

# Design and Implementation of Microcontroller Based Temperature Data Logging System

Vinay Divakar

**Abstract**— The term data logger (also sometimes referred to as a data recorder) is commonly used to describe a self-contained, standalone data acquisition system or device. These products are comprised of a number of analog and digital inputs that are monitored, and the results or conditions of these inputs is then stored on some type of local memory. In this paper, a Temperature Data Logging System is designed to record and display temperatures continuously. It uses a temperature sensor to sense the surrounding temperature and displays the temperature on the PC with the help of LPC 2148 ARM7 TDMI processor and the UART (Universal Asynchronous Receiver/Transmitter). The methodology for designing the temperature data logging system is discussed in detail. The code to implement the functionality of the logging system is modeled and simulated using the Software Tool Keilµ4. The characteristics of LM 35 temperature sensor and the hardware required to interface the output of LM 35 to the LPC 2148 is discussed in detail.

**Index Terms**— Temperature data logger, data logger, temperature recorder, temperature display, lm35 sensor, UART, ARM TDMI.

## I. INTRODUCTION

Temperature data loggers are devices that are used for retrieving and storing temperature values over a specific period of time.

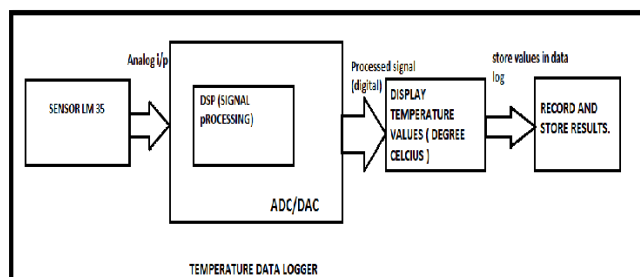


Fig. 1 Temperature Data Logging System

It will have the records of the data sensed and retrieved. Temperature data loggers generally make use of the lookup tables that consists the recorded data of the different temperatures that was continuously displayed and stored, this recorded data can be analyzed by the user. Temperature sensors used for sensing temperatures are LM 35 or

thermistors. The Fig (1) shows the block diagram of the working of a temperature data logger.

Initially the sensor senses the temperature and inputs analog signal to the ADC, which converts it into a digital values and are displayed and finally the sensed temperature values are recorded and stored in the memory in form of a log for further analysis. Temperature data loggers are used in fields, where it's important to keep a track and monitor temperatures over a definite period of time. Some of its applications are, it's used in refrigerators', freezers for medical and food storage, incubators, server rooms etc. LM35 is a temperature sensor designed by the national semiconductors. The output voltage generated by the LM35 sensor varies linearly and proportionally with the temperature and is very easy to predict the output voltage corresponding to the change in temperature. Some of the advantages of using LM35 sensor are that, it can measure temperatures more accurately than thermistors and the LM 35 can generate output voltages higher than that of thermocouples and generally does not require amplification of the output voltage.

## II. LM 35 AND UART WORKING PRINCIPLE

The LM 35 is a temperature sensor used to sense temperature and give an output in terms of voltage. LM35 is similar to a thermometer, but it's just that the LM 35 is very minute in size, like a thermometer in a chip, And instead of mercury like thermometer, the LM 35 relies on the property of diodes that causes the break down voltage of the diode to increase linearly with the increase in temperature. The temperature range of the LM 35 is from 55 °C to 150 °C. It has a sensitivity of 10mV/°C. At room temperature the LM 35 has accuracy of 0.25 °C and +/-0.75 °C over the full temperature range. The LM 35 can operate for voltages between 4V to 30Volts. The LM 35 sensor has three terminals, the first terminal is connected to 5 Volts Vcc supply and the third terminal is grounded. The centre terminal is terminal two and this is the output terminal Vout that gives the output voltage corresponding to the sensed temperature in analog form.

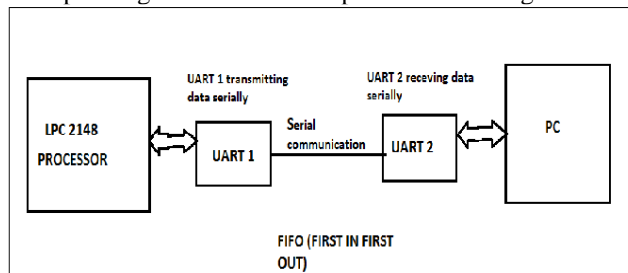


Fig. 2. UART Communication

Manuscript published on 30 December 2013.

