EFY TESTED ★ EFY TESTED S.C. DWIVEDI ★★★ EFY TESTED

TEMPERATURE BASED FAN SPEED CONTROL And MONITORING Using Arduino

BISWAJIT DAS

This project is a standalone automatic fan speed controller that controls the speed of an electric fan according to the requirement. Use of embedded technology makes this closed-loop feedback-control system efficient and reliable. The microcontroller (MCU) ATMega8/168/328 allows dynamic and faster control and the LCD makes the system user-friendly. Sensed temperature and fan speed levels are simultaneously displayed on the LCD panel.

The project is very compact and uses a few components only. It can be implemented for several applications including air-conditioners, water-heaters, snow-melters, ovens, heat-exchangers, mixers, furnaces, incubators, thermal baths and veterinary operating tables. The project will help save energy/electricity.

Circuit and working

Circuit diagram of the temperature fan speed control and monitoring is shown in Fig. 1. It is built around Arduino Uno board (Board1), 16x2 LCD (LCD1), temperature sensor LM35(IC1) and a few other components.

Arduino is at the heart of the circuit as it controls all functions. LM35 is a precision integrated circuit whose output voltage is linearly proportional to Celsius (Centigrade) temperature. It is rated to operate over a -55°C to 150°C temperature range. It has +10.0mV/Celsius linear-scale factor.

PARTS LIST

Semiconductors:	
Board1	- Arduino Uno
LCD1	- 16x2 LCD
IC1	- LM35 temperature sensor
T1	- BD139 npn transistor
D1	- 1N4007 rectifier diode
LED1	- 5mm LED
Resistors (all 1/4-watt, ±5% carbon):	
R1, R2	- 1-kilo-ohm
R3	- 470-ohm
VR1	- 10-kilo-ohm preset
Capacitor:	
C1	- 10µF, 16V electrolytic
Miscellaneous:	
CON1	- 2-pin terminal connector
CON2	- 3-pin connector
CON3	- 8-pin connector
M1	- 12V DC operated fan
	- 12V battery for fan

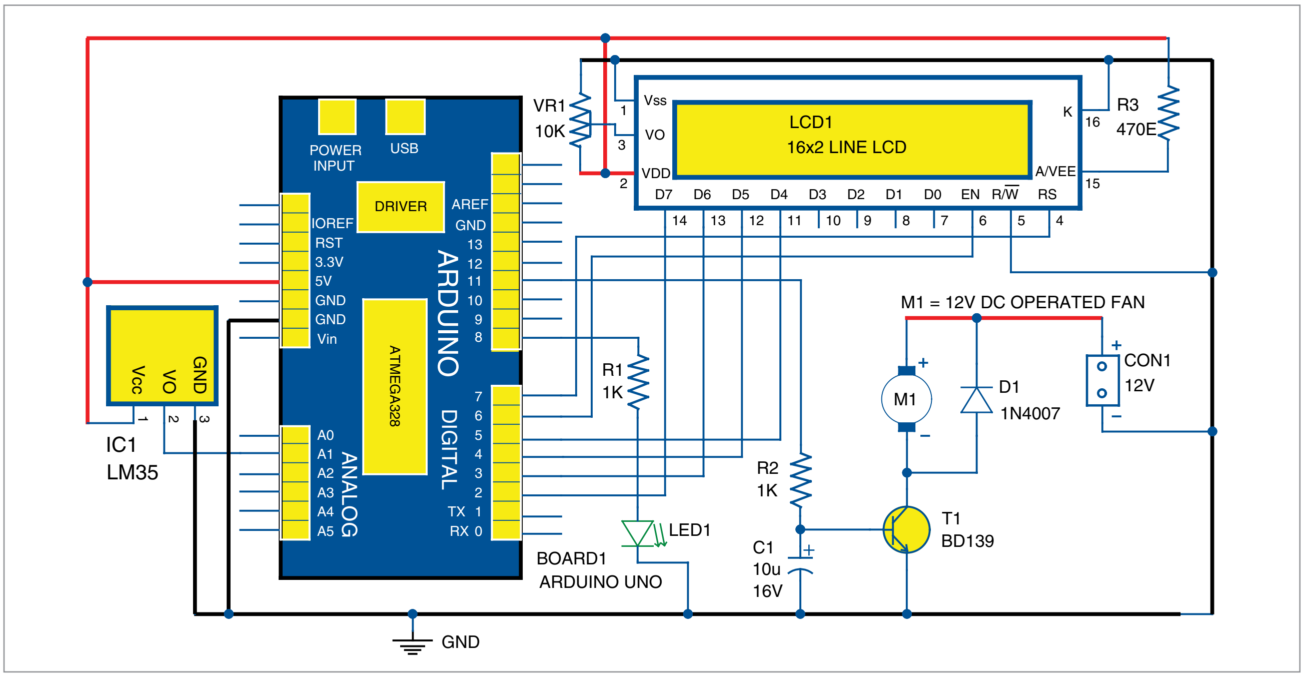
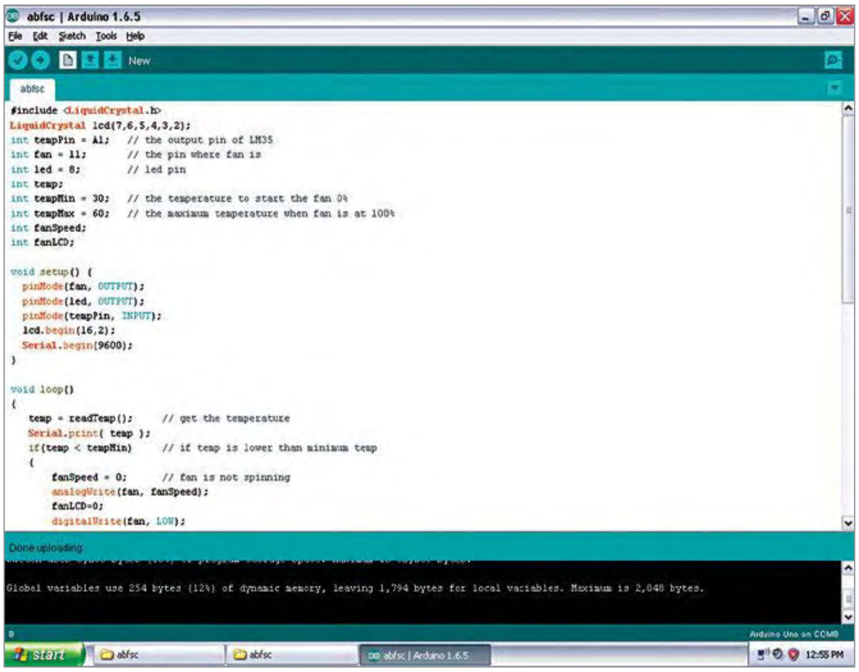


