Temperature Controlled Fan

by

Rumesh

Updated on

1 June, 2022

Closed loop control system

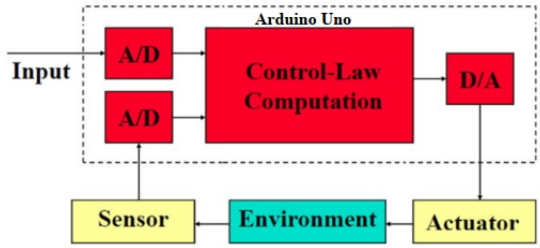

Figure 1: Simple Real-time Control system

To demonstrate closed loop feedback control system, temperature controlled fan system implemented. Based on the temperature in the environment fan's RPM (Revolutions Per Minute) change. In this Simple Real-Time Control System, measuring quantity is temperature, where DS18B20 used.

Temperature Controlled FAN

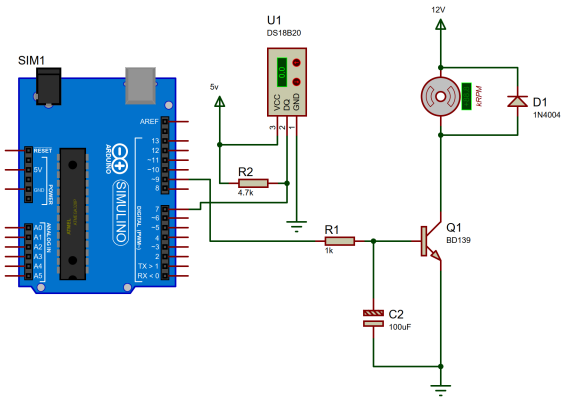

Figure 2: Temperature Controlled DC Fan

Module	Part No.
Temperature Sensor	DS18B20
Transistor	BD139
Resistors	1k, 5k
Capacitor	100µF
Diode	1N4007
Arduino	Arduino UNO R3
DC FAN	BDM8025S

Table 1: Components List

1 **Temperature Sensor DS18B20** :
The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements (0.5, 0.25, 0.125, 0.0625) and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply. Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems. Figure 2 shows the pin layout of the sensor. The 1-Wire bus requires an external pullup resistor of approximately 5kΩ, thus, the idle state for the 1-Wire bus is high.

Figure 3: Temperature Sensor

2 **Transistor & Diode** : Transistor act as a switch here. 1N4007 Diode connected parallel with Fan. DC Fan motor have armature inductance, It store energy, to recycle this energy and protect driver circuit diode used.

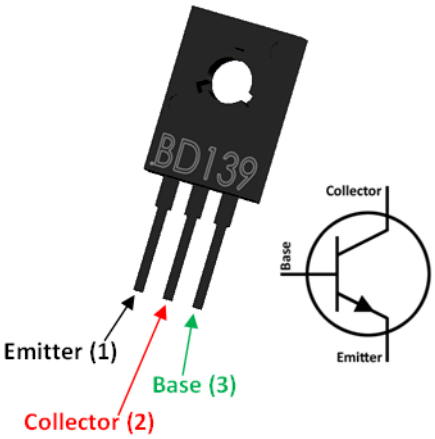

Figure 4: BD139

3 **Arduino Uno** : Arduino uno is equipped with the well-known ATmega328P (ATMega 16U2) Processor, 14 digital input/output pins, 6 analog inputs, USB connections, ICSP header and reset button. It is a 8-bit AVR RISC-based microcontroller. Power input is 2.7-5.5 volts, can be powered up by laptop USB or external power adapter. This board have a built-in bootloader which allows flashing the board via USB.

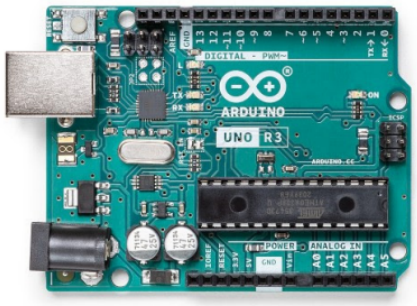

Figure 5: Arduino Uno

For this implementation only two digital pins (one for sensor data, another one for PWM) used.