\* Correspondence Author (s)

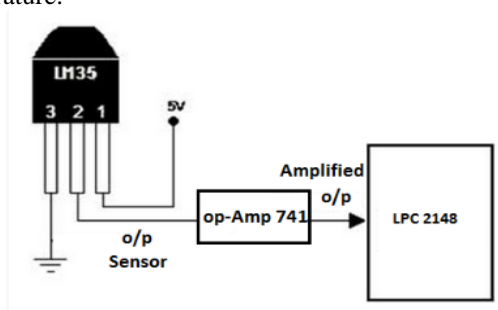
Mr. Vinay Divakar\*, is currently pursuing M.sc (Engg) degree program in Electronic System Design Engineering in M.S Ramaiah School of Advanced Studies Affiliated to Coventry University, U.K.

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC-BY-NC-ND license <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

The output signal generated by the LM 35 is then further processed for displaying. A UART is a universal asynchronous receiver/transmitter that performs the task of serial communications between the device and Computer as shown in the Fig (2). UART changes the incoming parallel bits to serial bits (data), which is transmitted serially through a communication line and the second UART is used at the receiver end that converts the serially receiving data back into parallel bits for the PC to read. The UART performs all the tasks such as timing, parity checking etc. needed for communication. For programming UART in different environments, registers are accessible to set and receive the communication parameters such as speed, parity check etc.

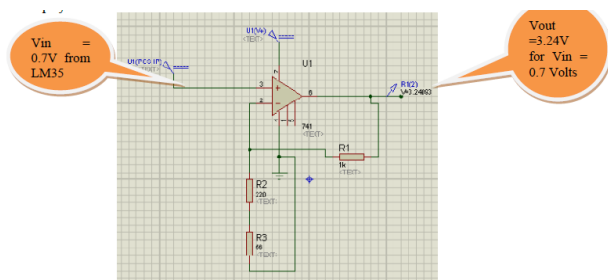
### III. INTERFACING OF LM35 TEMPERATURE SENSOR TO LPC2148

LM35 is a transducer or temperature sensor that converts heat energy into electrical energy i.e. it senses the temperature and gives an output voltage corresponding to the sensed temperature.



**Fig 3a. Interfacing LM 35 Sensor**

The Fig (3a) shows the hardware interfacing of LM35 sensor to the LPC2148 ARM 7 TDMI board. The output of the LM35 sensor is connected to the port P0.30, that is pin



**Fig. 3b Non-Inverting Amplifier**

AD0.3, through a non –inverting or inverting operational Amplifier in order to amplify the output voltage of the LM35, before passing it to the AD0.3(ADC) pin of the LPC2148.

The output of the sensor goes to the input of the op-Amp 741, which generates an amplified voltage as input to the LPC 2148, which further processes the signal to a form that it can be displayed. The Fig (3b) show's a non-inverting op-Amp whose input voltage ( $V_{in}$ ) at pin 3 is from the output terminal of the LM35. For an input of 0.7V corresponding to the maximum temperature of 70 °C (sensitivity of LM35 is 0.1V/°C) the output voltage ( $V_{out}$ ) from the op-Amp i.e. Pin 6 is 3.24V, which is the amplified voltage input to AD0.3 Pin of the LPC2148. Generally the resistance of the feedback resistor R1 is 1k, supply voltage to pin 7 is 5 to 12V and Pin 4 is grounded. The calculations to find resistance R2 is shown below,

$$\text{Gain } G = V_{out}/V_{in} \\ = 3.3/0.7$$

$$= 4.7 \\ G = 1 + R1/R2 \\ 4.7 = 1 + 1/R2 \quad (G = 4.7) \\ R2 = 270\Omega$$

Therefore, to get an output voltage 3.3V, for an input of 0.7V, we use a R2 of resistance 270Ω, since there is no standard resistance of 270, hence we use two resistors connected in series i.e. R2 = 220Ω and R1 = 56Ω, to obtain a resistance of approximately 270Ω.

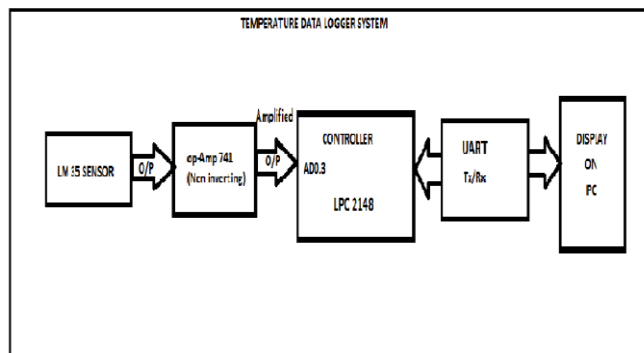
**//Code for ADC (AD0.3)//**

```
#include<LPC21xx.h>
void delay(void)
{
    int m;
    for(m=0;m<10000000;m++)
    }
int main(void)
{
    unsigned int result,result1;
    PINSEL1=0x10000000;
    ADOCR=0x00270608;
    IODIR0=0x000000ff;
    while(1)
    {
        do
        {
            result=ADOGDR;
        }
        while((result&0x80000000)==0)
        result1=((result>>6)&0x000003ff);
        IOSET0=result1;
        delay();
        IOCLR0=result1;
    }
}
```

The code is used to select the AD0.3 pin and then convert the analog input from the op-Amp to a digital form, which is displayed by means of serial communication using UART (Universal Asynchronous Receiver/Transmitter).

