

DEVELOPMENT OF COMPUTER AIDED MEASURING AND
MONITORING DEVICE FOR TEMPERATURE, VOLTAGE AND
RESISTANCE.

BY

CHIDI IWUANYANWU

(93/3548)

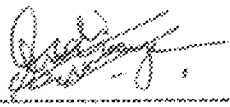
A PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE AWARD OF BACHELOR OF
ENGINEERING (B. ENG.)
DEGREE IN THE

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

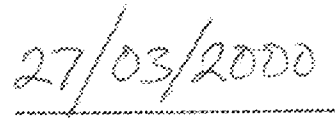
MARCH, 2000

DECLARATION

I hereby declare that this project work is my original work and has never been submitted elsewhere before.



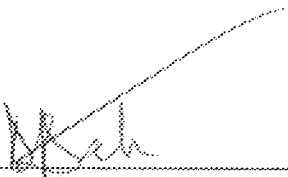
CHIDI IWUANYANWU



DATE

CERTIFICATION

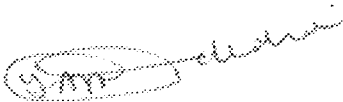
This is to certify that this project titled "Development of Computer Aided Measuring and Monitoring Device for Voltage, Resistance and Temperature" was carried out by Chidi Iwuanyanwu under the supervision of Mr. I. A. Danjuma and submitted to Electrical and Computer Engineering Department, Federal University of Technology, Minna. in partial fulfillment of the requirement for the award of Bachelor of Engineering (B. ENG) degree in Electrical and Computer Engineering.



MR. I. A. DANJUMA
Project Supervisor

27/03/2000

DATE



ENGR (DR) Y. A. ADEDIRAN
Head of Department

6/4/2000

DATE



DR. J. O. ONI
External Examiner

6/4/2000

DATE

DEDICATION

This project is dedicated to the Almighty and only true *God*, through *Jesus Christ my Lord*. To my beloved parents, *Mr and Mrs M. A. Iwuanyanwu*.

ACKNOWLEDGEMENT

First and foremost, my gratitude goes to God for his guidance, wisdom, provision and protection through out my course of study.

I like to say special thanks to my supervisor Mr. I. A. Danjuma for his guidance and contributions.

My gratitude goes to my parents Mr and Mrs M. A. Iwuanyanwu for their sacrifice and relentless effort, for the love and tender care in ensuring that I have a sound education. I will like to use this opportunity to thank Chukuma Ajah, Uzoma Onuachu, Fred Aganga, Chibuike Ekpo and others who helped me to see the project through.

ABSTRACT

This project presents a Computer Aided Measuring and Monitoring Device. It is an integration of a software and peripheral hardware via the printer parallel interface port of a Host microcomputer (the IBM PC). The process involves converting the Analogue quantity to be measured into its digital equivalent, which is read and processed by the software. The resultant value is displayed on the visual Display Unit of the Host microcomputer, as the measured value.

However, this development enhances greater flexibility and speed in terms of measurement.

Hence, it is employed to replace human observers as a matter of convenience, as in monitoring industrial processes.

TABLE OF CONTENTS

TITLE PAGE										
DECLARATION	-	-	-	-	-	-	-	-	-	i
CERTIFICATION	-	-	-	-	-	-	-	-	-	ii
DEDICATION	-	-	-	-	-	-	-	-	-	iii
ACKNOWLEDGMENT	-	-	-	-	-	-	-	-	-	iv
ABSTRACT	-	-	-	-	-	-	-	-	-	v
CHAPTER ONE										
1.0 INTRODUCTION	-	-	-	-	-	-	-	-	-	1
1.1 AIMS AND OBJECTIVES	-	-	-	-	-	-	-	-	-	1
1.2 LITERATURE REVIEW	-	-	-	-	-	-	-	-	-	2
1.3 PROJECT OUTLINE	-	-	-	-	-	-	-	-	-	5
CHAPTER TWO										
2.0 SYSTEM DESIGN	-	-	-	-	-	-	-	-	-	6
2.1 OPERATION OF THE PRINTER PORT	-	-	-	-	-	-	-	-	-	6
2.2 THE TRANSDUCER AND ITS CIRCUIT	-	-	-	-	-	-	-	-	-	10
2.3 SIGNAL CONDITIONING CIRCUIT	-	-	-	-	-	-	-	-	-	13
2.3.1 FOR TEMPERATURE	-	-	-	-	-	-	-	-	-	13
2.3.2 FOR VOLTAGE	-	-	-	-	-	-	-	-	-	15
2.3.3 FOR RESISTANCE	-	-	-	-	-	-	-	-	-	16
2.4 THE ANALOGUE MULTIPLEXER	-	-	-	-	-	-	-	-	-	17
2.5 ANALOGUE - TO - DIGITAL CONVERTER	-	-	-	-	-	-	-	-	-	19
2.5.1 PRACTICAL ADC CIRCUIT	-	-	-	-	-	-	-	-	-	19

2.5.2	AN A/D CONVERTER IC	-	-	-	-	-	-	-	20
2.6	THE DIGITAL MULTIPLEXER	-	-	-	-	-	-	-	23
2.7	THE SOFTWARE	-	-	-	-	-	-	-	25
2.7.1	THE INPUT/OUTPUT SUBROUTINE	-	-	-	-	-	-	-	25
2.8	POWER SUPPLY	-	-	-	-	-	-	-	30
CHAPTER THREE									
3.0	CONSTRUCTION PROCEDURE	-	-	-	-	-	-	-	34
3.1	TESTING	-	-	-	-	-	-	-	35
3.2	DISCUSSION OF RESULT	-	-	-	-	-	-	-	35
CHAPTER FOUR									
4.0	CONCLUSION	-	-	-	-	-	-	-	36
4.2	RECOMMENDATION	-	-	-	-	-	-	-	36
5	REFERENCE	-	-	-	-	-	-	-	37
	APPENDIX 1	-	-	-	-	-	-	-	38
	APPENDIX 2	-	-	-	-	-	-	-	48

CHAPTER ONE

1.0 INTRODUCTION

Mention 'Computers' to most people and they think of desk-top microcomputers (Personal Computers). Microcomputers vary from single-chip systems, such as pocket calculators through to industrial controllers and multi-user office computers. Computers are, of course, used in science and industry for applications which require the computers to accept input from sources other than a keyboard or disk, and respond by controlling electronic, electrical, or mechanical equipment directly. Here in this project we are going to look at a new way of using the computer for measurement through the printer parallel port.

Before we launch into a discussion on this project let us take some time to understand the topic of the project which is "Computer Aided Measuring / Monitoring Device". First what is measurement? Measurement is the process used to answer the questions: how many and how much? Measurements, broadly defined can be made by the unaided human senses and brain for example, in estimating distances, weights, temperatures, identifying and matching colors, estimating roughness by touch. In general, however, man's capabilities need to be both extended and refined by instruments. Measuring instruments refine human capabilities by providing greater sensibility, range, accuracy, precision, and speed.

With the use of computer to aid in measurement, greater flexibility and speed are obtained. However, most problems encountered in measurement such as nonlinearity of sensors, noise and interference are solved. Computer aided measurement are employed to replace human observers as a matter of convenience, as in monitoring industrial processes.

1.1 AIMS AND OBJECTIVES

The aim of this project is to produce a computer aided measuring and monitoring device which can be used:

- 1) As a multi-meter to measure voltage, resistance and temperature

2) To monitor the temperature of an enclosure or a system.

It also aims at revealing the deep secrets of interfacing IBM personal computer to the outside world through its parallel printer port. How all the pieces of the interfacing and control puzzle fit together, turning the parallel printer-port which was designed for outputting data only into an input and output port.

1.2 LITERATURE REVIEW

The Computer Aided Measuring/Monitoring Device is an intelligent instrument developed and used for monitoring and precise measurement. Although the sensory organs of the human body can be extremely sensitive and responsive, modern science and technology rely on the development of much more precise measuring and analytical tools for studying, monitoring or controlling all kind of phenomena.(page 334 of ref. 2).

