EMBEDDED SYSTEMS LAB PROJECT

INTERFACING A TEMPERATURE SENSOR WITH LP1768 MICROCONTROLLER

PROJECT BY:

MAHEK MISHRA – 190911110

SAVIN ANAND – 190911089

UTKARSH MISRA – 190911102

TEAM DETAILS:

SECTION-B

LAB BATCH-4

GROUP-3

INTRODUCTION

Temperature is one of the basic properties of nature. It is a physical quantity that expresses hot and cold and is the manifestation of thermal energy present in all matter. There are various different temperature sensors that have been used over the years like negative Temperature Coefficient (NTC) thermistor, resistance Temperature Detector (RTD), thermocouples, semiconductor-based temperature sensors just to name a few.

In this project, we have interfaced a LM35 temperature sensor with a LPC1768 ARM microcontroller.

PROBLEM STATEMENT

Write a program to interface a temperature sensor to LPC1768 and display the temperature on LCD.

AIM OF THE PROJECT

1) To discern the software development for ARM cortex microcontroller using embedded C language.

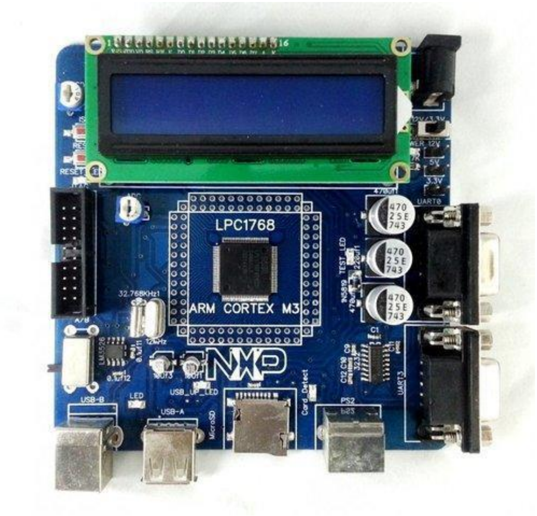
2) To design real world systems using ARM cortex-M embedded system.

3) To understand various interfacing circuits necessary for various applications and programming using ARM.

HARDWARE COMPONENTS USED

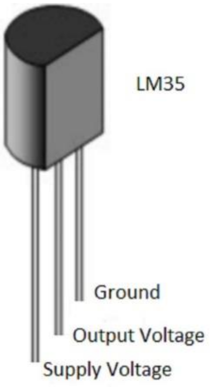
LPC1768

The LPC1768 is a Cortex®-M3 microcontroller for embedded applications featuring a high level of integration and low power consumption. The code has been written in Embedded C and tested out on an ARM7 Cortex M3 evaluation board.



LM35 temperature sensor

LM35 is a temperature measuring device having an analog output voltage linearly proportional to the temperature. It provides output voltage in Centigrade (Celsius) and does not require any external calibration circuitry. The sensitivity of LM35 is 10 mV/degree Celsius.

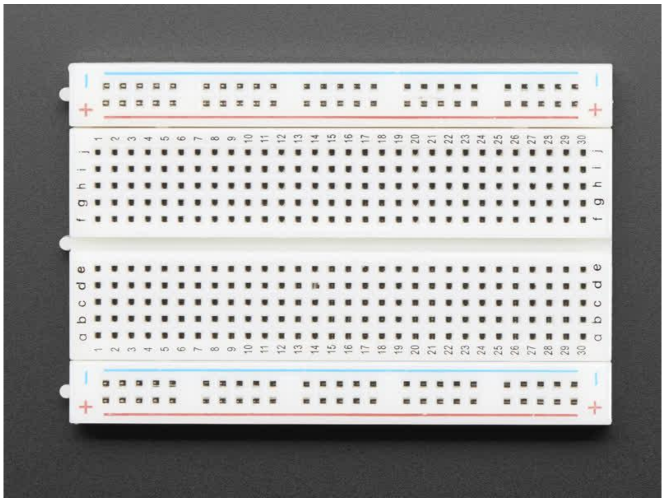


Zener Diode

A Zener diode is a heavily doped semiconductor device that is designed to operate in the reverse direction.

