PROJECT REPORT

TOPIC: DIGITAL THERMOMETER USING LM35

ABSTRACT:

Thermometers are widely used to measure temperature of human body and it is a commonly used device nowadays. This project involves a simple working of digital thermometer by using LM35 sensor. Future work of this project involves advancing by including sensors like DHT11/22 which gives humidity values which can be used in agricultural automations like measuring soil moisture, humidity, etc.,

COMPONENTS USED:

- Atmega328p
- LM35 sensor
- Lcd display (Hd44780)

ATMEGA328P :

- The ATmega328 is a single-chip microcontroller created by Atmel in the megaAVR .
- It has a modified Harvard architecture 8-bit RISC processor core.

LM35 SENSOR:

The LM35 is one kind of commonly used temperature sensor that can be used to measure temperature with an electrical o/p comparative to the temperature (in °C). It can measure temperature more correctly compare with a thermistor.

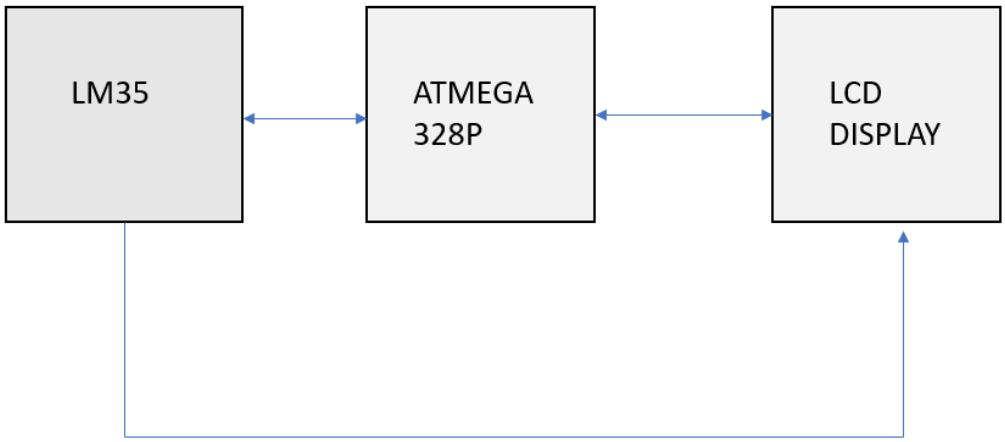
LCD DISPLAY (HD44780):

A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. The 16 x 2 intelligent alphanumeric dot matrix display is capable of displaying 224 different characters and symbols.

SOFTWARE USED:

- Simulide
- Visual Studio Code

BLOCK DIAGRAM:



WORKING:

- Here a mimic model of LM35 is used to measure the temperature.
- When the circuit is ON the display shows the name of the person which is passed in the form of string and the temperature is displayed.
- Only partial output is derived.

OUTPUT:

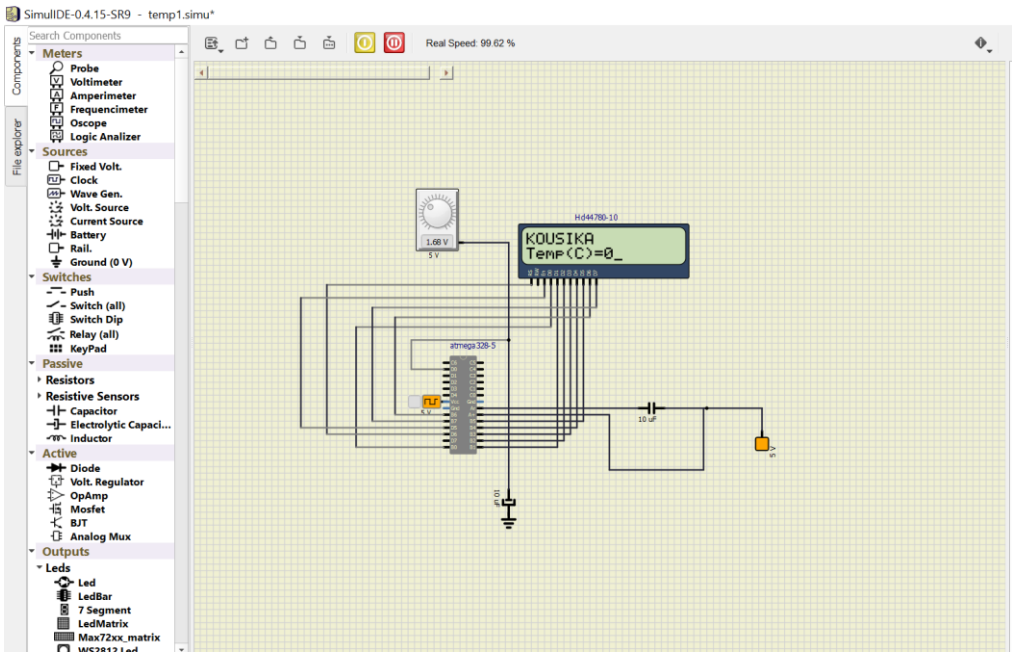
CODE:

```
#ifndef __AVR_ATmega328P__
  #define __AVR_ATmega328P__
#endif

#include <avr/io.h>
#define F_CPU 1000000
#include <util/delay.h>
#include <stdlib.h>

#define enable          5
#define registerselection 6
#define DDRA _SFR_IO8(0x01)

void send_a_command(unsigned char command);
void send_a_character(unsigned char character);
void send_a_string(char *string_of_characters);
```

```
int main(void)
{
   DDRB = 0xFF;
   DDRA = 0;
   DDRD = 0xFF;
   _delay_ms(50);

   ADMUX |=(1<<REFS0)|(1<<REFS1);
   ADCSRA |=(1<<ADEN)|(1<<ADATE)|(1<<ADPS0)|(1<<ADPS1)|(1<<ADPS2);

   int16_t COUNTA = 0;
   char SHOWA [3];


   send_a_command(0x01);                                  //Clear Screen 0x01 = 00000001
   _delay_ms(50);
   send_a_command(0x38);
   _delay_ms(50);
   send_a_command(0b00001111);
   _delay_ms(50);

   ADCSRA |=(1<<ADSC);
   while(1)
   {
      COUNTA = ADC/4;
      send_a_string ("KOUSIKA");
      send_a_command(0x80 + 0x40 + 0);
      send_a_string ("Temp(C)=");
      send_a_command(0x80 + 0x40 + 8);
      itoa(COUNTA,SHOWA,10);
      send_a_string(SHOWA);
      send_a_string ("     ");
      send_a_command(0x80 + 0);

   }
}

void send_a_command(unsigned char command)
{
   PORTB = command;
   PORTD &= ~ (1<<registerselection);
   PORTD |= 1<<enable;
   _delay_ms(20);
   PORTD &= ~1<<enable;
   PORTB = 0;
}

void send_a_character(unsigned char character)
{
   PORTB = character;
   PORTD |= 1<<registerselection;
```

```
    PORTD |= 1<<enable;
    _delay_ms(20);
    PORTD &= ~1<<enable;
    PORTB = 0;
}
void send_a_string(char *string_of_characters)
{
    while (*string_of_characters > 0)
    {
        send_a_character(*string_of_characters++);
    }
}
```

OUTPUT AFTER LOADING FIRMWARE (PARTIAL OUTPUT):



CONCLUSION:

- Thus the concept of embedded systems and their working is understood by constructing the circuit by using simulide and coding was done based on the requirement of the project and partial output is derived.