Experiment No. : 04

Experiment Name:

Making a LCD thermometer that shows temperature in Celsius and Fahrenheit using microcontroller and adding a push button which will change temperature between Celsius and Fahrenheit.

Objectives:

In this lab we will learn how we can design a thermometer using microcontroller which shows temperature in Celsius and Fahrenheit. The objective of this experiment includes:

- Using LM35 IC and integrate this with the arduino to measure temperature of the surrounding environment in degree celsius and in fahrenheit.
- Using a LCD to show temperature in Celsius and Fahrenheit in it's screen and a push button to change temperature between Celsius and Fahrenheit.

Theory:

Microcontroller:

A microcontroller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals. Microcontrollers are "embedded" inside some other device to control the features or actions of the device. Another name for a microcontroller, therefore, is "embedded controller." Microcontrollers are mostly designed for embedded applications and are heavily used in automatically controlled electronic devices such as cellphones, cameras, microwave ovens, washing machines, etc.

In our experiment we have used Arduino Uno R3. It is a microcontroller board based on a removable, dual-inline-package (DIP) ATmega328 AVR microcontroller. It has 20 digital input/output pins (of which 6 can be used as PWM outputs and 6 can be used as analog inputs).

As a temperature sensor, the circuit will read the temperature of the surrounding environment and relay the temperature to us back in degrees celsius.
The IC we will use to measure the temperature is the LM35 IC. We will integrate this with the arduino to measure the temperature. The arduino will then read this measured value from the LM35 and translate into degrees fahrenheit and celsius, which we will be able to read from the computer from the arduino serial monitor.

LM35 IC:

The LM35 is a low voltage IC which uses approximately +5VDC of power. This is ideal because the arduino's power pin gives out 5V of power. The IC has just 3 pins, 2 for the power supply and one for the analog output.

The output pin provides an analog voltage output that is linearly proportional to the celsius (centigrade) temperature. Pin 2 gives an output of 1 millivolt per 0.1°C (10mV per degree). So to get the degree value in celsius, all that must be done is to take the voltage output and divide it by 10- this give out the value degrees in celsius.

So, for example, if the output pin, pin 2, gives out a value of 315mV (0.315V), this is equivalent to a temperature of 31.5°C.

We can then easily convert this celsius value into fahrenheit by plugging in the appropriate conversion equation.

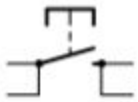
Temperature in Fahrenheit = 32 + (9/5) * Temperature in Celsius

Pin 1 receives positive DC voltage in order for the IC to work. This, again, is voltage approximately 5 volts. Pin 3 is the ground, so it receives the ground or negative terminal of the DC power supply. And Pin 2 is the output of the IC, outputting an analog voltage in porportion to the temperature it measures.

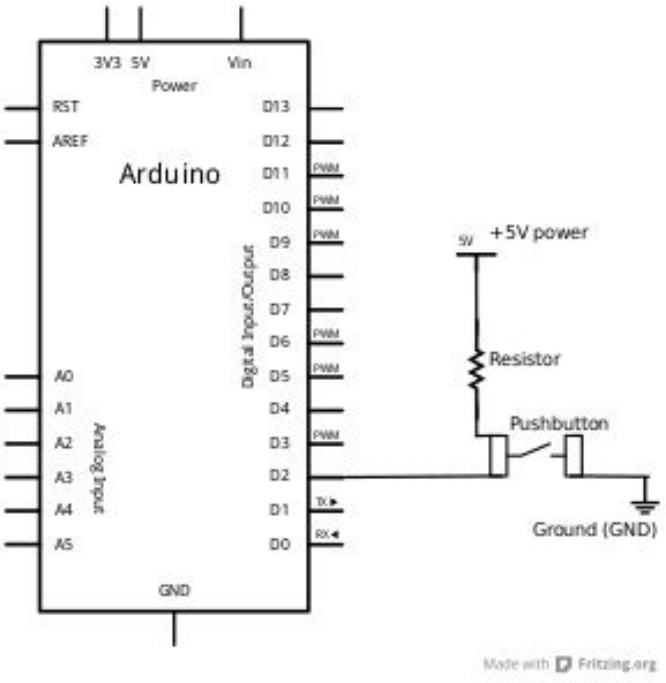
The arduino, with suitable code, can then interpret this measured analog voltage and output to us the temperature in degrees celsius and fahrenheit.

Push Button:

The pushbutton is a component that connects two points in a circuit when you press it. The example turns on an LED when you press the button. Pushing a button causes wires under the button to be connected, allowing current to flow. (called closed) When the button isn't pressed, no current can flow because the wires aren't touching (called open)



We connect three wires to the Arduino board. The first goes from one leg of the pushbutton through a pull-up resistor (here 2.2 KOhms) to the 5 volt supply. The second goes from the corresponding leg of the pushbutton to ground. The third connects to a digital i/o pin (here pin 7) which reads the button's state.



