DESIGN AND CONSTRUCTION OF A MICROCONTROLLER-BASED TEMPERATURE DATA ACQUISITION AND LOGGING SYSTEM

BY

SULE EZEKIEL ANDREW
2003/15363EE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING,
SCHOOL OF ENGINEERING AND ENGINEERING TECHNOLOGY,
FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA.

NOVEMBER 2008

DESIGN AND CONSTRUCTION OF A MICROCONTROLLER-BASED TEMPERATURE DATA ACQUISITION AND LOGGING SYSTEM

BY

SULE EZEKIEL ANDREW
2003/15363EE

A THESIS SUBMITTED TO DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING, FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA. IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE BACHELOR OF ENGINEERING(B.ENG) DEGREE IN ELECTRICAL AND COMPUTER ENGINEERING.
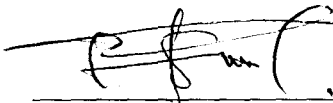
NOVEMBER 2008

DEDICATION

I dedicate this thesis to the Almighty God, the father of our lord and savior Jesus Christ, whom by his grace and mercy, I have enjoyed sufficient divine protection, guidance, provision, before, during and after this work. I also dedicate it to my able parent Mr. and Mrs. Sule Kalangu for both moral and material support given to me on the course of this work. Not forgetting my Uncle Mr. Dikko Kwabwaga and his family who help me in several ways during this work. I wish hem an abundant of God's blessings.

ATTESTATION/DECLARATION

I sule Ezekiel Andrew hereby declare that this project is presented in partial

fulfillment of the requirement for the award of bachelor of engineering(B.eng) degree., it

has not been presented before either wholly or partially for any other degree elsewhere.

Information hereby obtained from public or unpublished works of others are

acknowledged accordingly. I also relinquish he copyright to Federal University Of

Technology.

SULE EZEKIEL ADREW   ENGR. S.A MOHAMMED

(Name of student)   (Name of supervisor)

_____   _____ 10/11/08

(Signature and Date)   (Signature and Date)

ENGR(DR). Y.Adediran   _____

(H.O.D)   (Name of External Examiner)

_____   _____

(Signature and Date)   (Signature and Date)

iii

ACKNOWLEDGEMENT

ABSTRACT

Data acquisition is an ancient practice. Over the ages, data acquisitions have been implemented using different methods and equipments, like sensors around vehicles, telemetry, recording devices and so on. The design and construction of a microcontroller-based temperature data acquisition and logging system, as described in this project report, is intended to produce a simple means of acquiring, analyzing and storing data obtained from certain systems, in a simple better and more accurate way. The project is divided into six modules namely, the control unit, the sensing and conversion unit, the indicator unit and the power unit. The control unit is the central processing unit of the project. It has a microcontroller AT89C51 and the program software written in Assembly language, a logic level translator for a PC RS232C serial port (DB9). The sensing and conversion unit consist of a temperature sensor (LM35) and an analog to digital converter (ADC0804), the indicator unit is made up of driver transistors (A1015) in which relays with heaters are connected to.

Finally the power unit has a 240 to 12V step-down transformer and a full-wave bridge rectifier circuit as the voltage source. This was regulated to 5V using a LM 7805 voltage regulator IC. On integration, the forms the microcontroller-based temperature data acquisition and logging system.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

This is a project report on design and construction of a microcontroller-based temperature data acquisition and logging system. The report gives an account of the work carried out during the various stages of the project.

A data acquisition and logging system is a device designed to measure and log some parameters. The purpose of the data acquisition system is generally the analysis of the logged data and the improvement of the object of measurement. The system is normally electronic based and is made up of hardware and software.

Although, little was known about the detail of data acquisition in the prehistoric eras, but it was discovered that in every culture some people were preoccupied with the acquisition, analysis and storing on data's collected from certain activities. Ancient civilization relied on rainfall, use of sticks and other objects to collect data's.

For instance, an obelisk which was built by the Egyptians was used to determine time. The obelisk divided their day into parts some like our hours. Obelisks (slender tapering, four-sided monuments) were built as early as 3500 BC. Their moving shadows formed a kind of sundial, enabling people to partition the day into morning and afternoon [1].

Temperature measuring history can be traced from Anders Celsius who lived between the years 1701 — 1744. early, became engaged in the general problem of weights and measures, including temperature measurement [2].

Although, there have been a lot of effort by Nigerian researchers to develop a cost effective means of temperature data acquisition and logging system using digital electronic method. However, looking at most of these alternatives, it is obvious that there is still a need for a more and accurate temperature data acquisition and logging system design by our local engineers.

It is as a result of the above situation that this project is dimmed at developing a microcontroller- based temperature data acquisition system that shall operate to eliminate inefficiency in the previous and existing devices of this kind.

1.1 OBJECTIVES

1| The microcontroller-based temperature data acquisition and logging system is aimed at solving the problems of inaccuracy in the earlier methods of data acquisition and logging systems through the use of microcontroller and integrated circuits of high efficiency and accuracy.

2] Also, earlier data acquisition systems were large in size, complex in construction and expensive to build, but this project is designed to overcome the Problem of cost complexity, portability and accuracy.

3| To analyze and improve on the object of measurement.

1.2 METHODOLOGY

The method adopted in this project is based on the piezoelectric property of quartz crystal. If an electric field is applied to the crystal, it changes it shape and if it is squeezed

2

or bent it generates an electric field [3]. The pulse generated by the crystal is used to clock the microcontroller

The microcontroller is a programmable electronic component that determines the various temperature of the sensors based on the program written into it and implements the set-points which is reflected by the indicators and the data acquisition software. Assembly language is used in the programming of the microcontroller for easy programming and reprogramming. Thus, this project is designed to meet the universal need of an accurate temperature data acquisition and logging system.

1.3   SCOPE OF WORK

This project is designed to collect or obtain temperature values any system at different set-points depending on the nature of the system involved. Thus, a temperature set-point can be chosen to sooth the situation. This has a lot of application in food industries and places where the instantaneous temperature of the surrounding or cooling medium of machines are monitored.

A precision integrated-circuit temperature sensor(LM35) is used in this project due to the fact that the sensor has a range of temperature measurement from $-55^{\circ}C$ to $+150^{\circ}C$ and with an accuracy of $0.5^{\circ}C$ guarantable. Also, a microcontroller (AT89C52) with the logic translator forms the central processing unit. The microcontroller is programmed in assembly language. An analogue to digital converter (ADC 0804) with a four channel multiplexer (CD4052B) was also used to convert the output of the LM35's to digital signals. The heating unit is made up of heaters connected to relays which were driven by transistors.

1.4 IMPORTANCE OF THE PROJECT

Having a precise accurate data's of the temperature of an environment is of great necessity because of its wide application in places like, hospitals, in which the life's of premature babies depends on t temperature at which the incubator is `kept, laboratories require the use of precise readout in carrying out analysis and test which requires specific temperatures.

Poultry farms are meant to be kept at certain reasonable temperatures. Therefore, having a readout temperature is necessary. Industries like pharmaceuticals, petrochemicals and petroleum also require precise easily readout temperatures. The necessity of having a temperature data acquisition and logging system is so numerous that it cannot be exhausted in this report.

CHAPTER TWO

LITERATURE REVIEW/THEORETICAL BACKGROUND

2.1 BRIEF HISTORICAL BACKGROUND

Data acquisition and logging system is a practice that has being in existence for a

long time even right form the prehistoric era. This is reflected in the invention of

merchets, the oldest known astronomical tool, by the Egyptians around 600BC. A pair of

marchet was used to establish a north-south line (or meridian) by aligning them the pole

stand. They could then be used to mark off nighttime hours by determining when certain

other stars crossed the meridian.[2].

David A.Greer and David E. Schwelkert on August 27th, 1995 invented an event

recorder and a method for recording data relating to a distinct event which pertains to

operation of a mechanism [4].

2003 British Grand prix is another good method of data acquisition for critical

data acquisition system in motor sports, in which engineers in pit observe the loss of

pressure from one of coulthards tyres. The data acquisition system allowed the team to

recall him from his practice, resolving the fault before a dangerous situation occurred,

likely saving property and life[5].

Today various type of data of data acquisition are in existence such as:

Pico log data acquisition and logging system: this is for windows which work

in two modes: player mode for displaying previously recorded data and recorder mode,

for recording new data. Pico log can collect data from multiple converters at the same

time. This not only flows a mix of voltage input units to be used on the same PC, but also

allows a mix of voltage input units to be used on the same PC, but also allows other PC based instruments such as TC-08 (temperature measurement and the ADC-200 oscilloscope to be used at the same time) [6]

Another type DAS is the **precision instrumentation equipment (PIE)**: this utilizes hardware for real-time data gathering of fuel storage conditions under large underground (UST's) or above ground (AST's) tanks and software for the local and real display and analysis of this information. PIE systems performs the central function of collecting all of the live dynamic data relative to tanks. PIE collects its data real-time and stores it to a real-time RAM database for current, live data and to a hard disk as on historical database. This system works in a server/client mode over a secure, isolated LAN that transmit bidirectional from the server to the remote clients. PIE is completely operator independent, i.e. no operator interface is required whatsoever [7].

