Shri Ramdeobaba College of Engineering and Management, Nagpur

E-Circuit Design and Testing Laboratory

Session: January –May 2021

Project Report

Temperature Control System using ATMega32

Submitted By:

Ayushi Namdev (Roll No: 03)

Aanchal Gupta (Roll No: 10)

Abhinav Gautam (Roll No: 12)

Abhishek Raj (Roll No: 14)

Department of Electrical Engineering

1. Introduction-

Temperature is one of the main parameters to be controlled in most of the manufacturing industries like chemical, food processing, pharmaceutical etc. In these industries some product needs the required temperature to be maintained at highest priority or else the product will fail. So, the temperature controller is most widely used in almost all the industries. In this project we are going to design a circuit for measuring temperature. This circuit is developed using "PT100". Temperature is usually measured in "Centigrade" and "Fahrenheit".

The goal of this project is to design an ambient temperature measurement and control circuit. The actual temperature is sensed by the temperature sensor. It is displayed on LCD in both units of temperature, and then some additional functions are carried out automatically depending upon that temperature. Like turning heater on if temp falls below a certain point and turning fan on if it exceeds a certain limit. Or else keeping both of them off if the temp is in the range of normal temperature. There range values of turning these on are also adjustable by the user as per their need.

1.1. Components Used-

1. PT100-RTD (Temperature Sensor)
2. Microcontroller (ATmega32)
3. Resistors (99 Ohms (56+43), 330 Ohms and 10k Ohms)
4. Capacitors (0.1uF,1uF and 10uF)
5. Potentiometer (1k)
6. DC Power Supply (+5 Volts and +9 Volts)
7. LM317T
8. LCD 16x2 Module
9. DC Fan (5 V)
10. PTC Heating Element (5 V)

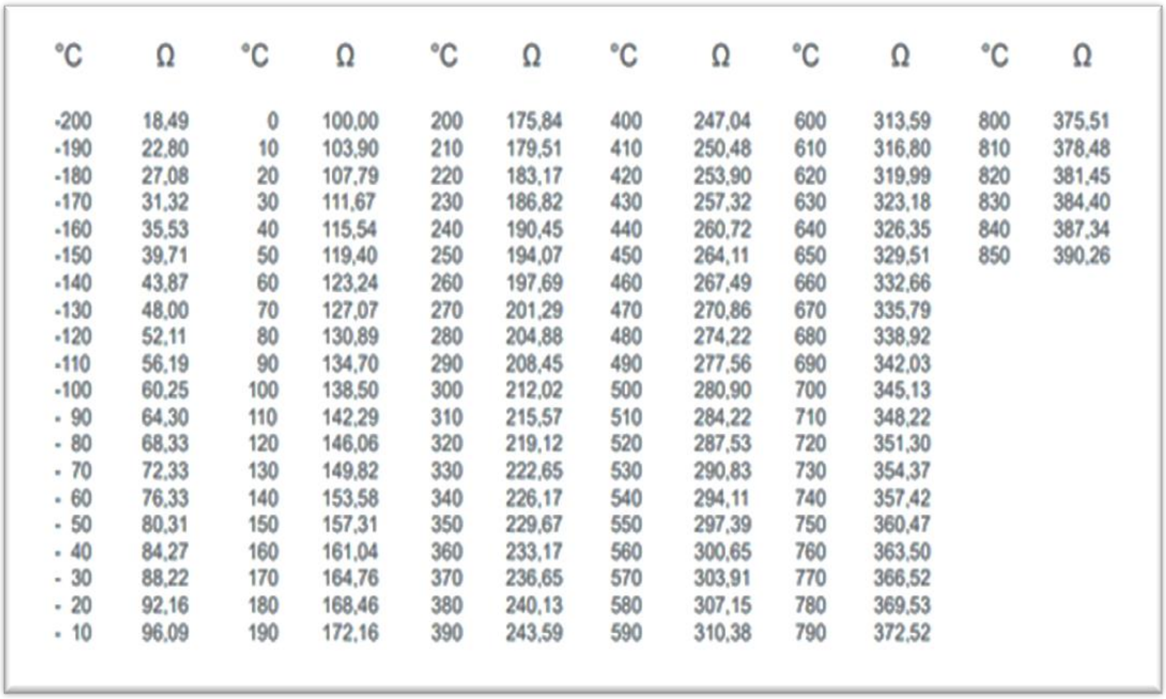
1.2. PT100-RTD -

The PT100 is a commonly used as industrial temperature sensor. It is known for its capability to measure high range temperature (200°C) with an accuracy of 0.1°C. The construction of the sensor is also simple and hence it can be used in rugged environments. One downside of this sensor is that it will not work out of the box. To get useful values of temperature one should use it along with a potential divider or a Wheatstone bridge. But since these sensors just work with variable voltage it is very easy to use them in projects. So, if you are looking for a sensor with good range and decent accuracy which is relatively cheap then PT100 would be a great choice.

It is one type of sensor which falls into a group called Resistance Temperature Detectors or RTD's. The sensor type, Pt100, indicates two important pieces of information about the sensor. The first part, Pt, is the chemical symbol for Platinum and this shows that the sensor is Platinum-based. The second part, 100, relates to the resistance of the device at 0°C.

