ECU
EGYPTIAN CHINESE UNIVERSITY
الجامعة المصرية الصينية
History Creates Your Future

Temperature control DC fan

Names:

Name	ID
Mohamed Ali Sayed	191900165
Kirolos Osama Ayad	191900016
Islam Osama Elsayed	191900044
Abanoub Ibrahem thabet	191900276
Alaa mostafa	191900082

Description of the project:

A Temperature Controlled DC Fan is a system which automatically turns on a DC Fan when the ambient temperature increases above a certain limit.

Generally, electronic devices produce more heat. So this heat should be reduced in order to protect the device. There are many ways to reduce this heat. One way is to switch on the fan spontaneously.

Components:

- Atmega16 Microcontroller: ATmega16 is an 8-bit high performance microcontroller from the Atmel's Mega AVR family. Atmega16 is a 40-pin microcontroller based on enhancedRISC (Reduced Instruction Set Computing) architecture with 131 powerful instructions.
- LM35 Temperature Sensor: LM35 is a temperature measuring device having an analog output voltage proportional to the temperature
- L298N Motor Driver: The L298N is a 16-pin Motor Driver IC which can control a setof two DC motors simultaneously in any direction.
- 12v DC Fan: Axial compact fans are suitable for high air performance with moderate pressure build-up.
- LED: LED is an electronic component and stands for Light Emitting Diode. It is a small light which has a very low power consumption.
- LCD: LCDs (Liquid Crystal Displays) are used in embedded system applications for displaying various parameters and status of the system
- Resistor: A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element.
- Potentiometer : is a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider.

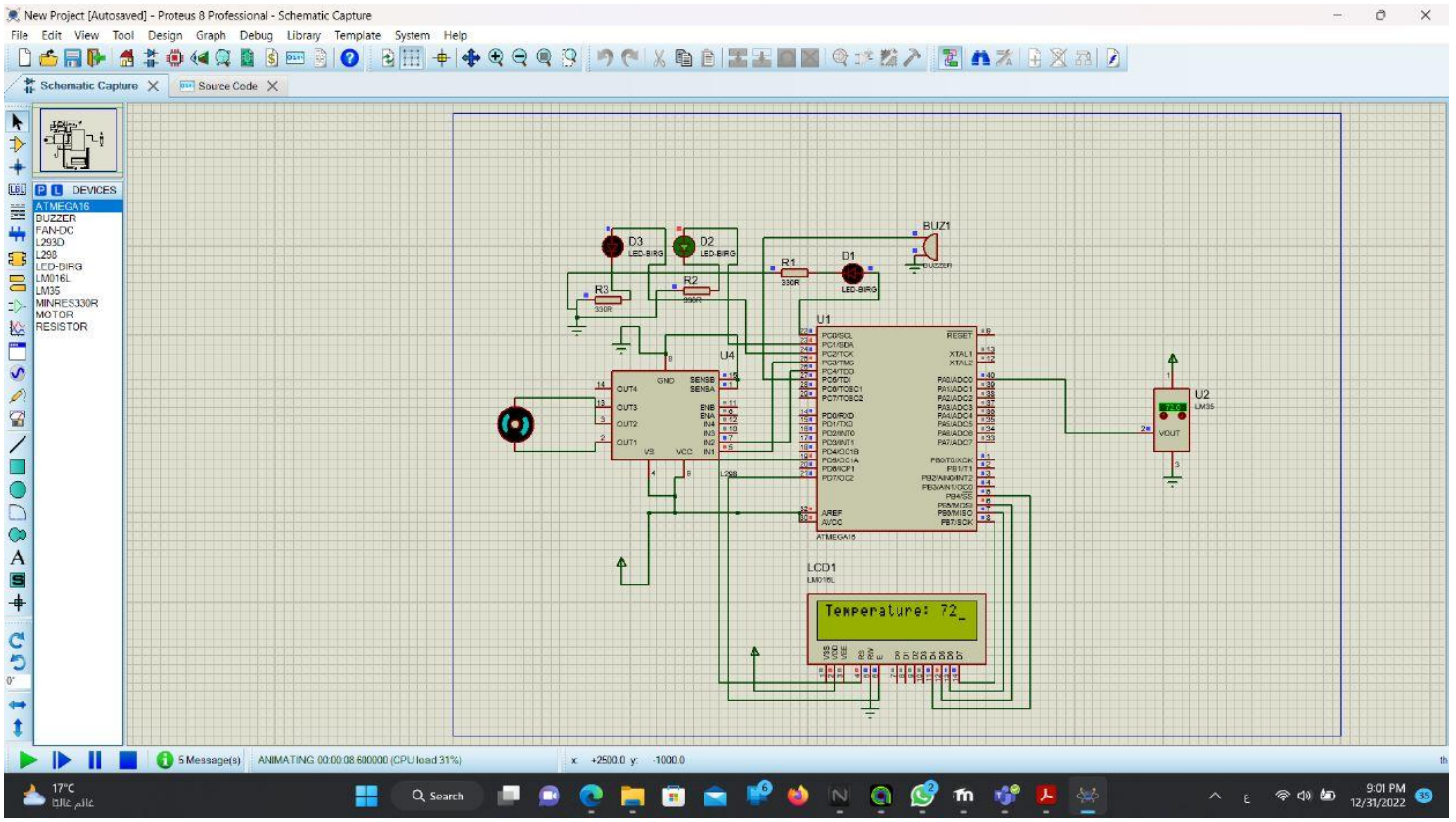
Operation:

- Fan that works by sensing heat from overheating into three levels according to the heat level sensed by (lm35) and display the result on LCD screen andafter passing 100 °C a buzzer turn on.
- Low speed: 1 - 40 °C
- mid speed: 40-80 °C
- High speed: 80-99 °C or higher
- At °C<100 : the buzzer goes on

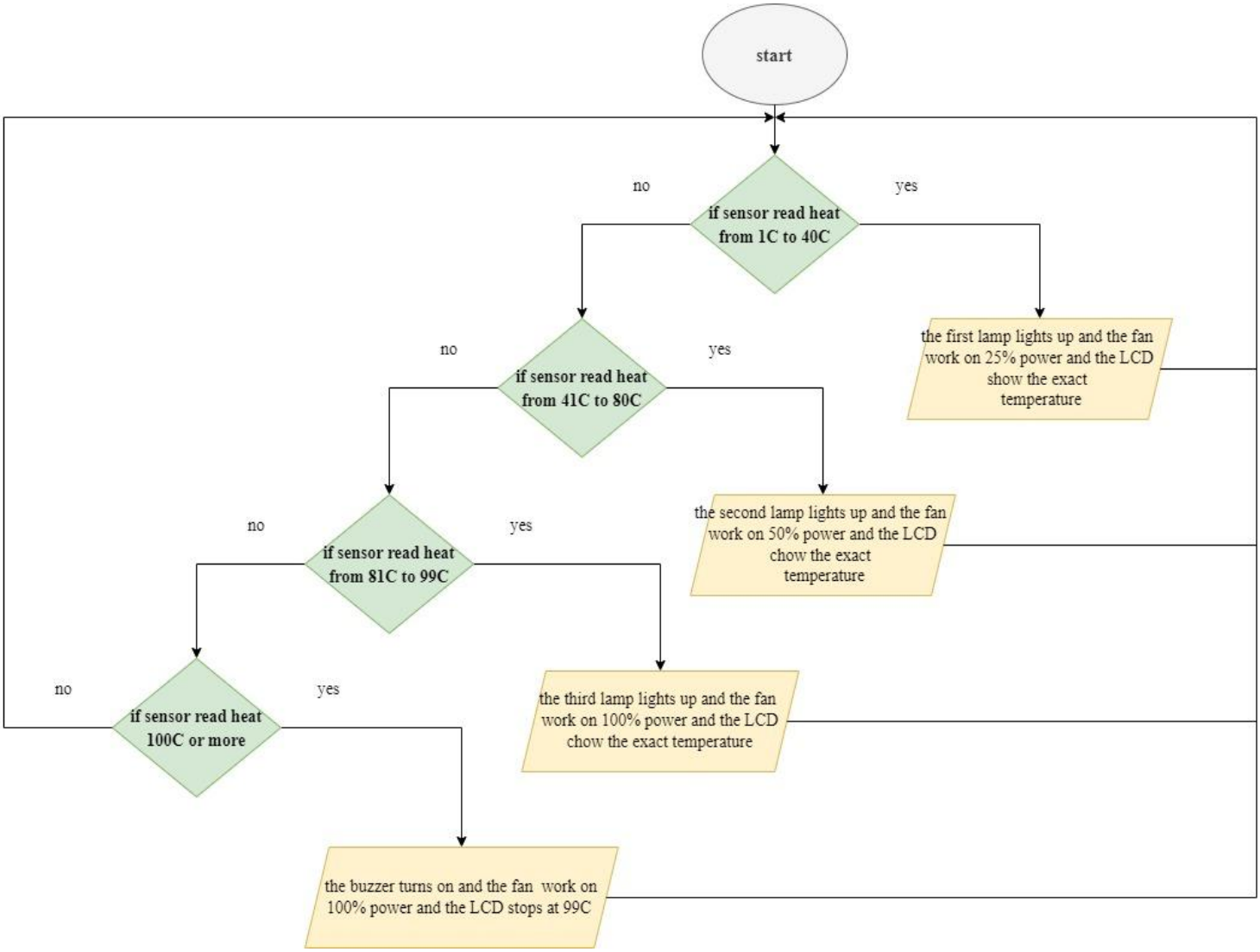
Code requirements (Pseudo code):