### IV. BLOCK DIAGRAM OF THE TEMPERATURE LOGGING SYSTEM

The temperature sensor used is LM35, an external hardware circuit is used to amplify the milli volts (mV) output to Volts (V), which is input to the ADC pin AD0.3 of the LPC2148, which displays the output volts in terms of temperature on



**Fig 4. Temperature Logging System Block Diagram**

the PC through UART. The Fig (4) shows the detailed block diagram of the whole system. The LM35 senses the temperature and generates an analog output signal as low as in millivolts (mV), which goes as input to the non-inverting input terminal of the op-AMP. The output of the op-Amp is the amplified voltage of the output generated by the LM35. The amplified output voltage from the op-Amp is the input to the Port P0.30/AD0.3 (pin) i.e. ADC (Analog to digital converter), which converts the output analog signal from the op-Amp to a digital form and generates discrete values that is displayed as hexadecimal. The selection of the Pins can be done through coding techniques. The output of the ADC is displayed on the PC through the UART that is connects the LPC2148 and the PC. The serial communication between the LPC2148 and the PC can be achieved by coding techniques.

//Code for serial communication using UART//

```
#include<LPC214x.h>
#include<string.h>
int main(void)
{
    int i,l;
    char c;
    char str1[]="Enter the character" ;
    char str2[]="Character entered is:";
    PINSEL0=0x00050000;
    U1LCR=0x03;
    U1DLL=0xC2;
    U1LCR=0x83;
    l= strlen (str1);
    for(i=0;i<l;i++)
    {
        while(!(U1LSR & 0x20))
        {
            U1THR=str1[i];
        }
        while(!(U1LSR & 0x01))
        {
            c=U1RBR;
        }
        l= strlen(str2);
        for(i=0;i<l;i++)
        {
            while(!(U1LSR & 0x20));
            U1THR=str2[i];
        }
        while(!(U1LSR & 0x20))
        {
            U1THR=c;
        }
    }
}
```

The above code is used for serial communication between the controller LPC2148 and the PC through the UART. The code can be used to check the functionality of the UART, such as serial communication i.e. receiving and transmission of data. The functionality of the code for the working of UART can be done using software tools such as keilµ3 or keilµ4. In Figure 2.4, LPC 2148 is the ARM V7 TDMI processor that is the controller that does the processing of the signal and the real time implementation of the code that is dumped within it.

## V. MODELING AND SIMULATION ON THE SYSTEM

Before implementing a system in hardware it is necessary to develop the software such as a code or program which will control the functionality of the hardware such as LPC 2148. Once the code for the desired functionality is developed, then the code is simulated to check whether it is meeting the desired functionality. Simulation of the code is done using software tools such as keilµ4.

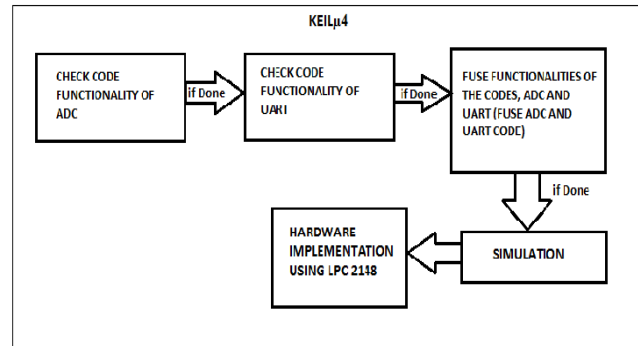


Fig. 5a System Modelling

The Fig (5a) shows methodology of modeling and simulating the temperature data logger system. Initially a code for ADC selecting the port p0.30/AD0.3 is developed with the desired functionality, and then the code for serial communication is developed for UART with proper functionality. Then the developed ADC and UART code is fused to form a code that converts analog input to digital, for serial communication and displaying. Once the fused code with the desired functionality is developed, then the code is simulated to analyze its performance and operation. The developed fused code is implemented and simulated using the software keil, if the functionality of code meets the desired requirement and if the simulation is successful, then the code is implemented on the hardware that is LPC 2148.