M7 data acquisition and logging system: this allows for the monitoring of digital cameras, giving 24hours surveillance as well as data collection and monitoring, this lend it self ideally to water boards and companies who need to be proactive office base personnel can monitor from a distance through attached camera system without the need of costly service personnel to interrogate false alarms or other issues with in plant equipment [8].

Data lynx data collection software is an inexpensive data collection and data logging and storage package for programmable logic controllers (PLCs) using a PC. This data acquisition software package is an idea data logger solution for collecting data from remote location or relaying data from multiple locations back to a single location to be stored. [9]

2.2 LITERATURE REVIEW

The microcontroller-based temperature data acquisition and logging system is an electronic device designed to measure and log some parameters usually temperature based. It is usually powered from a dc supply which may be a battery or the host device when in line with other devices or equipment for data acquisition e.g. machines, cars etc.

1. A microcontroller –based temperature data acquisition and logging system has it operation controlled by the microcontroller with the aid of the program written to control it operations.

There have been companies engaged in constructing and marketing data acquisition and logging systems, but their design methods or techniques used is usually not available for analysis. However, the use of microcontroller and provision for expansion of temperature channels is not very common, that is why this project is meant to reduce the cost of production and increase the efficiency of performance by the use of programmable microcontroller in the design because, fewer components are used.

In the course of designing the microcontroller-based temperature data acquisition and logging system, the project is analyzed. The project consists of the following unit:

The control unit, which is the central processing unit of the system comprising of microcontroller AT89C51 and the program software written in assembly, a logic level translator for PC RS232C serial port. The sensing and conversion unit made up of a temperature sensor LM35 and an analog – to digital converter (ADC0804). The heating unit consisting of transistor relays and heaters.

Finally, the power unit has a 240-12V step-down transformer, whose output is regulated to 5V by using LM7805 voltage regulator IC.

7

2.3 THEORETICAL BACKGROUND

2.3.1 POWER SUPPLY

In order to make the project portable and because of the need for constant power supply, the project was powered through a 240 -12V a.c step-down transformer. This was further regulated to 5V dc in order to meet the voltage range of the microprocessor which is 5V.

2.3.2 VOLTAGE REGULATOR

The regulation of a power supply is its ability to hold output steady under conditions of changing input or changing load [9]. Many simple DC power supplies regulate the voltage using regulator such as zener diode, avalanche breakdown diode, or voltage regulator tube [10]. Each of these devices begins conducting at a specified voltage and will conduct as much current as required to hold its terminal voltage to that specified voltage.

The power supply is designed to only supply a maximum amount of current that is within the safe operating capability of the shunt regulating device(commonly, by using a series resistor). In shunt regulators, the voltage reference is also regulating device. Similarly, an integrated-circuit voltage regulator (LM 7805) was used. This was supplied in the TO-220 case and has three leads. The pass transistor, the pass transistor, error amplifier, reference circuit, and protection circuitry are all on the chip [9].

2.3.3 MICROCONTROLLER

the microcontroller is an advanced electronic device which has risen out of logic integrated circuits. The rate of development has arguably been greater than any other electrical device. Its introduction commenced with the microelectronic developments for integrated circuits, but it has reached the point of creating circuitry with a density of 20,000 transistors per square centimeter of the semiconductor slice [11].

The microcontroller can perform the following principal ranges of operations:

❖ Control.

❖ Calculation.

❖ Administration.

2.3.4 MICROCONTROLLER OPERATION

The microcontroller consists of thousands of electronic switching circuits. As with any logic operations, each circuit can either be ON or OFF, i.e., HIGH or LOW. Instructions fed into a microcontroller are in binary digits known as **bits**. The form of the instruction must be a series of ones and zeros which relate to circuit being closed or open, i.e., ON or OFF, or HIGH or LOW. Typically, a circuit which is ON will supply a signal of about 5V and a circuit which is OFF will supply a signal of about 0V. an eight bit instruction is termed a **byte**. The byte is fed into the microcontroller either by a sequence of pulses, which is known as **asynchronous** action (or **serial operation**) or all pulses at once, which is known as **synchronous** action (or **parallel operation**).

Information is fed into a microprocessor in two distinct forms: **instructions** and **data.** Both are expressed in binary form. The instructions relate to a particular section of memory arrangement, called the **Read Only Memory (ROM)**. It has a set of instructions manufactured into it which cannot be changed, when the microcontroller is switched-off. It is called non-volatile memory. There is another memory in which information can be temporary stored, called **Random Access Memory (RAM)**. A RAM comprises of a larger number of stores, each of which has an address, therefore, when we insert an information byte, we require an associated address indicating the particular location in which it is to be stored. Similarly, we need to know the address should we need to recall the information byte. RAM is volatile. i.e., the information stored in it is lost when it is de-energized.

The microcontroller can be seen to operate on the interaction of a number of interacting processes. At the heart of these interactions is the **accumulator.** This can be considered as the section where the main activity takes place. Thus when it is proceeding through a series of operations, the changes in the information process takes place in the accumulator. Typically, a sequence of events could require two or three changes, at which stage the processor would have gone as far as it could. The result can then be stored in the RAM, clearing the accumulator ready for the next series of operations. The control of the sequence may come either from the ROM or from another section of the RAM in conjunction with the ROM.

Control bus

CPU

Adress
bus

R
O
M

Adress
bus

R
A
M

Adress
bus

P
O
R
T

Port

Data bus
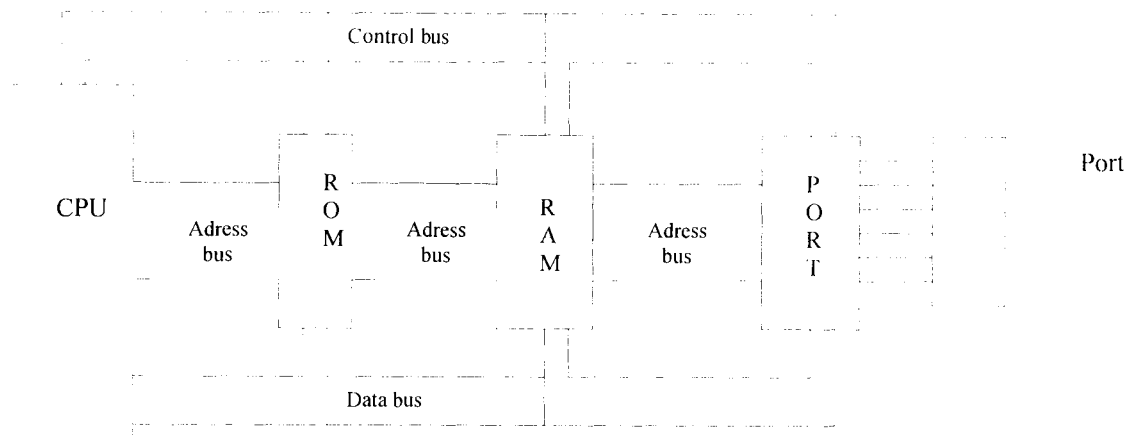
Figure 2.1 A simple block diagram of a microcontroller system.

In the diagram above, the central processing unit CPU is the micontroller chip containing the accumulator. It is connected to ROM and to the RAM by three sets of circuits or buses, the address bus and data bus, and the control bus.

2.3.5 OSCILLATOR

For an exceptionally high degree of frequency stability, use crystal oscillator is essential. The quartz crystal has peculiar properties when mechanical stress is applied across its two opposite faces, a potential difference is applied across them. It is called **piezoelectric effect.** Conversely, when a potential difference is applied across its two opposite faces, it causes the crystal to either expand or contract. If an alternating voltage is applied, the crystal wafer is set into vibrations. The frequency of vibration v equals the natural resonant frequency of the crystal, as determined by its structural characteristics.

Where the frequency of the applied a.c. voltage equals the natural resonant frequency of the crystal, the amplitude of vibration will be maximum.

As a general rule, the thinner the crystal, the higher its frequency of vibration. The electrical equivalent circuit, shown in the figure below, consist of series RLC $_1$ circuit in parallel with a capacitor C_2.
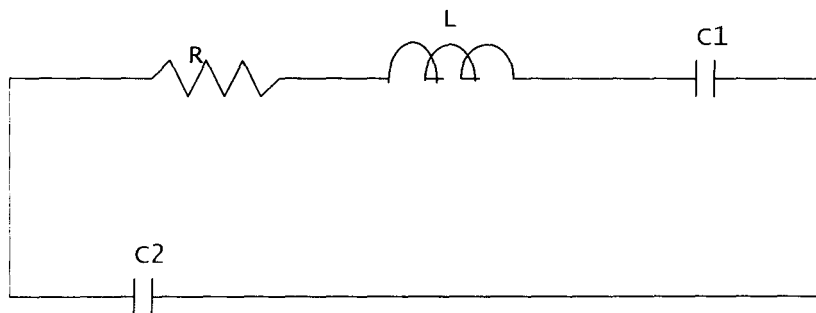
Figure 2.2 Electrical equivalent circuit of a crystal

The circuit has two resonant frequencies:

- The lower series resonance frequency f_1, which occurs when X_L X_C. In that case,

$$Z = R \; i.e$$

$$F1 = 1/2\pi\sqrt{LC1}$$

- The other is the parallel resonance frequency f_2 which occurs when reactance of the series leg equals the reactance of C_2. At this frequency, the crystal offers very high impudence to the external circuit.

$$F_2 = 1/2\pi\sqrt{LC}$$

$$C = C_1 C_2 / C_1 + C_2$$

12

Crystals are available at frequency 15KHz and above [12]. At frequency above 100MHz, they become so small that handling them becomes a problem.

