Interfacing laboratory

Report 1

Introduction to Arduino

Two wires interface I2C – multiplexed displays on arduino

Yara Al-Shafei

1120068

Teacher Assistant: Mohammed mudlal

Instructor: Dr Wasel Ghanem

Abstract

Nowadays, the world is trending toward using the microprocessors due to their many advantages like cost and size. Usually, sensors are connected to these microprocessors to get a system with complete functionality.

The aim of the experiments is to work with the Arduino microprocessor, two wires interface concept used in Arduino, LM75B temperature sensor and 7 segments display.

The report is going to give a brief introduction about Arduino, photocells, two wires interface, 7 segments and LM75B temperature sensor, the work has been done in the lab, the results obtained in pictures and a conclusion.

Table of contents

1. Introduction

1.1 Arduino

1.1.1 What is an Arduino?

Arduino is a board used to build many projects since it is programmable device. It is providing an open source software which helps you to develop your program with help of many others. Projects you can build with it starts with simple ones like lighting LEDs to building robots.

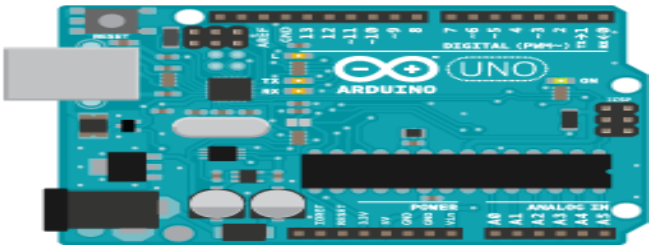

Figure 1.1.1 Arduino board

1.1.2 Advantages of using Arduino

Many advantages for arduino made the world use it, from it is low price, it is simplicity to it is compatibility to many OSs.

1.2 Photocells

1.2.1 What is a photocell?

The photocell is a variable resistor with light. When the light intensity on it is high the resistance will become low, and when the light intensity is low the resistance will be high. Due to this feature, it is used in many fields like in switching circuits.

Figure 1.2.1 Photocell sensor

The following graph shows the reverse relation between the resistance and light:

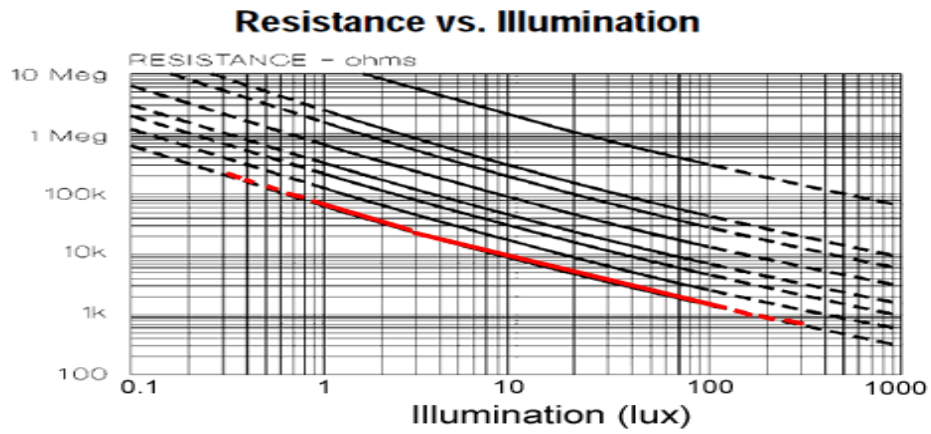

Figure 1.2.2 Photocell graph

1.3 Interrupts

Interrupts are signals sent to the CPU because an immediate event happened and needs an action, this action is in ISR. The interrupt is two types: software and hardware interrupts.

Software interrupt, happen because of instruction in the instruction set or something happened in the processor itself like dividing by zero.

But the hardware interrupts happen between devices in the PC, external devices like printers or both. This type is also called interrupt request.

1.4 Timer register

1.4.1 What is a timer?

Timer is a hardware device built inside the arduino for counting or timing. This timer is programmable by given it a prescaler. There is many types of timers like timer0 and timer 1 and they differ in their number of bits, uses and prescaler values. All these timers depend directly on the system clock of the arduino board.

There is an available manual for the timer registers bit for helping in configuring them.

1.5 Two wire interface – I2C bus

Two wire interface is a way for serial communication, it is used for bidirectional data transfer between devices. This is make it useful for many applications. It uses asynchronous mode. One of the communicating devices is master and the other is slave. It is name come from the two basic lines SDA and SCL. SCL is the clock line bus used for synchronization and is controlled by the master. SDA is known as the data transfer bus.

