**MICROCONTROLLERS AND INTERFACING TECHNIQUES TERM PROJECT**

**COURSE CODE** : 19CCE201 **SEMESTER** : 3 **BRANCH** : CCE

***TEMPERATURE CONTROLLED DC FAN***

|  |  |
| --- | --- |
| Kogoor Sai Hrithik | CB.EN.U4CCE20027 |
| Mohan V | CB.EN.U4CCE20033 |
| V.Parthiv | CB.EN.U4CCE20041 |
| Varun S | CB.EN.U4CCE20069 |

**1.Abstract**

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.

The aim of this project Is to design a temperature controlled fan using LPC2148 microcontroller, in which the *fan is automatically turned ON or OFF* according to the temperature and also the *temperature limit can be changed* anytime using input from *UART* .

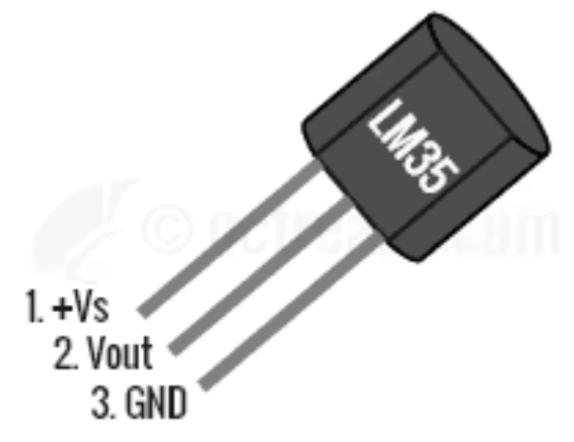
In this circuit, the *LM35* temperature sensor is used to get the *temperature sensed*. Then, the temperature is displayed on the LCD.

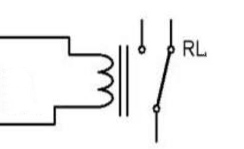
Like this, the microcontroller will continuously monitor the temperature. If the temperature exceeds *more than 35 degree Celsius* (upper safe temperature limit of laptops), the microcontroller will turn on the relay to *start the fan*.

If the temperature drops *below 35 degree Celsius*, the microcontroller will turn off the relay , but the *fan speed will slow down gradually* which further reduces the temperature.

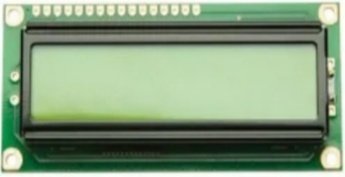
**2. Introduction**

*LPC2148* is a single-chip 16 bit or 32 bit ARM7 family based microcontroller 16/32-bit RISC Microcontroller with 512KB on-chip Flash ROM with In-System Programming (ISP) and Two 10bit ADCs with 14 channels, Two UARTs.

 ***Components used*** : LM35(Temperature Sensor) , Single Pole Double throw(SPDT) Relay, 12V DC Fan, Virtual terminal, LM016L (16 X 2 LCD Display), Push Button

****The *LM35* is an integrated circuit sensor that can be used to measure temperature with an output voltage that is proportional to temperature (in °C). When temperature increases, the voltage across a diode increases at a known rate.

*******Relay* is an electromagnetic switch which allow low power circuits to switch to a relatively high Voltage.

*DC fans* are basically cooling fans engineered to meet requirements such as higher air flows.

A *16x2 LCD display* is alphanumeric dot matrix display is capable of displaying it can display 16 characters per line and there are 2 such lines.

Configuration :

1. Pin 2 is the *data pin of LM35* which is connected to the analog input pin **P0.27** i.e. ADC0. Pin 1 to 5V power supply and Pin 3 to Ground.
2. As temperature changes, the output of the ADC is generated. The digital output of the ADC is given to Microcontroller to control the fan accordingly.
3. The *last 4 data pins of the LCD* are connected to **P0.12** to **P0.15** of the microcontroller. Pin 1 and 3 to Ground and Pin 2 to 5V power supply.
4. A *Single Pole Double throw(SPDT) relay* is connected to **P0.20** pin. The single pole is connected to 12V DC Source and Normally Open(NO) throw is connected to 12V DC Fan.
5. The *Rx of Virtual terminal* is connected to **P0.1**(TX) pin and *TX* is connected to **P0.2** (RX) pin.
6. The push button with DC supply is connected to **P0.16** pin, which is used to change the temperature limit based on the input from the UART.