2.3.6 TEMPERATURE SENSING TECHNIQUE

A temperature sensor is a device that senses temperature, the variation in an environment and gives it value in electrical form. The thermal sensors in use include , thermocouples, resistance temperature detectors (RTD), thermistor and integrated circuit sensors.

Thermocouples, measure temperature difference using voltage developed by the junction of two dissimilar metals. Another temperature sensor in use is the thermistor. It could either be negative temperature coefficient (NTC) or positive temperature coefficient (PTC) thermistor. The resistance of the NTC decreases with increase in temperature while that of PTC increases as the temperature as temperature increases. Thermistors are fabricated in disk/rods, beads and washers covering resistance 10^6 ohms and a variety of temperature coefficient. [10].

However in this project, the temperature sensor in use is The LM35 precision integrated - circuit temperature sensor, whose output voltage is linearly proportional to the Celsius(centigrade) temperature. The LM 35 does not require any essential calibration or timing to provide typical accuracies of $+1/4°C$ at room temperature and $+3/4°C$ over a full $-55°C$ to $+150°C$ temperature range. The LM 35 has a low output impedance, linear output, and precise inherent calibration which makes interfacing to readout or control circuitry especially easy. The LM35 has an advantage

over linear temperature sensors calibrated in Kelvin and the user is not required to subtract a large constant voltage form its output to obtain centigrade scaling. [13].

Features:

- ❖ Calibrated directly in Celsius(centigrade)
- ❖ Linear +10.0mV/°C scale factor
- ❖ 0.5oC accuracy guaranteable (at +25°C)
- ❖ Suitable for remote applications
- ❖ Operates from 4 to 30volts
- ❖ Less than 60uA current drain
- ❖ Low self-heating 0.08°C in still air
- ❖ Nonlinearity only +1/4°C or -1/4°C typical.
- ❖ Low impudence output 0.1 ohms for 1mA load.

2.3.7 ANALOG – TO-DIGITAL CONVERSION

An analog- to-digital converter (abbreviated AD, A/D) is an electronic device that converts an input analog voltage (or current) to a digital number (a series of 1s and 0s) and eventually to a digital number (base 10) for reading on a meter, monitor or chart [13]. There various types of ADC in use namely: **voltage-to-frequency ADC** which converts the arising input voltage pulse train with the frequency proportional to the amplitude of the input. The pulses are converted over a fixed period to determine the frequency, and the pulse counter output, in turn, represent the digital voltage. Another type of ADC is the **successive-approximations** converter which is compose of a digital-to-analog (DAC), a single comparator, some logic, some control logic and registers. When the analog voltage

14

to be measured is present at the input to the comparator, the system control logic initially set all bit to zero.[13].

In this report, ADC 0804 a monolithic complementary metal oxide semiconductor (CMOS) device is used. The ADC is an 8-bit converter, 1-channel multiplexer and microcontroller compatible control logic. It uses successive approximations as the conversion technique. the converter features a high impendence chopper stabi9lized comparator, a 256R voltage divider with analog switch tree and successive approximation register. The device eliminates the need for external zero and full-scale adjustment.

2.3.8 LOGIC LEVEL TRANSLATOR

The device is a distance part that does not respond to logic levels more than 0V, a means of interfacing the controller with the bipolarity signaling voltages on the RS232 interface was needed. A converter that translate the CMOS level 0V/5V signal to -12/+12V signal and the computer serial port.

2.3.9 INTERFACING UNIT

This is a serial (RS232) pin out and signals for the PC RS232 connector. It receives and transmit signals from and to the PC from an electronic system.

The pin arrangement is shown bellow:

15
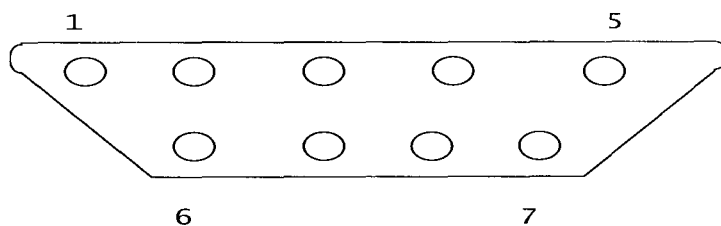
Fig 2.3 DB9 serial (RS232) port pin arrangement

Pin	Name	Direction	Description
1	CD	←—	Carrier Detect
2	RXD	←—	Receive data
3	TXD	—→	Transmit data
4	DTR	—→	Data terminal ready
5	GND		System ground
6	DSR	←—	Data set ready
7	RTS	—→	Request to send
8	CTS	←—	Clear to send
9	RI	←—	Ring indicator

Fig 2.4 pin description of RS232

2.3.10 PROGRAMMING

programming languages vary greatly in their sophistication and their

degree of versatility. Some programming languages are written to address a particular

kind of computing problem or for use on a particular model of computer system. For

instance programming languages such as **FORTRAN** and **COBOL** were written to solve

certain general types of programming problems-FORTRAN for scientific applications,

and COBOL for business applications.

Although these languages were designed to address specific categories

of computing problems, they are highly portable, meaning that they may be used to

16

program many types of computers. Other languages, such as machine languages, are designed to be used by one specific model of computer system, or even by one specific computer in certain research applications. The most commonly used programming language are highly portable and can be used to effectively solve diverse types of computing problems. Languages like C, PASCAL, and BASIC fall into this category.[14].

Programming languages are classified into three:

* ❖ Machine language

* ❖ Assembly language

* ❖ High level language

2.3.11 MACHINE CODE

In machine language, instructions are written as sequences of 1s and 0s. called **bits,** that a computer can understand directly. An instruction in machine language generally tells the computer four things: (1) where to find one or two numbers or simple pieces of data in the main computer memory (Random Access Memory, or RAM), (2) a simple operation to perform, such as adding the two numbers together, (3) where in the main memory to put the result of this simple operation, and (4) where to find the next instruction to perform. While all executable programs are eventually read by the computer in machine language, they are not all programmed in machine language. It is extremely difficult to program directly in machine language because the instructions are sequences of 1s and 0s.

2.3.12 ASSEMBLY LANGUAGE

Programmers use assembly language to make machine-language programs easier to write. In an assembly language, each statement corresponds roughly to one machine language instruction. An assembly language statement is composed with the aid of easy to remember commands. Assembly languages share certain features with machine languages. For instance, it is possible to manipulate specific bits in both assembly and machine languages. Programmers use assembly languages when it is important to minimize the time it takes to run a program, because the translation from assembly language to machine language is relatively simple.

2. 3.13 HIGH LEVEL LANGUAGE

High-level languages are relatively sophisticated set of statements utilizing words and syntax from human language. They are similar to normal human languages than assembly or machine languages and are therefore easier to use for writing complicated programs. These programming languages allow larger and more complicated programs to be developed faster.

However, high – level languages must be translated into machine language by another program called a compiler before a computer can understand them. For this reason, programs written in a high-level language may take longer time to execute and use up more memory than program written in an assembly language [14].

CHAPTER THREE

DESIGN AND IMPLEMENTATION

3.1 INTRODUCTION

The design of the microcontroller-based data acquisition and logging system consist of the following subunits:

I.   Regulated 5-volt systems of DC supply voltage.

II.   LM35 temperature to – voltage converters.

III. 4-channel CD4052 analog multiplexer

IV.   ADC0804 8-bit analog -to-digital converter.

V.   RS232 logic level translator

VI.   8-bit 8952 system controller.

VII. Load power switches

The design was done in six modules namely, power supply unit, temperature sensing unit, analog to digital conversion unit, control unit, and the heating unit logic level translator. This is shown in fig3.0 below:
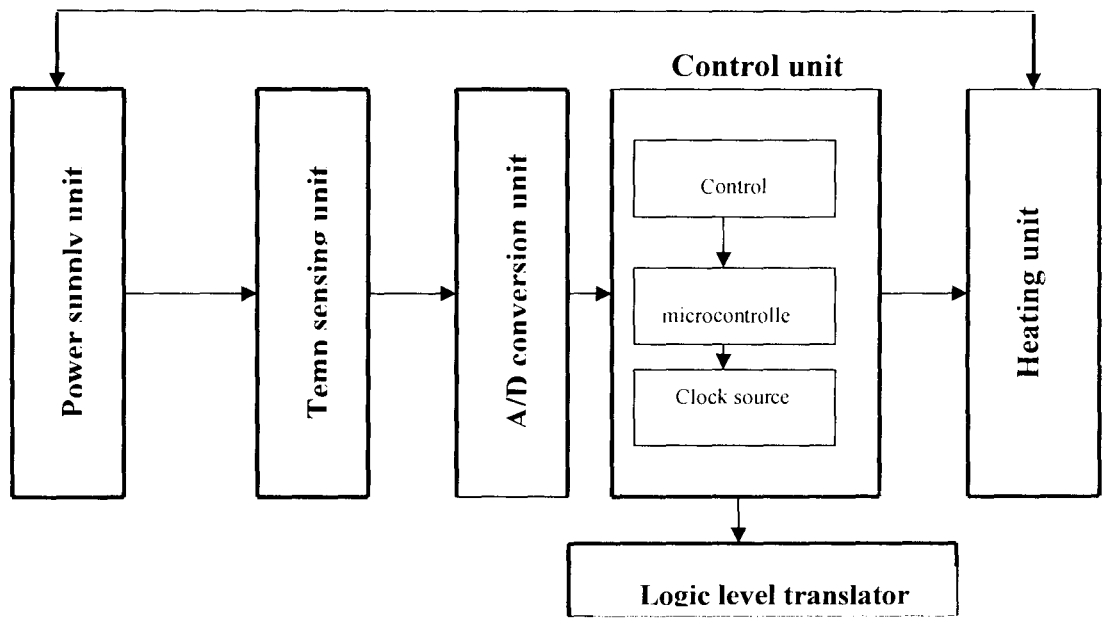
Fig 3.0 Block diagram of microcontroller –based- data acquisition and logging system.

