechanism of the 4 seven segment display. We have observed that all displays connected to the PORTD and they display same number when we open all displays. Therefore, we have activated them synchronously and we have solved this problem. When we want to display any number, we are activating each display separately and we show the digit for a some time interval. In infinite loop, our eyes cannot observe the transition between active time and not active time of the displays.

In the loop phase of our implementation, firstly we have read the value of the potentiometer and we have keep this in a value variable. Then, to parse digits of the out variable, we have used modulo and division arithmetic. We have taken modulo 10 of our value, we can reach the its last digit. After parsing last digit, we are activating our seven segment display that keeps our digit by manipulating PORTB. As we have mentioned in previous part, we have to give 0 to base pin of BJT to activate display. Therefore, we have assigned 0 value to necessary PORTB and we have assigned 1 to other ports. We have kept active our display for a short time (5ms) and it allows us to observe the digits. We repeat same operation for the other digits and we get correct implementation.

In essence, we parse our analog input we activate required display and show the digit for each digit of input. In infinite for loop, we have observed that the analog numbers can be seen in seven segment displays.

Our implementation is given below.

```
int potentiometer = A0; //keeps address of analog 0 in variable

//keeps binary configuration of numbers
//for seven segment display
byte sevensegment[10] = {
  0b00000001, //0
  0b01001111, //1
  0b00010010, //2
  0b00000110, //3
  0b01001100, //4
  0b00100100, //5
  0b00100000, //6
  0b00001111, //7
  0b00000000, //8
  0b00001100  //9
};

void setup()
{
        DDRD = 0b11111111; //define digital pins 0 to 7 as output
        DDRB = 0b1111; //define digital pins 8 to 11 as output
        pinMode(potentiometer, INPUT); //defines analog 0 as input
}

void loop()
{
        int value = analogRead(potentiometer);
        //reads the value of potentiometer

        int digit = value % 10;
        //takes last digit of number by modulo 10.

        PORTB = 0b1110;// activates most-right display
        PORTD = sevensegment[digit];
        // gives required configuration of digit to PORTD
        delay(5); //displays in seven segment display for 5ms
        value = (int)(value / 10); //integer divison with 10
        digit = value % 10;
        //takes second right-most digit of number by modulo 10.

        PORTB = 0b1101; //activates second most-right display
        PORTD = sevensegment[digit];
        // gives required configuration of digit to PORTD
        delay(5); //displays in seven segment display for 5ms

        value = (int)(value / 10); //integer divison with 10
        digit = value % 10;
        //takes second left-most digit of number by modulo 10.

        PORTB = 0b1011;//activates second most-left display
        PORTD = sevensegment[digit];
        // gives required configuration of digit to PORTD
        delay(5); //displays in seven segment display for 5ms

        digit = (int)(value / 10);
        //takes left-most digit of number by dividing with 10

        PORTB = 0b0111;//activates most-left display
        PORTD = sevensegment[digit];
        // gives required configuration of digit to PORTD
        delay(5); //displays in seven segment display for 5ms
}
```