**3. Algorithm**

1. Configure PINSEL0 pins as GPIO
2. Configure P0.20 as Output which is used as input for RELAY
3. Configure P0.16 as Input for getting input from switch
4. Make ADC and LCD initialisation
5. While (1)
6. Get Temperature value from ADC
7. If P0.16 is high
8. Get input from UART(PC)
9. End
10. Delay of 1 second
11. If previous value is not equal to current value
12. Convert Temperature into Fahrenheit
13. Display temperature in Celsius and Fahrenheit
14. If temperature is greater than temperature limit
15. Set P0.20 so that relay is and fan is on
16. Display “Normal” in LCD
17. Else
18. Clear P0.20 so that relay is and fan is off
19. Display “Overheat” in LCD
20. End
21. End
22. End

**4.Program code**

#include <LPC214X.H>

#include <stdio.h> // for sprintf function

#include <stdlib.h> // for atoi function

#define LCD (0xFFFF00FF)

#define RS (1<<4)

#define RW (1<<5)

#define EN (1<<6)

void ADC\_INIT(void); // Initial setup of ADC

int ADC(void); // Returns ADC Value

int UART(void); // UART interface

void LCD\_INIT(void); // Initial setup of LCD

void LCD\_CMD(char command); // Instruction to LCD

void LCD\_STRING (char\* msg); // String to LCD

void delay\_ms(int count); // 1 milli second delay

int main (void)

{

int ADC\_val, prev = 0 , check = 0, temp\_f, TEMP\_LIMIT = 35 ;

//35 C - laptop upper safe temperature limit

char val[10];

PINSEL0 = 0x00000000;

IODIR0 |= 0X00100000; // P0.20 as output for DC Fan

IODIR0 &= 0XFFFEFFFF; // P0.16 as input from switch

ADC\_INIT();

LCD\_INIT();

while (1)

{

ADC\_val = ADC(); // Returns ADC Value

if ((IOPIN0 & 0x00010000) == 0x00010000) // if P0.16 is high

{

TEMP\_LIMIT = UART(); // Returns temperature limit from PC

check = 1; //executes if statement to check again

}

delay\_ms(1000); // 1000 milli sec = 1 sec

if (prev != ADC\_val || check == 1)

{

temp\_f = (ADC\_val \*9/5)+32 ; // temperature in farenheit

sprintf(val," %dC / %dF",ADC\_val,temp\_f); // int to string

LCD\_CMD(0x01); // Display clear

LCD\_STRING(val);

LCD\_CMD(0xC0); // New line in display

if (ADC\_val >= TEMP\_LIMIT)

{

IOSET0 |= 0x00100000; // sets P0.20 for relay

LCD\_STRING(" OverHeat");

}

else

{

IOCLR0 |= 0x00100000; // clears P0.16 for relay

LCD\_STRING(" Normal");

}

prev = ADC\_val;

}

check = 0;

}

}

void ADC\_INIT(void)

{

PINSEL1 &= 0xFF7FFFFF; // (PINSEL1<23> = 0)

PINSEL1 |= 0x00400000; // (PINSEL1<22> = 1)

//P0.27 is Configured as ADC Pin AD0.0

PCONP |= (1<<12); // Enable Power/Clock to ADC0

}

int ADC(void)

{

unsigned int ADC\_data;

AD0CR = 0x00200700; //CLKDIV=(PCLK)/8, BURST=0, CLKS=11clks/10bits, PDN=1

AD0CR|= 0x01; // A/D Channel 0

AD0CR |= (1<<24); //Activate ADC Module (new conversion )

//Wait for conversion to get over by monitoring 28th bit of A/D register

while(!(AD0GDR & 0x80000000));

//Read 10 bit ADC Data ie RESULT = 10 Bit Data (15:6)

ADC\_data = (AD0GDR >> 6)& 0x3FF;

//Deactivate ADC Module ie START = 000 (Bits 26:24) (stop conversion)

AD0CR &= 0xF8FFFFFF;

return ADC\_data;

}

int UART(void)

{

unsigned int i=0 ,val;

char data = 0, str[4];