Breadboard

A thin plastic board used to hold electronic components (transistors, resistors, chips, etc.) that are wired together. Most electronic components in electronic circuits can be interconnected by inserting their leads or terminals into the holes and then making connections through wires where appropriate.



FLOWCHART AND EXPLANATION

Steps to get a temperature reading from the LM35 sensor:

Step 1
Configure pins, port pin 0.1-0.7 (configuring LCD 4-bit mode), p0.8 as Rs pin and p0.9 as enable pin.

Step 2
Configure ADC Registers

Step 3
Load addr0 register value into temp

Step 4

Extract value by loading temp with 0xfff0 and right shift by 4

Step 5

In final (a float variable) store the value of temperature after dividing it by 12.41

Step 6

Convert the final value to string and store into variable temper

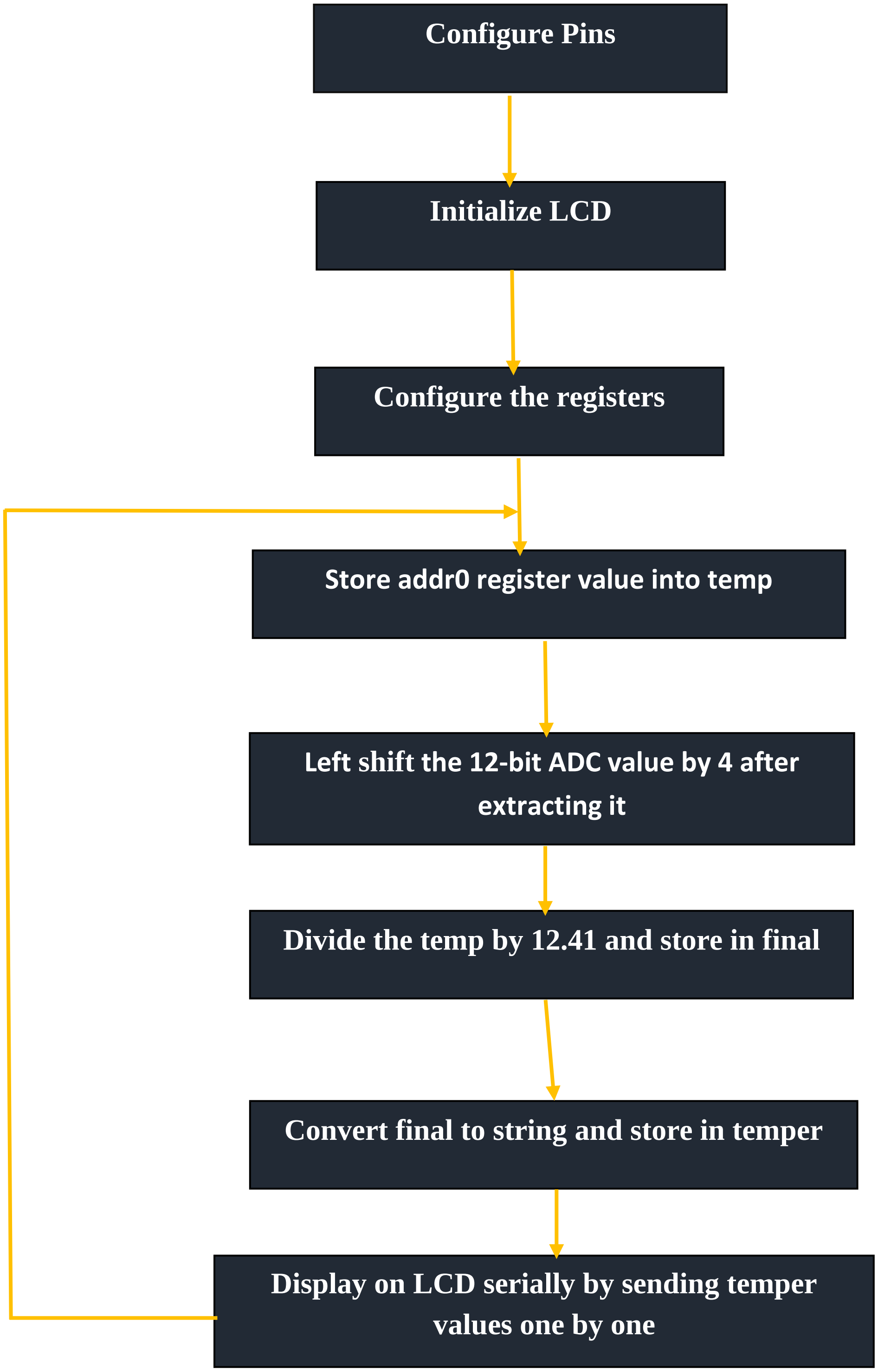
Step 7

Display string on LCD serially

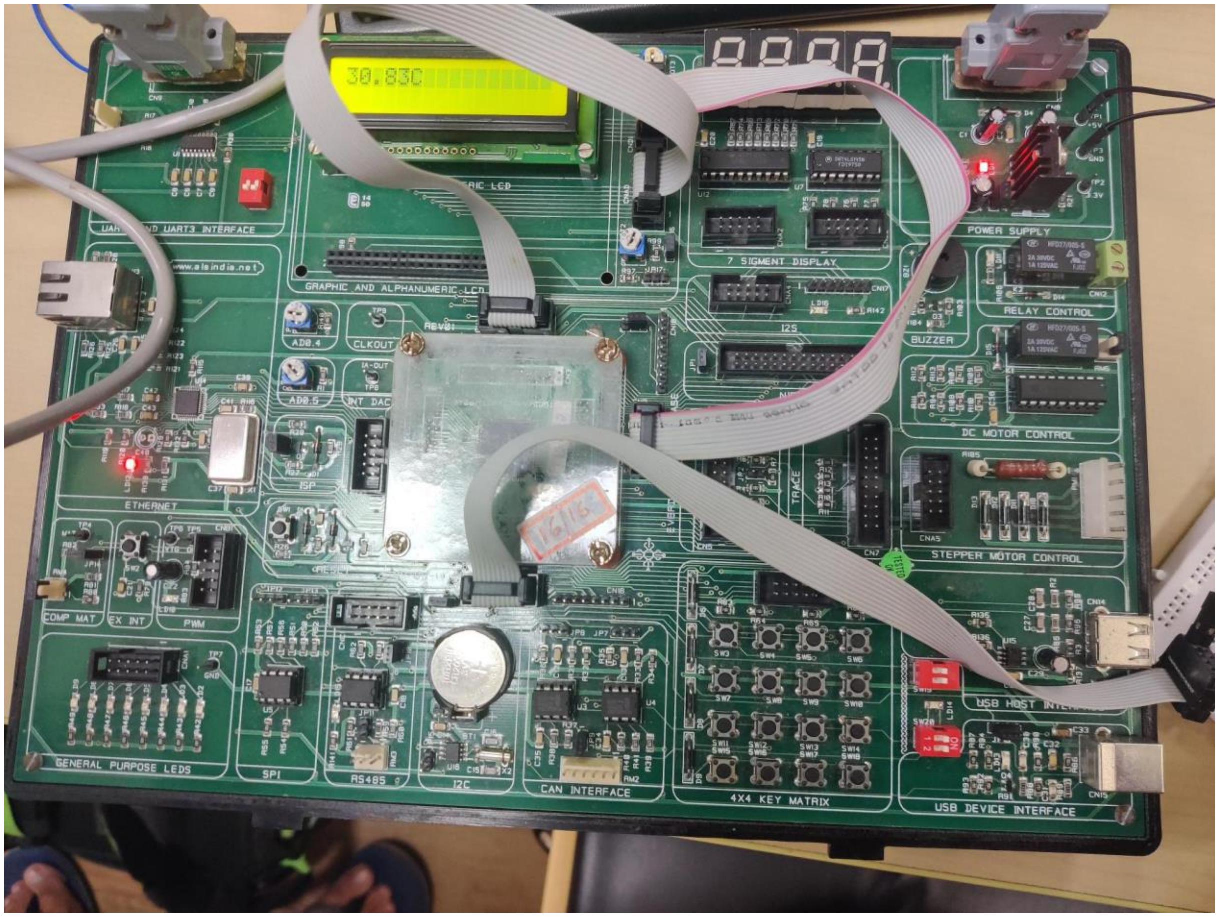
Step Size of ADC = 3.3 / (2^12) = 0.0806mV