Some of the earliest instruments of measurement were used in astronomy and navigation. the armillary sphere, the oldest known astronomical instrument, consist essentially of a skeletal celestial globe whose rings represent the great circles of the heavens. The armillary sphere was known in ancient China, the ancient Greeks were also familiar with it and modified it to produce the astrolabe, which could tell the time or length of day or night as well as measure solar and lunar altitudes. The compass, the earliest instrument for direction finding that did not make reference to the stars, was a striking advance in instrumentation made about the 11th century. The telescope, the primary astronomical instrument, was invented about 1608 by the dutch optician Hans Lippershery and first used extensively by Galileo.(page 334 of ref.2)

Instrumentation developed at rapid pace in the industrial revolution of the 18th and 19th centuries, particularly in the areas of dimensional measurement, electrical measurement, and physical analysis. Manufacturing processes of the time required instruments capable of achieving new standards of linear precision, met in part by the screw micrometer, special models of which could attain a precision of 0.00025mm.

The industrial application of electricity required instruments to measure current,

voltage, and resistance. Analytical methods using such instruments as the microscope and the spectroscope, became increasingly important; the light radiation given off by incandescent substances, began to be used to identify the composition of chemical substances and stars. (page 334 of ref.2).

In the 20th century the growth of modern industry, the introduction of computerization, and the advent of space exploration have spurred still greater development of instrumentation, particularly of electronic devices. Often a transducer, an instrument that changes energy from one form into another (such as thermistor, photocell, or microphone) is used to transform a sample of the energy to be measured into electrical impulses that are more easily processed and stored. The introduction of the electronic computer in the 1950s, with its great capacity for information processing and storage, virtually revolutionized methods of instrumentation, for it allowed the simultaneous comparison and analysis of large amounts of information. (page 334 of ref.2).

Today there are Lots of computerized measuring instruments used in industries. Instrument computerization has been via the internal expansion slot inside the microcomputer, which has the possibility of damaging the entire computer when the instrument is wrongly operated. (Chapter 32 page 288 of ref.4) computerization is also done via the printer port which is safer. But the limitation in using the printer port for instrument computerization is that: there are a maximum of five Lines (5-bits) feeding back into the computer. Which means instrument computerization involving the use of Analogue - to - Digital converters, which translates analogue information into eight - bit or more bytes, can not be efficient.

This Limitation was broken in this project through the use of the 2-input 4-bit digital multiplexer, the 74157 Integrated circuit (IC), together with the low level feature of the C++ programming language used in this project. Which now opens the way to instrument computerization through the IBM PC printer port as presented in this project. (page 286 of ref.4, pages 274-279 of ref.5).

1.3 PROJECT OUTLINE

This project is written to report how a computer Aided Measuring / Monitoring Device (CAMMD) was developed. CAMMD is a development of peripheral hardware and software, for voltage resistance and temperature measurement via the IBM personal computer.

All the information on how the CAMMD was developed is presented in four chapters.

- Chapter one, gives the general introduction of the project. Here, an insight on the whole research is discussed and this includes Introduction, Aims and objectives of the project, literature review and project outline.
- Chapter two, deals with the system design. Discussed here, is the procedures taken in designing the various stages and modules of the project.
- Chapter three, discussed the construction and testing of the project, and also the results obtained are discussed.
- Chapter four contains the conclusion, recommendations and also references of textbooks consulted.

2.0 SYSTEM DESIGN

The Computer Aided Measuring / Monitoring Device (CAMMD) consists of a peripheral hardware, connected to the printer interface port of the IBM PC, and a software which the PC runs to operate this hardware. So it has two principal aspects, Hardware and software. The block diagram of the hardware aspect is shown in Figure 2.0.

The quantity to be measured, if not voltage, is first converted to a voltage signal by the required conversion circuits as shown in Figure 2.0. The voltage signal is then conditioned to the analogue multiplexer by the signal conditioning circuits. Typical functions performed by the signal-conditioning circuits are noise filtering and voltage amplification. Next, the analogue multiplexer, under software control through the decoder, selects the appropriate channel for the signal to be fed to the Analogue-to-Digital Converter (ADC). The ADC is also software controlled, it converts the signal to its digital equivalent. Then, with the aid of the Digital multiplexer, the eight bit digital signal output of the ADC, is fed 4 bits by 4 bits as required into the printer port. The digital data is then processed by the software and the value measured is displayed on the monitor screen.

In this chapter, how the various block diagrams were designed will be treated in more details. However, the principles employed during the design were based on the working limits of the printer interface. That is why a knowledge of the printer interface operation will be made known before further discussions on the various block diagrams.

2.1 OPERATION OF THE PRINTER INTERFACE PORT

The printer interface port is found on any of these two plug-in-cards:

- Display adapter card
- Multi-input/output (I/O) card

(Some PC's will have part of this on the mother board rather than as plug-in-cards). Slots in the motherboard accept these auxiliary cards and you are relatively free to put any type of card in any slot.

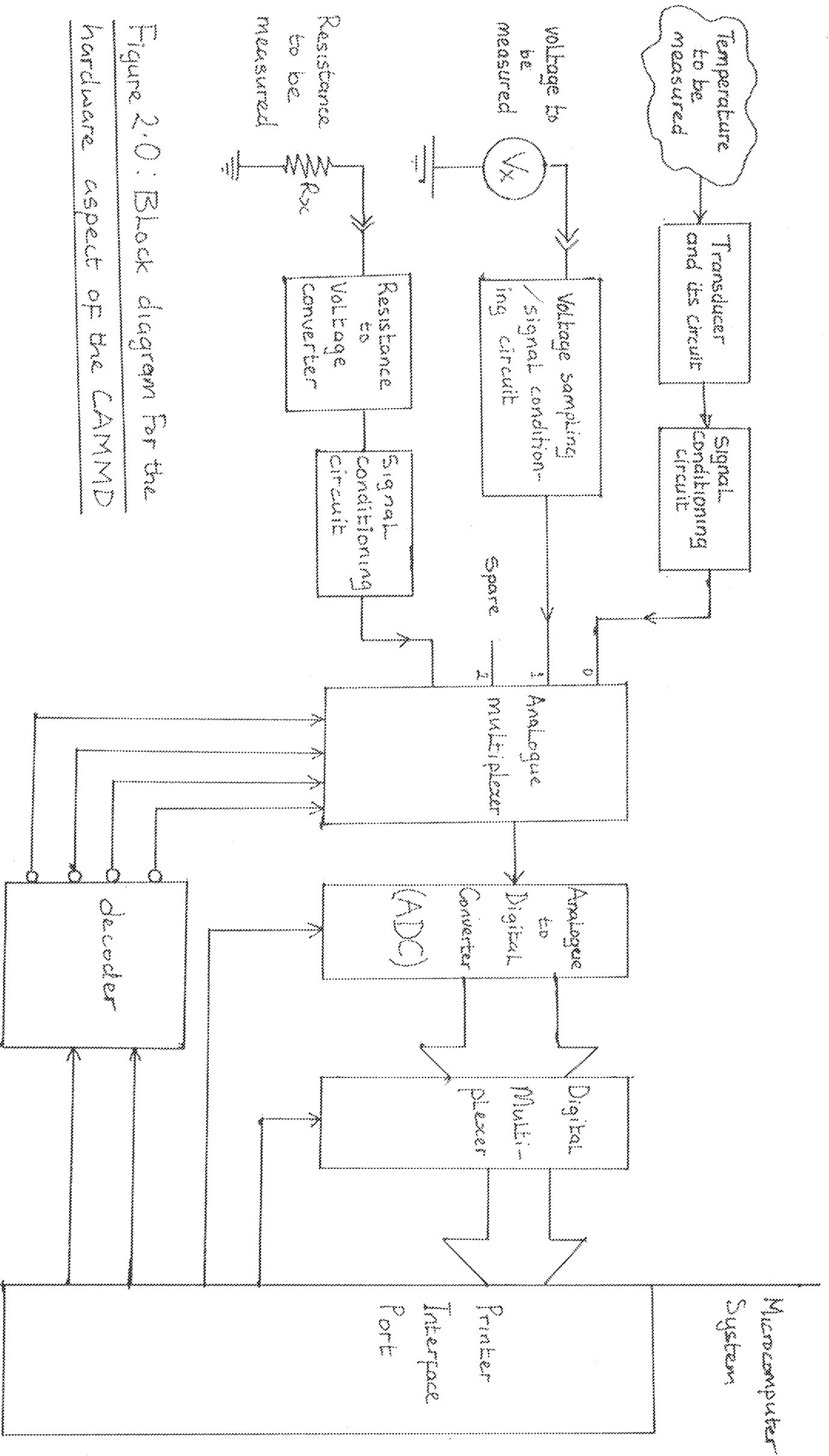


Figure 2.0: Block diagram For the hardware aspect of the CAMMD

The PC addresses, or accesses, its various I/O ports by using a unique address code. Each device or board in the computer has an address that no one else in the PC shares with it.