PINSEL0 |= 0x00000005; // Enable RxD0 and TxD0

U0LCR = 0x83; // 8 bits, no parity , 1 stop bits, DLAB = 1

U0DLL = 97; // 9600 Baud Rate @ 15MHZ VPB Clock

U0LCR = 0x03; // DLAB = 0

while(data != 44) // if data received is not comma

{

// Wait until reception is over and UART0 is ready with data

while(!(U0LSR & 0x01));

data = U0RBR; //Receive character

// Wait until UART0 ready to send character

while(!(U0LSR & 0x20));

U0THR = data; //Send character

str[i] = data;

i++;

}

val = atoi(str); //converts string into integer

return val;

}

void LCD\_INIT(void)

{

//P0.12 to P0.15 LCD Data. P0.4,5,6 as RS RW and EN

IO0DIR |= 0x0000F070;

delay\_ms(20);

LCD\_CMD(0x02); // Initialize cursor to home position

LCD\_CMD(0x28); // 4 - bit interface, 2 line, 5x8 dots

LCD\_CMD(0x06); // Auto increment cursor

LCD\_CMD(0x0C); // Display on cursor off

LCD\_CMD(0x01); // Display clear

LCD\_CMD(0x80); // First line first position

}

void LCD\_CMD(char command)

{

IO0PIN = ( (IO0PIN & LCD) | ((command & 0xF0)<<8) ); //Upper nibble

IO0SET = EN; // EN = 1

IO0CLR = (RS|RW); // RS = 0, RW = 0

delay\_ms(5); // 5 milli sec

IO0CLR = EN; // EN = 0, RS and RW unchanged(RS = RW = 0)

delay\_ms(5);

IO0PIN = ( (IO0PIN & LCD) | ((command & 0x0F)<<12) ); //Lower nibble

IO0SET = EN; // EN = 1

IO0CLR = (RS|RW); // RS = 0, RW = 0

delay\_ms(5);

IO0CLR = EN; // EN = 0, RS and RW unchanged(RS = RW = 0)

delay\_ms(5);

}

void LCD\_STRING (char\* msg)

{

unsigned int i=0;

while(msg[i]!=0)

{

IO0PIN = ( (IO0PIN & LCD) | ((msg[i] & 0xF0)<<8) ); //Upper nibble

IO0SET = (RS|EN); // RS = 1, EN = 1

IO0CLR = RW; // RW = 0

delay\_ms(2); // 2 milli sec

IO0CLR = EN; // EN = 0, RS and RW unchanged(RS = 1, RW = 0)

delay\_ms(5);

IO0PIN = ( (IO0PIN & LCD) | ((msg[i] & 0x0F)<<12) ); //lower nibble

IO0SET = (RS|EN); // RS = 1, EN = 1

IO0CLR = RW; // RW = 0

delay\_ms(2);

IO0CLR = EN; // EN = 0, RS and RW unchanged(RS = 1, RW = 0)

delay\_ms(5);

i++;

}

}

void delay\_ms(int count)

{

int j=0,i=0;

for(j=0;j<count;j++)

{

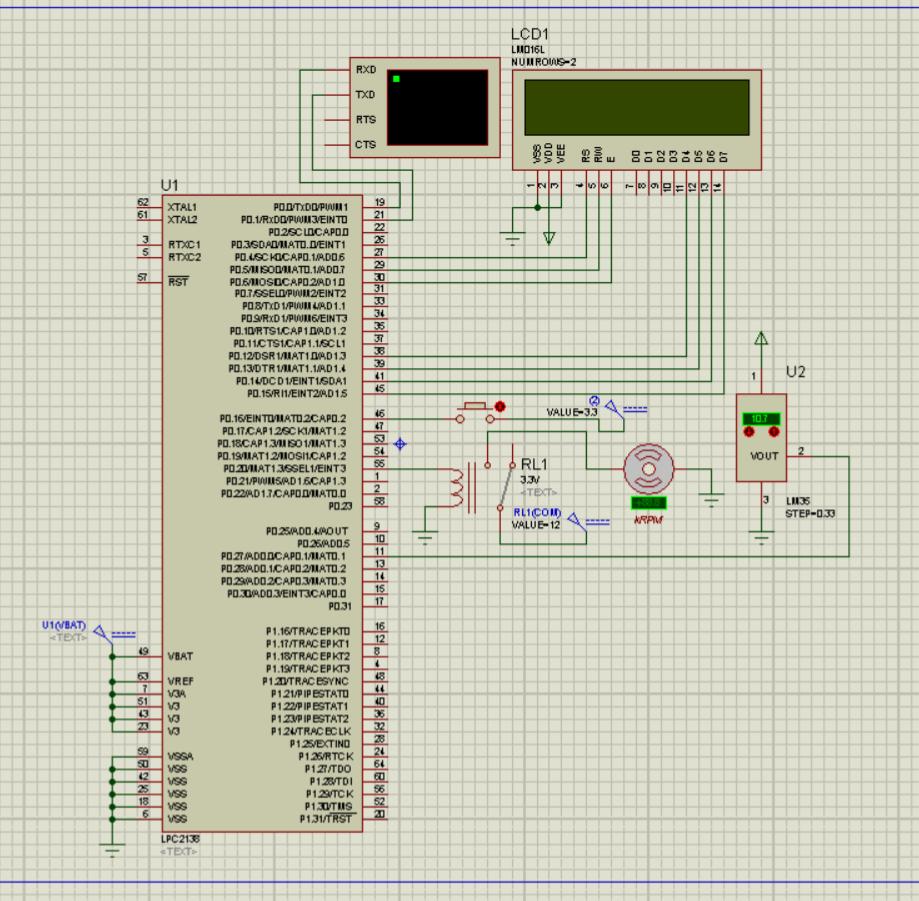
// At 60Mhz, the below loop introduces delay of 1 milli sec