- Temperature Range: -200°C to 850°C
- Resistance Range: 18.49 Ohms to 390.26 Ohms
- Accuracy: ±0.1°C
- Nominal Resistance: 100Ω at 0°C

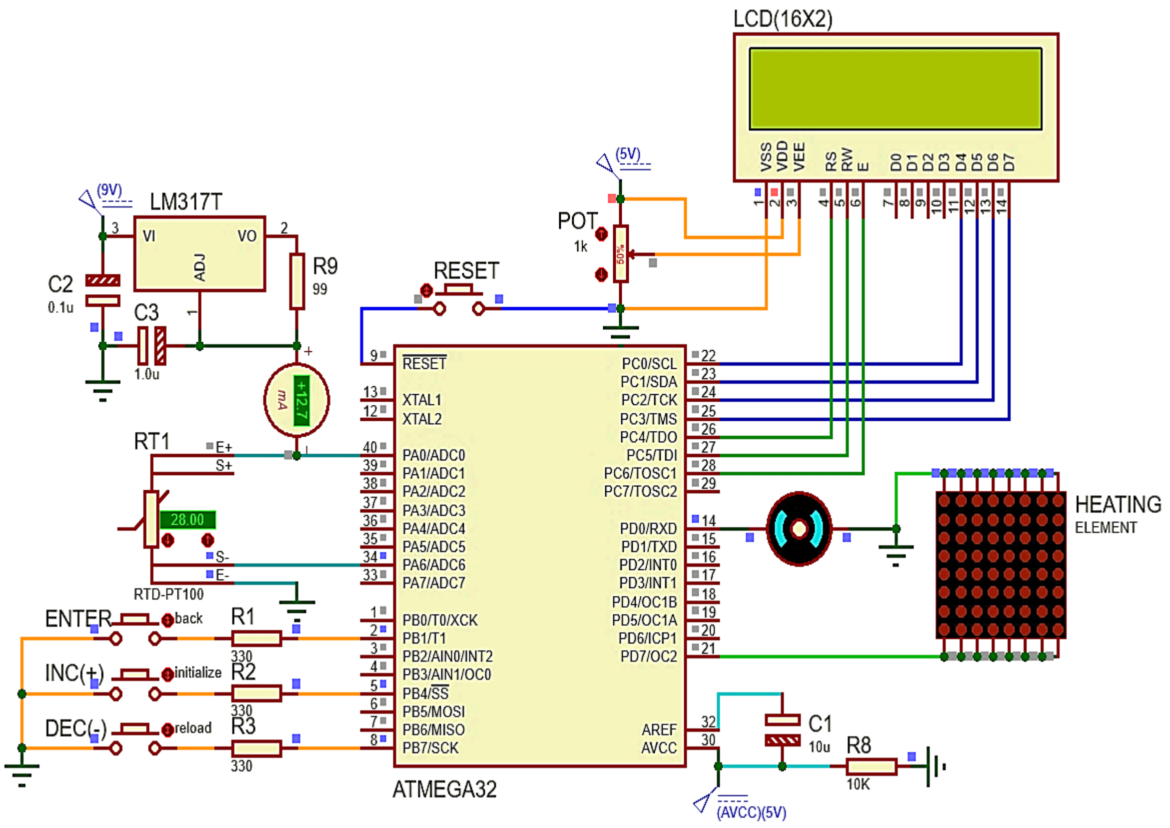
Datasheet Representing the variation in resistance of the PT100-RTD with respect to change in surrounding temperature.



°C	Ω	°C	Ω	°C	Ω	°C	Ω	°C	Ω	°C	Ω
-200	18,49	0	100,00	200	175,84	400	247,04	600	313,59	800	375,51
-190	22,80	10	103,90	210	179,51	410	250,48	610	316,80	810	378,48
-180	27,08	20	107,79	220	183,17	420	253,90	620	319,99	820	381,45
-170	31,32	30	111,67	230	186,82	430	257,32	630	323,18	830	384,40
-160	35,53	40	115,54	240	190,45	440	260,72	640	326,35	840	387,34
-150	39,71	50	119,40	250	194,07	450	264,11	650	329,51	850	390,26
-140	43,87	60	123,24	260	197,69	460	267,49	660	332,66		
-130	48,00	70	127,07	270	201,29	470	270,86	670	335,79		
-120	52,11	80	130,89	280	204,88	480	274,22	680	338,92		
-110	56,19	90	134,70	290	208,45	490	277,56	690	342,03		
-100	60,25	100	138,50	300	212,02	500	280,90	700	345,13		
- 90	64,30	110	142,29	310	215,57	510	284,22	710	348,22		
- 80	68,33	120	146,06	320	219,12	520	287,53	720	351,30		
- 70	72,33	130	149,82	330	222,65	530	290,83	730	354,37		
- 60	76,33	140	153,58	340	226,17	540	294,11	740	357,42		
- 50	80,31	150	157,31	350	229,67	550	297,39	750	360,47		
- 40	84,27	160	161,04	360	233,17	560	300,65	760	363,50		
- 30	88,22	170	164,76	370	236,65	570	303,91	770	366,52		
- 20	92,16	180	168,46	380	240,13	580	307,15	780	369,53		
- 10	96,09	190	172,16	390	243,59	590	310,38	790	372,52		

2. <u>Circuit Diagram-</u>

As shown below is the circuit diagram of our project:



