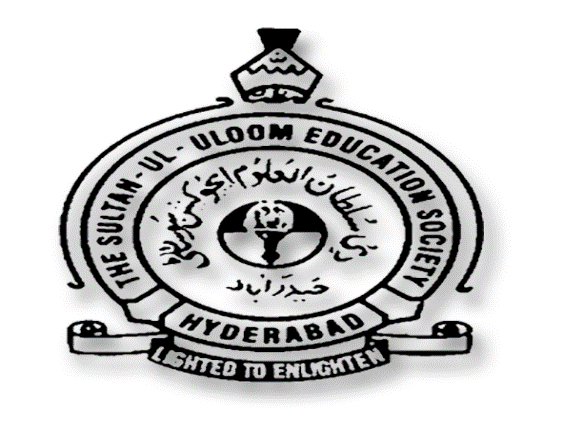
**Arduino Based Temperature Controlled Fan**



Working: -

In this Arduino based project, we are going to build a **temperature-controlled fan using Arduino.**With this circuit, we will be able to adjust the fan speed in our home or office according to the room temperature and also show the temperature and fan speed changes on an LCD display. To do this we will be using an Arduino UNO Board, LCD, DHT11 sensor Module, and DC fan that is controlled by using PWM.

This project consists of three sections. One senses the temperature by using humidity and temperature sensor namely **DHT11**.

The second section reads the dht11 sensor module’s output and extractstemperature value into a suitable number in Celsius scale and control the fan speed by using **PWM**. PWM is Pulse width modulation reduces the average power delivered by an electrical signal by converting the signal into discrete parts. In the PWM technique, the signal’s energy is distributed through a series of pulses rather than a continuously varying (analog) signal. Output Voltage of PWM signal. We are creating PWM at pwm pin of Arduino and applied it at base terminal of the transistor. Then transistor creates a voltage according to the PWM input. The output voltage of the PWM signal will be the percentage of the duty cycle. For example, for a 100% duty cycle, if the operating voltage is 5 V, then the output voltage will also be 5 V. If the duty cycle is 50%, then the output voltage will be 2.5 V. We have created PWM at pwm pin of Arduino and applied it at base terminal of the transistor. Then transistor creates a voltage according to the PWM input.

Temperature sensor LM35 senses the temperature and converts it into an electrical (analog) signal, which is applied to the ATmega328 microcontroller of the Arduino UNO Board. The analog value is converted into a digital value. Thus, the sensed values of the temperature and speed of the fan are displayed on the LCD. When the temperature exceeds 30°C the fan starts rotating.

A low-frequency pulse-width modulation (PWM) signal, whose duty cycle is varied to adjust the fan’s speed is used. An inexpensive, single, small pass transistor-like 2N222 or BD139 can be used here. It is efficient because the pass transistor is used as a switch.

A picture containing text, electronics

Description automatically generated

A picture containing electronics

Description automatically generated

A picture containing text

Description automatically generated

LM35 sensor: -

The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 device does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55°C to 150°C temperature range.

PWM: -

Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between the full Vcc of the board (e.g., 5 V on Uno, 3.3 V on a MKR board) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off. The duration of "on time" is called the pulse width. To get varying analog values, you change, or modulate, that pulse width. If you repeat this on-off pattern fast enough with an LED for example, the result is as if the signal is a steady voltage between 0 and Vcc controlling the brightness of the LED.

In the graphic below, the green lines represent a regular time period. This duration or period is the inverse of the PWM frequency. In other words, with Arduino's PWM frequency at about 500Hz, the green lines would measure 2 milliseconds each. A call to [analog Write](https://www.arduino.cc/en/Reference/AnalogWrite)() is on a scale of 0 - 255, such that analog Write(255) requests a 100% duty cycle (always on), and analog Write(127) is a 50% duty cycle (on half the time) for example.

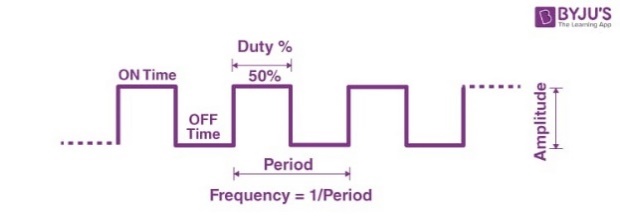
Diagram

Description automatically generated

**Formula for duty cycle given below:**

Duty Cycle= Ton/T Where T= total time or Ton + Toff

and Ton= On time of pulse (means 1)

And Toff= Off time of pulse (means 0)

And last part of system shows humidity and temperature on LCD and Fan driver. Here we are using this DHT sensor for sensing temperature, and then programmed our Arduino according to our requirements.

Code: -

#include <LiquidCrystal.h>

LiquidCrystal lcd(2,3,4,5,6,7);

int tempPin = A0; // the output pin of LM35

int fan = 11; // the pin where fan is

int led = 8; // led pin

int temp;

int tempMin = 30; // the temperature to start the fan 0%

int tempMax = 60; // the maximum temperature when fan is at 100%

int fanSpeed;

int fanLCD;

void setup() {

pinMode(fan, OUTPUT);

pinMode(led, OUTPUT);

pinMode(tempPin, INPUT);

lcd.begin(16,2);

Serial.begin(9600);

}

void loop()

{

temp = readTemp(); // get the temperature

Serial.print( temp );

if(temp < tempMin) // if temp is lower than minimum temp

{

fanSpeed = 0; // fan is not spinning

analogWrite(fan, fanSpeed);

fanLCD=0;

digitalWrite(fan, LOW);

}

if((temp >= tempMin) && (temp <= tempMax)) // if temperature is higher than minimum temp

{

fanSpeed = temp;//map(temp, tempMin, tempMax, 0, 100); // the actual speed of fan//map(temp, tempMin, tempMax, 32, 255);

fanSpeed=1.5\*fanSpeed;

fanLCD = map(temp, tempMin, tempMax, 0, 100); // speed of fan to display on LCD100

analogWrite(fan, fanSpeed); // spin the fan at the fanSpeed speed

}

if(temp > tempMax) // if temp is higher than tempMax

{

digitalWrite(led, HIGH); // turn on led

}

else // else turn of led

{

digitalWrite(led, LOW);

}

lcd.print("TEMP: ");

lcd.print(temp); // display the temperature

lcd.print("C ");

lcd.setCursor(0,1); // move cursor to next line

lcd.print("FANS: ");

lcd.print(fanLCD); // display the fan speed

lcd.print("%");

delay(200);

lcd.clear();

}

int readTemp() { // get the temperature and convert it to celsius

temp = analogRead(tempPin);

return temp \* 0.48828125;

}

**Circuit Diagram: -**

Diagram, schematic

Description automatically generated

\Diagram

Description automatically generated