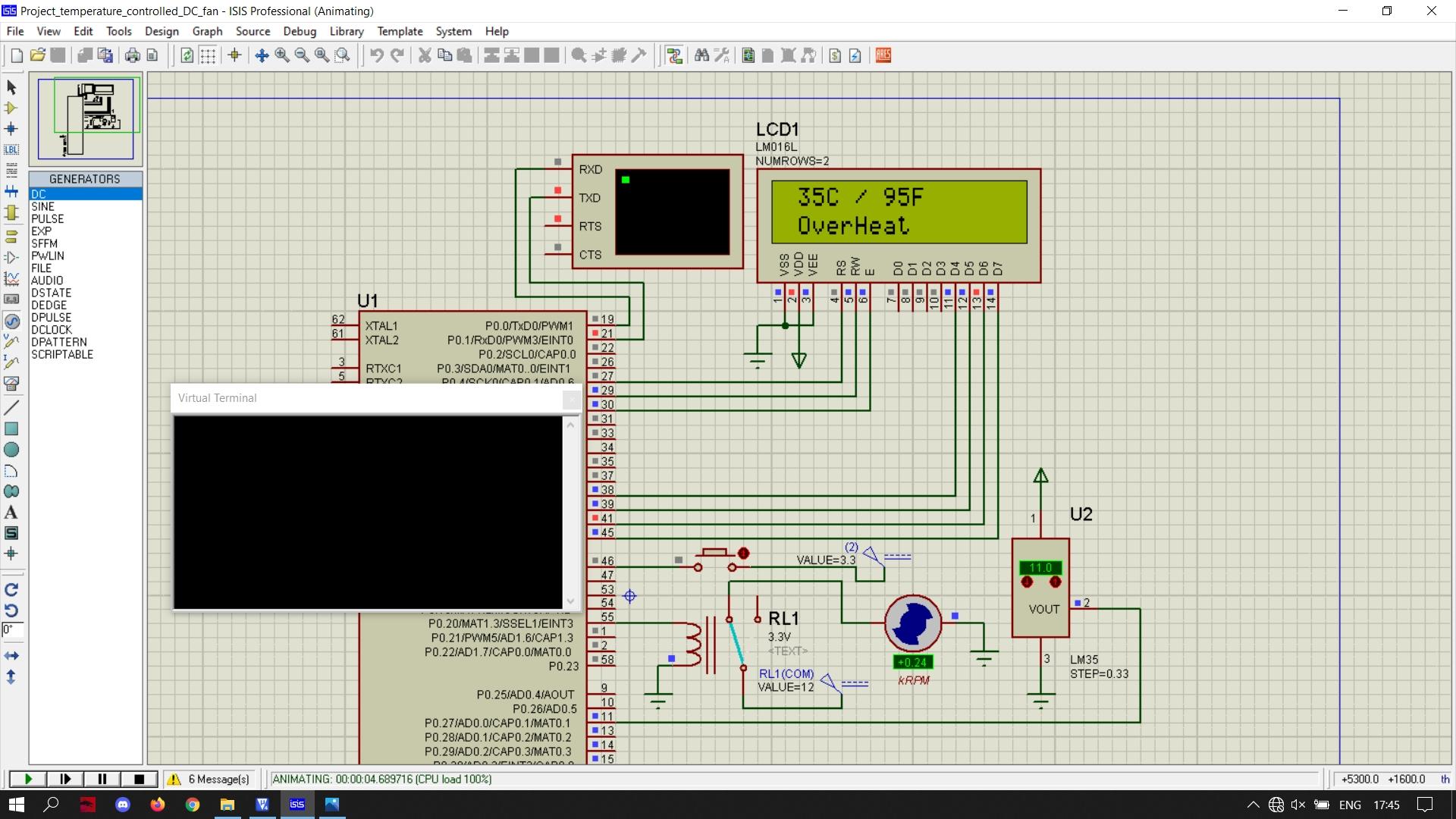
