Temperature Controlled DC Fan using Microcontroller

1. Introduction

The objectives of this project are to:-

(i) Enable the electric fan to consequently change the rate level as indicated by temperature changes.

(ii) Develop an automatic fan framework that can change the speed level because of the environment temperature changes.

The microcontroller base programmed fan framework displayed in this venture is obliged to satisfy the necessity of advances "tomorrow will be more exceptional than today". The electric fan naturally switches the pace as per the earth temperature changes. This electric fan framework contains mix of sensor, controller, driver and engine with incorporation of installed controlled programming.

1.1 Problem Statement

Most SITES feels the badly designed about changing the fan rate level physically when the room temperature changes. Along these lines, the programmed fan framework that consequently changes the velocity level as indicated by temperature changes is prescribed to be fabricated for tackling this issue.

2. Work Done:-

2.1 Circuit Principle:-

In circuit principle there are three electronics devices used which are temperature sensor, micro controller and transistor IC. Basically, the function of the temperature is to sense the temperature from the environment and to give an analog output to the ADC pin of the microcontroller. Temperature sensor (thermistor) acts as a transducer. The function of the ADC pin of the microcontroller is to convert the analog signal into digital signal as the microcontroller can read only digital signals. ATmega8 microcontroller is used here. It has 6 multiplexed ADC channels which has 10-bit determination. This analog signal is compared to the threshold value programmed in the microcontroller. If the analog signal is greater than the threshold value or set value then the fan will be switched on. Motor driver runs the DC fan. It has two output signal and two enable pins. It is designed so that two DC motors will run at the same time.

2.2 Use of ADC Registers:-

ADC, ADCSRA and ADMUX registers are contained inside the ATmega8 microcontroller and these registers are declared for analog to digital conversion. 10 bit resolution is used in analog to digital conversion.

1) Reference voltage is selected by utilizing ADCMUX register to the ADC.

2) Reference voltage is set by selecting REFS1 and REFS0 values in ADMUX.

3) ADC channel is selected by the MUX0 to MUX3.

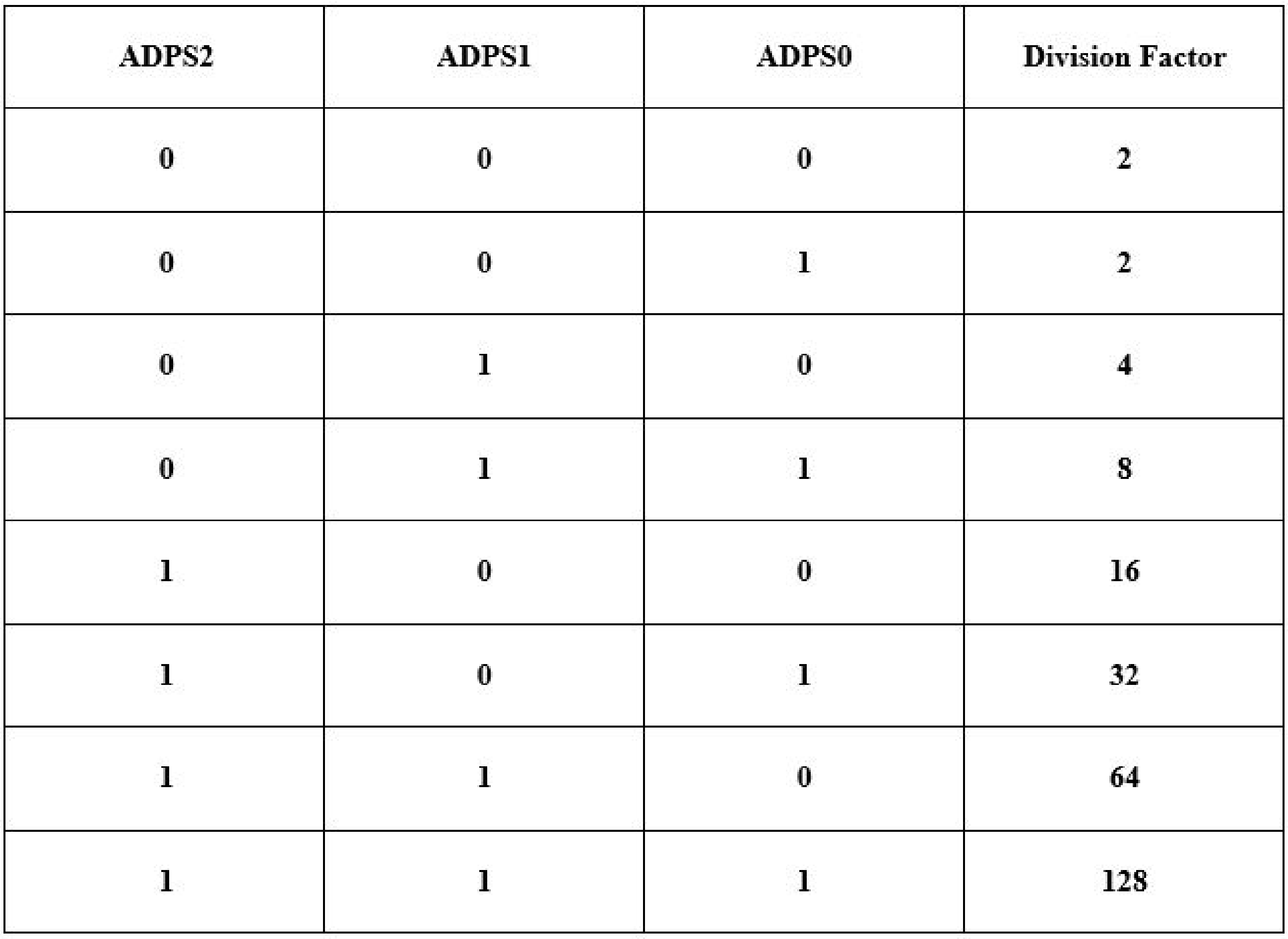
According to the set value of the MUX ADC values are shown in the below table:-