3.2 POWER SUPPLY UNIT

The digital and analog subsystems required a highly regulated 5 volt Dc supply for operation. This supply voltage was derived from a 12V, 0.5A step-down transformer, a 4-diode full-wave bridge rectifier and a 7805 5 V Dc electronic stabilizer as shown in fig 3.1
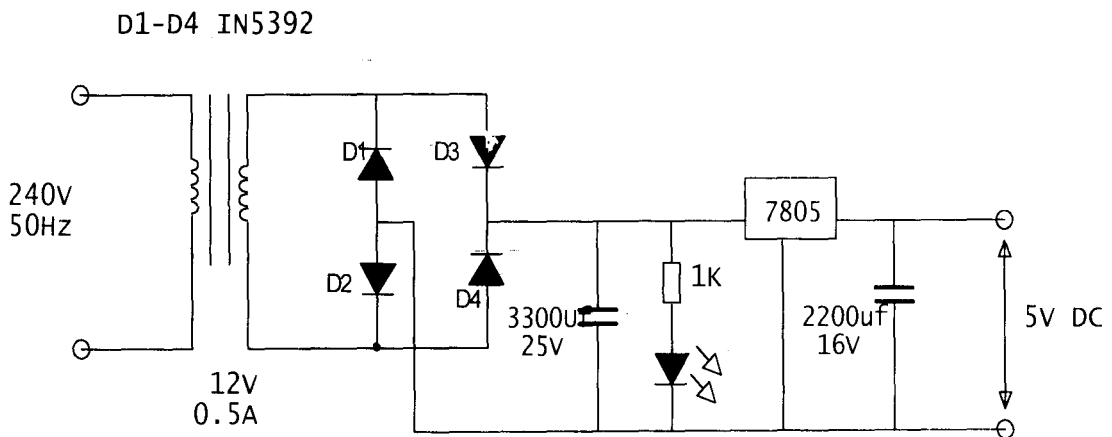
Fig 3.1 system power supply unit circuit.

The 12V rms AC voltage on the secondary winding of T1 was converted to a pulsating DC voltage at 100 Hz (twice the frequency of the means), by a 4-diode full-wave bridge rectifier. The amplitude of the DC voltage was given by the relation:

$$V_{DC\ (peak)} = [(V_{rms}\sqrt{2}) - 1.4]\ V \qquad \dots\dots\dots\dots\dots\dots\dots (1)$$

For a 12V (at 240V line voltage),

$$V_{DC\ (peak)} = (12\sqrt{2}) - 1.4 \approx 15.5V$$

The DC voltage is pulsating, going from zero to maximum to zero every 10ms.

The DC voltage was smoothened by a capacitance of value deduced from the expression:

$$CV = It \dots\dots\dots\dots\dots\dots\dots\dots\dots (II)$$

A 7805 regulator was used to provide the required 5 volt DC supply. The minimum input voltage for the regulator for a 5V regulated output was 7V. On a 15.5 peak input voltage, this corresponds to a maximum allowable AC ripple voltage of 15.5 - 7 = 8.5V

Using CV = It where

C = value of the smoothening capacitance

V = maximum load current

i = maximum load current

t = period of the pulsating DC voltage

21

(1/F: HWR; 1/2F: FWR).

The peak system operating current was computed by the summation of the current drawn by the principal sub systems:

- ❖ Microcontroller   15mA (max)

- ❖ ADC0804   5mA (max)

- ❖ CD4052   ____

- ❖ Relays(4)   60mA (max)

- ❖ Logic level converter   ____

- ❖ Power indicator   12mA

$\sum I = 90mA$

Hence: $C = It/V = 0.09 * 1/100$

$$\frac{}{8.5}$$

$= 0.0009/8.5 = 106 \ \mu F$

This value of capacitance was increased to 3300μF to improve system performance on low AC line voltages.

A capacitor was chosen for a 25V working voltage. The smoothened DC voltage was regulated down to +5V DC by 7805 5v DC regulator. The 5V DC supply was buffered by a 16V, 2200μF capacitance and fed into the circuit.

3.21 POWER ON INDICATOR

A LED power on indicator was provided across the rectifier output. The LED was connected to a current limiting resistor as depicted in fig 3.1
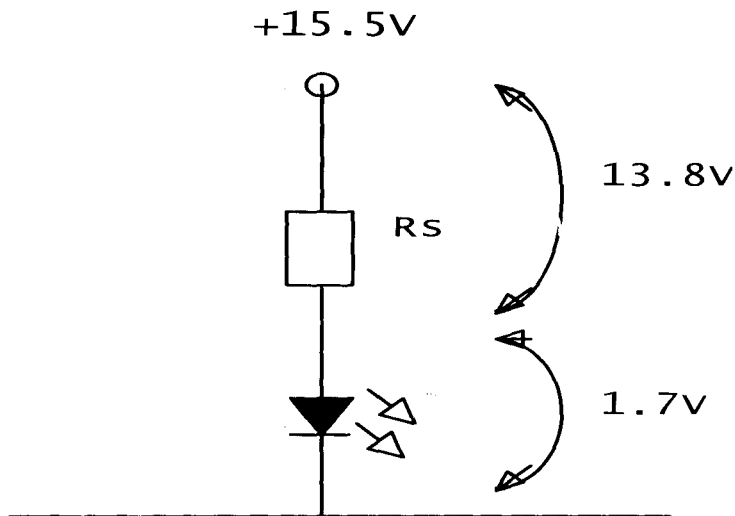
Fig 3.2 power on indictor

The LED was operated on a 13mA continuous forward current (minimum: 5mA, maximum: 20mA).

R_S were calculated using:

$$R_S = V_S - V_{LED} / I_{LED}$$

$$= 15.5 - 1.7 / 0.013 = 1061\Omega$$

Thus 1 kΩ resistance was used.

3.3 TEMPERATURE SENSING-TO- VOLTAGE CONVERSION UNIT

Four LM35 temperature centigrade –to- voltage transducer were used as the system surrounding interface. The LM35 is a precision integrated circuit whose output voltage is inversely proportional to the Celsius (centigrade) temperature. The subunit was used to detect the ambient temperature in the monitored location and has the electrical specifications stated below:

❖ Calibrated directly in Celsius (centigrade)

❖ Linear +10.0mV/°C scale

❖ 0.5°C accuracy guarateable (of +25°C)

❖ Rated for full -55°C to + 150°C range

❖ Suitable for remote applications

❖ Operates from 4 to 30 volts

❖ Less than 60µA current drain

❖ Low self – heating 0.08°C in still air.

❖ Non linearity only ±¼°C typical.

❖ Low impedance output 0.1Ω for 1mA load.

Fig 3.3 diagram of an LM35 +Vc Vout GND showing the pins. The four sensors were interfaced

with an ADC0804 single-channel analog-to-digital converter via a CD4052 dual 1 to 4

analog switch. The sensors were wired to the multiplexer as shown in fig3.3 below:

Fig 3.4 LM35 interfaced connection

3.31 DUAL 4-CHANNEL 1 TO 4 MULTIPLEXER

Since a single- channel ADC was used and 4 sensors were required to be

interfaced, a means of individually addressing and selecting the sensors was needed.

A CD4052 low-speed dual 1 of 4 analog multiplexer was utilized. The device is

16 in DIP, and has the pin out shown in fig 3.4 below:

Fig 3.5 CD4052 Avilos multiplexer

        The device has a two bit address inputs (A_0, A_1), the logical state

of which determines the sensor that is connected to the converter input. The

multiplexer was run on a +5/0V supply. It was connected to the system controller

over P3, the controller generating the required logic levels required for device control.

The inhibit pin of the device was disabled permanently as it was not needed.