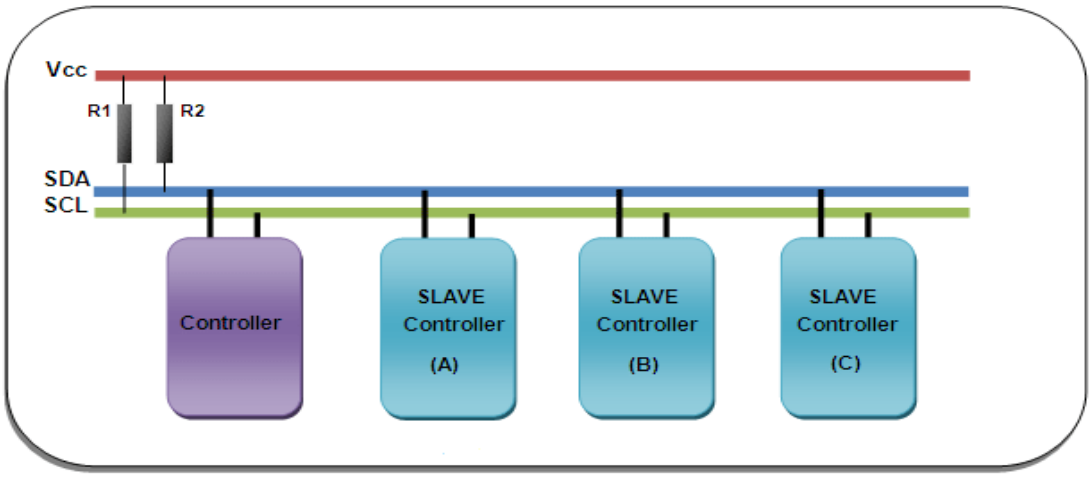

Figure 1.5.1 Block diagram of I2C

1.6 Multiplexed displays

Multiplexed displays are electronic display devices where the entire display is not driven at one time. It is main application when connecting multiple 7-segment displays for working like clocks or stop watch.

Many advantages for using this way, less wires, less cost and low power consumption.

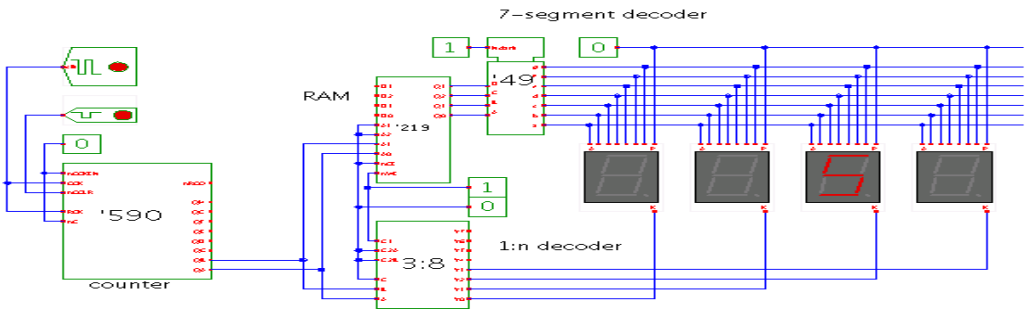

Figure 1.6.1 multiplexed displays

1.7 LM75B Temperature sensor

this sensor is digital sensor used for measuring the temperature. Usually it is used with I2C which we described about before. This combination makes them ideal to be used with many applications like systems the thermal management is critical for them.

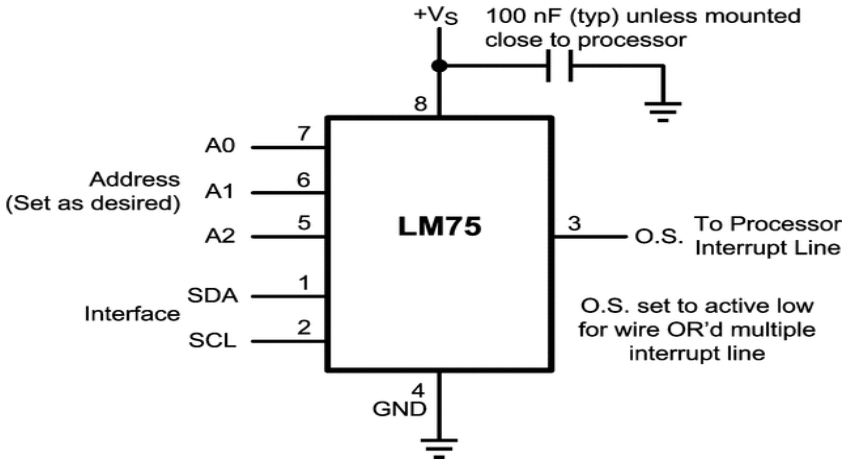

Figure 1.7.1 LM75 Pins

1.8 7-Segment display

Seven segment display used for displaying decimal numbers, it is used in many applications like digital clocks, timers, calculaters and stop watches.

The LEDs in the seven segments are referred to by letters A to G, and the decimal point is optional to display integer numbers.

Figure 1.8.1 7-segment

2. Procedure and Discussion

In this part, the steps followed to do the tasks asked in the lab will be shown.

2.1 Arduino with photocell

In this part, we want to print on the serial monitor in the arduino software the analog voltage readings we read from the photocell and see this amount of light on the led.

We connected the photocell and the led with the arduino, two resistors were connected with the photocell to limit the amount of current flows to it so it won't burn.