MUX 3-0	**ADC CHANNEL**
0	**ADC0**
1	**ADC1**
10	**ADC2**
11	**ADC3**
100	**ADC4**
101	**ADC5**

4) To select any ADC channel from ADC0 to ADC7 ADCMUX is defined by ADMUX=0b00000111

5) To start the analog to digital conversion REFS0, ADEN, ADSP2, ADSP1 are set to 1 where ADSP1 and ADSP2 are used as dividing factor of the internal clock cycle.

6) ADSRA and ADSC are set to 1 for conversion



ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

7) An interrupt signal is introduced when the conversion is completed.

8) After a delay ADIF = 1 is set.

3. ATmega-8:-

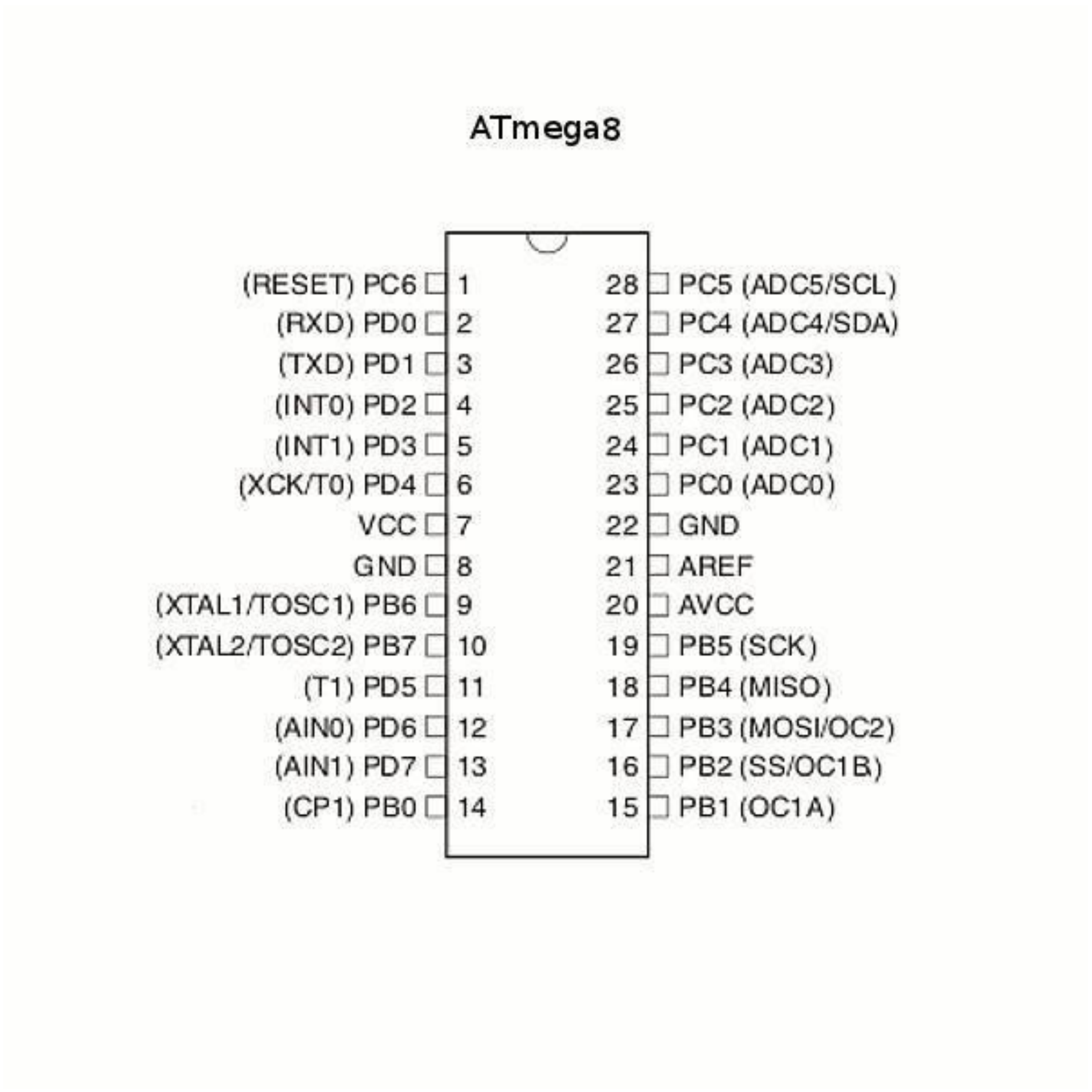
3.1 ATmega8 Specification :-

The ATmega8 microcontroller contains 32 general purpose working registers. As shown in the below figure these registers are directly connected to ALU. Two registers can carry one single instruction consequently in one clock cycle.

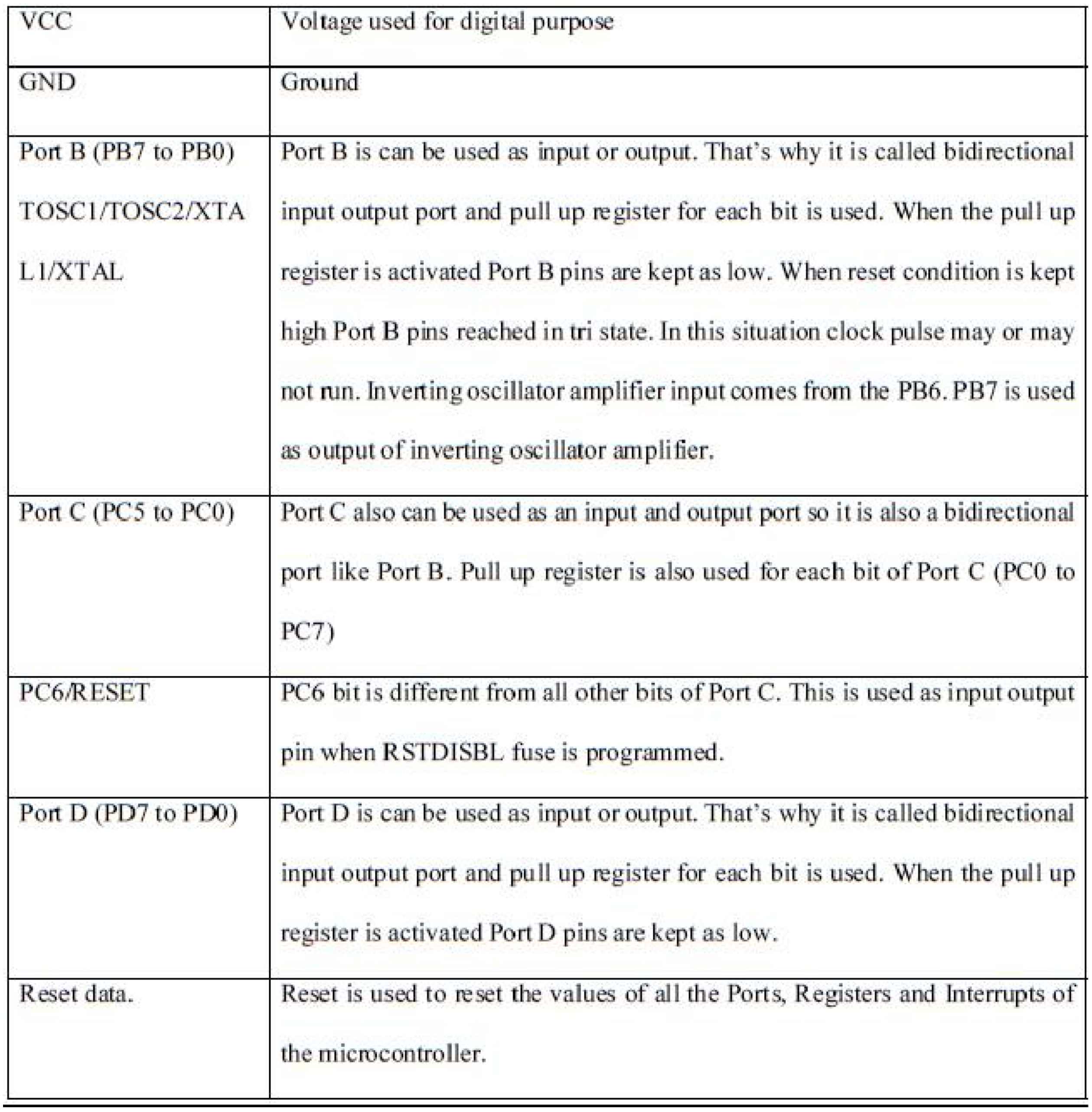
The following feature are comprised in ATmega8: -