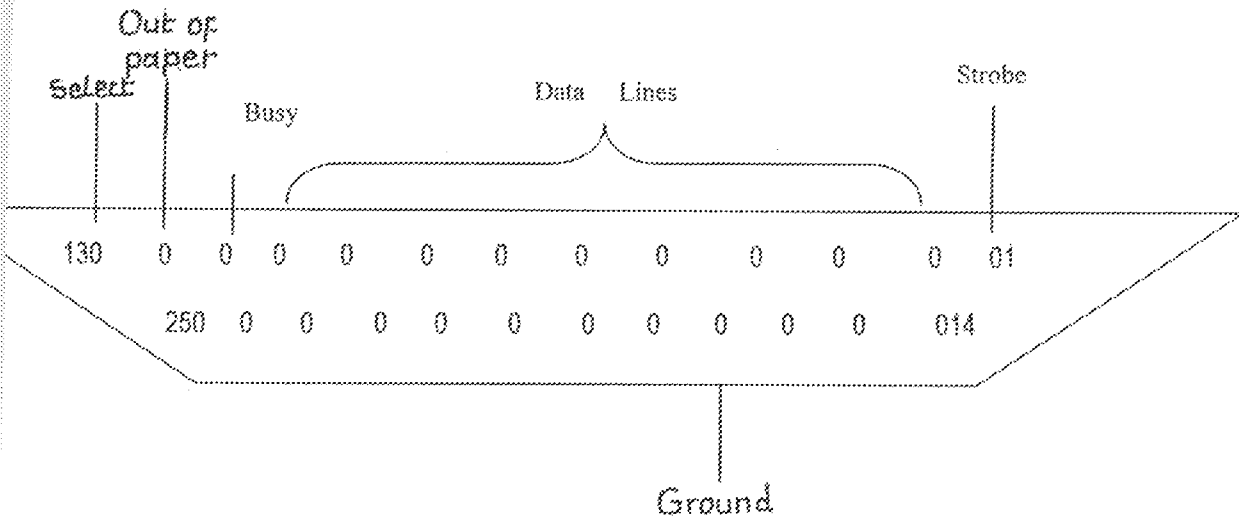
The printer port that comes on the monochrome display adapter has a starting address of 956 in base 10 numbering form. The parallel port contained on the multi-I/O card has a starting address of 888 or 632 all in base 10 numbering. Usually, you specify the address of the port when you install the board. So depending on the adapter or card, the starting address can be 956, 888 or 632. There are up to three ports in the printer interface - output, status and control ports. Therefore the starting address as mentioned above is used for the output port. The next address 957, 889 or 633 is used for the status port. While the next address to the status address, 958, 890 or 634 is used for the control port. Figure 2.1 shows the port map of the printer interface port.

ADDRESS (Dec)	PORT
956 or 888 or 632	<p>Printer Output Port</p> <p>(data bits 7-0)</p>
957 or 889 or 633	<p>Printer status Port</p>
958 or 890 or 634	<p>Printer control port</p>

Figure 2.1 Printer port map

The printer port on the IBM PC is a 25 pin connector, often referred to as a DB-25 connector. Figure 2.2 shows the pin-out designations for the connector and the meaning of the pins. Note that only a little more than half of the pins are use. The others are not connected inside the computer or are grounded to the chassis.

Figure 2.2: PARALLEL PRINTER CONNECTOR

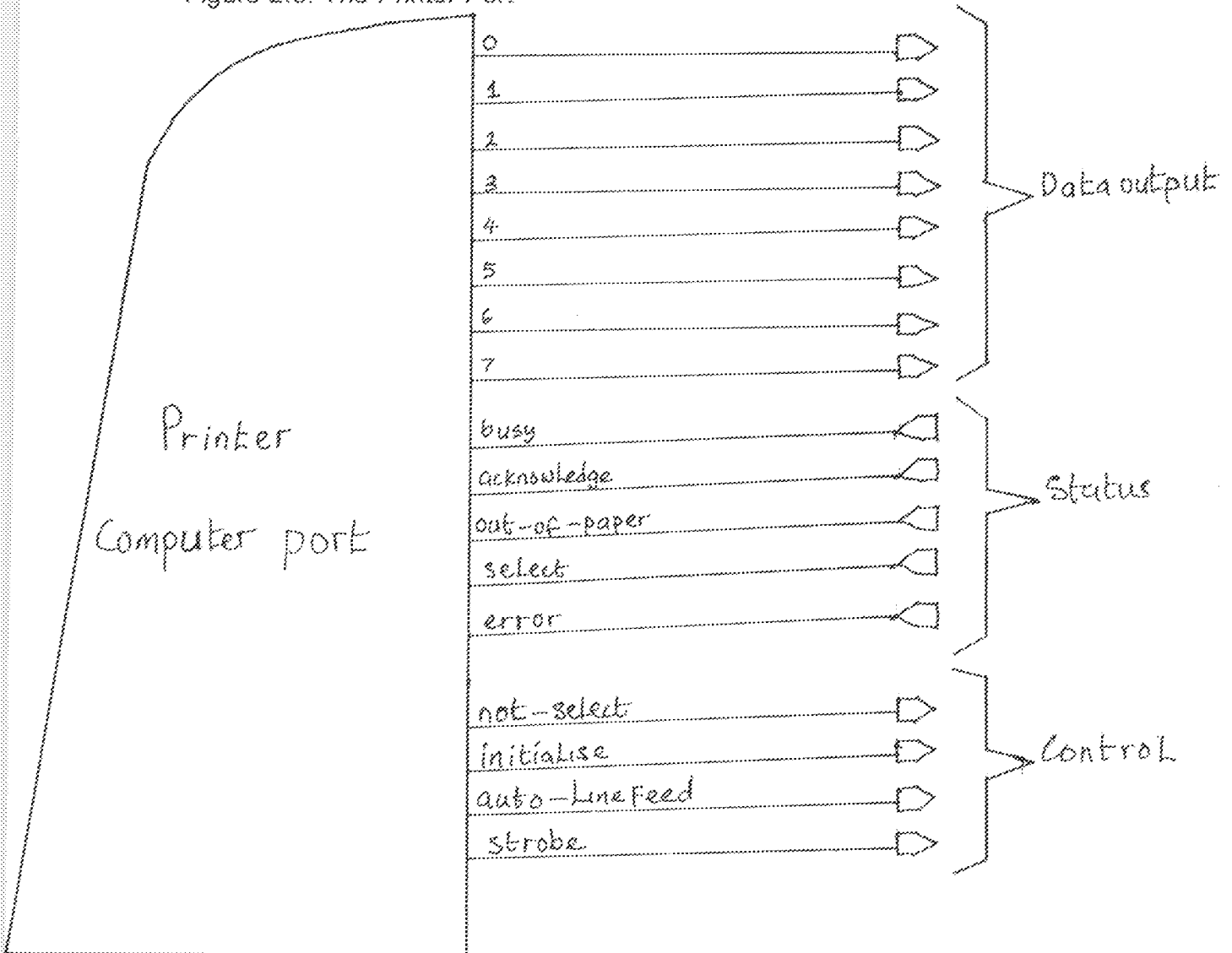


FUNCTION:

1	Strobe (status)
2	Data bit - 0
3	Data bit - 1
4	Data bit - 2
5	Data bit - 3
6	Data bit - 4
7	Data bit - 5
8	Data bit - 6
9	Data bit - 7
10	Acknowledge (status)
11	Busy (status)
12	Out of paper (status)
13	Selected (status)
14	Auto line feed (control)
15	Error (status)
16	Initialise printer (control)
17	Select printer (control)
18	Ground
19	Ground
20	Ground
21	Ground
22	Ground
23	Ground
24	Ground
25	Ground

The data output is comprised of eight binary bits as shown in figure 2.3. The control port is also an output port. The output lines are Latched, meaning that whatever data you place on them stay there until you change it or turn off the computer. As shown in figure 2.3, the control has four out going lines. The status lines are the only ones that feed back into the computer. There are five status lines as shown in figure 2.3.