2.1. Design Description-

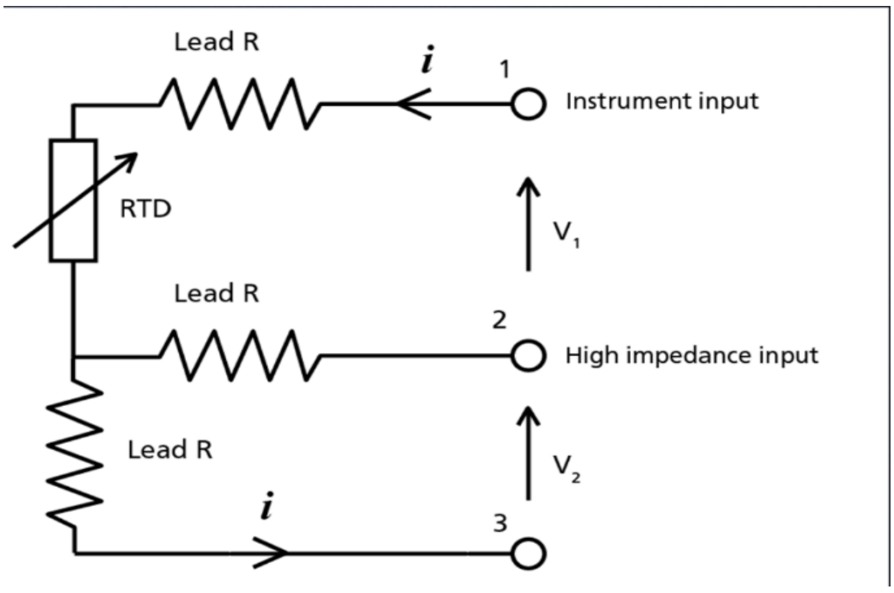
In this project as we are required to measure the temperature using PT100-RTD and display the measured temperature in degrees Celsius and degree Fahrenheit. So, for that purpose first of all the PT100-RTD is connected to a constant current source as we couldn't have simply connected it to voltage source as our resistance of the sensor changes as the temperature increases. Hence a constant current source was necessary to for proper measurement of ambient temperature. The value of current from current source is also chosen such that it gives proper change in voltage provided into the ADC pins so that it can increase the ADC values read from the ADC pins. Also, the proper value of voltage is connected to the AVCC Pin as per out requirement of resolution of ADC input. Further the code is programmed in such a way that the ADC value we obtain is properly converted in temperature value. Also, we have connected a dc fan and a PTC Heating Element so as when the temperature exceeds a certain threshold of temperature (As Set by the user), the signal is given to port B pin 0 and the fan is turned on. Similarly, as the temperature decreases below a certain temperature (As Set by the user), so the PTC Heating Element is turned on by giving signal to Port B Pin7. Now to display the temperature value we have calculated and stored in atmega32 is displayed through LCD16x2 which properly displays the value of temperature in degree Celsius and degree Fahrenheit. Furthermore, a message is also displayed along with the temperature like "optimal temperature" for the temperature range of Low temp value<t<High temp value, "Low temp: heat on" for temperature range t< Low temp value and lastly, "high temp: fan on" for temperature range t> Low temp value. A 1k potentiometer is also attached to the Power input and contrast pins of the lcd which serves the purpose of setting the proper contrast in the lcd display.
The high and low range values of the ambient temperature can be set by the user using the Enter Increase and Decrease buttons. User can also reload the values used in previous operation by pressing reload button and directly go to the temperature measurement or can initialize it if they wish to do so and set the values as per their preference. Also, if during temperature measurement if the user wishes to change the range values of temperature pressing enter would allow them to do so as well.
Lastly some hardware is also connected in the circuit like capacitor and resistor along the AREF input terminal and also in parallel with the PT100-RTD which serves the purpose of reducing the noise from the circuit by acting as a filter. Also, a button is connected to the reset pin in series with ground (as reset pin is active low) which is can be used as a hardware reset of the microcontroller.

3. Working of circuit-

3.1. Obtaining temperature values from ADC values-

Point 1 is connected to constant current source in circuit and point 3 is connected to ground

Let Voltage at point 1 be V', Voltage at point 2 be V", lead resistances be RL and RTD resistance be Rt (V1, V2 and i are as shown in diagram)



V’= i*(Rt+2RL)

V” = i*(RL)

V1 = V’-V” = i*(Rt+RL)

V2 = V”

V1-V2 = V’- 2*V” = i*(Rt)

As current is constant so this value is always directly proportional to PT100 resistance and it is directly proportional to temperatures with a degree increase in Celsius gives 0.385 increase in Ohms.

3.2. Selecting current source for PT100 RTD-

Resolution = 10 bits (1024 values range from 0 to 1023)

AVCC Voltage = 5 V (Range from 0 to 5 volts)

Hence, voltage change for changing 1 unit of ADC=

5V/1023 = 4.8875mV

So, if i/p voltage increase by 4.8875mv, the ADC value will increase by 1.

As we know, our resistance increases by 0.385 ohm per degree increase in temperature

=> 4.8875mV/0.385Ohm = 12.695mA (Approx. 12.7mA)

(This is the current source we require to excite the PT100-RTD)

Also, at 0 degree Celsius,

Voltage = 100Ohms*12.7mA = 1.27Volts

ADC value at this voltage = 1.27 Volts/4.8875mV=259.7

As 0.7 part will have no significance so to get temp value from ADC, we subtract this 259

So, in this way we get temp. value in Celsius to convert it into Fahrenheit we use

Temp far = (9/5) *Temp C + 32

3.3. LM317T as a constant current source-

As we required a constant current source so for that purpose this circuit is used it just needs a supply voltage to be connected at its VI Pin. Now we just need to decide the resistor value to get our desired current. The current supplied by it is
I out=1.25V/R