1. 8 Kb of programmable flash memory which can read and write 10,000 times.
2. 512 bytes of EEPROM which can also read and write but 10 times more than the programmable flash memory.
3. 1 Kb of static RAM.
4. 23 general purpose input/output lines.
5. 32 general purpose working registers.
6. Counter and Timer with comparison mode.
7. Internal and external interrupts.
8. 3 types of communication interface which are SPI (serial port interface), a serial programmable USART and two wire interfaces.
9. A six channel ADC with 10-bit resolution with programmable watchdog timer with internal oscillators.

3.2 Pin Diagram of ATmega8:-



3.3 Descriptions of the Pin of ATmega8:-



VCC	Voltage used for digital purpose
GND	Ground
Port B (PB7 to PB0) TOSC1/TOSC2/XTAL1/XTAL	Port B is can be used as input or output. That's why it is called bidirectional input output port and pull up register for each bit is used. When the pull up register is activated Port B pins are kept as low. When reset condition is kept high Port B pins reached in tri state. In this situation clock pulse may or may not run. Inverting oscillator amplifier input comes from the PB6. PB7 is used as output of inverting oscillator amplifier.
Port C (PC5 to PC0)	Port C also can be used as an input and output port so it is also a bidirectional port like Port B. Pull up register is also used for each bit of Port C (PC0 to PC7)
PC6/RESET	PC6 bit is different from all other bits of Port C. This is used as input output pin when RSTDISBL fuse is programmed.
Port D (PD7 to PD0)	Port D is can be used as input or output. That's why it is called bidirectional input output port and pull up register for each bit is used. When the pull up register is activated Port D pins are kept as low.
Reset data.	Reset is used to reset the values of all the Ports, Registers and Interrupts of the microcontroller.