3.3.2 ANALOG-TO-DIGITAL CONVERTER

To translate the analog voltage equivalent of the sensed temperature to a digital value

that can be manipulated by software, an interfaced device capable of translating the

measured quantity (temperature) from the analog domain to digital is needed. This

was realized using an ADC0804 8-bit analog–to-digital converter. The ADC0804 is a

CMOS successive approximation register A/D converter that uses a differential

potentiometric ladder for conversion. The device has a logic interface that allows

direct connection to most processors/controllers, requiring no intervening the logic.

```
   1 CS        Vcc  20
   2 RD        CLKR 19
   3 WR        B0   18
   4 CLKIN     B1   17
   5 INTR      B2   16
   6 Vin-      B3   15
   7 Vin+      B4   14
   8 AGND      B5   13
   9 Vref/2    B6   12
  10 DGND      B7   11
```
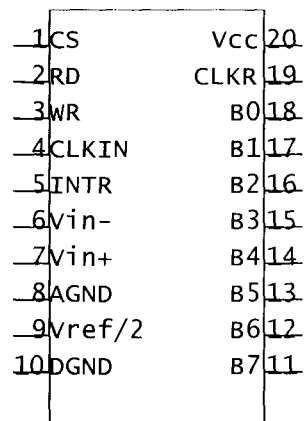
Fig 3.6 diagram of ADC0804

The device has the specification listed below:

- ❖ 8-bits resolution

- ❖ Total error of ±¼ LSB, ±½LSB and ±1 LSB

- ❖ Conversion time of 100μs

- ❖ 1-input channel

- ❖ Parallel interfaced type

- ❖ Minimum supply voltage of 4.5V

- ❖ Maximum supply voltage of 6.3V

❖ External reference source

❖ Minimum temperature of -40°C[15]

The converter was connected to the CD4052 multiplexer as shown in fig 3.5 below:

It was also interfaced with the system controller over P1,(B0 – B7), with P3.2, P3.3

providing the /ADC_CS and /ADC_WR control functionalities respectively.
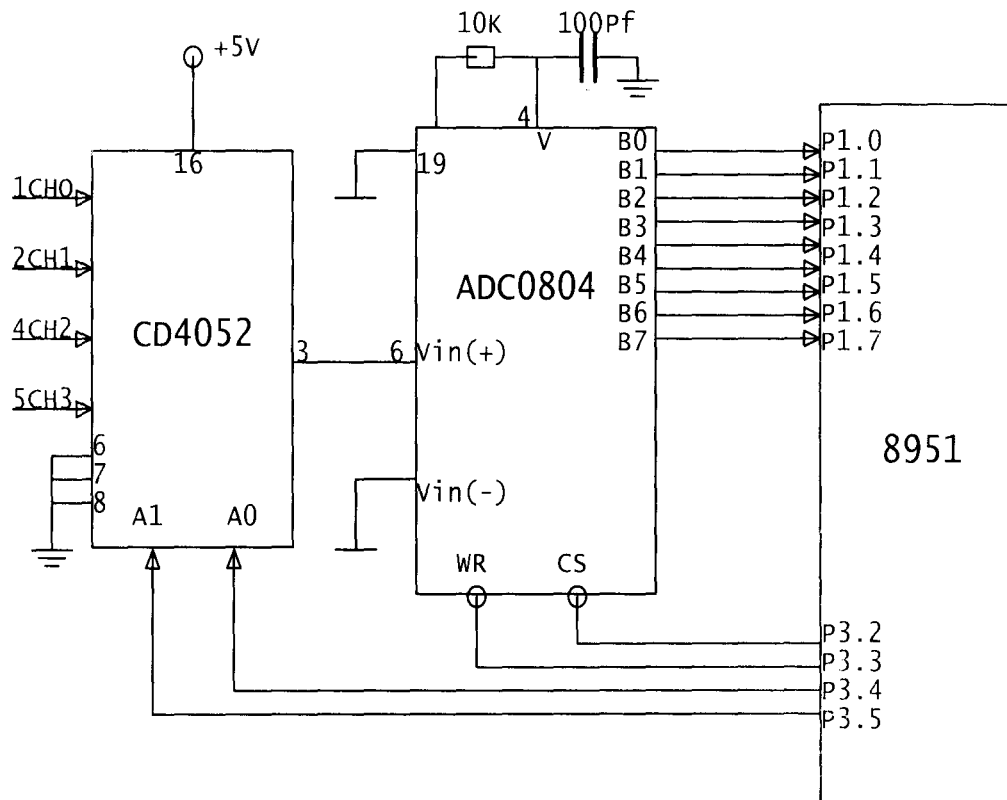
Fig 3.7 circuit connection between CD4052, ADC0804 and 8952 controller

28

The converter was wired for operation on a clock frequency determined by the

RC components connected to Pin 4.

The frequency of operation is obtained thus:

F = 1 / 1.1RC

R = typically 10kΩ, connected between pin 19 and 4

C = 100pF connected between pin4 and 0V

$F = 1/1.1 *10^4*10^2*10^{-12} =14Hz$

The converter was set up for operation with a full scale (11111111_2) input

voltage of 2.56V. this was realized by a potential divider connected to pin 9 (V_{ref})

of the ADC as shown in fig3.8 below:
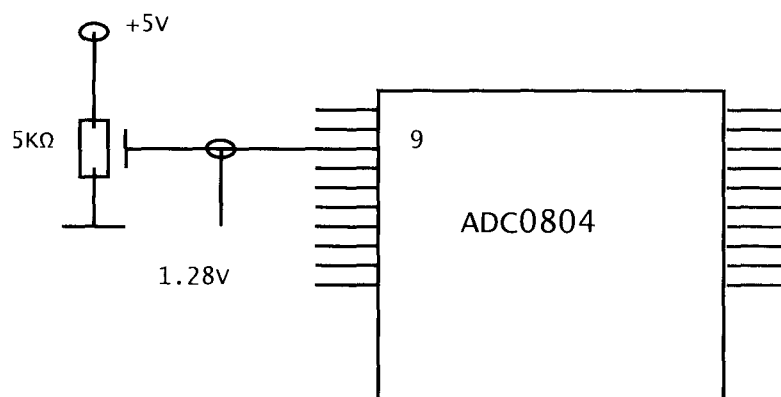
Fig 3.8 ADC reference voltage

With a 2.56 input voltage corresponding to FFh hexadecimal, the ADC output

voltage therefore, has a 1:1 relationship between any applied input Dc voltage and

the output binary value i.e., the ADC exhibits a LSB output change for a 10mV input charge corresponding to a LSB charge/°C.

The temperature value from the converter was converted to ASCII decimal value and sent over the serial link to the host system where it was compared against preset upper and lower thresholds respectively. The result of the comparison is then translated into commands that are sent down the serial interface from the host PC to the control box. This is to either keep the heater unit in the location under monitoring on or hot.

3.4   RS232 LOGIC LEVEL TRANSLATOR

Because the device is a distance part that does not respond to logic levels more negative than 0V, a means of interfacing the controller with the bipolarity signaling voltages on the RS232 interface was needed. A converter that translate the CMOS level 0V/ 5V signal to -12 / +12V signal and the computer serial port.

A transistor implementation was used as depicted in fig3.9 below:
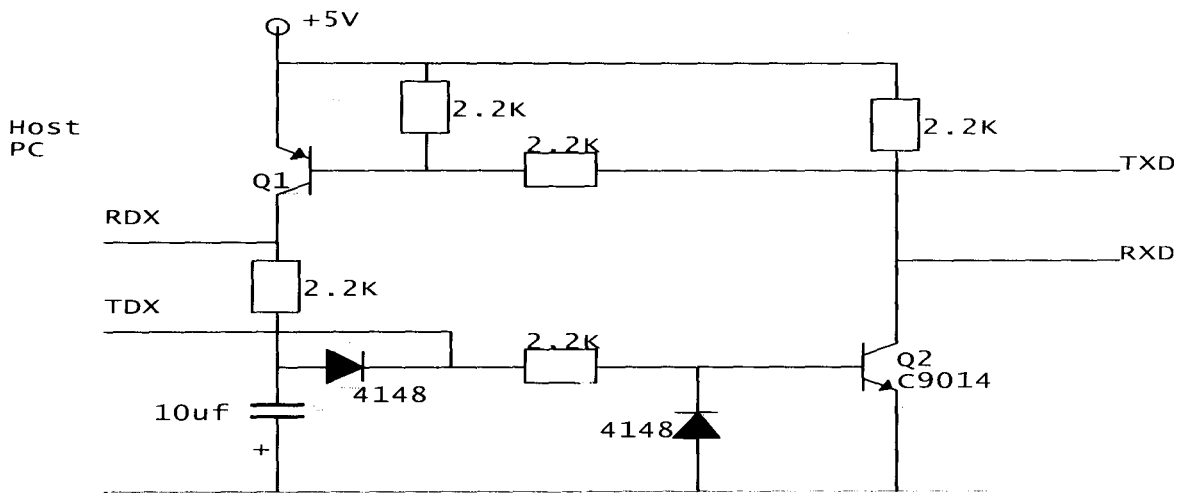
Fig 3.9 RS232 logic level translator.