Implementation

By programming ATMega328P Processor using Arduino IDE system going to work. Inorder to integrate sensor with Arduino OneWire & Dallas Temperature libraries should be installed. At first environment temperature detected by DS18B20 is 32°C [DS18B20.ino]. So fixed the reference temperature as 20°C and temperature max set as 100°C (But sensor can be used to measure temperatures from -55°C to +125°C). The circuit shown in Figure 5 Implemented. Code[A01.ino] is uploaded to the arduino. FAN speed (0 - 100%) going to map to (20°C - 100°C). So the Fan will rotate at first, if environment temperature is 20°C or below FAN will not rotate. Also we can set reference temperature as 32°C. so Fan will not rotate at first. it will start rotating when the temperature increases. Here PWM used, where reading from sensor compared with the reference and output will change based on that. Baud rate set to 9600 in code so that received data can be observed Arduio Serial monitor in this configuration.

***Relevant codes are included in Appendix section.**

Output on Serial Monitor

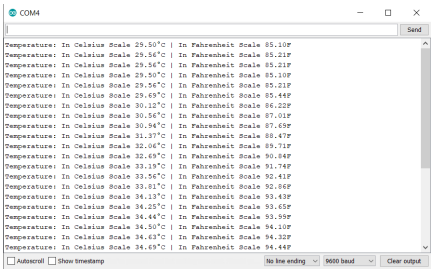

Figure 6: Serial Monitor Output - Reading

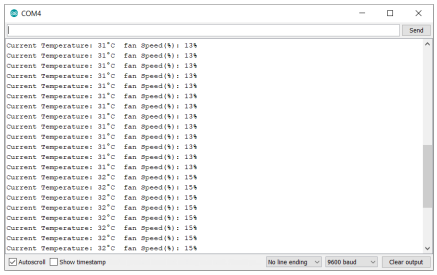

Figure 7: Serial Monitor Output - Fan speed

Appendix

1. Arduino Code - DS18B20.ino

```
#include <OneWire.h>
#include <DallasTemperature.h>

#define ONE_WIRE_BUS 7 // Sensor Data Pin connected to Digital Pin 7 in Arduino
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

float Celsius = 0;
float Fahrenheit = 0;

void setup() {
  Serial.begin(9600); //Baud rate for Serial Monitor
  sensors.begin();
}


void loop() {
  sensors.requestTemperatures();
  Celsius = sensors.getTempCByIndex(0);       // Using only one sensor
  Fahrenheit = sensors.toFahrenheit(Celsius);

  Serial.print("Temperature: In Celsius Scale ");
  Serial.print(Celsius);                      // To get temperature in Celsius
  Serial.print("C | In Fahrenheit Scale ");  //C & F text
  Serial.print(Fahrenheit);                   // To get temperature in Fahrenheit
  Serial.println("F");
  delay(200);

}
```

2. Arduino Code - A01.ino

```
#include "Arduino.h"
#include <OneWire.h>
#include <DallasTemperature.h>

#define ONE_WIRE_BUS 7 //Temperature Sensor Connected to 7 of Arduino
#define fan 9 // Fan connected to PWM 9 pin of Arduino
//Can also use one of PWM 3,5,6,9,10,11 pins
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

int temp;           // Measured Temperature
int tempMin = 20;   // Minimum Temperature to start the FAN
int tempMax = 100;  // Maximum Temperature to rotate FAN at full speed
int fanSpeed;
int fanP;           // Fan Speed Percentage

void setup()
{
  pinMode(fan, OUTPUT); // setting fan pin as output
  Serial.begin(9600);
  sensors.begin();
}
void loop()
{
  sensors.requestTemperatures();
  temp = sensors.getTempCByIndex(0);  // Read Temperature in Celsius scale

   if(temp  <= tempMin) { //If temp is lower than min temperature
        fanSpeed = 0;   // Fan speed = 0 when temp lower than reference temp
        fanP = 0;
        analogWrite(fan, 0); // digitalWrite(fan, LOW);
   }
   if((temp  > tempMin) && (temp <= tempMax)) {
         fanSpeed = map(temp, tempMin, tempMax, 0, 1023);
         // Arduino Uno Resolution ==> volts / 1024 units
          analogWrite(fan, fanSpeed); // spin the fan at based on temperature value
          fanP = map(temp, tempMin, tempMax, 0, 100); // speed of fan to display
   }

     Serial.print("Current Temperature: ");
     Serial.print(temp);
     Serial.print("C ");
     Serial.print("| fan Speed(%): ");
     Serial.print(fanP);
     Serial.println("%  ");
     delay(200);
}
```

References

1. https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf
2. https://www.arduino.cc/reference/en/libraries/onewire/
3. https://www.arduino.cc/reference/en/libraries/dallastemperature/
4. https://www.st.com/resource/en/datasheet/cd00001225.pdf
5. https://www.electroschematics.com/arduino-fan-speed-controlled-temperature/