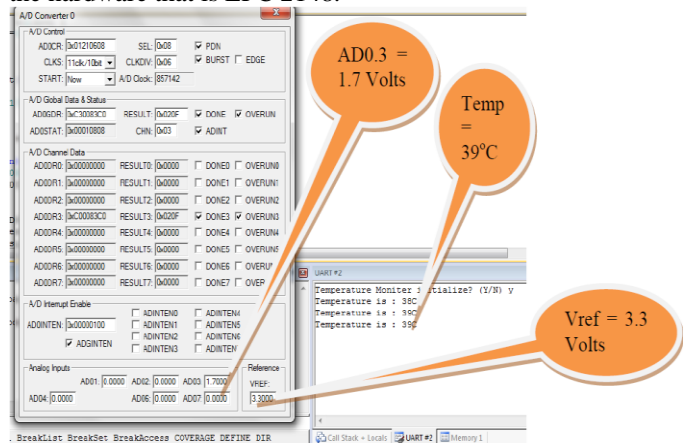
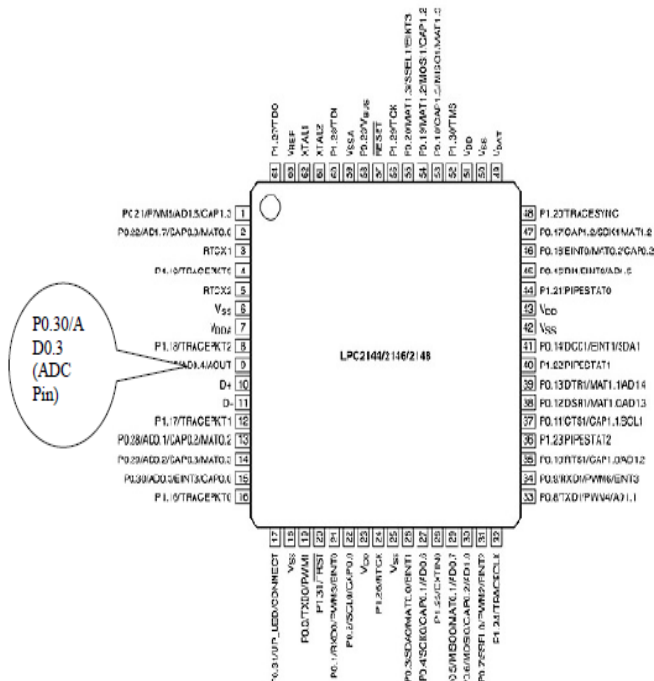


Fig. 5b ADC and UART functions

The Fig (5b) shows the functionality of the code developed for the temperature data logging system. It is analyzed in the keil software. In the Figure it is seen that, VREF is the maximum voltage (3.3V) and for a input voltage of 1.7V at the AD0.3, the output temperature displayed through UART2 is 39 °C. Once the desired results are obtained, then next phase is the hardware implementation of the code using the LPC 2148. The detailed code and hardware implementation will be discussed further.

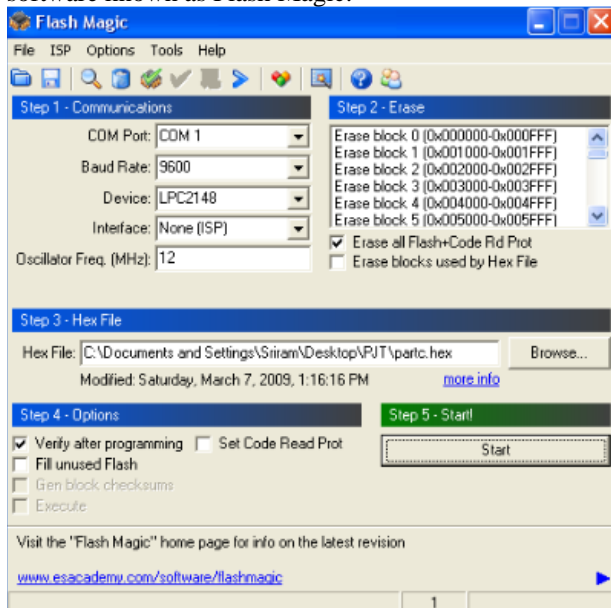
## VI. HARDWARE IMPLEMENTATION OF TEMPERATURE DATA LOGGING SYSTEM USING LPC2148

Once the simulation and modeling of the code is completed, then it is time to implement the code on the hardware LPC2148. The LM35 sensor output is interfaced to the AD0.3 pin of the LPC2148 through the non-inverting op-Amp. The output of the non-inverting op-Amp is the amplified voltage of the output of LM35 sensor.