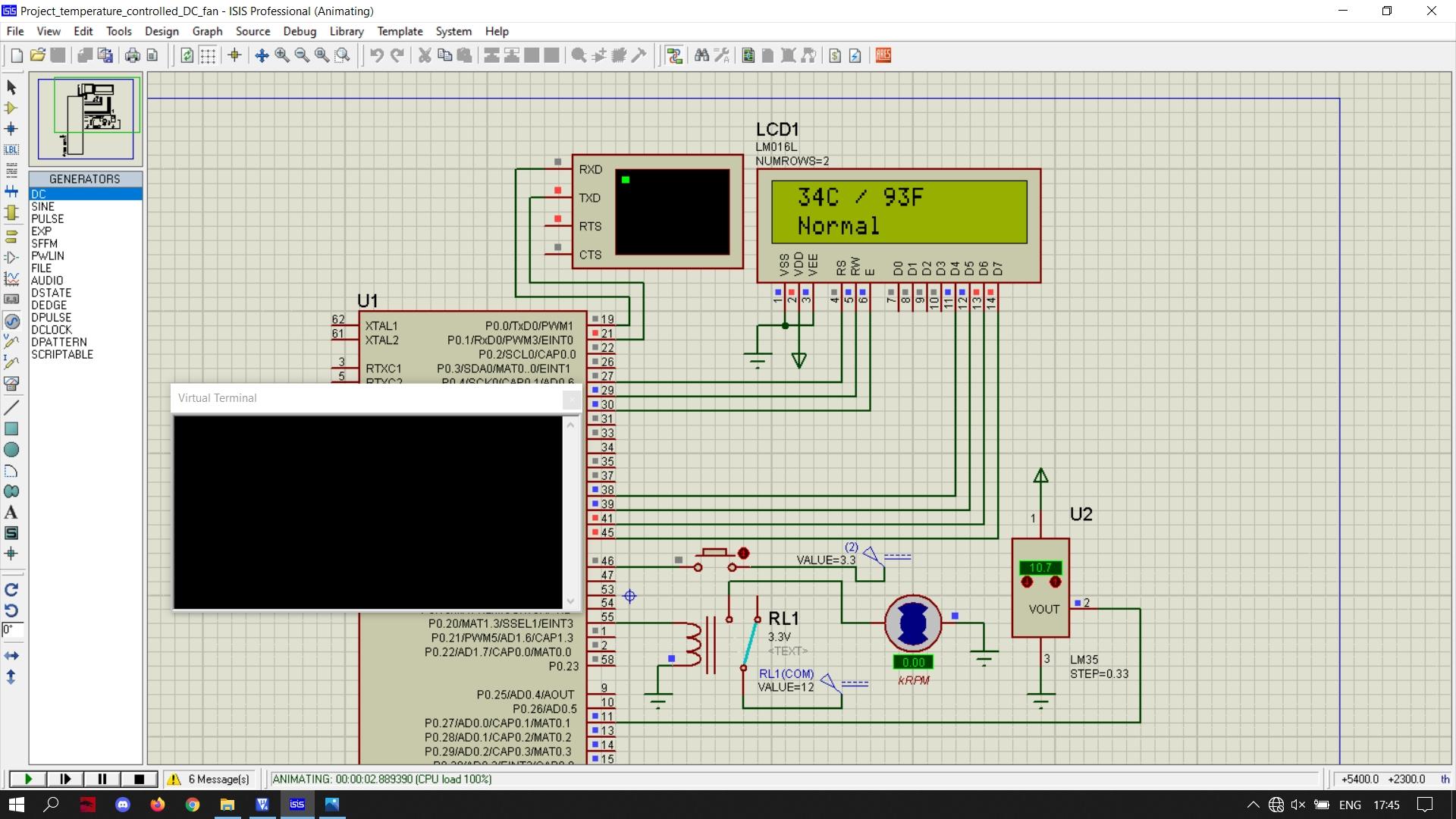
for(i=0;i<1250;i++);