Where, R is the resistance connected between VO and ADJ Pins of the LM317T

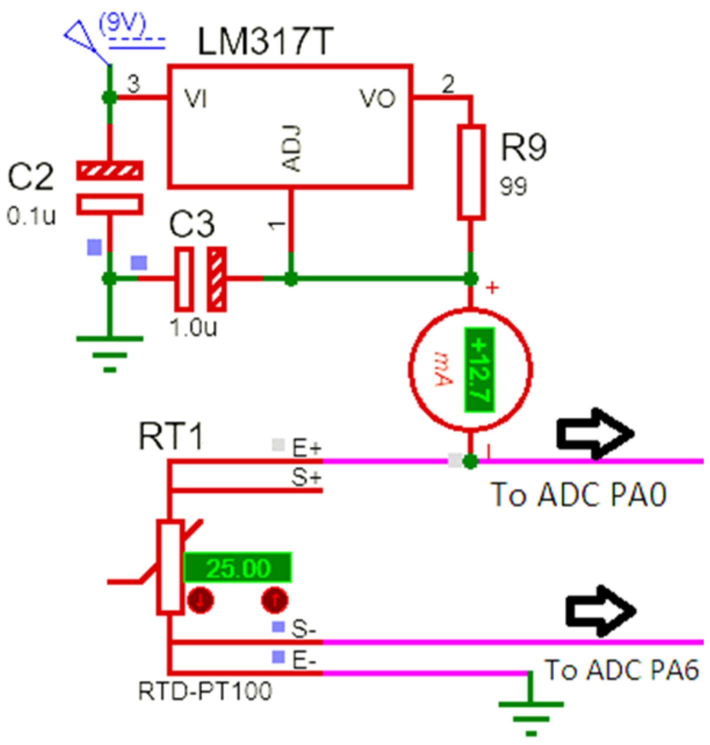
Now, as we require current =12.7mA so,

R=1.25/12.7mA

R=98.8Ohms

R=99 Ohms (Approx.) (56 and 43 connected in series)

And this branch is connected to PT100 RTD in order to supply constant current to it.

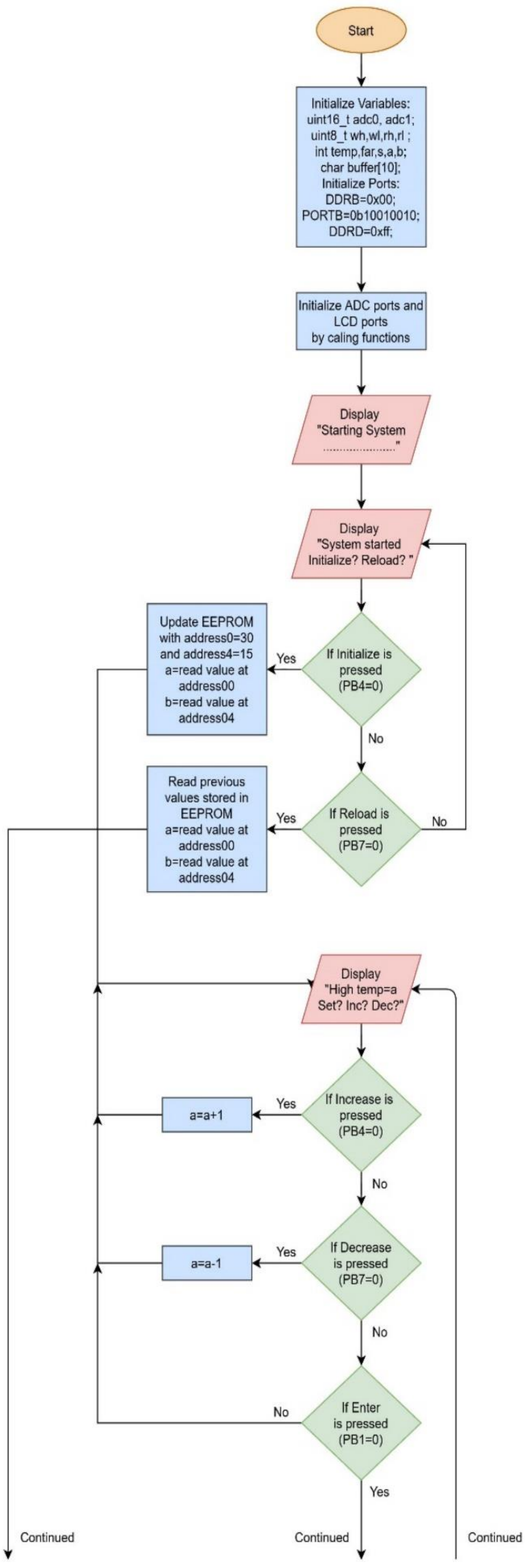


4. Program code along with Flow chart-

CODE:

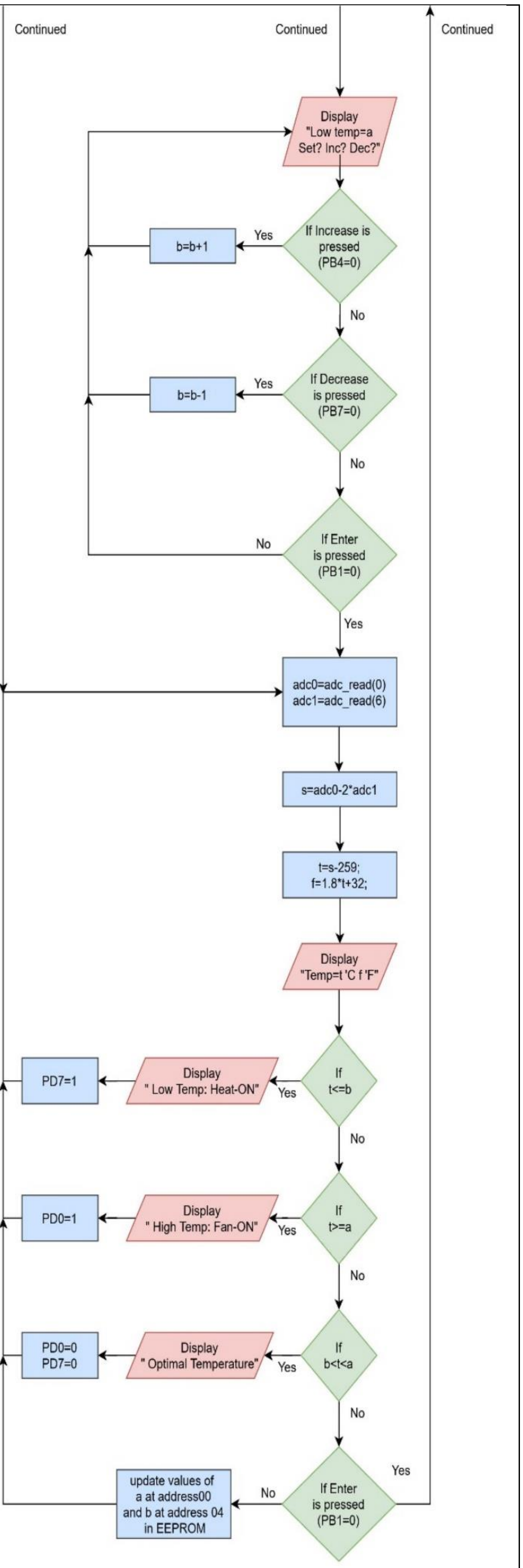
```c
#define F_CPU 1000000UL
#include <stdio.h>
#include <avr/io.h>
#include <avr/eeprom.h>
#include <util/delay.h>
#include "Lcdlbr/lcd.h"
void adc_init()
{
ADMUX = (1<<REFS0);
ADCSRA =
(1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(0<<ADPS0);
}
uint16_t adc_read(uint8_t ch)
{
ch &= 0b00000111;
ADMUX = (ADMUX & 0xF8)|ch;
ADCSRA |= (1<<ADSC);
while(ADCSRA & (1<<ADSC));
return (ADC);
}
int main()
{
DDRB=0x00;
PORTB=0b10010010;
DDRD=0xff;
uint16_t adc0, adc1;
uint8_t wh,wl,rh,rl ;
int temp,far,s,a,b;
char buffer[10];
adc_init();
lcd_init(LCD_DISP_ON_CURSOR);
lcd_clrscr();
lcd_gotoxy(0,0);
lcd_puts("Starting System");
lcd_gotoxy(0,1);
lcd_puts(".");
_delay_ms(100);
lcd_puts("..");
_delay_ms(100);
lcd_puts("...");
_delay_ms(100);
lcd_puts("....");
_delay_ms(100);
lcd_puts(".....");
_delay_ms(100);
while (1)
{
lcd_clrscr();
lcd_gotoxy(0,0);
```

FLOWCHART:



Start
Initialize Variables: uint16_t adc0, adc1; uint8_t wh,wl,rh,rl ; int temp,far,s,a,b; char buffer[10]; Initialize Ports: DDRB=0x00; PORTB=0b10010010; DDRD=0xff;
Initialize ADC ports and LCD ports by caling functions
Display "Starting System ...."
Display "System started Initialize? Reload? "
If Initialize is pressed (PB4=0)
Yes
Update EEPROM with address0=30 and address4=15 a=read value at address00 b=read value at address04
No
If Reload is pressed (PB7=0)
Yes
Read previous values stored in EEPROM a=read value at address00 b=read value at address04
No
Display "High temp=a Set? Inc? Dec?"
If Increase is pressed (PB4=0)
Yes
a=a+1
No
If Decrease is pressed (PB7=0)
Yes
a=a-1
No
If Enter is pressed (PB1=0)
No
Yes
Continued
Continued
Continued

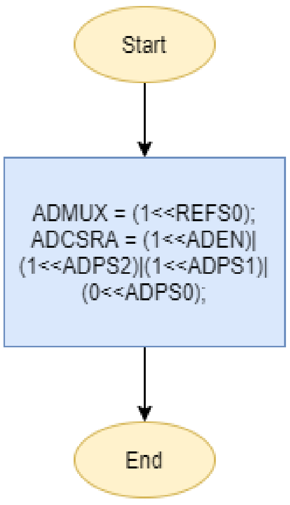
```
lcd_puts("System Started");
lcd_gotoxy(0,1);
lcd_puts("Initial?Reload?");
_delay_ms(300);
if((PINB & (1<<7))==0)//rel
{
rh = eeprom_read_byte((uint8_t*)0);
a=rh;
rl = eeprom_read_byte((uint8_t*)4);
b=rl;
_delay_ms(300);
goto measure;
}
if((PINB & (1<<4))==0)//ini
{
eeprom_update_byte ((uint8_t*) 0, 30);
rh = eeprom_read_byte((uint8_t*)0);
a=rh;
eeprom_update_byte ((uint8_t*) 4, 15);
rl = eeprom_read_byte((uint8_t*)4);
b=rl;
while(1)
{
high:
lcd_clrscr();
lcd_gotoxy(0,0);
lcd_puts("High temp=");
itoa(a,buffer,10);
lcd_puts(buffer);
lcd_gotoxy(12,0);
lcd_puts("'C");
lcd_gotoxy(0,1);
lcd_puts("Set? Inc? Dec?");
_delay_ms(300);
while((PINB & (1<<4))==0)
{
_delay_ms(300);
a++;
}
while((PINB & (1<<7))==0)
{
_delay_ms(300);
a--;
}
while((PINB & (1<<1))==0)//enter
{
while(1)
{
lcd_clrscr();
lcd_home();
```