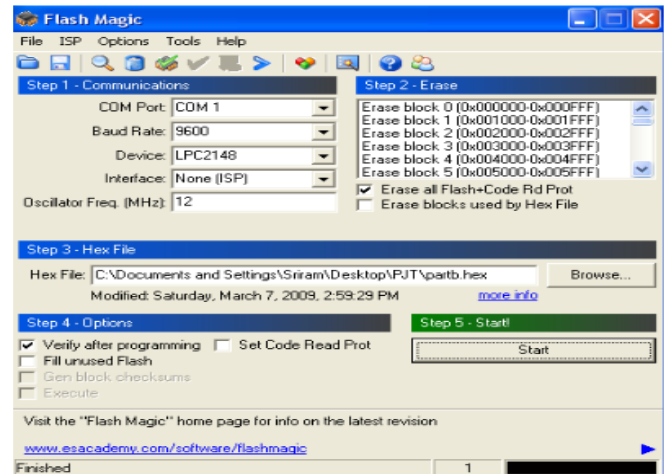
**Fig. 6a ADC Pin Highlighted**

The Fig (6a) shows the pin configuration of LPC2148. It shows the AD0.3 pin, which is the ADC (Analog to Digital Converter) to which the output of the op-Amp is connected. The pin configuration is used to check the location of the pins on the LPC2148. In the Figure 2.11, the pin configuration of LPC2148 shows a round notch which is used as a reference to select and interface to the desired pin on the LPC2148. Once the code is developed, it is dumped into the LPC2148 using a software known as Flash Magic.



**Fig. 6b Code Dumping**

The Fig (6b) shows how to dump the code using Flash Magic software. Before dumping the code it is necessary to set the baud rate, COM port, device type and frequency in the Flash Magic. Once the code is simulated in keilµ4, then a hex file of the code is generated which is dumped into the LPC2148.



**Fig. 6c Baud Rate**

The Fig (6c) shows the baud rate set to 9600, frequency to 12MHz, device type to LPC2148 and the COM port to COM1. Once all the fields are set in the Flash Magic then the generated hex file is browsed and selected, then the start icon is clicked and thus it begins dumping the code into the LPC2148. The Fig (6c) shows the completion of the dumping of code into the LPC2148.

```
//Code for Temperature data logging system//
#include<LPC214x.h>
#include<string.h>
Const unsigned int temp_value[2][32] =
{{0,0x03e0,0x03a2,0x0396,0x0367,0x035e,0x0345,0x0326,
0x0307,0x02e8,0x02cc,0x02ba,0x029b,0x028b,0x027c,0x0
25d,0x023e,0x020f,0x0200,0x01f0,0x0164,0x01bb,0x01d7,
0x0193,0x0199,0x019c,0x019f,0x01a2,0x01a6,0x01a9,0x01
ac},{0,70,68,66,64,62,60,58,56,54,52,50,48,47,46,44,40,39,
38,36,30,22,25,30,31,32,33,34,35,36,37,38}}; //Look-up
Table
void delay (void)
{
    int m,n;
    for(n=0; n<2;n++) {
        for(m=0;m<=1000000;m++); //delay of 1sec to
        capture readings
    }
}
void adc_init(void)
{
    PINSEL1=0x10000000; //Select P0.30 pin for ADC
}
unsigned int adc_read(void)
{
    unsigned int result;
    AD0CR = 0x00210608; // Select AD0.3, 10bit, Burst mode
    AD0CR = AD0CR | 0x01000000; //Start ADC, PDN=1
}
do
{

```

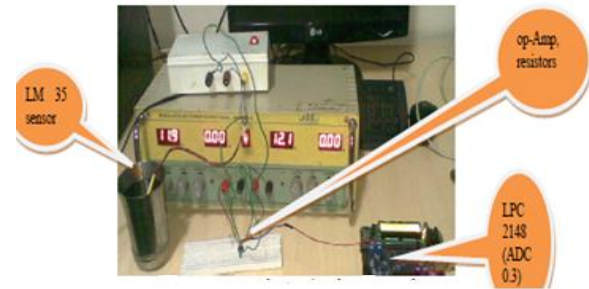


```

    result=AD0GDR; //Results from ADC
}
while(!(result & 0x80000000)==0);
return((result>>6) & 0x000003FF); // Return result as 32-bit
int
}
void uart_init(void)
{
    VPBDIV=0x00000002; //VPB Bus freq set to 30MHz (UART
standards)
    PINSEL0 = 0x00050000; // Select pin for UART
    U1LCR = 0x83;
    U1DLL = 0xC2; // Baud set to 9600
    U1LCR = 0x03;
}
int main()
{
    char preem[] = "Temperature is : ";
    char str1[] = "Start Temperature Monitor? (Y/N)";
    char c;
    unsigned int temp,a[2]={0,0};
    int i,l,l1;
    l=strlen(preem);
    uart_init(); //Initializing UART
    l1=strlen(str1);
    for(i=0;i<=l1;i++)
    {
        while(!(U1LSR & 0x20));
        U1THR = str1[i];
    }
    while(!(U1LSR & 0x01));
    c=U1RBR;
    while(!(U1LSR & 0x20));
    U1THR = c;
    if (c == 'Y' || c == 'y')
    {
        while(1)
        {
            adc_init(); //Initialize ADC
            temp = adc_read();
            for(i=0;i<32;i++) {
                if (temp == temp_value[0][i]) {
                    a[0]=temp_value[1][i] /10; //display the tenth element
                    a[1] = temp_value[1][i] % 10; //display the reminder
                    //a = temp_value[1][i]; //Searching for value in LUT
                }
            }
            while(!(U1LSR & 0x20));
            U1THR = '\n'; //Carriage return for next reading
            for (i=0;i<1;i++) {
                while(!(U1LSR & 0x20));
                U1THR = preem[i]; //Print preamble
            }
            for(i=0;i<2;i++) {
                while(!(U1LSR & 0x20));
                U1THR = a[i] + '0'; //Print temp value in ASCII
            }
            while(!(U1LSR & 0x20));
            U1THR = 'C'; //Print Centigrade/ Celcius after the
temp value
            delay(); //Delay
        }
    }
}