3.40 CONTROL UNIT

3.41 MICROCONTROLLER

An 8-bit AT8952 microcontroller was used for coordinating system functionality. The device has the following specifications:

❖ Compatible with MCS-51^{TM} products

❖ 8kbytes of in-system reprogrammable flash memeory

❖ Endurance of 1000 write/erase cycles

❖ Fully static operation : 0Hz to 24MHz

❖ Three-level program memory lock

- ❖ 256*8-bit internal RAM

- ❖ 32 programmable I/O lines

- ❖ Three 16-bit timer/counters

- ❖ Eight interrupt sources

- ❖ Programmable serial channels

- ❖ Low-power idle and power-down modes.[14]

The controller pin out is shown in fig 3.10 below:

1	p1.0(T2)	Vcc	40
2	p1.1(T2Ex)	(AD0) p0.0	39
3	p1.2	(AD1) p0.1	38
4	p1.3	(AD2) p0.2	37
5	p1.4	(AD3) p0.3	36
6	p1.5	(AD4) p0.4	35
7	p1.6	(AD5) p0.5	34
8	p1.7	(AD6) p0.6	33
9	RST	(AD7) p0.7	32
10	p3.0(RXD)	$\overline{EA}$ / Vpp	31
11	p3.1(TXD)	ALE /$\overline{PROG}$	30
12	p3.2(INT0)	PSEN	29
13	p3.3(INT1)	(A15) p2.7	28
14	p3.4(T0)	(A14) p2.6	27
15	p3.5(T1)	(A13) p2.5	26
16	p3.6($\overline{WR}$)	(A12) p2.4	25
17	p3.7($\overline{RD}$)	(A11) p2.3	24
18	XTAL1	(A10) p2.2	23
19	XTAL2	(A 9) p2.1	22
20	GND	(A 8)p2.0	21

Fig 3.10 AT89C52 pin out.

32

He device was interfaced with the ADC over P1, the PC serial port over P3.0,P3.1 and the load control switches over P2.

It was connected to an 11.0592MHz crystal with a baud rate of 9600bps corresponding to 960 character/second. The flow chart for the program is shown:

```
         ┌─────────────┐
        (    START      )
         └──────┬──────┘
                │
                ▼
       ┌─────────────────┐
       │   SENSOR = 0    │
       └────────┬────────┘
                │
                ▼
  ┌────────────────────────────┐
  │  DIGITIZE SENSOR VOLTAGE   │
  └──────────────┬─────────────┘
                 │
                 ▼
  ┌────────────────────────────┐
  │       SEND TO SYSTEM       │
  └──────────────┬─────────────┘
                 │
                 ▼
  ┌────────────────────────────┐
  │    SENSOR = SENSOR +1      │
  └──────────────┬─────────────┘
                 │
                 ▼
              ◇ SENSOR≥ 8 ◇        NO
                 │
                YES
                 │
                 ▼
       ┌─────────────────┐
       │   SENSOR = 0    │
       └────────┬────────┘
                │
                ▼
  ┌────────────────────────────┐
  │   SEND "N" TO SYSTEM       │
  └────────────────────────────┘
```

34

"A" — YES → RELAY 1=OFF

NO ↓

"B" — YES → RELAY 1= ON

NO ↓

"C" — YES → RELAY 2= OFF

NO ↓

"D" — YES → RELAY2 =ON

NO ↓

"E" — YES → RELAY3=OFf

NO ↓

"F" — YES → RELAY3= ON

NO ↓

"G" — YES → RELAY4=OFF

NO ↓

"H" — YES → RELAY4=ON

35
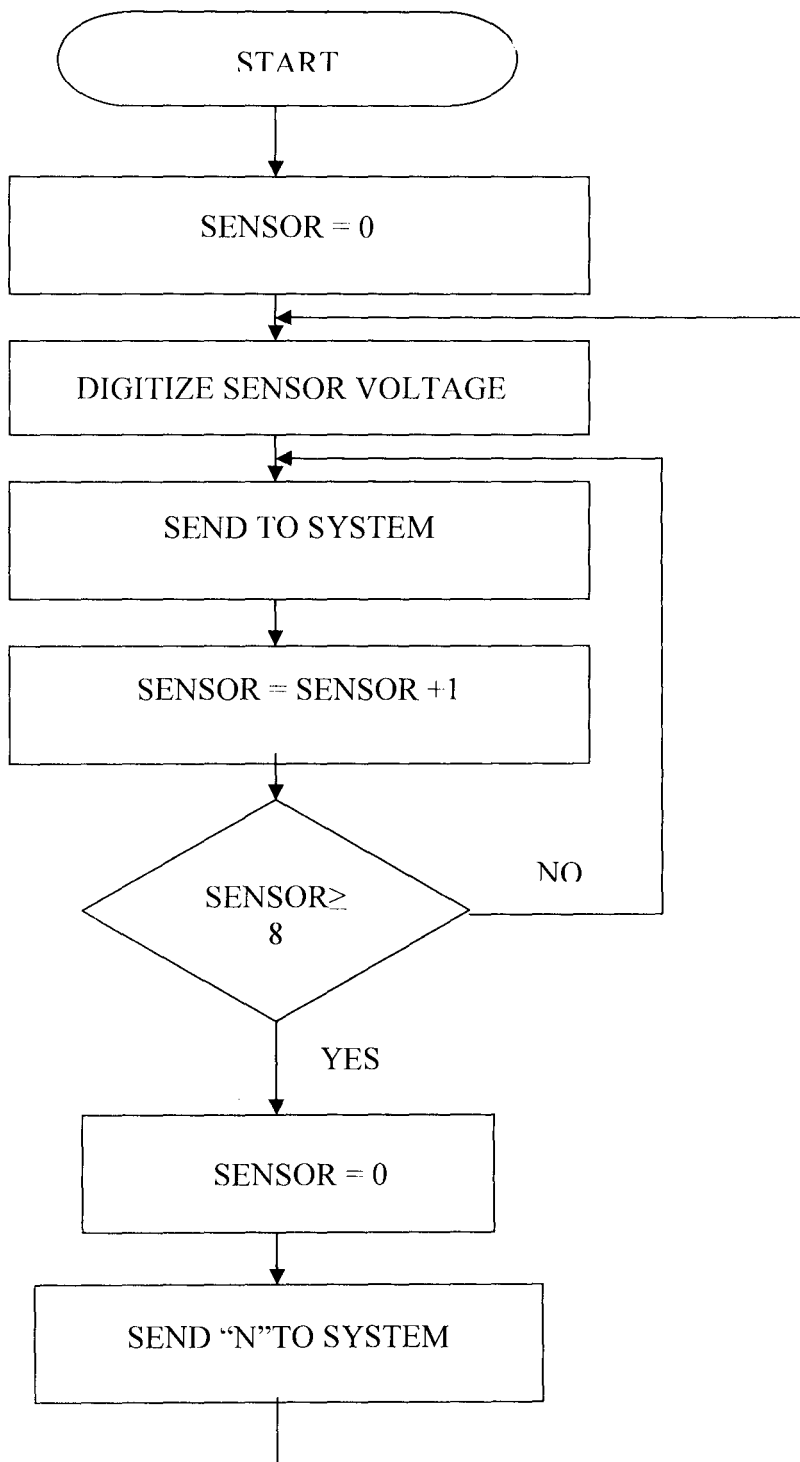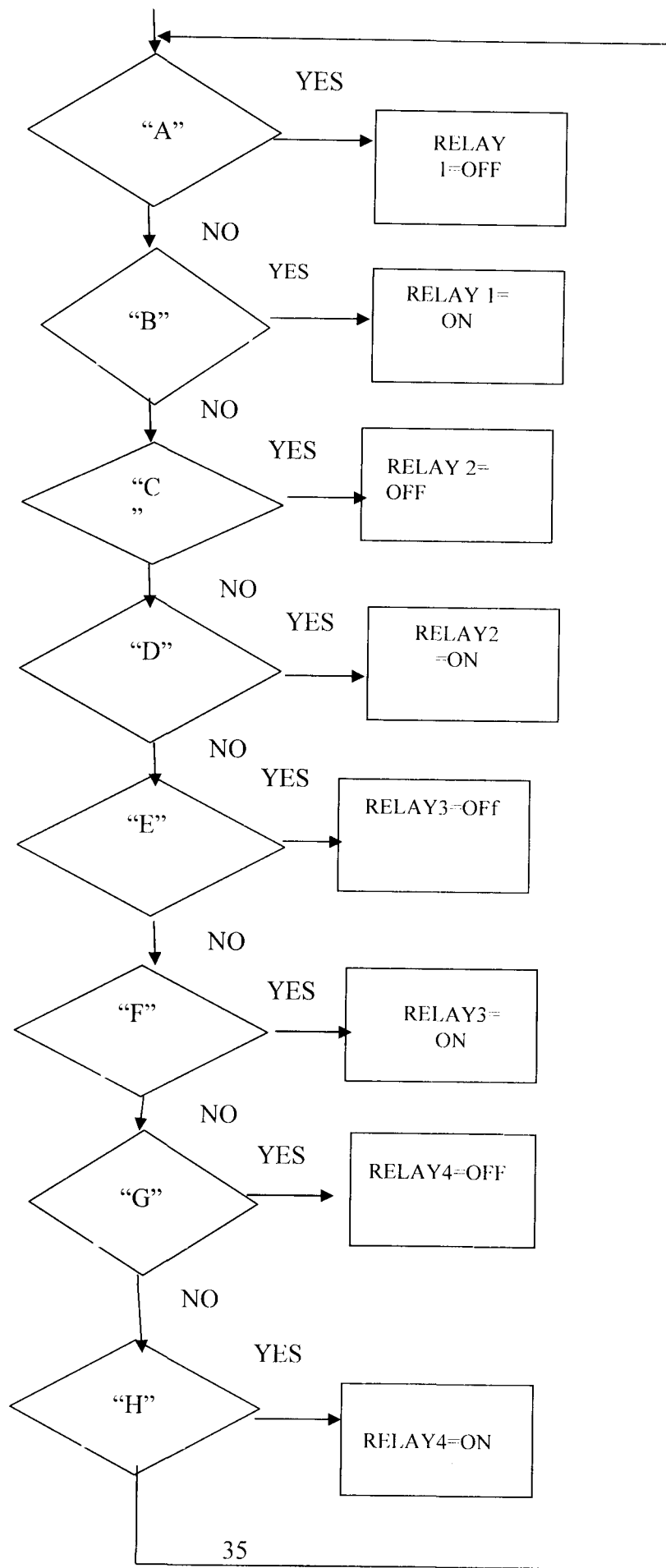
The controller was used to:

1. Make a correction of the ambient temperature from the analog to digital domain via the ADC0804.

2. Execute heater ON/OFF sequence under host PC control.

3.4.2 CLOCK SOURCE

The clock source unit is made up of a 11.0592MHz crystal. This unit generates the clock pulse for the microcontroller to execute its instructions.