The connection was like the picture below:

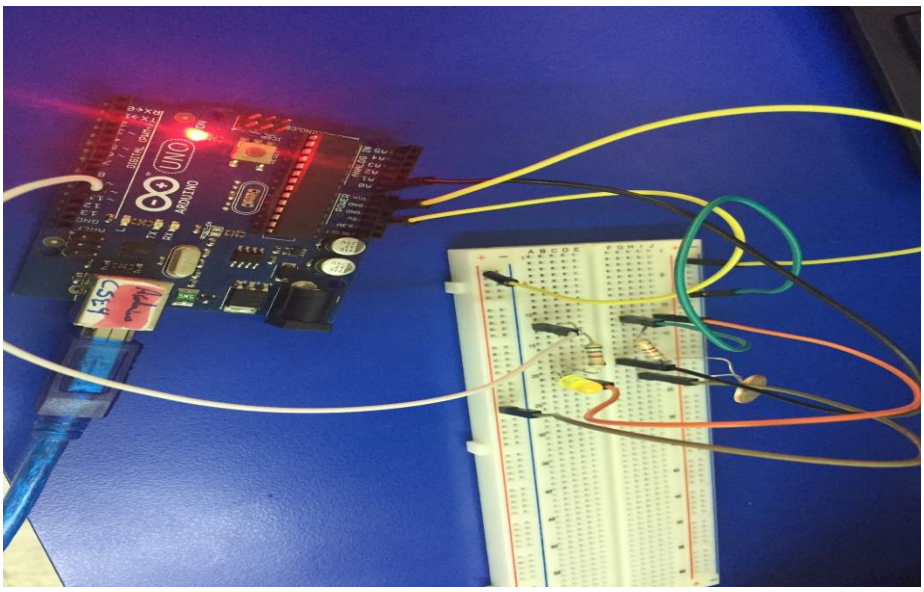

Figure 2.1.1 Demonstration of use

After connecting, we wrote the given code on the arduino software but we added a threshold were the photocell will give dark or light, we tried different values for the threshold starting from 100 until we reached the appropriate one which is 1000 for dark and 100 for light, as the screenshots show:

:

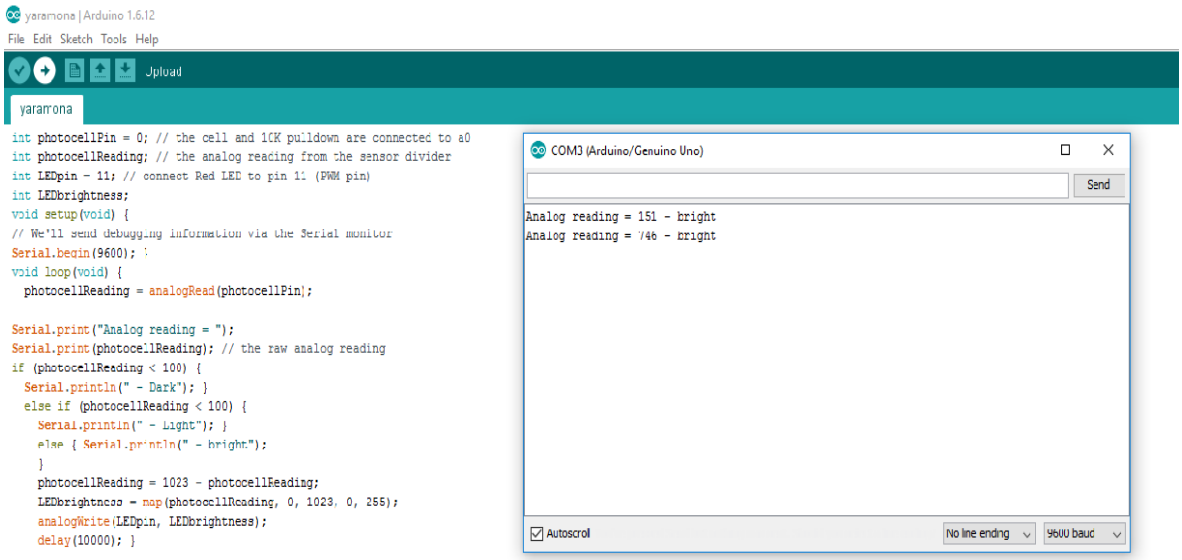

Figure 2.2.2 Photocell threshold 100

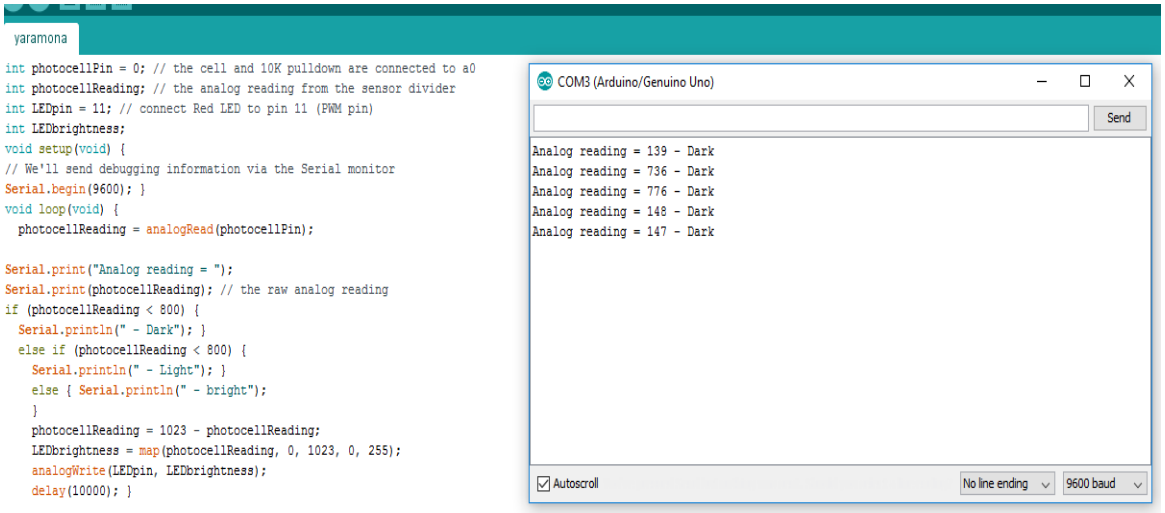

Figure 2.2.3 Photocell threshold 800

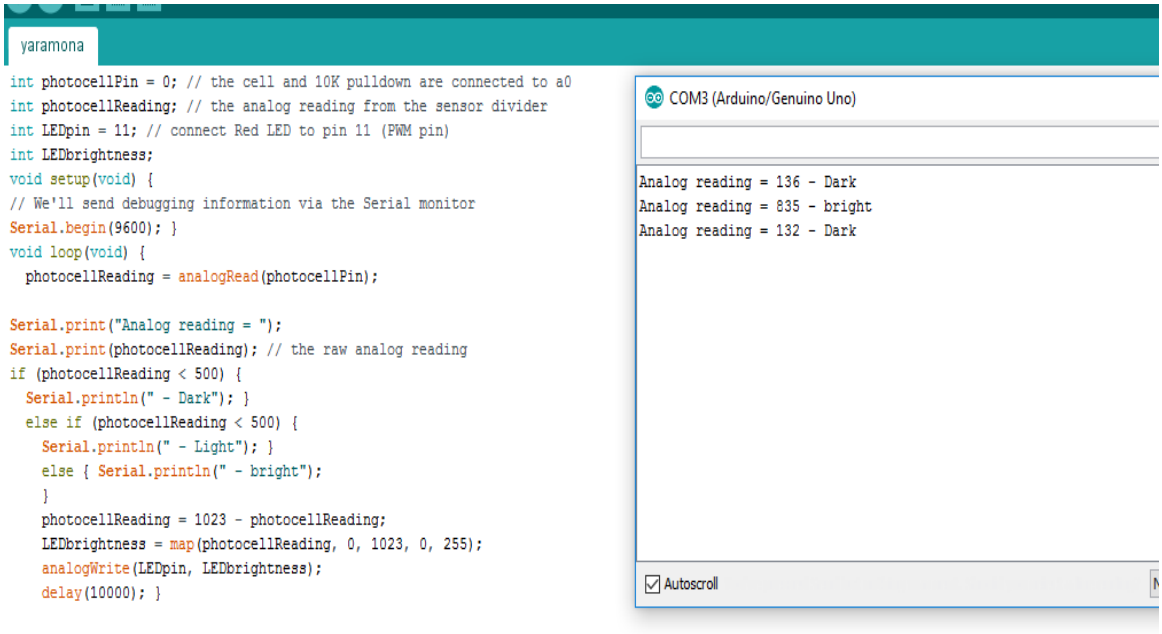

Figure 2.2.4 Photocell threshold 500

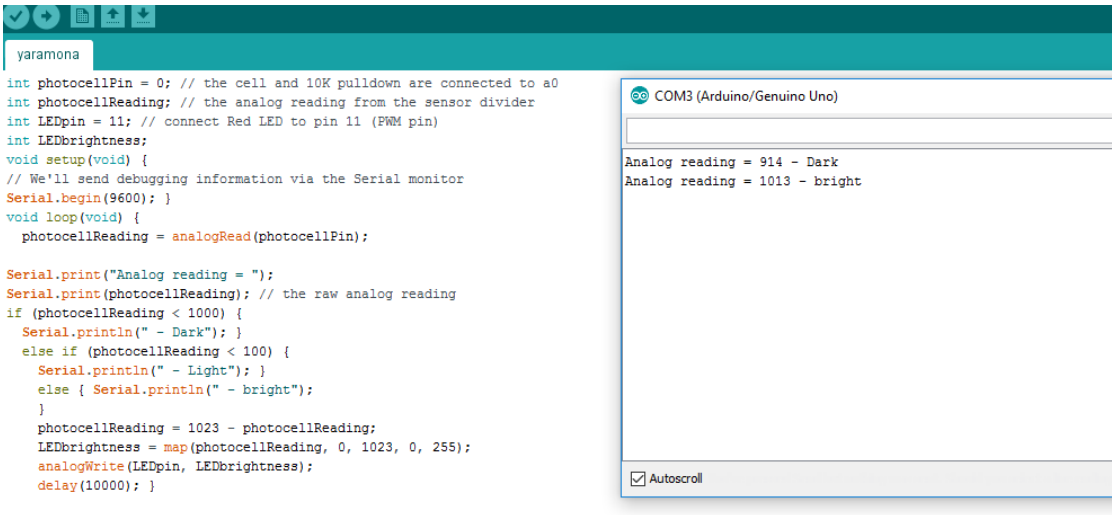

Figure 2.2.5 Photocell threshold 1000 and 100

2.2 Arduino with push button

In this part we saw the hardware interrupt by connecting arduino with a push button. When pushing on the button an interrupt should happen and so the led will turn on.

The connection was as in the following picture:

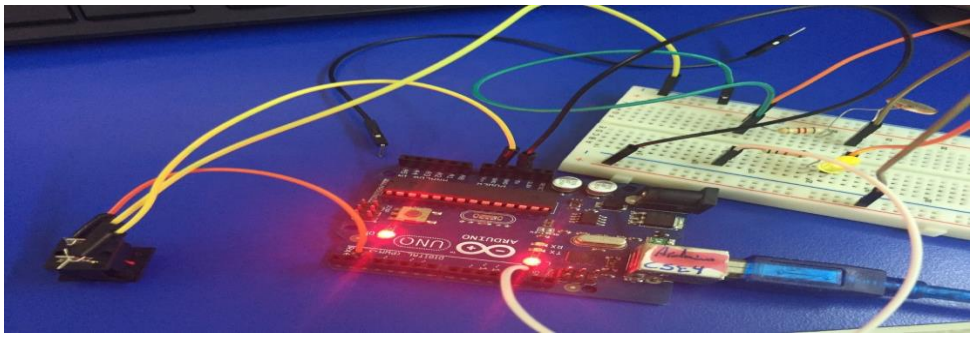

Figure 2.2.1 Arduino with push button

The code shown in the following screenshot was given in the experiment:

```
// Hardware interrupt example
int pin = 13;