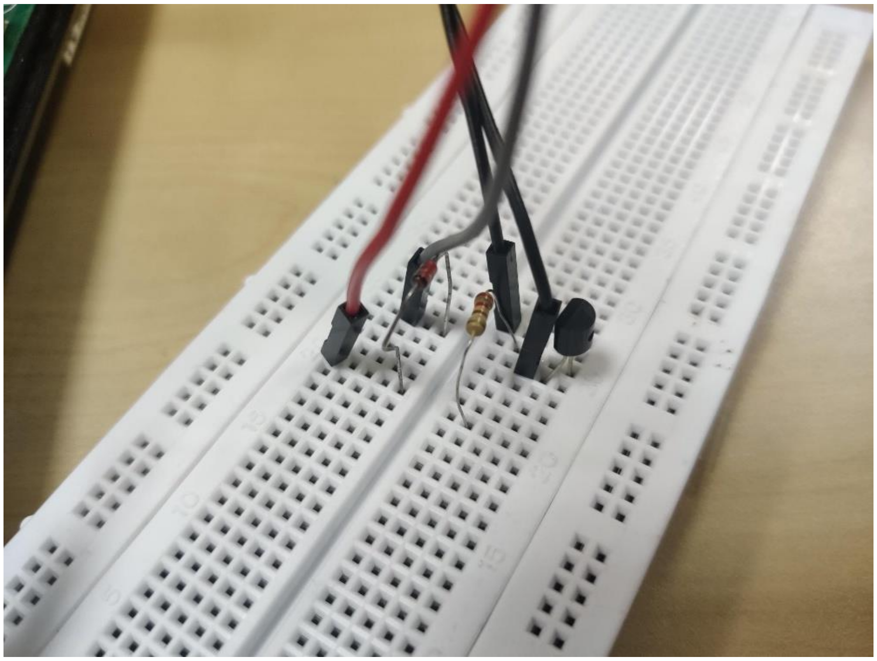
For every degree Celsius, LM35 provides 10mV of change.

Hence, ADC Value is divided by 12.41 to get Temperature.