The crystal 11.0592MHz generates pulses in one second. Normally an 8051 compatible microcontroller executes one instruction in 12 clock pulses. Therefore, the microcontroller will be able to perform 11059200 divided by 12 instructions in one second.

3.4.3 HOST PC CONTROL SOFTWARE

A control software was written for the host PC. The software was written in **VISUAL BASIC (VB)** for user interactivity.

The software was written basically to:

I.   Read the rates of temperature sensed by the four sensors.

II.   Compare the sensed temperature with the user upper and lower preset temperatures.

If the sensed temperature violates the user-defined temperature, the host PC

sends control commands to open or close the relay element associated with the

heating unit.

Also, a log file was created for storing the temperature data sent from

the control unit for safe keeping.

3.5 HEATER CONTROL UNITS

Four heaters were designed for the interface, hence four power

switching units were interfaced with 89C52. the power units were realized

using 6V 10A DC electromechanical relays driven by pnp transistors as
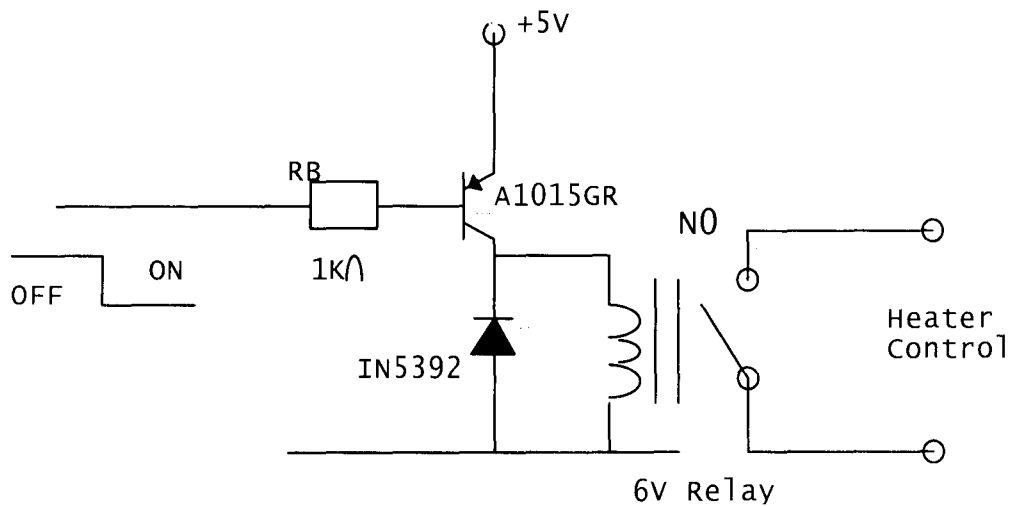
shown in fig3.11 below.

Fig 3.11 6V 10A relay.

PNP silicon transistors were used as activated switches to source current

into the relay coil winding causing activation.

The relay current was deduced from the expression:

$$I_{coil} = V_{coil} / R_{coil} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots III$$

V_{coil} was set by the manufacturer, R_{coil} was measured as 400Ω

$$I_{coil} = 6/400 = 15mA \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots IV$$

This coil current therefore becomes he collector current of Q1 - Q4. However relays typically need to have about ½ their rated coil voltages maintain closure after initial energizing. Thus the minimum collector current needed for ther relays was used.

$$I_{min} = 3/400 = 7.5mA \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots V$$

The above value of coil current is thus he minimum needed to keep the contacts closed at 3V coil voltage. The four transistors were biased on by base resistances R_B of value deduced for the expression.

$$R_B = V_{CC} - V_{BE}/I_B \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots VI$$

However, $I_B = I_C/H_{fe} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots VII$

The transistor selected has a typical DC current gain of 200

$$I_B = 0.0075/200 = 38\mu A \dots\dots\dots\dots\dots\dots\dots\dots\dots VIII$$

$$R_B = V_{CC} - V_{BE}/I_B$$

$$= 5 - 0.7/38 = 113k\Omega \dots\dots\dots\dots\dots\dots\dots\dots\dots IX$$

The value of R_B above is the maximum resistance usable to meet the stipulated relay switching requirement.

3.6 IMPLEMENTAION

The implementation of this project was done by the connection of the various units as explained above. This and other connections are indicated on the project circuit diagram shown in fig3.12 below:
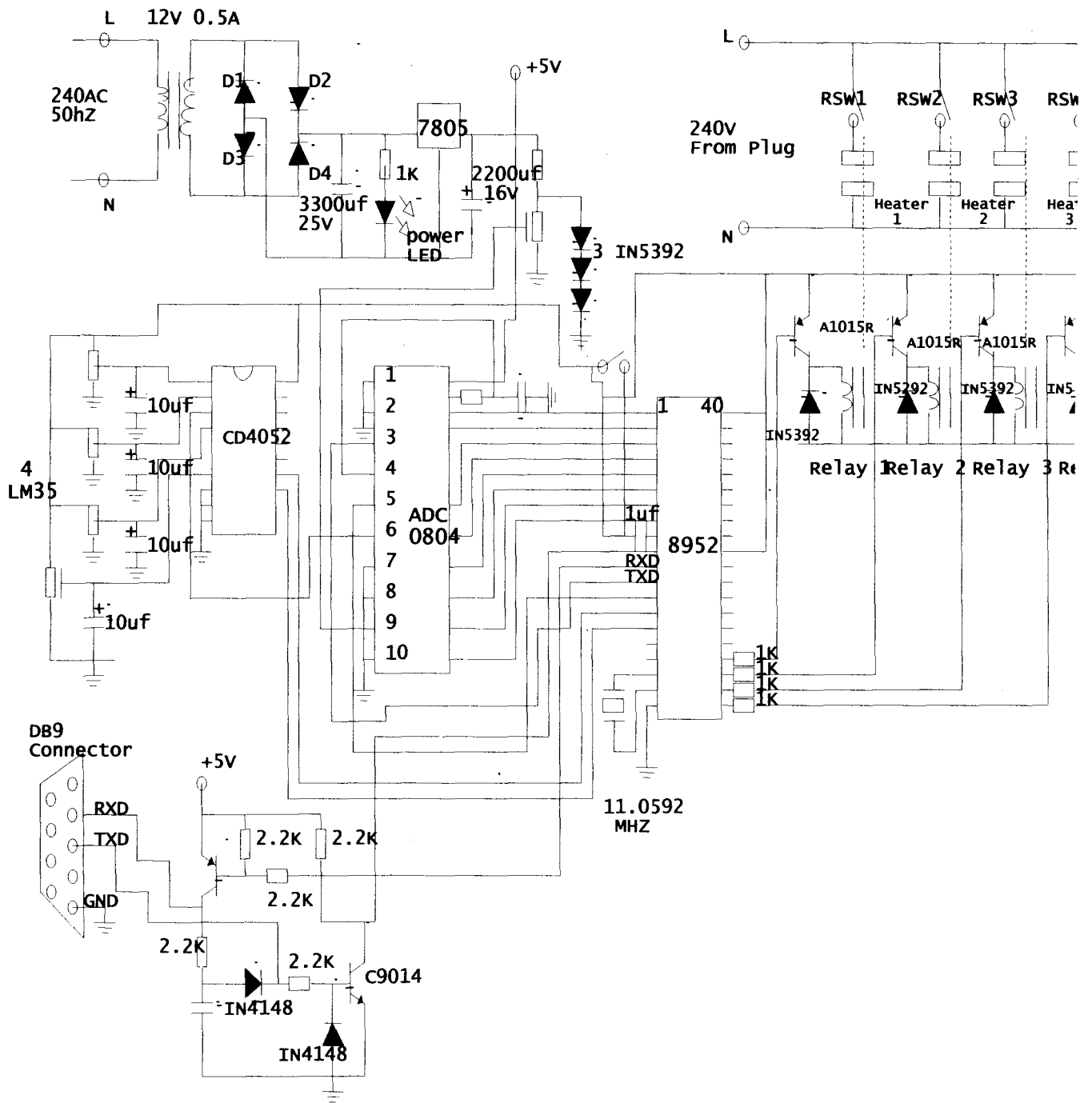
fig 3.12 circuit diagram of microcontroller-based-temperature data acquisition and logging system.

CHAPTER FOUR

TESTING, RESULTS AND DISCUSSION

4.1 TESTING

The software for the AT89C51 microcontroller was tested by running I prog studio 6.09 assembler. The program was debugged after which, it was burnt into the microcontroller chip. The program for the interface with the computer was also tested and compiled. After proper design of the circuit, the project was initially constructed on a breadboard and tested, using a digital multimeter, before it was finally soldered to a veroboard.