volatile int state = LOW;
void setup() {
  pinMode(pin, OUTPUT);
  attachInterrupt(0, blink, CHANGE);
  }

  void loop() {
    digitalWrite(pin, state);
    }
void blink() {
  state = !state;
  }
```

Figure 2.2.2 Code

The following pictures show the led after an interrupt happen:

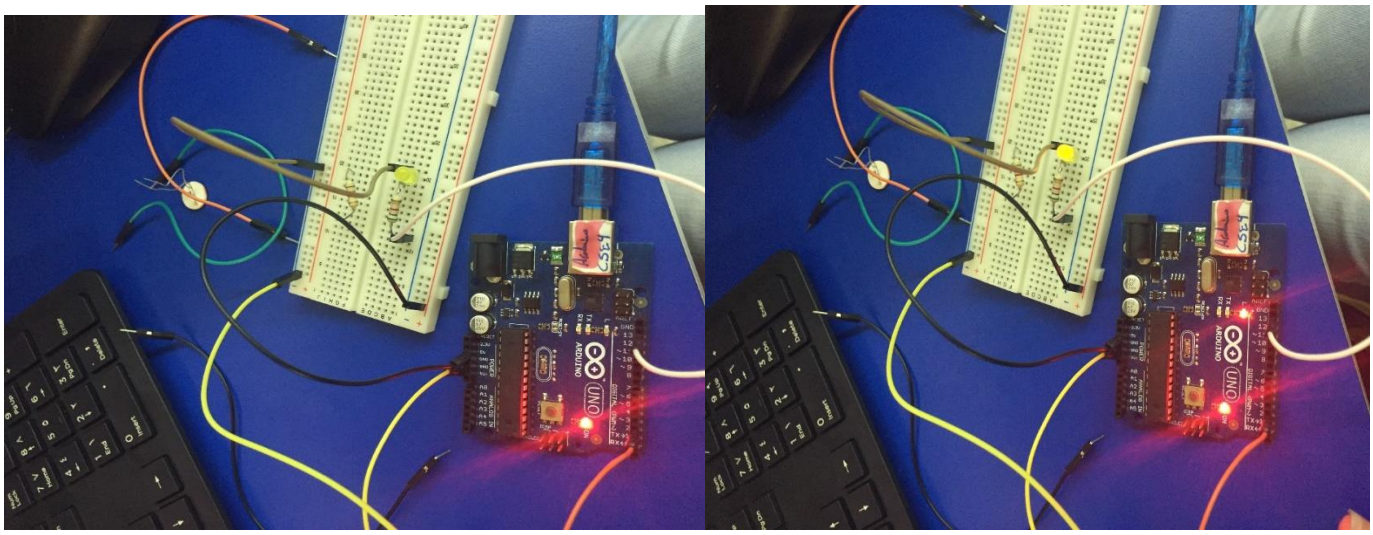

Figure 2.2.3 The led is off and on after an interrupt

2.3 TO DO (PWM)

In this part was asked to generate a square wave. The frequency was given equal to 1 KHz all what we should do is to calculate the prescaler if we suppose that the duty cycle is 0.5 then:

Timer value = 2^16 – (16 MHz* duty cycle / frequency * prescaler)

After finding the prescaler we connect the signal from the arduino to signal generator to ensure that we got the square wave.