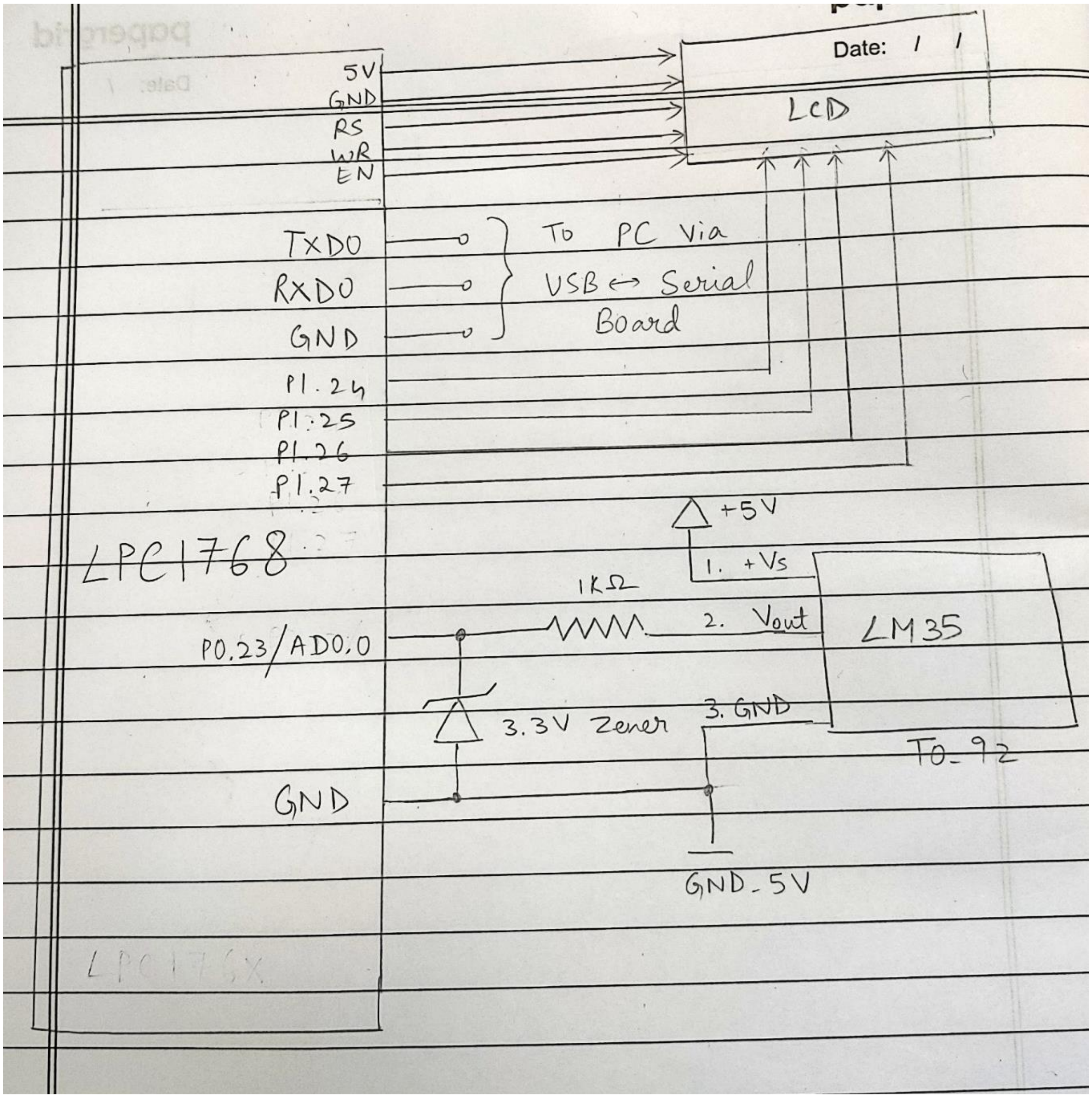

Configure Pins
Initialize LCD
Configure the registers
Store addr0 register value into temp
Left shift the 12-bit ADC value by 4 after extracting it
Divide the temp by 12.41 and store in final
Convert final to string and store in temper
Display on LCD serially by sending temper values one by one

RESULT


30.83C
UART2 AND UART3 INTERFACE
GRAPHIC AND ALPHANUMERIC LCD
7 SEGMENT DISPLAY
POWER SUPPLY
RELAY CONTROL
BUZZER
DC MOTOR CONTROL
STEPPER MOTOR CONTROL
ETHERNET
COMP MAT
EX INT
PWM
GENERAL PURPOSE LEDS
SPI
RS485
I2C
CAN INTERFACE
4X4 KEY MATRIX
USB HOST INTERFACE
USB DEVICE INTERFACE
www.alsindia.net