Figure 2.3: The Printer Port



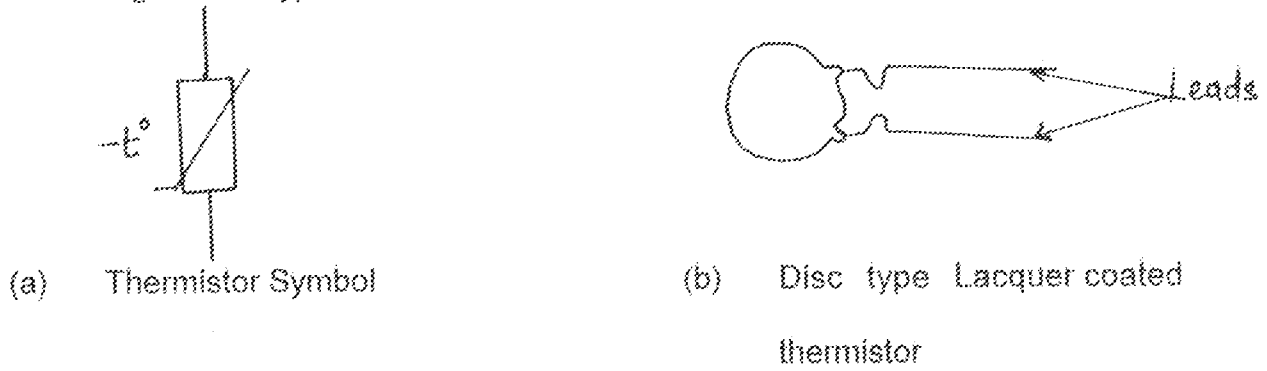
2.2 THE TRANSDUCER AND ITS CIRCUIT

Transducers have the function of converting non-electric input variables to electrical, usually analogue form. As mentioned in previous sections the quantities being measured are: temperature, resistance and voltage. Only temperature is non-electric, therefore a transducer called thermistor, whose resistance changes with temperature, is used.

Thermistors are made by sintering mixtures of oxides of nickel and

manganese, bonding two electrical leads to the sintered material, and enclosing the unit in a protective coating see Fig 2.4. The devices

Figure 2.4 Typical thermistor for temperature sensing purposes



(the $-t^\circ$ indicates a negative temperature characteristic)

are made in a wide variety of shapes (beads, discs, rods, and flakes), are very inexpensive, and very compact. Thermistors are semiconducting devices whose resistance depends exponentially on temperature. This means that the thermistor is very sensitive to temperature, but it also means that its temperature range is limited. Most types have a negative temperature coefficient and some with a positive temperature coefficient are available.

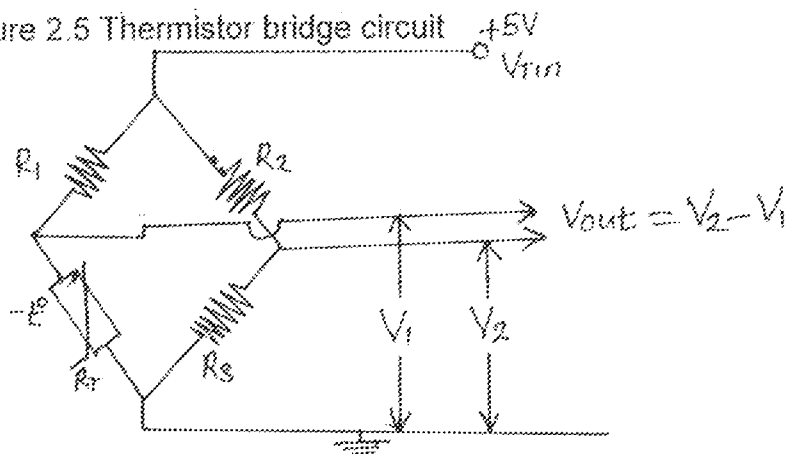
The thermistor used in this project have a negative temperature coefficient. Which means that the resistance of the thermistor decreases when temperature increases.

One of the features of thermistors is that the resulting electrical output per unit change of temperature is invariably tiny. It is about a few millivolts, at best, 1.1mV. This small, possibly slowly varying signal will be contained within noise and interference at frequencies beyond 1Hz. Therefore the signal is handled as differential signal, which require two signal wires. That is the information bearing signal is the difference between the voltages on the two wires, each measured with respect to ground.

Consequently the thermistor is wired into one arm of a Wheatstone Bridge, as shown in figure 2.5, and the circuit is supplied with a voltage, $V_{T_{in}} = +5V$. The output voltage is given by

$$V_{out} = \left(\frac{R_3}{R_3 + R_2} \right) V_{TH} - \left(\frac{R_T}{R_T + R_1} \right) V_{TH}$$

Figure 2.5 Thermistor bridge circuit



Offset adjustment in a bridge arrangement is straightforward. R_3 or R_2 can be trimmed so that at 40°C , the chosen lower limit for the project, the output from the bridge is exactly zero volts. Hence

at 40°C :

$$V_{out} = \left(\frac{R_3}{R_3 + R_2} \right) V_{TH} - \left(\frac{R_T}{R_T + R_1} \right) V_{TH} = 0$$

therefore,

$$\frac{R_3}{R_3 + R_2} = \frac{R_T}{R_T + R_1}$$

This implies that

$$\frac{R_3}{R_2} = \frac{R_T}{R_1}$$

Therefore if $R_2 = R_1$, then $R_3 = R_T$

But at 40°C $R_T = 12910\Omega$, Therefore $R_3 = 12910\Omega = 12.9\text{k}\Omega$

with $R_1 = 10\text{k}\Omega$, the current flowing through the thermistor is given by

$$I_T = \frac{V_{TH}}{R_1 + R_T} = \frac{4.99}{10000 + 12910} = 0.22\text{mA}$$

Therefore the maximum power dissipated across the thermistor is given by

$$P_{T_{max}} = I_T^2 R_T = (0.22 \times 10^{-3})^2 \times 12910 \text{ watts} = 0.612\text{mw}$$

Since this does not exceed the maximum power rating of the thermistor which is 12mw. Therefore $R_1 = R_2 = 10K\Omega$ is used.

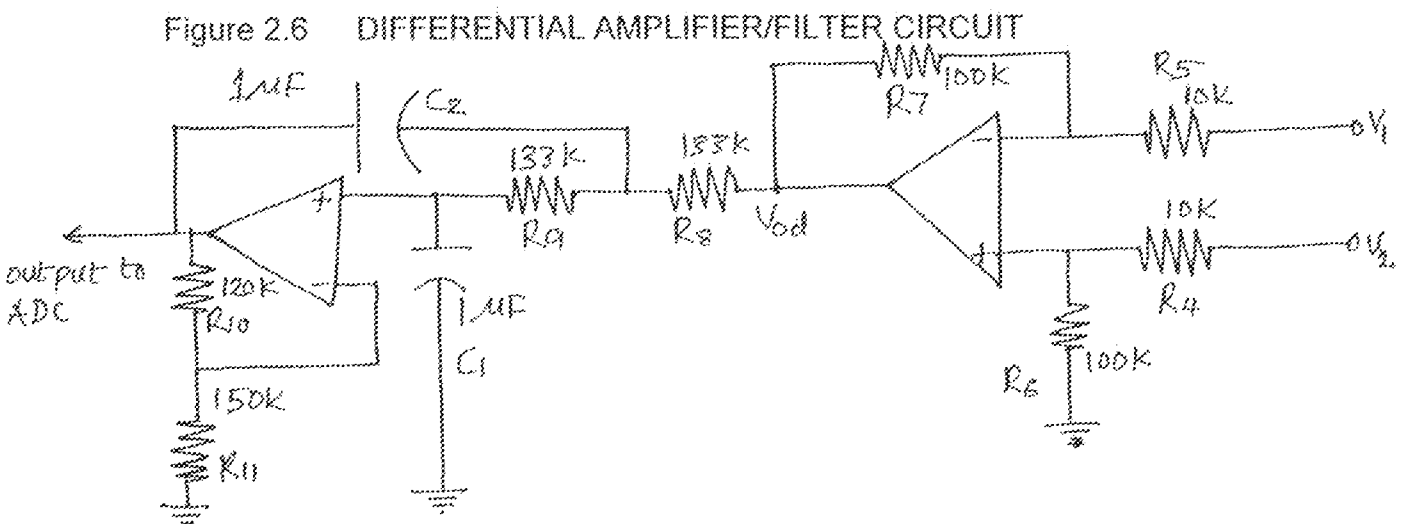
2.3 SIGNAL CONDITIONING CIRCUIT

In the context of digital systems the expression "signal conditioning" covers the shaping, amplification, buffering and filtering of any signal so that it comes within the range and bandwidth requirements of the system input. Normally this entails bringing the signal level up to the input span of an ADC, typically 0V to + 5V. How these were achieved for the three measuring quantities in this project will be discussed in this section.