Continued
Continued
Continued
Display "Low temp=a Set? Inc? Dec?"
If Increase is pressed (PB4=0)
Yes
b=b+1
No
If Decrease is pressed (PB7=0)
Yes
b=b-1
No
If Enter is pressed (PB1=0)
No
Yes
adc0=adc_read(0) adc1=adc_read(6)
s=adc0-2*adc1
t=s-259; f=1.8*t+32;
Display "Temp=t'C f'F"
If t<=b
Yes
Display " Low Temp: Heat-ON"
PD7=1
No
If t>=a
Yes
Display " High Temp: Fan-ON"
PD0=1
No
If b<t<a
Yes
Display " Optimal Temperature"
PD0=0 PD7=0
No
If Enter is pressed (PB1=0)
No
update values of a at address00 and b at address 04 in EEPROM
Yes

```
lcd_gotoxy(0,0);
lcd_puts("Low temp=");
itoa(b,buffer,10);
lcd_puts(buffer);
lcd_gotoxy(11,0);
lcd_puts("'C");
lcd_gotoxy(0,1);
lcd_puts("Set? Inc? Dec?");
_delay_ms(300);
while((PINB & (1<<4))==0)
{
_delay_ms(300);
b++;
}
while((PINB & (1<<7))==0)
{
_delay_ms(300);
b--;
}
while((PINB & (1<<1))==0)//enter
{
while(1)//3is
{
measure:
adc0 = adc_read(0);
adc1 = adc_read(6);
s = adc0 - 2 * adc1;
temp = s - 259;
far=(1.8*temp)+32;
lcd_gotoxy(0,0);
itoa(temp,buffer,10);
lcd_puts("Temp=");
lcd_puts(buffer);
lcd_gotoxy(7,0);
lcd_puts("'C");
lcd_gotoxy(11,0);
itoa(far,buffer,10);
lcd_puts(buffer);
lcd_gotoxy(14,0);
lcd_puts("'F");
_delay_ms(300);
if(temp>=a)
{
lcd_clrscr();
lcd_home();
lcd_gotoxy(0,1);
lcd_puts("High Temp:FAN ON ");
PORTD=(1<<PIND0);
}
else if(temp<=b)
```

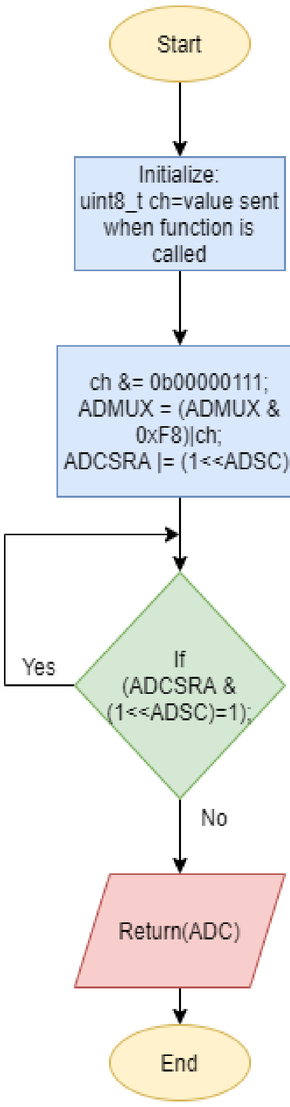
ADC INITIALIZATION FUNCTION

This function is called in our program to initialize the ADC channel when our system starts as ADC pins are used to measure the level of voltage from its pins and convert it into appropriate value. It is called by writing:
adc_init();
in the code.



ADC Read Function

It is the function used to read the value at the ADC pins of the microcontroller .
It us called by writing:
adcx=adc_read(x);
where:
adcx = unsigned integer in which the value from ADC pin is stored .
x=pin from which the ADC value is to be read .

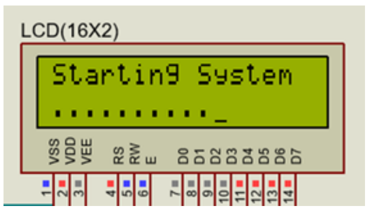


Start
Initialize:
uint8_t ch=value sent when function is called
ch &= 0b00000111;
ADMUX = (ADMUX & 0xF8)|ch;
ADCSRA |= (1<<ADSC);
If (ADCSRA & (1<<ADSC)=1);
Yes
No
Return(ADC)
End

```
{
lcd_clrscr();
lcd_home();
lcd_gotoxy(0,1);
lcd_puts("Low Temp:HEAT ON");
PORTD=(1<<PIND7);
}
else
{
lcd_clrscr();
lcd_home();
lcd_gotoxy(0,1);
lcd_puts("Optimal Temp");
PORTD=(0<<PIND0);
PORTD=(0<<PIND7);
}
if((PINB & (1<<1))==0)
{
_delay_ms(300);
goto high;
}
wh=a;
eeprom_update_byte ((uint8_t*) 0, wh);
wl=b;
eeprom_update_byte ((uint8_t*) 4, wl);
}//Temp measure
}//lt set
}//Low temp
}//ht set
}//High temp
}
}
}
```

5. Simulation-

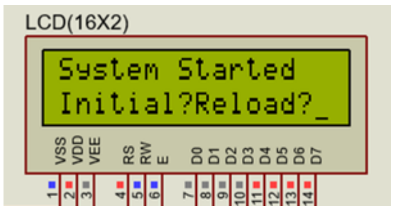
Step 1-



```
Starting System
..........._
```

Message displayed when system starts.