LM35 INTERFACING SCHEMATIC



CODE

```
#include <LPC17xx.h>
#include <stdio.h>

#define rsctrl 0x00000100       // p0.8
#define enctrl 0x00000200       // p0.9 the lcd enable
#define dtctrl 0x000000f0       // p0.4-7 4 bit mode

#define ADC_CHANNEL_SEL (1<<0)  // The channel selecting on ADC
#define ADC_ENABLE (1<<21)      // The ADC component Enabling
#define ADC_START (1<<24)       // The ADC is loading into software mode
#define ADC_DONE (1U<<31)       // defining it as compiler will throw #61-D warning or an unsigned
value

void lcd_init();
void wr_cn();
void clr_disp();
void delay();
void lcd_com();
void wr_dn();
void lcd_data();
void clear_ports();
void lcd_puts(unsigned char *);

unsigned int i, j, temp, temp1=0, temp2=0;
float final;
char temper[20];
unsigned long LED = 0x00000010;

int main(void)
{
    SystemInit();
    SystemCoreClockUpdate();

    LPC_PINCON->PINSEL1 |= (1 << 14);

    while (1) {
        LPC_ADC->ADCR = ADC_CHANNEL_SEL | ADC_ENABLE | ADC_START;

        while (((temp = LPC_ADC->ADDR0) & ADC_DONE) == 0);
        temp = LPC_ADC->ADDR0;
        temp &= 0xFFF0;
        temp >>= 4;
        final = (float)(temp / 12.41);
        sprintf(temper, "%3.2fC", final);

        lcd_init();             //lcd_init function called
        temp1 = 0x80;

        lcd_com();           //lcd_com function called
        delay(800);          //delay function called

        lcd_puts( & temper[0]);
    }

}

void lcd_init() {
    LPC_PINCON->PINSEL0 = 0;
    LPC_GPIO0->FIODIR |= dtctrl;
    LPC_GPIO0->FIODIR |= rsctrl;
    LPC_GPIO0->FIODIR |= enctrl;
    clear_ports();            //clear_ports function called
    delay(3200);             //delay function called
    for (i = 0; i < 3; i++) {
        temp2 = (0x30);
        wr_cn();            //wr_cn function called
        delay(30000);        //delay function called
    }

    temp2 = (0x20);
    wr_cn();                //wr_cn function called
    delay(30000);            //delay function called

    temp1 = 0x28;
    lcd_com();                //lcd_com function called
    delay(30000);            //delay function called

    temp1 = 0x0c;
    lcd_com();                //lcd_com function called
    delay(800);            //delay function called

    temp1 = 0x06;
    lcd_com();                //lcd_com function called
    delay(800);            //delay function called

    temp1 = 0x01;
    lcd_com();               //lcd_com function called
    delay(10000);            //delay function called

    temp1 = 0x80;
    lcd_com();               //lcd_com function called
    delay(800);             //delay function called
}

void lcd_com() {
    temp2 = temp1 & 0xf0;
    temp2 = temp2;
    wr_cn();                //wr_cn function called

    temp2 = temp1 & 0x0f;
    temp2 = temp2 << 4;
    wr_cn();                //wr_cn function called

    delay(1000);            //delay function called
}

void wr_cn() {
    clear_ports();           //clear_ports function called
    LPC_GPIO0->FIOPIN = temp2;
    LPC_GPIO0->FIOCLR = rsctrl;
    LPC_GPIO0->FIOSET = enctrl;
    delay(25);               //delay function called
    LPC_GPIO0->FIOCLR = enctrl;
}

void lcd_data() {
    temp2 = temp1 & 0xf0;
    temp2 = temp2;
    wr_dn();                //wr_dn function called
    temp2 = temp1 & 0x0f;
    temp2 = temp2 << 4;
    wr_dn();                //wr_dn function called
    delay(1000);            //delay function called
}

void wr_dn() {
    clear_ports();           //clear_ports function called
    LPC_GPIO0->FIOPIN = temp2;
    LPC_GPIO0->FIOSET = rsctrl;
    LPC_GPIO0->FIOSET = enctrl;
    delay(25);               //delay function called
    LPC_GPIO0->FIOCLR = enctrl;
}

void delay(unsigned int r1) {
    unsigned int r;
    for (r = 0; r < r1; r++);
}

void clr_disp() {
    temp1 = 0x01;
    lcd_com();                  //lcd_com function called
    delay(10000);               //delay function called
}

void clear_ports() {
    LPC_GPIO0->FIOCLR |= rsctrl;
    LPC_GPIO0->FIOCLR |= enctrl;
    LPC_GPIO0->FIOCLR |= dtctrl;
}

void lcd_puts(unsigned char * buff) {
    unsigned int i = 0;
    while (buff[i] != '\0') {
        temp1 = buff[i];
        i++;
        lcd_data();             //lcd_data function called
    }
}
```