When the pushbutton is open (unpressed) there is no connection between the two legs of the pushbutton, so the pin is connected to 5 volts (through the pull-up resistor) and we read a HIGH. When the button is closed (pressed), it makes a connection between its two legs, connecting the pin to ground, so that we read a LOW. (The pin is still connected to 5 volts, but the resistor in-between them means that the pin is "closer" to ground.)For electricity to flow, there has to be a complete circuit from the power to the ground. If a microcontroller pin is not connected to

anything, it is said to be “floating” and you can’t know ahead of time what the value will be when you read it. It can also change between times that it is read. When we add a pull-up resistor, we get a circuit like the following:When a push button is pushed down, the circuit is complete and ground is connected to pin 2. (The +5V goes through the closed switch to ground as well.) When it is not pushed down the circuit is from the +5V through the resistor and the microcontroller sees the +5V. (HIGH)

Code:
In the setup function we need to set pin of the push button to INPUT using the pinMode(pinNo, INPUT) function. Then we can use digitalRead(pinNo) function to check whether the push button is pressed or not. If the button is pressed then digitalRead(pinNo) function will return HIGH otherwise it will return LOW.

LCD Display:

LCD stands for Liquid Crystal Display. We will refer to it as either an LCD or simply a display.

In this experiment we used a LCD display to show when the LED brightness is intensifying or dimming as we press the push button.

The LCDs have a parallel interface, meaning that the microcontroller has to manipulate several interface pins at once to control the display. The interface consists of the following pins:

A register select (RS) pin that controls where in the LCD's memory you're writing data to. You can select either the data register, which holds what goes on the screen, or an instruction register, which is where the LCD's controller looks for instructions on what to do next.

A Read/Write (R/W) pin that selects reading mode or writing mode

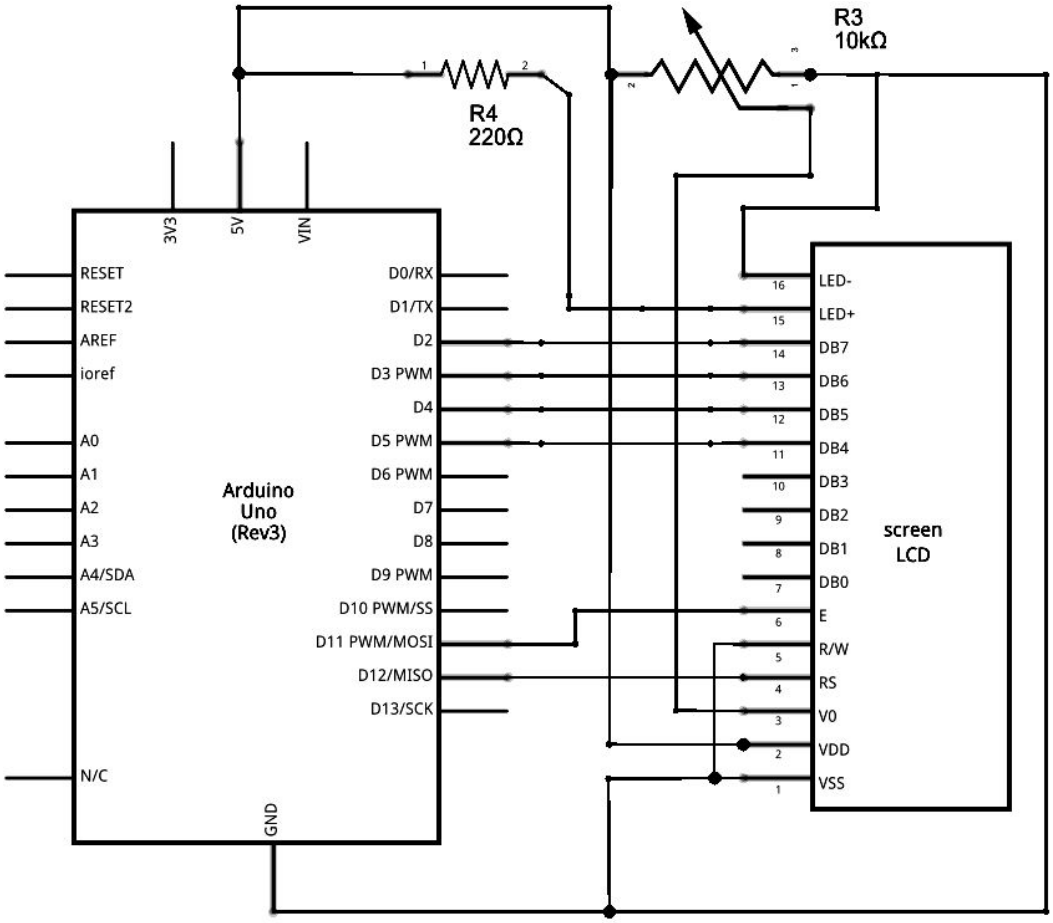
An Enable pin that enables writing to the registers

8 data pins (D0 -D7). The states of these pins (high or low) are the bits that you're writing to a register when you write, or the values you're reading when you read.