- ✓ Configure pins (Input and Output).
- ✓ Check If the temperature that the sensor reads bigger than or equal to 1°C and less than or equal to 40°C then play the fan on the lowest speed.
- ✓ Elseif temperature that the sensor reads bigger than or equal to41°C and less than or equal to 80°C then play the fan on the middle speed.
- ✓ Elseif temperature that the sensor reads bigger than or equal to 81°C then play the fan on the Hight speed.
- ✓ Else the buzzer work as the temperature is gretar than than 100°C.

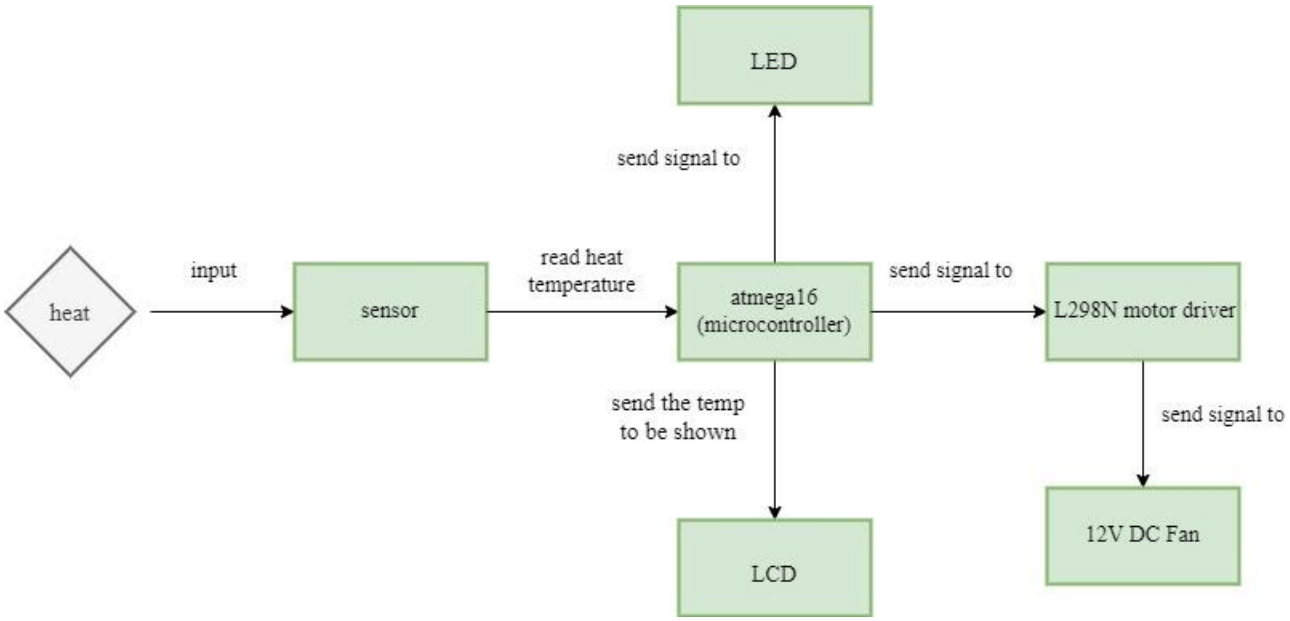
Simulation:


New Project [Autosaved] - Proteus 8 Professional - Schematic Capture
File Edit View Tool Design Graph Debug Library Template System Help
Schematic Capture
Source Code
DEVICES
ATMEGA16
BUZZER
FAN-DC
L293D
L298
LED-BIRG
LM016L
LM35
MINRES330R
MOTOR
RESISTOR
D3
D2
D1
R1
R2
R3
BUZ1
U1
U2
U4
LCD1
Temperature: 72_
5 Message(s)
ANIMATING: 00:00:08.600000 (CPU load 31%)
17°C
Search
9:01 PM
12/31/2022

Flow chart:


start
if sensor read heat from 1C to 40C
no
yes
the first lamp lights up and the fan work on 25% power and the LCD show the exact temperature
if sensor read heat from 41C to 80C
no
yes
the second lamp lights up and the fan work on 50% power and the LCD chow the exact temperature
if sensor read heat from 81C to 99C
no
yes
the third lamp lights up and the fan work on 100% power and the LCD chow the exact temperature
if sensor read heat 100C or more
no
yes
the buzzer turns on and the fan work on 100% power and the LCD stops at 99C

Block digram:



code:

```
#define F_CPU 1600000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <string.h>
#include <stdio.h>
#define LCD PORTB
#define EN 7
#define RS 5
#define RW 6



void lcdcmd(unsigned char cmd){
	PORTD &= ~(1<<RS);
	PORTD &= ~(1<<RW);
	LCD=cmd& 0xF0;
	PORTD |=(1<<EN);
	_delay_ms(1);
	PORTD &= ~(1<<EN);

	LCD=cmd<<4;
	PORTD |=(1<<EN);
	_delay_ms(1);
	PORTD &= ~(1<<EN);
}
```

```
void lcddata(unsigned char data){
	PORTD |= (1<<RS);
	PORTD &= ~(1<<RW);
	LCD=data& 0xF0;
	PORTD |=(1<<EN);
	_delay_ms(1);
	PORTD &= ~(1<<EN);

	LCD=data<<4;
	PORTD |=(1<<EN);
	_delay_ms(1);
	PORTD &= ~(1<<EN);
}
void lcd_init(){
	DDRB =0XFF;
	DDRD =0xFF;
	PORTD &= ~(1<<EN);
	lcdcmd(0x33);
	lcdcmd(0x32);
	lcdcmd(0x28);
	lcdcmd(0x0E);
	lcdcmd(0x01);
	_delay_ms(2);

}
void lcd_print(char *str){
	unsigned char i=0;
	while(str[i]!=0){
		lcddata(str[i]);
		i++;
	}
}

void convertndisplay(unsigned char value)
{
	unsigned char x,d1=0,d2=0;
	x=(value/10);
	d1=x;
	d2=value%10;
	lcdcmd(0x8D);
	lcddata(d1+0x30);
	lcddata(d2+0x30);
}

int main()
{
	lcd_init();
	lcd_print("Temperature:");
	DDRA |= (1<<0);
	ADCSRA=0x87;
	ADMUX=0xE0;
	 DDRC=0xFF;
	while(1)

	{
		unsigned char Temp;
		unsigned char i;

		ADCSRA |= (1<<ADSC);
		while((ADCSRA&(1<<ADIF))==0);
		Temp=ADCH;
		convertndisplay(Temp);
		_delay_ms(50);
		//PORTC=0b11101111; // for rotate clock wise
		/*_delay_ms(200);
		PORTC=0xFF; // for anti clockwise
```

```
_delay_ms(200);*/
i=64;
OCR0=i;
TCCR0=0x61;
GICR=(1<<INT0);
MCUCR=0x02;
sei();


if (Temp>0 && Temp<41)
{
	PORTC=0b00001001;

	_delay_ms(10);
	PORTC=0b00000001;
	_delay_ms(30);
}
else if(Temp>=41 && Temp<81)
{
	PORTC=0b00001010;

	_delay_ms(40);
	PORTC=0b00000010;
	_delay_ms(40);


}
else if(Temp>=81 && Temp<100)
{
	PORTC=0b00001100;



}
else if(Temp>=100)
{
	PORTC=0b11101100;

}
else{
	PORTC=0x00;
}
```

Submitted to:

Dr.Amr Zamel