2.3.1 FOR TEMPERATURE

It has been mentioned in section 2.2, the reason why the thermistor signal is handled as a differential signal and how it was shifted to zero at 40°C which is optionally the lowest temperature to be measured.

The design of the signal conditioning circuit depending on the measurement requirements. Here it is required to produce a highly sensitive temperature measuring device. Therefore the change in output of the thermistor circuit for a unit change in temperature, 1.1mV/°C, is amplified to the smallest change in voltage, 20mV which the ADC can detect. This, couple with the fact that noise interferes with the required signal, lead to the use of a differential amplifier system with a filter circuit as shown in figure 2.6



The output voltage of the differential amplifier is given by:

$$V_{od} = \left(\frac{R_6}{R_3 + R_4} \right) \left(1 + \frac{R_7}{R_5} \right) V_2 - \left(\frac{R_7}{R_5} \right) V_1$$

if the ratio of $\frac{R_7}{R_5}$ and $\frac{R_6}{R_4}$ are made equal we have

$$V_{od} = \frac{R_7}{R_5} (V_2 - V_1)$$

Therefore the gain A_d of the differential amplifier is given by

$$A_d = \frac{R_7}{R_5} = -2.2$$

Since the signal is contained within noise and interference signals, the bandwidth of the signal must be limited to reduce noise and minimize any aliasing effect. So, bandwidth limiting of the differentially amplified system is set to 1Hz by a Sallen - and - key second-order low-pass filter named after its inventors, where the characteristic frequency is given by

$$\begin{aligned} F_c &= \frac{1}{2\pi \sqrt{C_1 C_2 R_3 R_6}} \\ &= \frac{1}{2(3.1416)(1 \times 10^{-6} \times 133 \times 10^3)} \text{ (Hz)} \\ &= 1.2 \text{ Hz} \end{aligned}$$

with this circuit a transfer function that is maximally flat in the passband is achieved.

The gain of the filter circuit which is a non-inverting amplifier is given by:

$$A_v = 1 + \frac{R_{10}}{R_{11}}$$

Therefore the overall gain of the amplifier system from equation (2.2) is given

by

$$A = A_d \times A_v = \frac{R_7}{R_5} \left(1 + \frac{R_{10}}{R_{11}} \right)$$

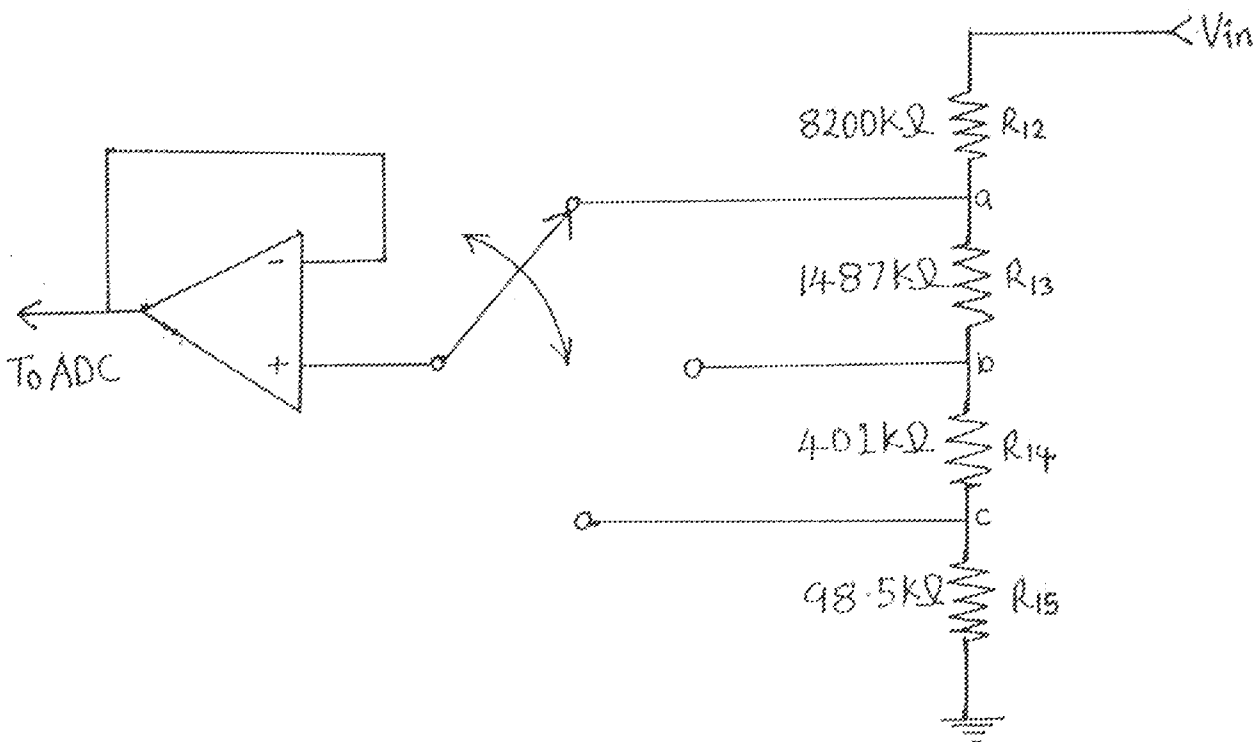
$$= \frac{100k}{10k} \left(1 + \frac{120k}{150k} \right) = 18$$

with $A = 18$, the final output is then $20\text{mV}/^\circ\text{C}$ which is suitable to drive an 8-bit ADC. Although, there exist an appreciable common mode voltage output of about 110mV in this case, it does not impose any problem since software takes care of its effect through the use of look up table. Another limitation of this differential amplifier is its low differential input impedance which is only $2R_s$. This impedance loads the differential signal source (the bridge circuit) and can attenuate it. This problem was solved by the provision of two buffer amplifiers between the bridge circuit and the differential amplifier. The two input buffer amplifiers provide very high impedance to the bridge circuit.

2.3.2 FOR VOLTAGE

The work of the signal conditioning circuit here is to reduce any high voltage range signal, so that it comes within the range of the ADC, 0 to 0.5V . So a precision metal-film resistor network is used. This permits higher voltages to be measured. The network and how it is switched is shown in figure 2.7.

Figure 2.7 Range switching/Potential divider resistor Network



The voltage division down the network is not uniform, since the design was based on availability of components. But, this poses no problem since the correct voltage is calculated in the software. Voltage divider rule is used to calculate the voltage at any position down the resistor network, depending on the range selected through the software program. The voltage at position "a" is given by

$$V_a = \left(\frac{R_{13} + R_{14} + R_{15}}{R_{12} + R_{13} + R_{14} + R_{15}} \right) V_{in}$$

$$= \left(\frac{1487k + 40k + 98.5k}{8200k + 1487k + 40k + 98.5k} \right) V_{in}$$

$$= (0.1950) V_{in}$$

Therefore, the input voltage (V_{in}) that will produce a maximum of 5 volts as required by the ADC is given by

$$V_{in} = \frac{V_a}{0.195} = \frac{5}{0.195}$$

$$= 25.6V$$

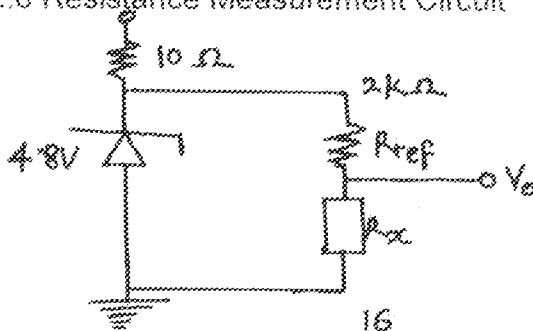
consequently to measure voltages less or equal to 25V, position "a" is selected. Like wise, to measure voltages less or equal to 100v or 500v, position "b" or "c" respectively is selected. The voltage measurement, then has four ranges: 5, 25, 100 and 500V.