There's also a display constrast pin (Vo), power supply pins (+5V and Gnd) and LED Backlight (Bklt+ and BKlt-) pins that you can use to power the LCD, control the display contrast, and turn on and off the LED backlight, respectively.

The process of controlling the display involves putting the data that form the image of what you want to display into the data registers, then putting instructions in the instruction register. The LiquidCrystal Library simplifies this for you so you don't need to know the low-level instructions.

Circuit:



To wire your LCD screen to your board, connect the following pins:

LCD RS pin to digital pin 12
LCD Enable pin to digital pin 11
LCD D4 pin to digital pin 5
LCD D5 pin to digital pin 4
LCD D6 pin to digital pin 3
LCD D7 pin to digital pin 2

Additionally, wire a 10k pot to +5V and GND, with it's wiper (output) to LCD screens VO pin (pin3). A 220 ohm resistor is used to power the backlight of the display, usually on pin 15 and 16 of the LCD connector

Equipment:

1.A computer (Windows, Mac, or Linux).
2.An Arduino compatible microcontroller.
2. Arduino Software
3. A breadboard
4. A USB A-to-B cable
5. Male/Male Jumper Wires
6. A LM35 IC
7. Resistors(220 Ω)
8. Push Button
9. LCD Display
10. Potentiometer

Experiment:

Circuit Diagram:

Program Source Code:

Making a thermometer that shows temperature in celsius and fahrenheit using a microcontroller and adding a push button which helps to change temperature between celsius and fahrenheit.

```
#include<LiquidCrystal.h>

float temp;
const int buttonpin = 8;
bool flag;
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);



void setup() {
  flag = true;
  pinMode(buttonpin, INPUT);
//  Serial.begin(9600);
 lcd.begin(16,2);
}

void loop() {

  if (digitalRead(buttonpin) == HIGH) {
    flag = !flag;
    delay(300);
  }
  temp = analogRead(A0);
  temp = temp * 0.48828125;
  if(flag)
  {
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Temp: ");
    lcd.print(temp,2);
    lcd.print("*C");
  }
```

```
  else
  {
    temp = 5.0*temp;
    temp /= 9.0;
    temp += 32.0;

    lcd.clear();

    lcd.setCursor(0,0);
    lcd.print("Temp: ");
    lcd.print(temp,2);
    lcd.print("*F");
  }
  delay (500);
}
```

Procedure:

Part1: Designing the circuit:

We designed the circuit by connecting the LM35 temperature IC, push button, LCD display, potentiometer and resistors as defined above. We used the LCD display to show the temperature calculated from the output analog voltage of the LM35 IC, proportional to the temperature that it measures. We also used a push button to change temperature between celsius and fahrenheit.

Part 2: Connecting Arduino to the computer

We launched the Arduino IDE from the computer then created a new sketch. Then, we connected the Arduino UNO board with the computer through a USB A-to-B cable.

Part 3: Compiling and Uploading the sketch to the arduino

Now we need to write the source code for the experiment.

We included the LiquidCrystal.h library so that we can use the LCD display. Then we set up the pins of the LCD display as follows

RS to pin 12
EN to pin 11
D4 to pin 5
D5 to pin 4
D6 to pin 3
D7 to pin 2

We set pin 8 for the push button, then we setup the pins in the setup() function. We also declared a boolean type variable “flag” to decide whether the temperature should be shown in celsius(flag = true) or in fahrenheit(flag = false).

When we push the push button then the value of the variable named “flag” is inverted and consequently temperature is changed between celsius and fahrenheit. We also used delay so that we can observe the temperature change properly.

Conclusion:

From this experiment, we learnt about LM35 IC which provides analog output voltage that is proportional to the measured temperature of the surroundings. We also learnt how we can output digital data as text in a LCD display and used it to display the temperature. Every time the push button is pressed the unit of measured temperature of the thermometer is changed between celsius and fahrenheit. But if we press the push button for a long time delay then the temperature may fluctuate between it’s celsius and fahrenheit value. Here, we used delay to solve this problem and the temperature doesn’t change within 300 millisecond once it is changed. Finally, we were successful designing the thermometer using the microcontroller, show the temperature on the display of the LCD monitor and also change the temperature between celsius and fahrenheit by pushing the push button.