```

The above code represents the code that is required to implement the temperature data logging system on the LPC2148. This code is dumped into the LPC2148 for real time implementation of the hardware

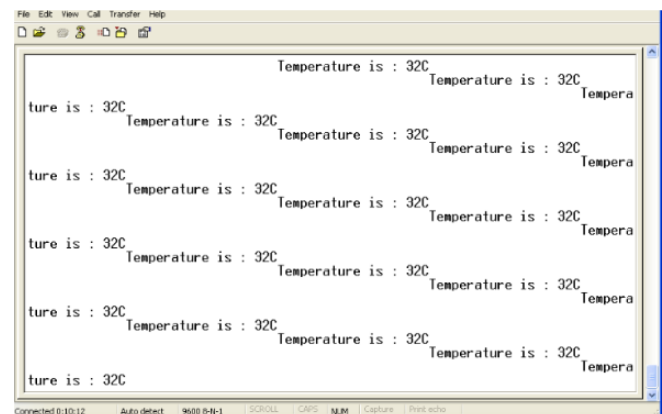


**Fig. 7 Hardware Implementation of Logging System**

The Fig (7) shows the hardware setup for the temperature data logging system. It consists of an op-Amp 741 IC connected on the breadboard. The output of the LM35 sensor dipped in a glass of water, whose output is given as input ( $V_{in}$ ) to the op-Amp and the output of the op-Amp is given to the LPC2148 which displays the output result on the PC using serial communication through the UART. A power supply of 12V is given to the pin 7 of the op-Amp and the pin 4 of the op-Amp is grounded. Once the appropriate connections to the op-Amp, LPC2148 and PC are done, then the output from the LPC2148 is displayed in terms of temperature ( $^{\circ}\text{C}$ ) on the HyperTerminal through the UART.

## VII. TEMPERATURE DETECTION AND LOGGING ON PC

Once the hardware is setup implementing the functionality of the code, then it is necessary to test the functionality of the code that is implemented using the LPC2148 by analyzing the output results of the LPC2148 that is displayed on the HyperTerminal through the UART. The Fig (8) shows the results generated by the LPC2148 that is displayed on the



**Fig. 8 Temperature Displayed on Hyper Terminal**

HyperTerminal. In the Fig (8), the temperature displayed is  $39^{\circ}\text{C}$  for the room temperature sensed by the temperature sensor LM35. The designed hardware generates a continuous value of temperatures ( $^{\circ}\text{C}$ ) that is displayed on the HyperTerminal with the delay of one second. Therefore the LM35 is used to sense different temperatures over a definite period of time and displays it continuously on the HyperTerminal.

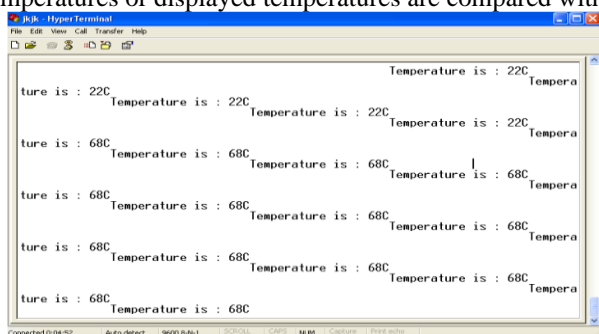
**Table. 1 Temperature Data Log**

Temperature (°C)	Voltage (V)
68	3.2
66	2.96
64	2.81
62	2.78
60	2.70
58	2.60
56	2.5
54	2.4
52	2.31
50	2.25
48	2.15
47	2.10
46	2.05
44	1.95
40	1.85
39	1.80
38	1.70
36	1.65
30	1.60
25	1.43
22	1.15

The Table 1 shows the different temperatures displayed for the corresponding voltage from the LM35 to the LPC2148, which is amplified using an op-Amp 741. The table 2.1 represents the data log of the temperatures that is displayed continuously on the HyperTerminal. The table represents the V\_Lookup table that has a record of different temperatures sensed by the LM35. A V\_Lookup functions can be used to achieve the desired data or temperature corresponding to their voltages from the temperature data log. The look up tables can be implemented using Microsoft office Excel 2007.