4.:- LM2576 TRANSITOR IC

4.1 Features of lm-2576:-

- 3.3-V, 5-V, 12-V, 15-V, and Adjustable Output Versions
- Adjustable Version Output Voltage Range,1.23 V to 37 V (57 V for HV Version) ±4% Maximum Over Line and Load Conditions
- Specified 3-A Output Current
- Wide Input Voltage Range: 40 V Up to 60 V for HV Version
- Requires Only 4 External Components
- 52-kHz Fixed-Frequency Internal Oscillator
- TTL-Shutdown Capability, Low-Power Standby Mode
- High Efficiency
- Uses Readily Available Standard Inductors
- Create a Custom Design with WEBENCH Tools
- Thermal Shutdown and Current Limit Protection

4.2 Application

- Simple High-Efficiency Step-Down (Buck) Regulator
- Efficient Preregulator for Linear Regulators
- On-Card Switching Regulators
- Positive-to-Negative Converter (Buck-Boost)

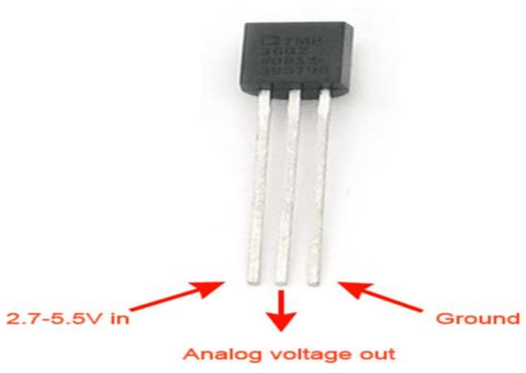
5. LM35:-

LM35 is a device which converts the physical signal into electrical signal. That's why this is known as the transducer. It is calibrated with the environmental temperature and it is linearly varies with the temperature and its output is in volt. There is no need of external calibration to provide the accuracy of the LM35 at room temperature which is about ±¼°C. Minimum temperature that can be measured by the LM35 device is -55°C. And maximum temperature that can be measured by LM35 is 150°C. Calibration of LM35 is done by trimming at the water level. To make the interfacing of control circuitry and readout circuitry very easy, low impedance at output side, output which is linear and precise inherent calibration of LM35 plays an important role. Temperature sensor takes a very low current of order 60 µA from the input supply. Heat loss in the LM35 is very less degree of around 0.1°C. LM35 can work in the range of -50°C to +150° which is the rated value. Another device which is also a temperature sensor of the family of LM35 known as LM35C which ranges from -40°C to +110°C. LM35 costs around 10 rupees in India and is easily available in the market which anyone can buy at any convenience store or electronics store.

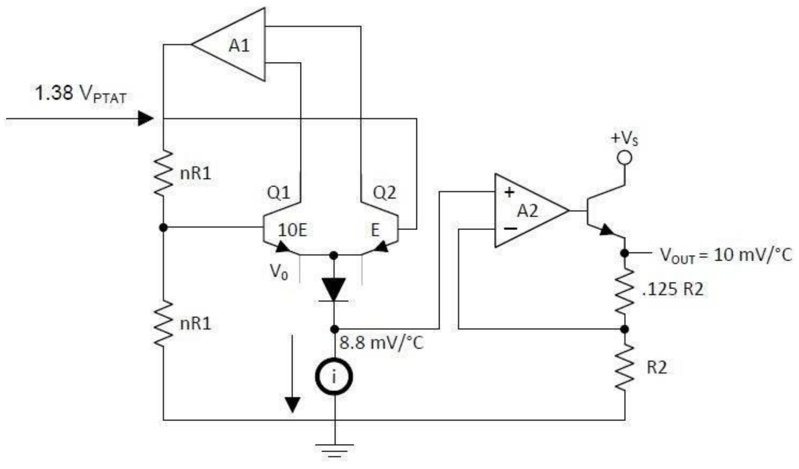
5.1 Features:-

- Low cost
- Accuracy is about ±¼°C
- Linearly varies with temperature(in centigrade)
- It can measure the temperature from -55°C to 150°C
- The current drawn from the supply is very less about 60 µA
- There is negligible heat in LM35
- Low impedance output; for a load of 1 mA about 0.1 ohm.
- Calibrated linearly with Celsius.
- Input voltage can vary from 4 volts to 30 volts.

5.2 Pin Diagram of LM35:-



5.3 Circuit Diagram of LM35:-



7. Overview of Temperature Controlled DC Fan using microcontroller:-

In this project we used ATmega8 microcontroller, driver IC, temperature sensor and DC motor. The function of each equipment depends on each other. The function of temperature sensor is to sense the temperature from the environment and give the analog input to the microcontroller at the Port C where ADC pin converts the analog signal into digital signals. Microcontroller has 28 pins. Some of the pins are known as VCC, GND, AVCC, etc. AVCC is used for ADC.