Fig. 1: Circuit diagram of the temperature based fan speed control and monitoring using Arduino

EFY Note

The source code of this project is included in this month's EFY DVD and is also available for free download at *source.efymag.com*



```
#include <LiquidCrystal.h>
LiquidCrystal lcd(7,6,5,4,3,2);
int tempPin = A1;   // 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);
```

```
Done uploading

Global variables use 254 bytes (12%) of dynamic memory, leaving 1,794 bytes for local variables. Maximum is 2,048 bytes.
```

Fig. 2: Screenshot of the source code on Arduino IDE

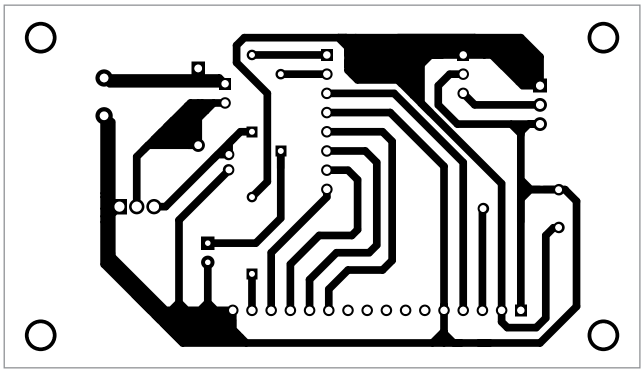


Fig. 3: Actual-size PCB pattern of the temperature based fan speed control and monitoring circuit using Arduino

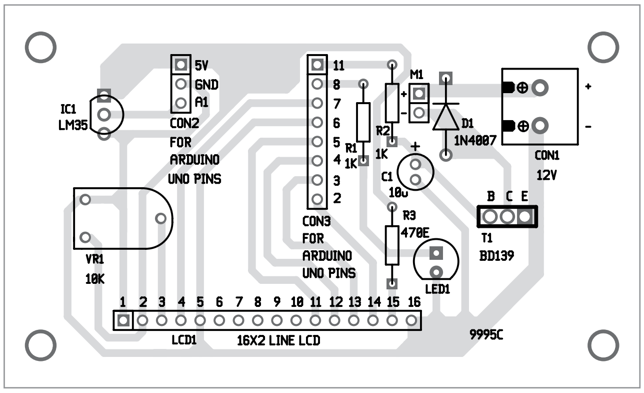


Fig. 4: Component layout of the PCB

Temperature sensor LM35 senses the temperature and converts it into an electrical (analogue) signal, which is applied to the MCU through an analogue-to-digital converter (ADC). The analogue signal is converted into digital format by the ADC. Sensed values of the temperature and speed of the fan are displayed on the LCD. The MCU on Arduino drives the motor driver to control fan speed.

Fan speed control technique. A low-frequency pulse-width modulation (PWM) signal, usually in the range of about 30Hz, whose duty cycle is varied to adjust the fan's speed is used. An inexpensive, single, small pass transistor can be used here. It is efficient because the pass transistor is used as a switch.

One disadvantage of this approach, however, is that it can make the fan noisy because of the pulsed nature of the signal. The PWM waveform's sharp edges cause the fan's mechanical structure to move (like a badly-designed loudspeaker), which can easily be audible.

Construction and testing

An actual-size, single-side PCB for the temperature based fan speed control and monitoring circuit is shown in Fig. 3 and its component layout in Fig. 4. Assemble the circuit on the PCB. CON2 and CON3 are used to connect Board1 (Arduino UNO board) through external connectors. A 12V battery is used to drive the 12V DC-operated fan.

Software

Software for the automatic temperature controller and monitor circuit is written in Arduino programming language. Arduino Uno is programmed using Arduino IDE software. ATmega328P on Arduino Uno comes with a pre-programmed bootloader that allows users to upload a new code to it without using an external hardware programmer.

Connect Arduino board to the PC and select the correct COM port in Arduino IDE. Compile the program (sketch). Then select the correct board from Tools→Board menu in Arduino IDE and upload the sketch (abfc.ino) to Arduino through standard USB port. EFY

Biswajit Das was manager - R&D, EFY Labs till recently