## VIII. GENERATION OF TEST CASES AND TABULATION OF RESULTS

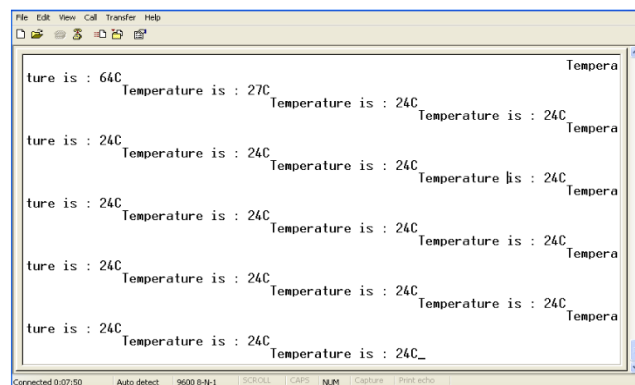
When the hardware has generated a set of results such as temperatures that is sensed or measured by the LM 35 and is displayed successfully on the hyper terminal, then we analyze the results by noting the higher temperatures detected and displayed on the hyper terminals. Similarly we note down the intermediate temperatures and lower temperatures sensed and displayed by the LM 35 and UART. The resulting temperatures or displayed temperatures are compared with



**Fig. 9a Display of Higher Temperatures**

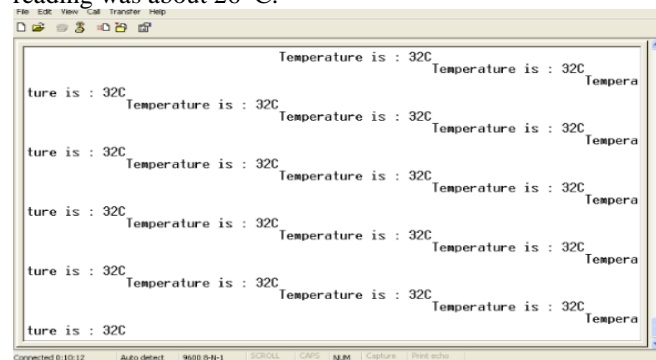
the temperatures shown by the thermometer in order to analyze the accuracy of the displayed temperatures. The Fig (9a) shows the maximum temperature of hot water detected

and displayed on the hyper terminal. The maximum temperature detected by the LM35 dipped in the hot water is 68 °C and the temperature measured by a thermometer dipped in the same hot water as that of LM 35 is 70 °C. So analyzing the temperatures displayed with respect to the temperatures measured by the thermometer we can check the accuracy of the temperature data logging system keeping the temperatures measured by the thermometer as the reference temperature for the displayed result or temperature on the hyperterminal.



**Fig. 9b Lower Temperature Displayed**

The Fig (9b) shows the lower range temperature i.e. 24 °C, sensed by the temperature sensor LM 35. The LM 35 sensor was removed from the hot water and placed in a tumbler of cool water along with a thermometer. The thermometer reading was about 26 °C.



**Fig. 9c Room Temperature Displayed**

The Fig (9c) shows the room temperature sensed by the LM 35 and displayed on the HyperTerminal. The LM 35 was removed from the cool water along with the thermometer and kept in air for about three minutes to get a stable output from the sensor of 32 °C, whereas the thermometer showed a reading of 30 °C.

**Table. 2 Temperature Data Logger Test Results**

Temperature Range	Temperature Displayed in °C (LM 35)	Thermometer Reading in °C (Reference)	Accuracy (+/- °C)
Hot	68	70	-2
Cold	24	26	-2
Room Temperature	32	30	+2

The Table 2 shows the practical temperature measurements of the temperature data logging system with reference to the thermometer temperature readings. For hot and cold temperature measurements of temperature log system, there is an accuracy of  $-2^{\circ}\text{C}$  with reference to the thermometer reading. And at room temperature there's an accuracy of  $+2^{\circ}\text{C}$ .