There are three ports in the ATmega8 microcontroller which are Port B, Port C and Port D. each port can be used as input or output ports. In the project we used Port B as an output and output from the port is given to the motor driver. Pin PB1 is connected to the input 1 and pin PB2 is connected to the input 2 of the driver IC. The output of the motor driver IC is connected to the DC motor. DC motor runs when input 1 and input 2 is either 01(low high) or 10(high low).

7.1 Working of the simulation Circuit: -

1. First power supply is given to the Arduino board.
2. Temperature sensor starts sensing the temperature from the environment.
3. The temperature sensor starts giving analog signal to the microcontroller.
4. ADC pins of the microcontroller starts converting analog signal into digital signal.
5. Using the internal successive approximation method the conversion of analog value to digital value is done by the microcontroller.
6. When the temperature is greater than the threshold or set value then the microcontroller gives output to the motor driver for starting the DC fan.
7. Now the motor starts running.

7.2 Applications:-

- This can be used in home applications.
- The circuit can be used in CPU to reduce the heat.
- BTS temperature control application.

8. Program and Result:-

8.1 Programming:-

```
#include <avr/io.h>
#include <util/delay.h>
int val;
long R;
double Thermister;

void ADC_init()
{
        ADMUX|=(1<<MUX0)|(1<<REFS0)|(1<<REFS1);
        ADCSRA|=(1<<ADEN);
}

int adc_read()
{

        ADCSRA|=(1<<ADSC);// Set ADSC

        while(!(ADCSRA&(1<<ADIF)));//Wait for ADIF bit of ADCSRA to be enabled

        ADCSRA|=(1<<ADIF);//Clear ADIF by writing 1 to it

        return(ADC);
}

double getTemp()
{
        val = adc_read();                    // store adc value on val register
        R=((10230000/val) - 10000);  // calculate the resistance
        Thermister = log(R);      // calculate natural log of resistance
        Thermister = 1 / (0.001129148 + (0.000234125 * Thermister) + (0.0000000876741 * Thermister *
Thermister * Thermister));
        Thermister = Thermister - 273.15;  // convert kelvin to °C
        return Thermister;
}
```

```
void main()
{
        DDRD=0xff;  //lcd interface 00001000
        DDRC=0x00;   // adc intterface
        DDRB |= (1<<1)|(1<<0);
        // PB1 as output

        // set PWM for 50% duty cycle at 10bit
        TCCR1A |= (1 << COM1A1);
        // set non-inverting mode
        TCCR1A |= (1 << WGM11) | (1 << WGM10);
        // set 10bit phase corrected PWM Mode
        TCCR1B |= (1 << CS11);

        char array[20],ohm=0xF4;
        double temp;

ADC_init() ;                // initialize ADC

      while (1)
        {
        temp = getTemp() ;   // store temperature value on temp resistor
        memset (array,0,20) ;
        dtostrf(temp,3,2,array);

        if (temp>=8 && temp<=20)
        {
           OCR1A=60000;// fan speed 20%  -65535
          PORTB&=~(1<<0);
        }
        else if (temp>=21 && temp<=30)
        {
           OCR1A= 26215;// fan speed 40% -26215
           PORTB&=~(1<<0);
        }
        else if (temp>=31 && temp<=40)
        {
           OCR1A= 39321 ;// fan speed 60%
           PORTB&=~(1<<0);

        }
        else if (temp>=41 && temp<=50)
        {
           OCR1A= 52428 ;       // fan speed 80%  -52428
           PORTB&=~ (1<<0);
        }
        else if (temp>=51 && temp<=55)
        {
          OCR1A=0;      // fan speed   90% - 0
          PORTB&=~ (1<<0);
        }
        else if(temp>=56)
        {
          OCR1A=65535;// fan speed
           PORTB|=(1<<0);

        }

    }


}
```

8.2 Simulation result:-

9. Conclusion:-

Basic idea of this project is to run the DC motor fan when temperature sensed by the temperature sensor is greater than threshold value. In this project we have used Arduino board for the programming of microcontroller through the USB. The microcontroller uses the hex file to execute the program. The temperature sensor output is connected to the microcontroller and it gives the output to the motor driver IC which runs the motor. In this way our main objective of the project is achieve