2.4 LM75B

In this part we connected the temperature sensor with the arduino to measure the temperature in the rome in both Celsius and Fahrenheit and print them on the serial monitor in the arduino software.

We connected the sensor with the arduino as in the picture:

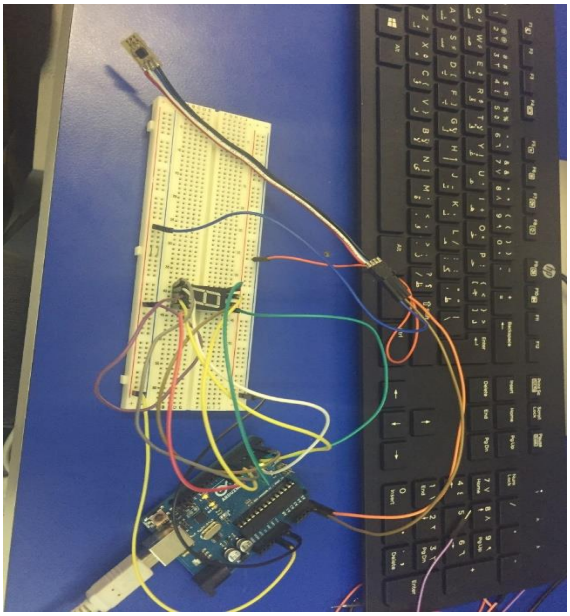

Figure 2.4.1 Arduino with temperature sensor

The code is shown in the following screenshot:

lm75b | Arduino 1.6.6

File Edit Sketch Tools Help

lm75b

```
//Simple code for the LM75B, simply prints temperature via serial
#include <Wire.h>
int LM75BAddress =0x48;
void setup(){
  Serial.begin(9600);
  Wire.begin(); //communication start

}
void loop(){
  float celsius = getTemperature();
  Serial.print("Celsius: ");
  Serial.println(celsius);
  float fahrenheit = (1.8 * celsius) + 32;
  Serial.print("Fahrenheit: ");
  Serial.println(fahrenheit);
  delay(200); //just here to slow down the output. You can remove this

}
float getTemperature(){
  Wire.requestFrom(LM75BAddress,2);
  byte MSB = Wire.read();
  byte LSB = Wire.read(); //it's a 12bit int, using two's compliment for negative
  int TemperatureSum = ((MSB << 8) | LSB) >> 4;
  float celsius = TemperatureSum*0.0625;
  return celsius;
}
```

Figure 2.4.2 Code

The results were as the screenshot shows:

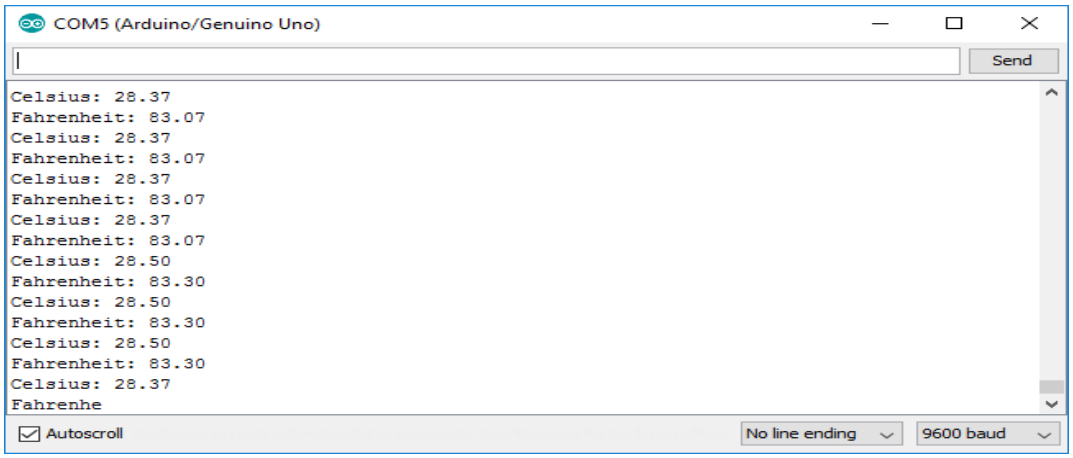

Figure 2.4.3 Results

2.5 7-segement Display

Here we will connect the 7-segenent display with arduino and show numbers on it. The connection will be as showed in the experiment and the given code in it will be used also.