#### IX. MERITS AND DEMERITS OF THE SYSTEM WITH IMPROVEMENT SUGGESTIONS

The advantages of the designed temperature data logging system is that the LPC 2148 uses a voltage as low as 3.3V and the LM 35 can operate at a low voltage of 4V, the power consumed is low. The LM 35 has capability to sense temperature for a wide range i.e.  $-55^{\circ}\text{C}$  to  $+150^{\circ}\text{C}$ . From analysis of the designed system it was understood that the system was more accurate in sensing temperatures in the range of  $30^{\circ}\text{C}$  to  $60^{\circ}\text{C}$ . Using the sensitivity information of the LM 35 i.e.  $0.1\text{V}/^{\circ}\text{C}$ , it was easier to calibrate the values of LM 35 and mark the accuracy of the designed temperature data loggers. Some disadvantages of the system is that, according to the data sheet of LM 35 by Texas instruments, the accuracy of LM 35 is  $0.5^{\circ}\text{C}$ , but the designed system shows an accuracy of  $\pm 2^{\circ}\text{C}$ . Therefore for better accuracy of the system, it is necessary to know the voltage required to generate for all temperatures starting from  $-3^{\circ}\text{C}$  to about  $100^{\circ}\text{C}$  in steps of  $1^{\circ}\text{C}$  rather than taking about twenty or thirty values. This method is time consuming but very accurate. Another demerit is that the LM 35 sensor used cannot be dipped into the water to take temperature measurements because its open three terminals, therefore it is necessary to insulate these three terminals i.e. Ground,  $V_{\text{cc}}$  and  $V_{\text{out}}$  of the LM 35 sensor and then take measurements in the water or to get rid of these problems, we just use a thermistors. And demerit is that the LM35 requires an op-Amp to amplify its output and this increases the complexity to some extent, the system would become simpler if the LM 35 could be directly connected to the AD0.3 pin of the LPC 2148.

#### X. CONCLUSION

The code for the temperature data logging system is developed using the software keil $\mu$ 4. The developed code is synthesized and the simulation of the operation of ADC and UART is done before dumping the code into the LPC 2148. LM 35 sensor has a sensitivity of  $0.1\text{V}/^{\circ}\text{C}$ . The implementation of the code on LPC 2148 is displayed on HyperTerminal, continuous display of temperatures with a delay of one second for different temperature range i.e.  $68^{\circ}\text{C}$ ,  $24^{\circ}\text{C}$ ,  $39^{\circ}\text{C}$ . The code is dumped into LPC 2148 using Flash Magic. The designed temperature data logging system has an accuracy of  $\pm 2^{\circ}\text{C}$ . A non-inverting op-Amp 741 is used to amplify the output of the LM 35. For an input voltage from the LM 35 of 0.7V to the op-Amp, gives an amplified output of 3.2V which is the input to LPC 2148 and displays a maximum temperature of  $68^{\circ}\text{C}$ . For input voltage of 1.15V, the minimum temperature of  $22^{\circ}\text{C}$  is displayed on the HyperTerminal.

#### REFERENCES

- [1] Deepika, Mr. Nirbhawjap Singh (2006), 'Designing a Microcontroller Based Temperature Data Logger' Thapar Institute of Engineering & Technology, July 2006.
- [2] Ž. Nakutis (2009) 'Embedded Systems Power Consumption Measurement Methods Overview' MATAVIMAI, December 18.

- [3] Hla Hla Htay (2011), "Design and Construction of Microcontroller Based Data Logger" Department of Physics, Yadanabon University
- [4] M. Moghavvemi, "Remote Sensing of Relative Humidity", University of Malaya Kuala Lumpur, Malaysia.
- [5] Radek Kuchta, Radimir Vrba, "Wireless and Wired Temperature Data System", Second International Conference on Systems Brno University of Technology, Brno.
- [6] M. Moghavvemi, "A Simple Low Cost Data Acquisition System for Remote Sensing of Relative Humidity and Temperature", Dept. of Electrical Engineering University of Malaya, Kuala Lumpur, Malaysia.
- [7] B. Niharika, Sayara Sultana, B.Prasanna, L.Harika "Data Logging To Collect And Display Temperature With Time And Day", Gokaraju Rangaraju Institute of Engineering & Technology.
- [8] Theophilus Wellem (2012) "A Microcontroller- based Room Temperature Monitoring System", International Journal of Computer Applications (0975 – 8887) Volume 53– No.1, September.
- [9] Cytron Technologies (2008) "Temperature Control System using LM35", Cytron Technologies Sdn Bhd, January.
- [10] BiPOM Electronics (2006) "Microcontroller to Sensor Interfacing Techniques", Document Revision: 1.01, February.



**Vinay Divakar** received Bachelor of Technology in Electronics and Communication Engineering in July 2012 from S.t Joseph Engineering College, Mangalore, India and currently pursuing M.sc (Engg) degree program in Electronic System Design Engineering in M.S Ramaiah School of Advanced Studies (Bangalore, India) Affiliated to Coventry University, U.K. He is a student and his research interests are in systems engineering, systems design and development, engineering design management etc. Recently he

has published two papers 1) systems engineering based process for development of UAV and 2) Design and Simulation of BPF and LNA in IJSER for Volume 4, Issue 12, December-2013. Now presently he's working on a research based project that involves a structured approach in the design and development of a physiotherapy hand trainer kit using Labview and NI DAQ.