To avoid loading of the network, a unit-gain buffer amplifier through a switching system, passes on the required voltage to the next stage for further processing. The switching system does the range switching by computer control of some relay switches arranged to select the required range position.

2.3.3 FOR RESISTANCE

Resistance can be measured with the circuit of Figure 2.8

Figure 2.8 Resistance Measurement Circuit



In this circuit $V_o = V_z R_x / (R_x + R_{ref})$, so since $V_z = 4.8V$ and up to $90k\Omega$ is to be measured that is $R_x = 90k\Omega$, Therefore R_{ref} is given by

$$\begin{aligned} R_{ref} &= \frac{V_z R_x}{V_o} - R_x \\ &= \left(\frac{4.8 \times 90}{4.7} - 90 \right) k\Omega \\ &= 2k\Omega \end{aligned}$$

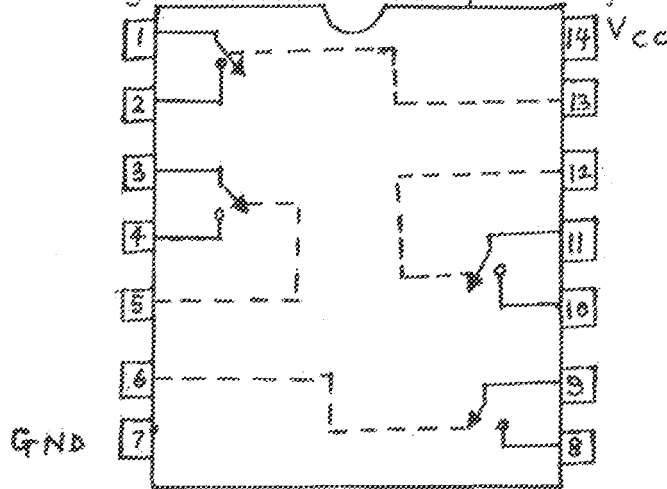
This means that with this circuit a maximum output voltage (V_o) of 4.7 occurs at the maximum measurable resistance of $90k\Omega$

2.4 THE ANALOGUE MULTIPLEXER

In a multichannel data acquisition system, as in this project, the various analogue signals from each of the output of the signal conditioning circuits, are to be converted to their various digital equivalents for further processing by the subsequent digital systems. An analogue multiplexer selects a particular signal for its conversion to digital code by an analogue to digital converter.

The analogue multiplexer employed in this project was designed with the 4-pole 1 way analogue switches integrated circuit (IC) shown in Figure 2.9. The IC contains four separate bi-directional

Figure 2.9 4068 CMOS 4-pole 1 way analogue switches IC



off/on switches in one package each with its own control input, as shown in figure 2.9.

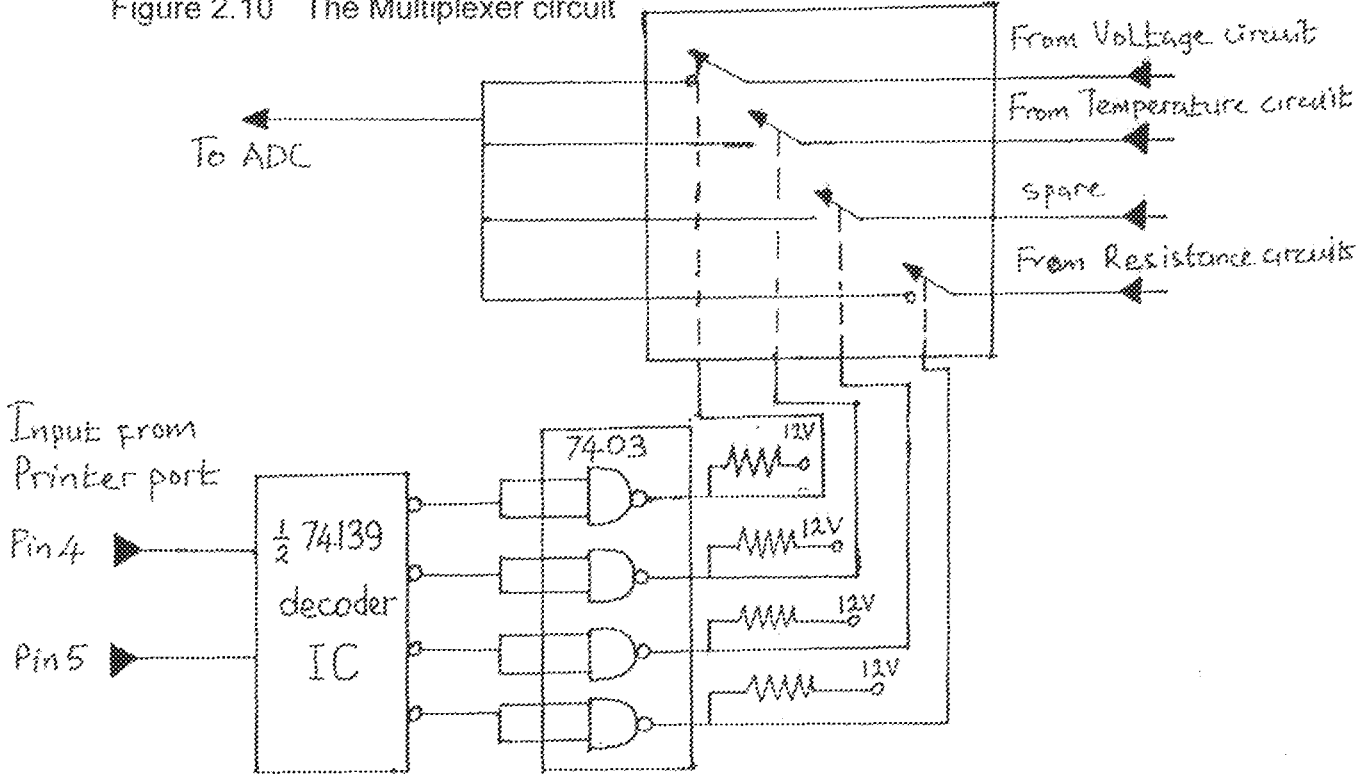
A switch is off with its control wire at low level and on at high level.

The resistance across each switch when closed is called its on resistance. This on resistance depends on the supply voltage and should be as low as possible

to avoid dropping part of the signal, so, the IC is supplied with a high voltage of + 12V to obtain an on resistance of 80Ω. But this requires a control voltage up to the supply voltage which is then higher than the normal Transistor Transistor logic (TTL) high state voltage level employed in this project. Consequently an open collector output logic control circuit is required to pull up the control voltage to about +11V, and ensure that only one switch is closed at a time as recommended by the manufacturer of the chip. These led to the use of a 1-out-of-4 decoder and an inverter with open collector output, in driving the multiplexer.

The overall analogue multiplexer circuit is shown in figure 2.10. The open collector output inverter here is determined from the 7403 NAND gate with open collector output, by the two input lines of each NAND gate connected together as shown in figure 2.10. The decoder used is one of the two decoders contained in the 74139 Dual 1 - out - of -4 Decoder IC. The

Figure 2.10 The Multiplexer circuit



decoder ensures that only one of its outputs is active in the low state at a time depending on the combined state of its input lines, which are set by the software. While the NAND gates, doing the work of inverters, invert the decoder's output so that only one is active in the high state of + 12V, to close only one switch at a time, as required by the manufacturer. With this multiplexer arrangement the required analogue input or channel, under software control is selected to pass on its signal to

the next stage of processing.

2.5 ANALOGUE - TO - DIGITAL CONVERTER

An ADC takes the instantaneous value of an analogue input signal and then produces as its output a coded digital word with a weight that corresponds to the level of the analogue. ADC will always contain some uncertainty over the conversion. This is because the analogue input is a continuous signal and can take any value within a defined range, whereas the digital output can only exist as a fixed number of codes. The uncertainty for an ADC is called quantising error and will be ± 0.5 LSB.

Consider an 8-bit ADC with an input span of 5V used in the project. The digital output, an 8-bit code, takes values from 0000 0000, 0000 0001, 0000 0010 and so on up to 1111 1111, which means that it has 256 quantisation levels. Suppose the analogue input is 1.25V, then the digital code is 0011 1111. When the input is 1.27V, this digital code changes to 0100 0000, that is one LSB step. However an analogue value of 1.26V, just halfway between 1.25V, and 1.27V, could give a digital code of 0011 1111 or 0100 0000. This is what is meant by the uncertainty of an ADC.