Further testing was carried out using a set point of $37^{o}C$ to measure the temperature of four students. Another test was carried out on the project by examining the accuracy of the microcontroller base temperature acquisition and logging system with a standard data acquisition and logging system.

4.2   RESULT

The display on the data acquisition panel on the computer is shown below at different

set points of temperature.
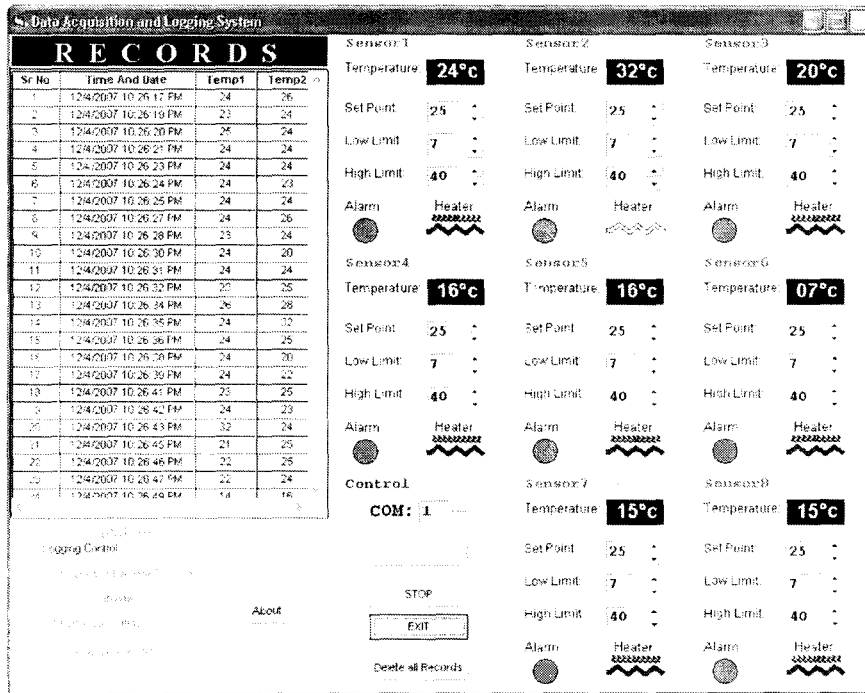
Fig 4.1 project output display

4.3   DISCUSSION OF RESULT

The output display of the project shows that the project is suitable for

the desired purpose of collecting, monitoring, analyzing and storing datas of system's

temperature for the purpose of improvement and modification.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 CONCLUSION

The performance of the design and construction of this project met the design specifications. It accomplishes he aims and objectives of the project, which is to design construct a prototype that can enable one to collect the various temperature values of four system, monitor them, analyze them and store them for improvement in efficiency of performance of the systems.

The project was constructed in such a way that it could be easily maintained and repaired. Moreover, the use of a microcontroller and a crystal oscillator in the design made the whole system efficient and accurate.

5.2 RECOMMENDATION

This project wish to make the following recommendations based on some areas of the design that can be improved upon, they are:

1) An alarm circuit should be included in the design to make the project more informative in place of the heaters.

2) More sensors can be added to expand the numbers of systems to be monitored by introducing more channels through the use of a multiple channel analog to digital converter (ADC) such as 0808(an eight channel ADC) so as to be able monitor more systems.

3) A backup power supply system can be incorporated in case of any PHCN power

interruption.

REFERENCES

[1]   hpp:// physics.nist.gov/GenInt/Time/ancient.html

[2]   hpp:// physics.nist.gov/GenInt/Temp/ early.html

[3]   hpp:// physics.nist.gov/GenInt/Time/revol.html

[4]   hpp:// en.wikipedia.org/wiki/Data acquisition system

[5]   hpp:// en.event history.org/Isead/Data acquisition

[6]   hpp:// data acquisition toolbox

[7]   www.picotech.com/remote_data_acquisition.html

[8]   www.pelago.co.uk

[9]   Charles A. Schuler, Electronic principles and applications, 5th edition, Glenco McGraw-hill, Columbus,1999, page 415

[10]   hpp:// en.wikipedia.org/wiki/voltage_regulator

[11]   Edward Hughes Electrical and Electronics technology, 8th edition, Pearson education (Singapore) pte.Ltd, India, 2004,page 509.

[12]   B.L Theraja and A.K Theraja, a textbook of Electrical Technology, Revised 23rd edition, S.Chand and company limited, New Delli;2003 page 2182.

[13]   www.raional.com/precision/temp/sensors.pdf

[14]   Atmel AT89C52 data sheet.

[15]   national semiconductor ADC0804 data sheet.

APPENDIX 1

SOURCE CODE FOR THE MICROCONTROLLER AT89C52

```
INCLUDE 89c51.mc

adc_write BIT p3.2

adc_intr BIT p3.3

relay1 BIT  p2.0

relay2 BIT p2.1

relay3 BIT p2.2

relay4 BIT P2.3

stack EQU 60h

adc_port EQU p1

temp DATA 9

count1 DATA 10

count2 DATA 11

temp2 DATA 12

sensor DATA 13

reading DATA 14

hi DATA 15

lo DATA 16
```

```
;**********************************************************

org 0000h

LJMP start

;**********************************************************

org 0003h

LJMP start

;**********************************************************

org 000bh

LJMP start

;**********************************************************

org 0013h

LJMP start

;**********************************************************

org 001bh

LJMP start

;**********************************************************

org 0023h


serial_isr:         JB ri,go_a

                            RETI


go_a:                       PUSH acc
```

```
                    MOV A, sbuf

                    CJNE A,#"a", go_1

                    CLR relay1

                    SJMP exit_isr

go_1:               CJNE A,#"b", go_2

                    SETB relay1

                    SJMP exit_isr

go_2:               CJNE A,#"c", go_3

                    CLR relay2

                    SJMP exit_isr

go_3:               CJNE A,#"d", go_4

                    SETB relay2

                    SJMP exit_isr

go_4:               CJNE A,#"e", go_5

                    CLR relay3

                    SJMP exit_isr

go_5:               CJNE A,#"f", go_6

                    SETB relay3

                    SJMP exit_isr

go_6:               CJNE A,#"g", go_7

                    CLR relay4

                    SJMP exit_isr

go_7:               CJNE A,#"h", go_8
```

```asm
                    SETB relay4

go_8:

exit_isr:        POP ACC

                 CLR ri

                 RETI

;*********************************************

delay:           mOV count1,#0

delay_lp:        MOV count2,#0

                 NOP

                 NOP

                 djnz count2,$

                 DJNZ count1, delay_lp

                 RET



;*********************************************************

init_serial:     MOV tmod,#22h

                 MOV th1,#0fdh

                 MOV tl1,#0fdh

                 MOV scon,#50h

                 SETB tr1

                 SETB es
```

```
                    SETB ea

                    SETB REN

                    CLR TI

                    RET




;*****************************************************

send_Char:      JNB ti,$

                        CLR ti

                        MOV sbuf,temp2

                        RET

;********************************************

reset_Delay:    MOV R7,#0

reset_loop:             MOV R6,#0

loop_1:                 NOP

                            NOP

                            NOP

                            NOP

                            DJNZ R6,loop_1

                            DJNZ R7, reset_loop

                            RET

;*************************************************


start:          CLR ea
```

```
                MOV sp,#stack

                ACALL reset_Delay

                MOV p1,#0ffh

                MOV p2,#0ffh

                MOV p0,#0ffh

                ANL p3,#00001111b

                MOV sensor,#0

                ACALL init_Serial

                ACALL delay_2_send

                ACALL delay_2_Send

                ACALL delay_2_Send


;*********************************************


mainloop:       MOV A, sensor

                CJNE A,#8, chk_Again

skip_back:      MOV sensor,#0

                SJMP skip_on

chk_again:      JNC skip_back

skip_on:        MOV A, sensor

                JNZ skip_2

                ACALL send_n
```

```asm
skip_2:         ANL p3,#00001111b

                MOV A, sensor

                RL A

                RL A

                RL A

                RL A

                ORL p3,A

                MOV R0,#0

                DJNZ R0,$

                CLR adc_Write

                NOP

                NOP

                SETB adc_Write

                MOV R0,#0

                DJNZ R0,$

                MOV reading,adc_port

                ACALL send_Reading

                INC sensor

                SJMP mainloop
```

;***
;

```
send_n:          MOV temp2,#"N"

                 ACALL send_char

                 ACALL delay

                 RET

;*****************************************************

array: DB "0","1","2","3","4","5","6","7","8","9"

;*************************************************


send_Reading:    MOV DPTR,#array

                 MOV A,reading

                 MOV B,#10

                 DIV ab

                 MOVC A,@a+dptr

                 MOV hi,A

                 MOV A,B

                 MOVC A,@a+dptr

                 MOV lo,A

                 MOV temp2, hi

                 ACALL send_char

                 ACALL delay_2_send

                 MOV temp2, lo

                 ACALL send_char

                 ACALL delay_2_send
```

```
            MOV temp2,#","

            ACALL send_Char

            ACALL delay_2_send

            RET


delay_2_send:    ACALL delay

            RET
```