3 RESULTS

3.1 PART 1

After our code implementation, we observed the LED's color change

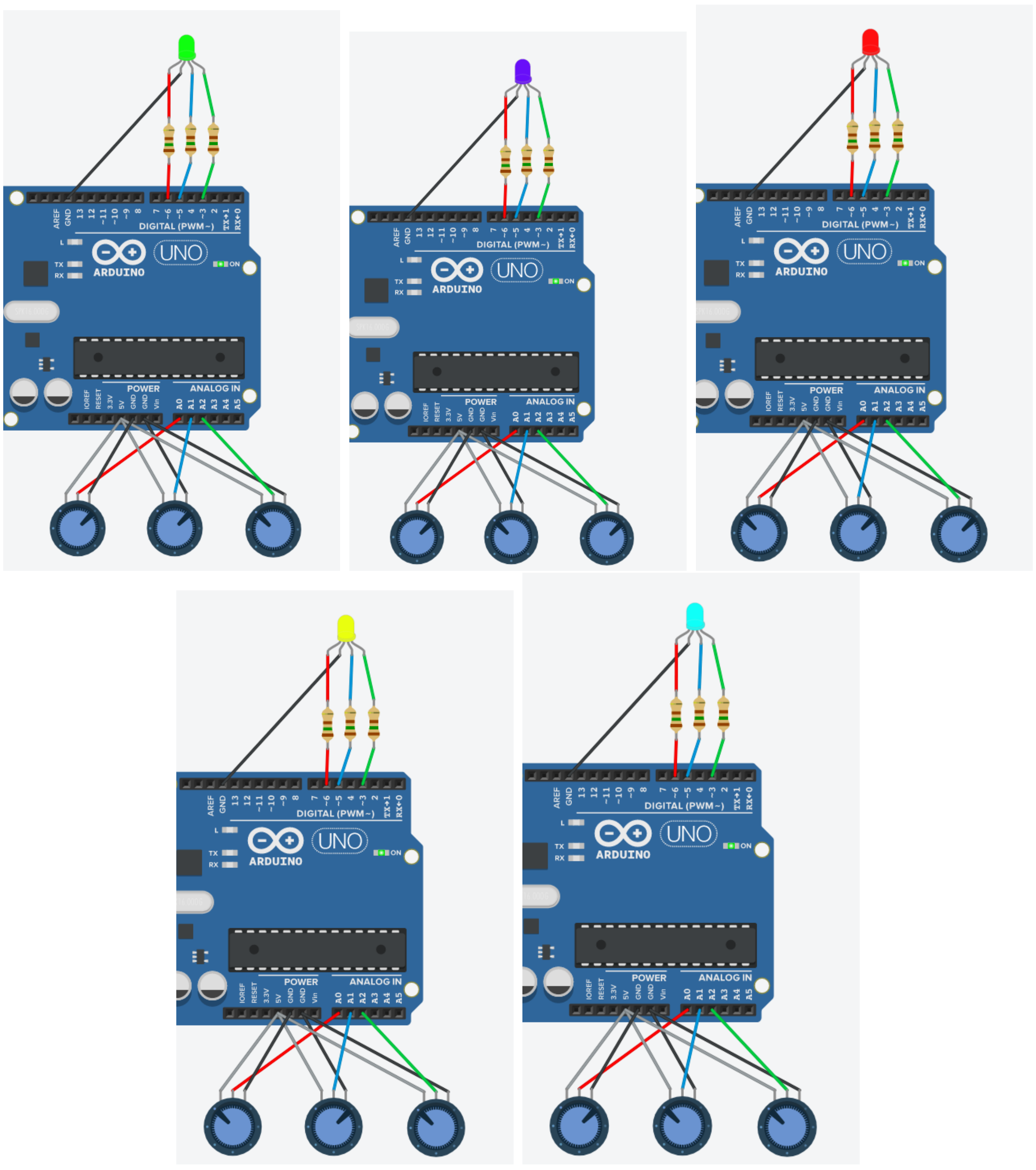

Figure 1: LED color's according to Potentiometer's value

3.2 PART 2

After our code implementation, we can see that our potentiometer's last digit shows up at the 7 segment display.

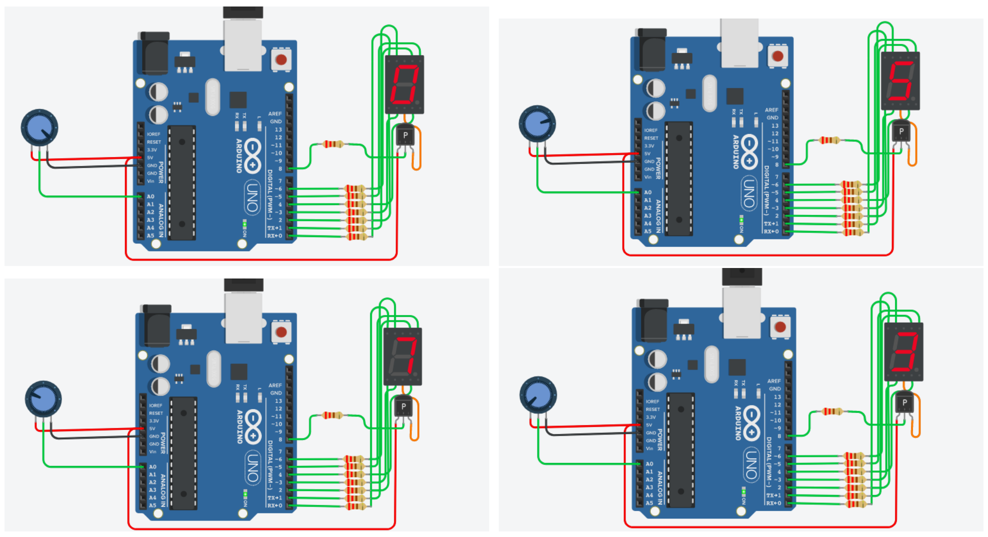

Figure 2: Last Digit value of the Potentiometer

3.3 PART 3

After our code implementation, we can see our potentiometers value from 4 7 segment displays.