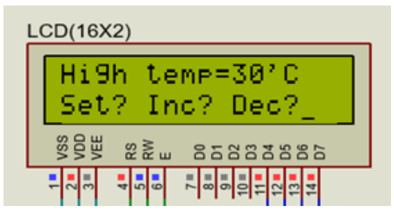
Step 2-



```
System Started
Initial?Reload?_
```

User has to press initialize or reload as per their requirement of using the system .

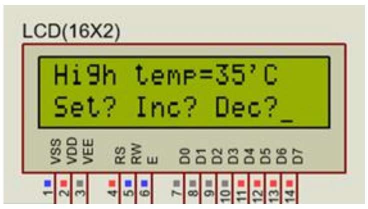
Step 3-



```
High temp=30’C
Set? Inc? Dec?_
```

If user presses initialize key this option would appear where they can increase or decrease the range value of high temperature.

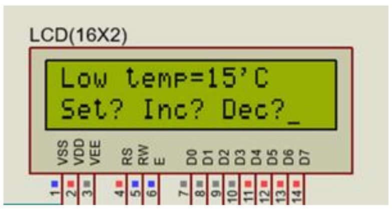
Step 4-



```
High temp=35’C
Set? Inc? Dec?_
```

Let us set this high value range as 35’C now as we press enter this value is saved for our system.

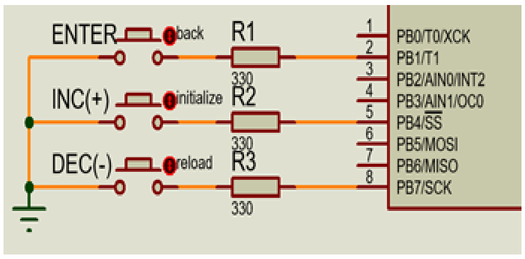
Step 5-



```
Low temp=15’C
Set? Inc? Dec?_
```

Similarly, here also we have the option of changing this value of low temperature for our system. when we press set for this value our system starts with low temp range as set

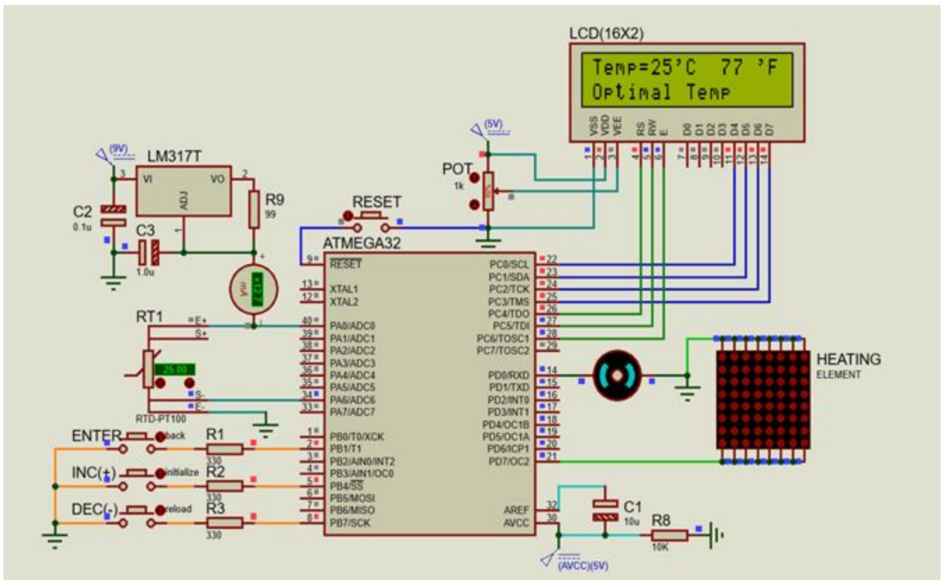
Operating Switches-



These are the keys used in the system to perform the functions of increasing or decreasing the range values, setting the temp, directly go to measuring temp with previously stored range values and even going back to setting temp while it is in continuous process of displaying the measured temperature.

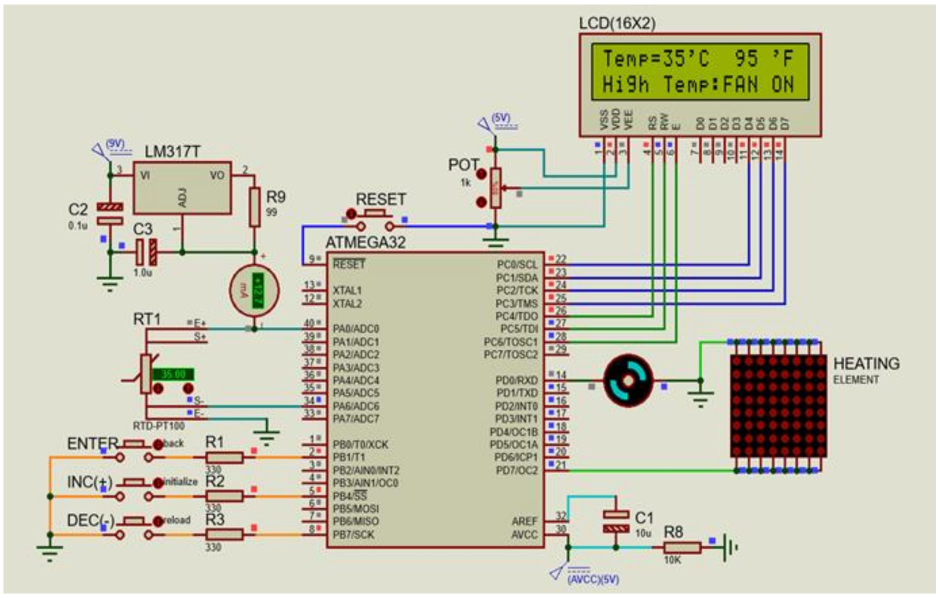
After both range values of temp are set the system goes to continuously measuring the ambient temperature and carry out its functions as shown below:

Case 1-



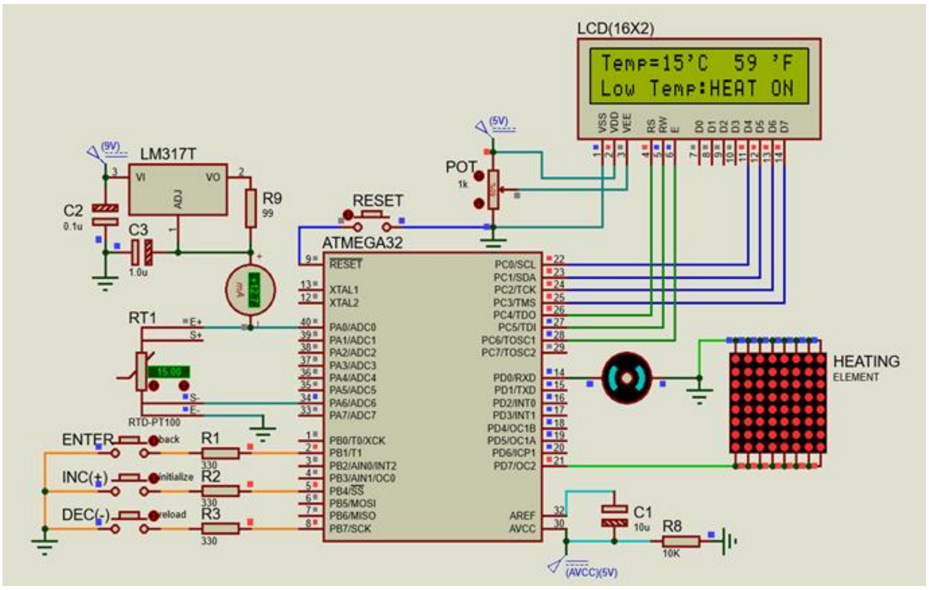
When ambient temperature is in between the set high and low range values of the temperature

Case 2-



When ambient temperature is higher than or equal to set high range value of temperature, the message is displayed and fan turns on

Case 3-



```
Temp=15'C  59 'F
Low Temp:HEAT ON
```

When ambient temperature is lower than or equal to set low range value of temperature, the message is displayed and heating turns on

6. Simulation Results-

So, through this simulation we were successfully able to check our circuit design and carry out the operation of our system. Code was perfectly executed through our simulation and the aim of our mini project was fulfilled.

7. Applications-

1. Within our homes, temperature sensors are used in many electrical appliances, from our refrigerators and freezers to help regulate and maintain cold temperatures as well as within stoves and ovens to ensure that they heat to the required levels for cooking, air confectioners/heaters. Even our common battery chargers use them, in order to prevent overcharging or undercharging based on measuring the battery's temperature.
2. Vehicles use temperature sensors within their radiator. This is important because they warn you when the water circulating through your automotive engine is approaching a dangerously high temperature, which could potentially cause the engine to break if exceeded, also climate control inside the car. By automatically adjusting parameters based on temperature, this is effectively avoided without putting the driver at risk.
3. HVAC systems require the measuring of temperatures so that it can help provide the optimal temperature to a room or building. Air conditioning is used in almost every home and office, so temperature sensors are essential for these units and systems. They can also be used to detect leaks by spotting unexpected anomalies in temperature.
4. Temperature Control Systems have saved the day in many places where they are being used. Electrical Fires have been minimized because these Control Systems turn off heating and cooling units when they are not necessarily in use, thus preserving lives and property.
5. Temperature Control Systems are economically important as they have the duty of ensuring that the energy supply reaching the house/office/industry is economized and

managed as much as possible. This it achieves by making sure that the heating and cooling units are only functioning when they are needed. The Control System further ensures that the energy bills that would be paid by the company or individual are "efficient", because the bill would only cover energy that was efficiently utilized.

6. In Homes and Industries alike, much wastage of useful resource has been prevented as a result of the use of Temperature Control Systems. These Systems efficiently manage the temperature of storage areas, thus keeping stored items at temperatures that extend their value period.

8. Conclusion-

In this project, microcontroller-based temperature measurement and controlling system has been designed which contains few basic elements having couple of lines control code using assembler/C. This system measures temperature using PT100-RTD as temperature sensor device and compares the results so that required optimal temperature can be maintained automatically as per the surrounding temperature.

9. References-

1. https://www.electronicwings.com/avr-atmega/atmega1632-adc
2. https://exploreembedded.com/wiki/Permanent_Data_Storage_with_AVR_Internal_EEPROM
3. https://www.omega.com/en-us/resources/rtd-2-3-4-wire-connections
4. https://embeddedcenter.wordpress.com/ece-study-centre/display-module/lcd-16x2-lm016l/

10. Bibliography-

1. The AVR Microcontroller
2. Beginners Programming in AVR Assembler
3. Practical AVR Microcontrollers