}

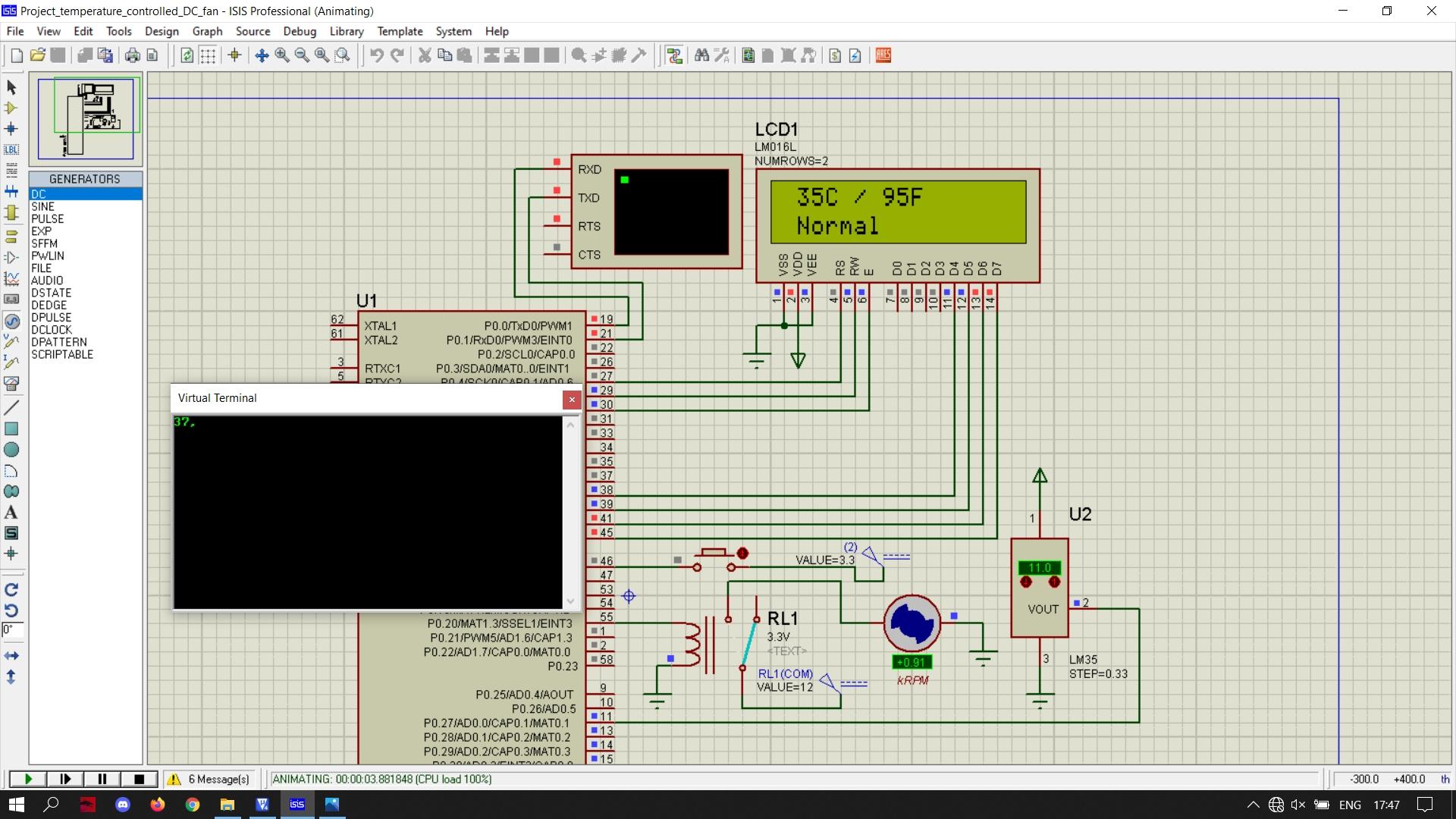
}

**5. Simulation results**

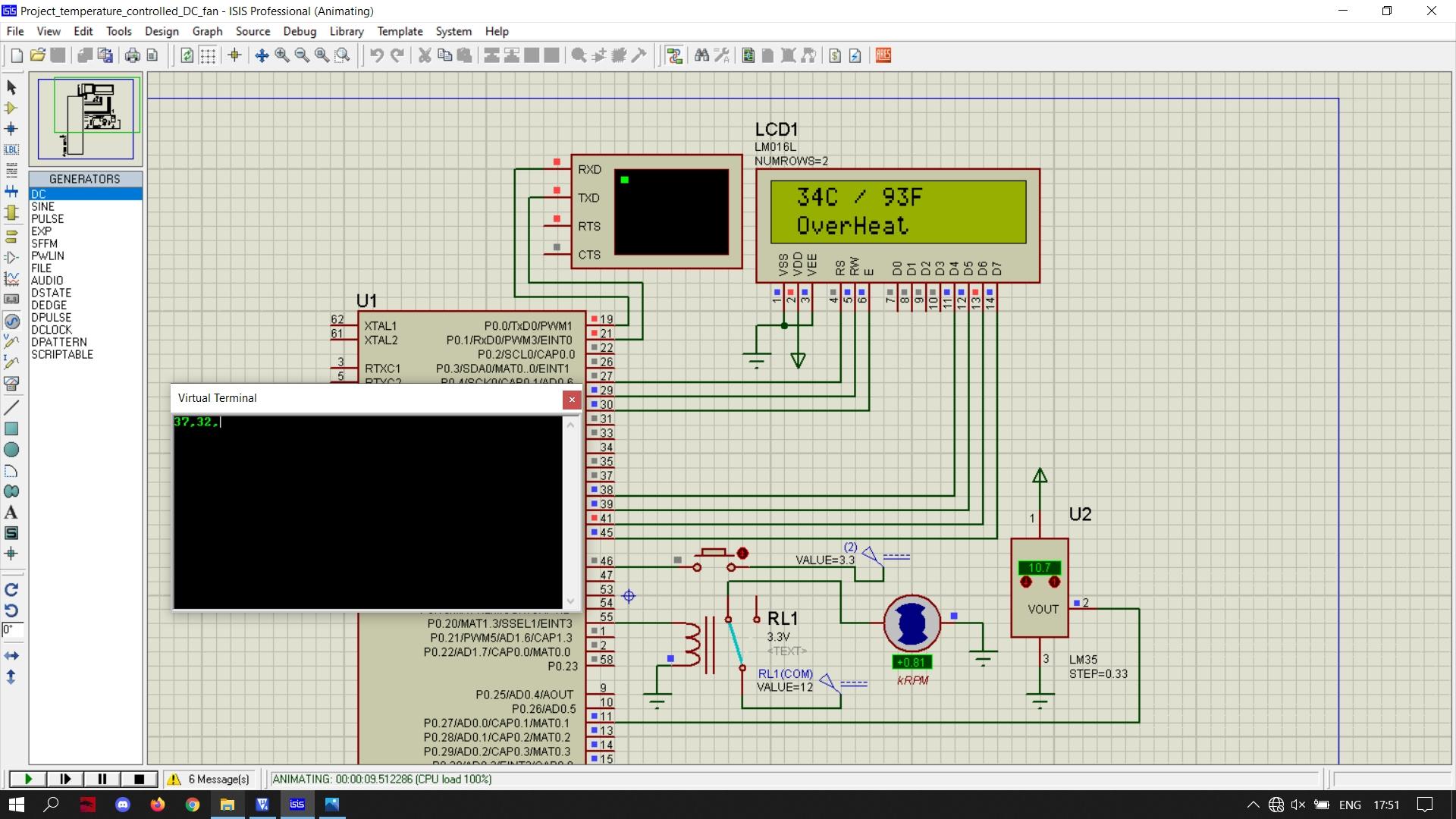
****

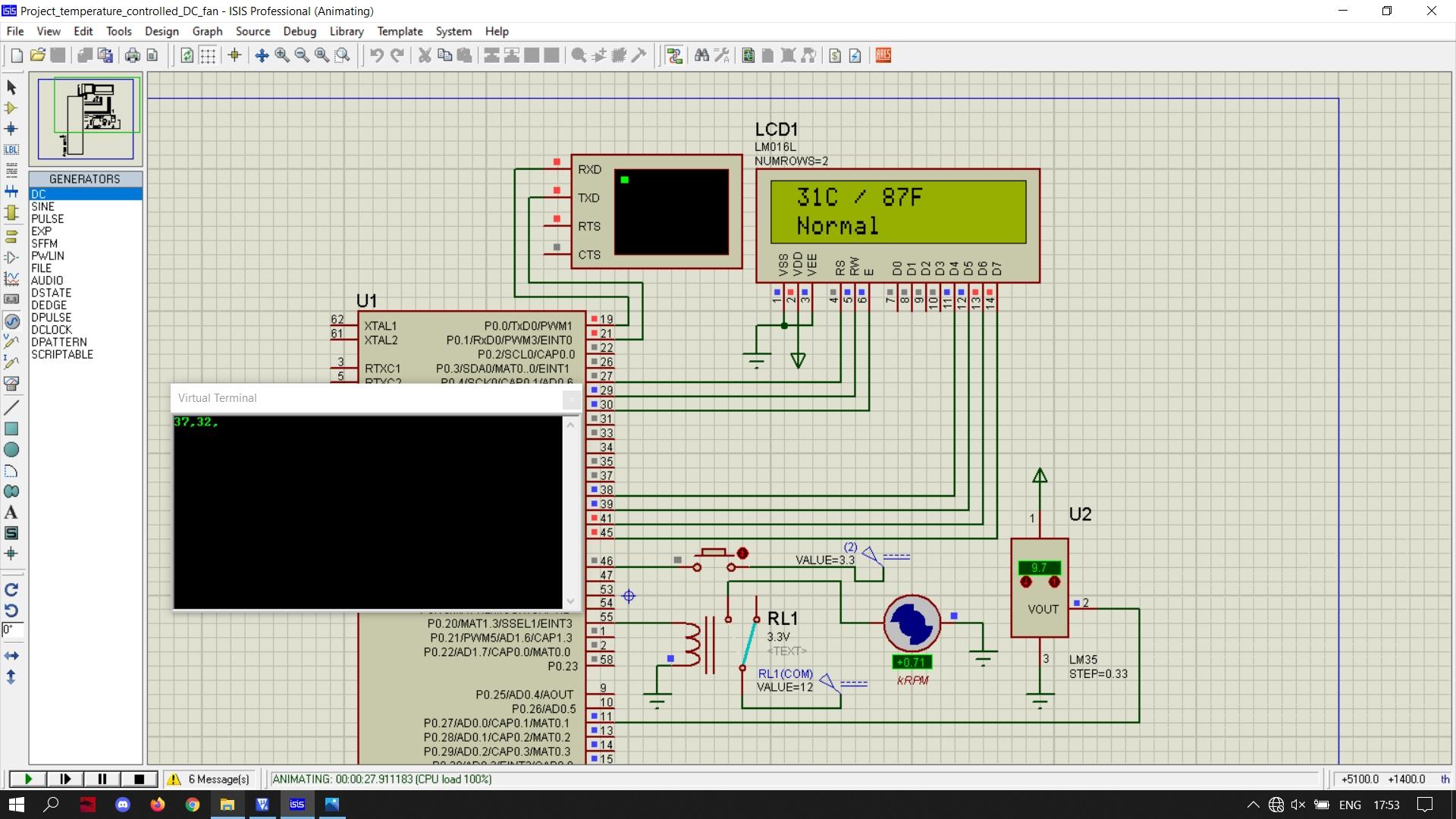
****Initially, temperature is 34°C , Fan is off. The temperature became 35°C , Fan is on. The LCD display shows “Normal” The LCD display shows “Overheat”

With temperature unchanged , switch is pressed and “37,” is entered in the virtual terminal. Now, the LCD display shows “Normal” because the TEMP\_LIMIT is 37 now but the current temperature is 35. Relay is off.

****

With temperature unchanged , switch Is pressed and “32,” is entered in the virtual terminal. Now, the LCD display shows “Overheat” because the TEMP\_LIMIT is 32 now but the current temperature is 34. Relay is on.

****

****The TEMP\_LIMIT with 32 as unchanged , temperature changes to 31. Now, the LCD display shows “Normal” because the TEMP\_LIMIT is 32 now but the current temperature is 31. Relay is off.