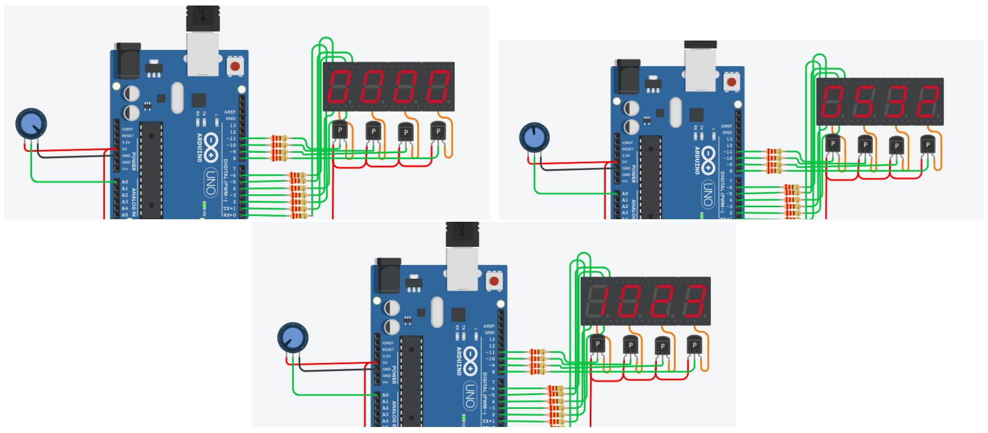

Figure 3: Value of the Potentiometer

4 DISCUSSION

4.1 PART 1

In this part of the experiment, we have implemented a circuit that can control RGB LED using potentiometers. Key points of this part are understanding the working principle of Pulse-width modulation mechanism and mapping the analog value. Firstly, we have implemented this circuit without thinking the principles behind it but we have realize that this approach is not true for our engineering properties even if the circuit works. We have searched for the PWM and we have combined our knowledge about color representation methods with it. Thus, we have understood the working mechanism of our circuit.

Reading analog values with Arduino was new for us. We have learned and used functions that allow us to manipulate analog values. We thought that this part of experiment provides us to make an introduction to analog values and pulse-width modulation principles of Arduino programming. We are happy with our result and what we have learned. Also, it was an good exercise to understand analog values in Arduino.

4.2 PART 2

In this part of the experiment, we have gone back to Arduino's C-like programming language. We were required read an analogue value from the potentiometer between "0" and "1023", then display this value's last digit to the seven-segment display panel. In order to achieve that we have learned how the potentiometer works. From our recitation session in microprocessor system course, we knew how the seven segment display works so we have used our information on these topic.

After grasping the hardware side of the circuitry, we have easily implemented our solution in code thanks to C-like programming language of Arduino and did not have a hard time as in the previous week with Assembly.

4.3 PART 3

In this part of the experiment, we were required to display all digits of the value read from the potentiometer using 4 different seven-segment displays. We have followed the same mentality in part 2 of the experiment but as opposing to part 2 we have used the electric-switch effect of the transistors consecutively. At each step we have activated the corresponding seven-segment display thanks to our transistors.

Once again, we have combined our knowledge from electronics course and simple programming skills to conduct our experiment. We did not have a hard time.

5 CONCLUSION

Achievement of this week's experiment is to make us learn reading analog inputs and manipulating our Arduino circuitry. As a new concept this week, we have learned about potentiometers which are variable resistors with a third adjustable terminal and seven-segment displays. We learned how to control LED's by using potentiometer and we manipulated LED's color and brightness. At the same time, we have learned how to use BJT in the circuitry and control our seven-segment displays. It was a good exercise to refresh our electronic knowledge. This experience was very productive for us we learned new things that we can use in real life.

References

[1] Kadir Ozlem. *BLG 351E – Microcomputer Laboratory Slides*. 2020.

[2] Arduino. Arduino documentation. https://www.arduino.cc/reference/en/, 2018.

[3] Autodesk. Tinkercad. https://www.tinkercad.com/, 2011.