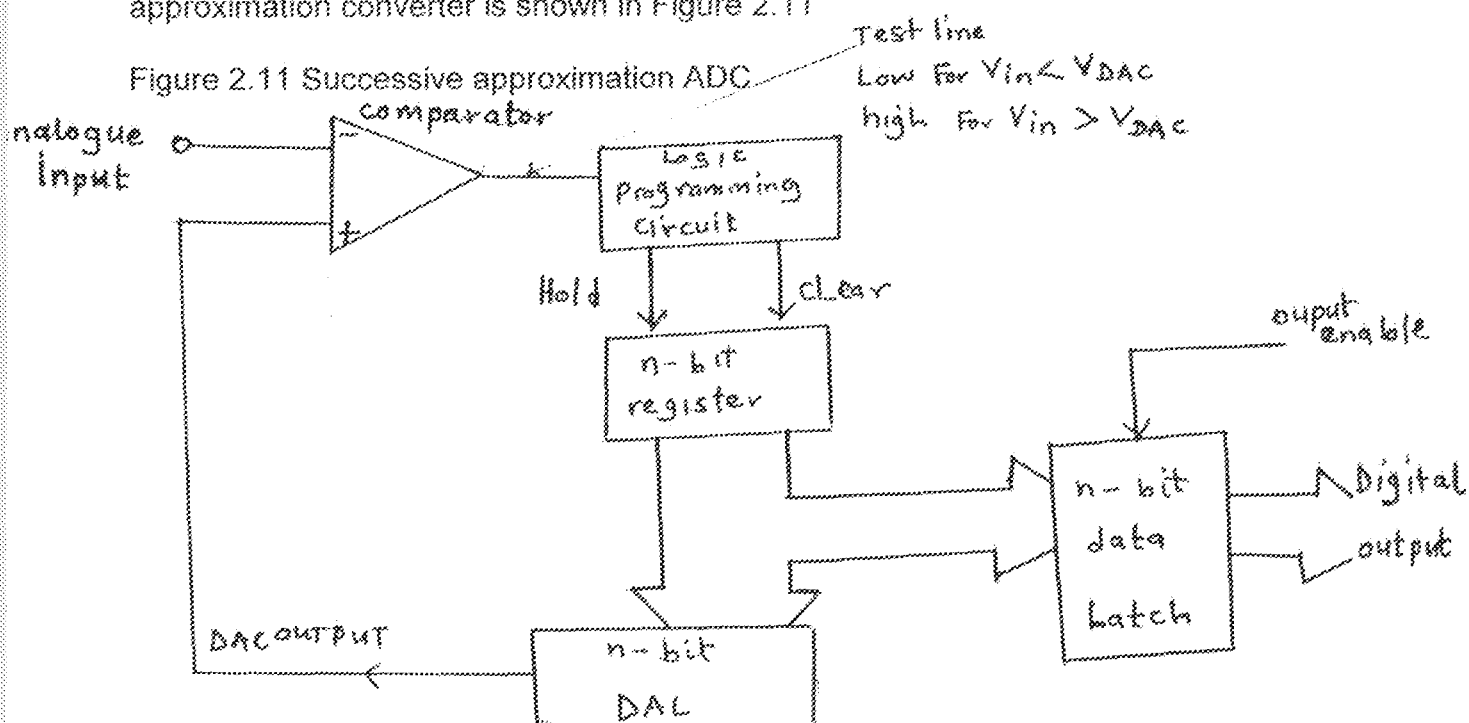
An important parameter of an ADC is the conversion time: the time interval between the command, being given to the ADC to begin the conversion, and the appearance at the output, of the complete digital equivalent of the analogue value. The speed of conversion varies with the type of ADC and can be as short as a few nano seconds for the ultra fast types or as slow as several milliseconds.

2.5.1 PRACTICAL ADC CIRCUIT

A wide variety of methods are used in analogue - to-digital conversion. These range from the slow and inexpensive to the very fast types which are therefore relatively costly. The common methods are

- voltage to frequency
- Parallel or flash conversion
- Single ramp and counter
- Dual/triple ramp
- Successive approximation

The method used in this project is the successive approximation converter. This is a popular method for use in microprocessor systems since it is relatively fast, has good accuracy, and can be software controlled. A typical successive approximation converter is shown in Figure 2.11



The method requires some programming logic (software in the microprocessor if required), a register to hold the result, a DAC and a fast microprocessor, as shown in figure 2.11

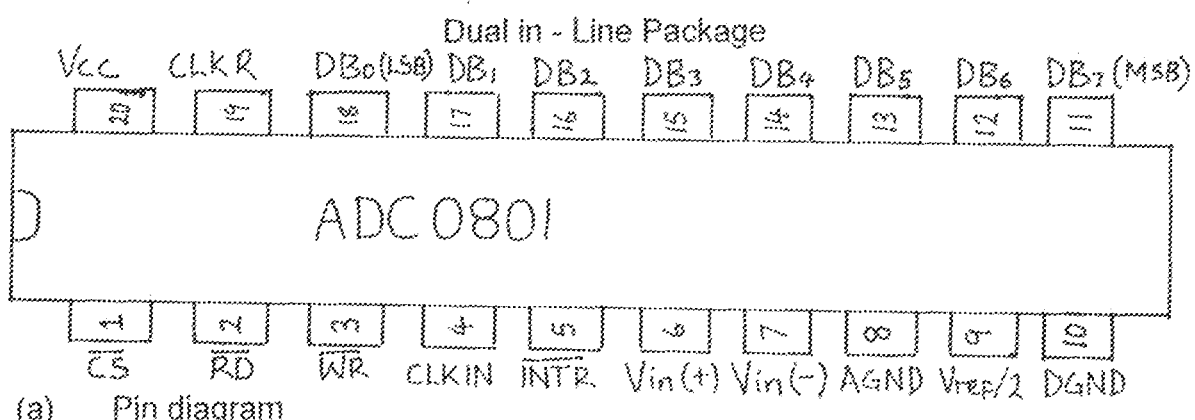
At the start of the conversion, the MSB of the register is set to logic 1 by the programming logic so that for 8-bits the register reads 1000 0000. This in turn is converted by the DAC and the value compared with the analogue input. Suppose it is less than V_{in} , then the logic 1 in that position is retained and the next most significant bit is also set to 1 (register holds 0110 0000) and used for comparison. This process continues until all bits have been tried and a point of balance is reached; that is, when V_{in} is just greater than the DAC output voltage. Then, the resultant digital code on the register is latched to the output of the A/D converter

2.5.2 AN A/D CONVERTER IC

In this project ADC0801 8-bit A/D converter IC was used. The pin diagram of the IC is shown in Figure 2.12 (a) and a list of the name and function of each pin on the IC is shown in figure 2.12 (b). The ADC 0801 A/D converter was designed to interface directly with the 8080, 8085, or Z80 microprocessors. Some pin labels on the ADC0801 IC correspond to pins on popular microprocessors. For instance, the

ADC0801 uses RD, WR and INTR as pin labels which correspond to the RD, WR, and INTR pins on the 8085 microprocessor. The ADC 0801 can also be interfaced with other popular 8-bit microprocessors such as the 6800 and 6502. The chip select input to the ADC0801 A/D converter receives its signal (chip select) from the microprocessor address-decode.