The connection as in the picture:

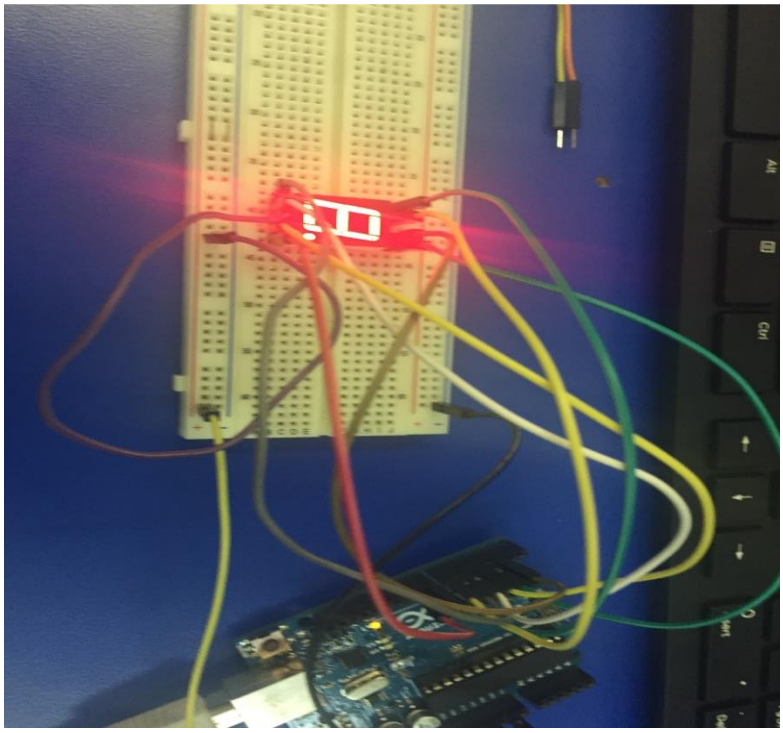

Figure 2.5.1 Arduino with 7-segment

The code used is:

Vseg | Arduino 1.6.6_

File Edit Sketch Tools Help

_7seg

```
byte seven_seg_digits[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 },
 { 0,0,0,0,1,1,0 },
 { 1,0,0,1,1,0,0 },
 { 0,1,0,0,1,0,0 },
 { 0,1,0,0,0,0,0 },
 { 0,0,0,1,1,1,1 },
 { 0,0,0,0,0,0,0 },
 { 0,0,0,1,1,0,0 }};
byte count=0;
void setup() {
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
pinMode(9, OUTPUT);
writeDot(0); // start with the "dot" off
}
void writeDot(byte dot) {
  digitalWrite(9, dot); }
  void sevenSegWrite(byte digit)


{ byte pin = 2;
for (byte segCount = 0; segCount < 7; ++segCount) {
  digitalWrite(pin, seven_seg_digits[digit][segCount]); ++pin; }
  }
  void loop() {
    for (byte count = 10; count > 0; --count) {
      delay(1000); sevenSegWrite(count - 1); }
delay(4000); }
```

Figure 2.5. Code

2.6 To Do

In this to do, we connected the temperature sensor with the arduino with 2 7-segments, the aim of this to do is to let the sensor sense the temperature and show it in the 7-segments.

First, we tried to show only one number of the temperature on one 7-segment, then, we tried to show the two numbers in the same 7-segement by showing them sequentially on the 7-segemnt like if the temperature is 27 we showed 2 then 7, as in the following picture:

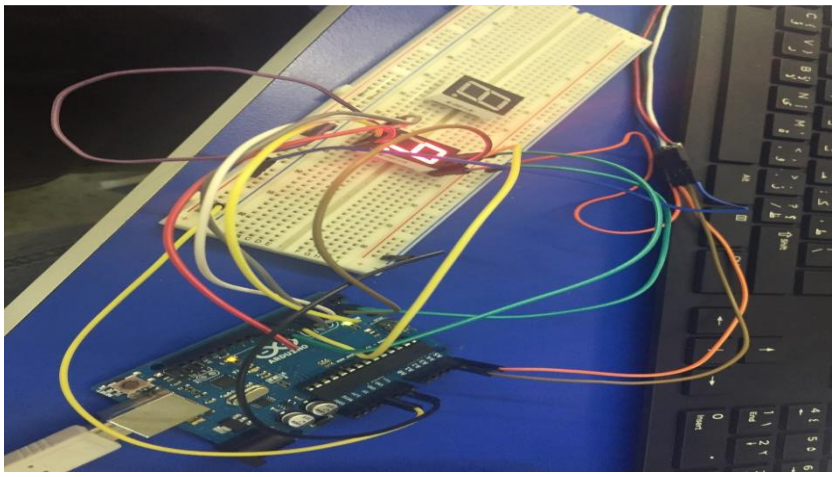

Figure 2.7.1 temperature on one 7-segement

finally, we showed the two numbers each on a different 7-segement as in the picture:

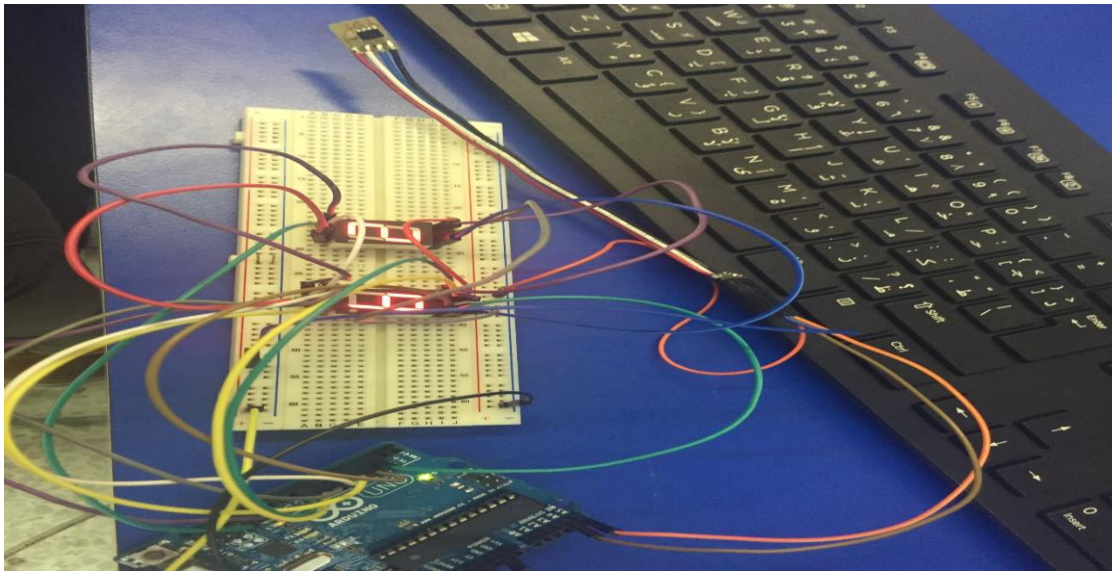

Figure 2.7.2 Temperature on different 7-segments

The code we wrote:

File Edit Sketch Tools Help

todo

```
//Simple code for the LM75B, simply prints temperature via serial
#include <Wire.h>
int LM75BAddress =0x48;
byte seven_seg_digits[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 },
 { 0,0,0,0,1,1,0 },
 { 1,0,0,1,1,0,0 },
 { 0,1,0,0,1,0,0 },
 { 0,1,0,0,0,0,0 },
 { 0,0,0,1,1,1,1 },
 { 0,0,0,0,0,0,0 },
 { 0,0,0,1,1,0,0 }};
void setup(){
  Serial.begin(9600);
  Wire.begin(); //communication start
   pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
pinMode(9, OUTPUT);
pinMode(12, OUTPUT);
pinMode(13, OUTPUT);
Wire.begin();
```

```
 }

   void sevenSegWrite(byte digit)
{ byte pin = 2;
for (byte segCount = 0; segCount < 7; ++segCount) {
  digitalWrite(pin, seven_seg_digits[digit][segCount]);
  ++pin; }
  }

void loop(){
  byte pin12 = 12,pin13 = 13;
  byte celsius = byte(getTemperature())/10;
  byte celsius1 = byte(getTemperature())%10;
 digitalWrite(pin12,0);
   digitalWrite(pin13,1);
   sevenSegWrite(celsius)   ;
  delay(10);
  digitalWrite(pin12,1);
  digitalWrite(pin13,0);
  sevenSegWrite(celsius1)   ;

  delay(10);
```

Figure 2.7.3 Code

3. Conclusion

We enjoyed These two experiments because they were full of useful and simple applications we can use in our lives. Many applications can be build using Arduino and the other components we used like the 7-segement and LM75B temperature sensor and in a short time. The results we got were acceptable and like what we expect.

4. References

[1] "Arduino - Introduction." Arduino - Introduction. Accessed October 09, 2016. https://www.arduino.cc/en/Guide/Introduction.

[2] "Photoresistor." Wikipedia. Accessed October 09, 2016. https://en.wikipedia.org/wiki/Photoresistor.

[3] I2C Interface or TWI (Two Wire Interface). (n.d.). Retrieved October 14, 2016, from http://www.engineersgarage.com/tutorials/twi-i2c-interface

[4] Multiplexed display. (n.d.). Retrieved October 14, 2016, from https://en.wikipedia.org/wiki/Multiplexed_display

[5] LM75B (ACTIVE). (n.d.). Retrieved October 14, 2016, from http://www.ti.com/product/lm75b/description

[6] (n.d.). Retrieved October 14, 2016, from https://en.wikipedia.org/wiki/Seven-segment_display#Implementations

[7] By default in the Arduino firmware interrupts are enabled. Interrupt masks are enabled / disabled by setting / clearing bits in the Interrupt mask register (TIMSKx). (n.d.). Timers-Arduino. Retrieved October 17, 2016, from https://arduino-info.wikispaces.com/Timers-Arduino