Figure 2.12 ADC0801 A/D CONVERTER IC



Pin No.	Symbol	Input/Output or Power	Description
1	CS	Input	Chip select Line from up-control
2	RD	Input	Read Line From up - control
3	WR	Input	Write from up-control
4	CLKIN	Input	Clock
5	INTR	Output	Interrupt line goes to up interrupt line
6	Vin(+)	Input	Analogue voltage (+)
7	Vin(-)	Input	Analogue voltage (-)
8	AGND	Power	Analogue ground
9	Vref/2	Input	Alternative voltage reference (+)
10	DGND	Power	Digital ground
11	DB7	Output	MSB data output
12	DB6	Output	Data output
13	DB5	Output	Data output
14	DB4	Output	Data output
15	DB3	Output	Data output
Data	DB2	Output	Data output
17	DB1	Output	Data output
18	DB0	Output	LSB data output
19	Input	Output	Connect external resistor for clock

20	V _{cc} (or ref)	Power	Positive of 5V power supply and primary reference voltage
----	--------------------------	-------	---

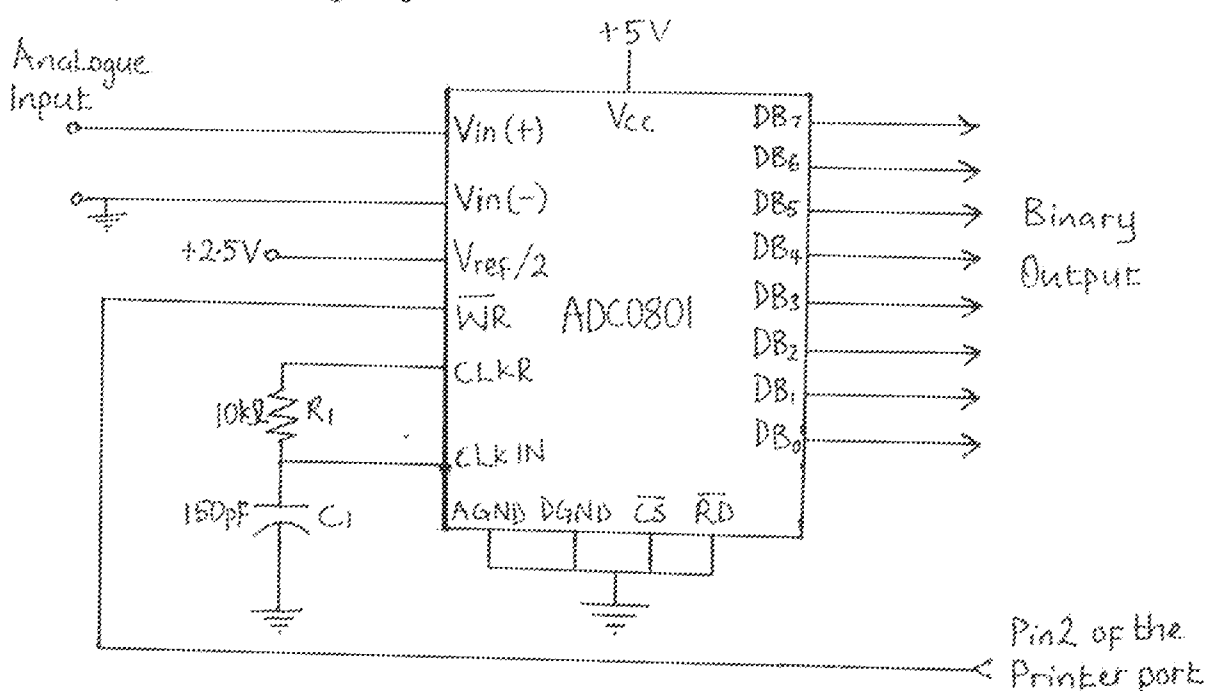
(b) Pin Labels and Functions

ing circuitry.

THE ADC0801 is a CMOS 8-bit successive approximation A/D converter. It has three state output so that it can interface directly with a microprocessor based system data bus. The ADC0801 has binary outputs and features a short conversion time of only 100 μ s. Its inputs and outputs are both MOS - and TTL - compatible it has an on - chip clock generator. The on-chip generator does need two external component (resistor or capacitor) to operate. The ADC0801 IC operates on a standard + 5V dc power supply and can encode input analogue voltages ranging from 0 to 5V.

The ADC0801 A/D converter IC in this project was configured as shown in figure 2.13. The function.

Figure 2.13 Wiring diagram for the ADC0801 CMOS ADC IC



of the circuit is to encode the difference in voltage between Vin (+) and Vin(-) compared to the reference voltage, 5V, to a corresponding binary value. For instance, the resolution of the ADC0801 IC is 8-bits or 0.39 percent. This means that for each 0.02V ($5V \times 0.39 \text{ percent} = 0.02V$) increase in voltage at the analogue inputs the binary count increases by 1. To start conversion, a signal of L to H transition is

applied at the WR input, this starts the ADC conversion process. When the conversion is finished, the binary output is updated. In the project the WR is clocked by one of the data Lines of the parallel printer port (Pin 2) as shown in figure 2.13. The ADC0801 has a high conversion rate because it uses the successive approximation technique in the conversion process. The resistor (R1) and capacitor (C1) connected to the CLKR and CLK IN inputs to ADC0801 IC cause the internal clock to operate. The frequency of the clock can be calculated as follows

$$F = \frac{1}{R_1 C_1} \quad (H_z)$$

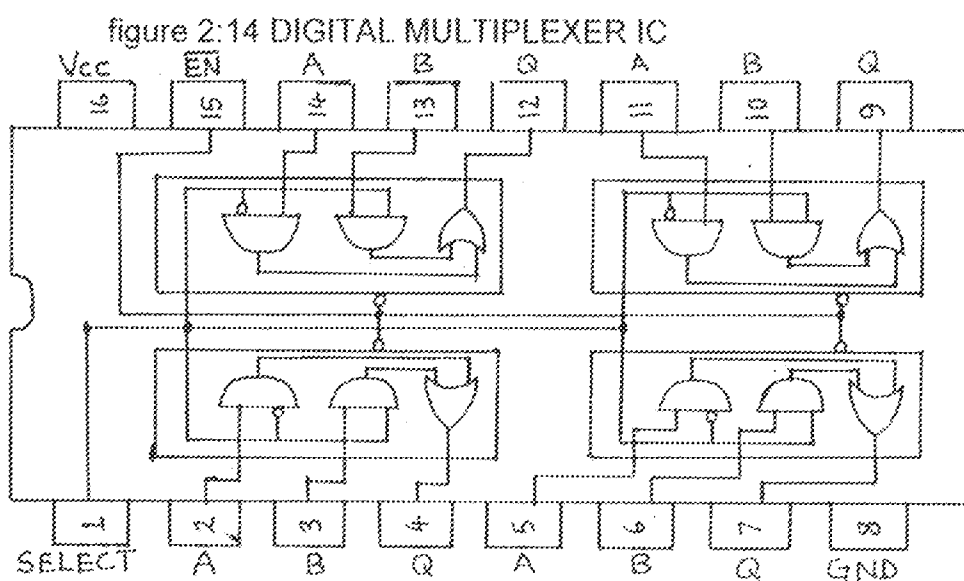
$$= \frac{1}{10 \times 10^3 \times 150 \times 10^{-12}} \quad (H_z)$$

$$= 666.7 \text{ KHz}$$

Accuracy of the process is assured at clock frequencies up to 640KHz

2.6 THE DIGITAL MULTIPLEXER

The interface characteristics of the input port of the printer interface, the status port, is such that only five lines feed back into the computer. The ADC converter translates analogue information into eight-bit bytes. This means an interface circuit is required to connect the ADC to the printer port. This was achieved in this project by using a 2-input 4-bit multiplexer, the 74157 IC shown in figure 2.14



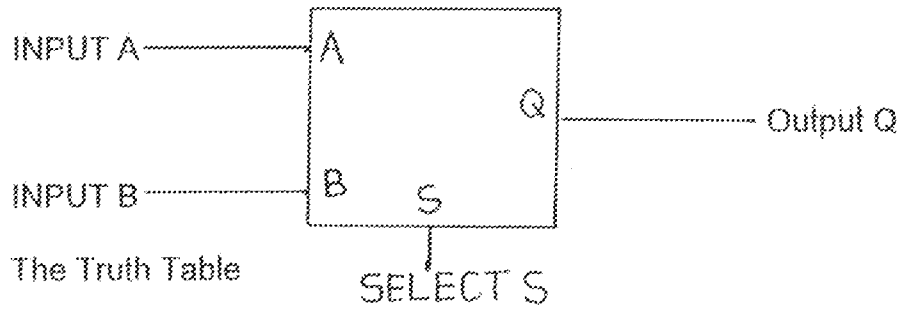
which takes the 8-bit byte of the ADC, 4 bits into the printer port with the aid of the input-output program executed by the computer.

The 74157, 2 - input 4-bit multiplexer was designed from the smallest

multiplexer, the 2-input 1-bit multiplexer. Considering the block diagram of a 2-input 1 bit multiplexer shown in figure 2.15,

Figure 2.15 A 2- input 1-bit multiplexer

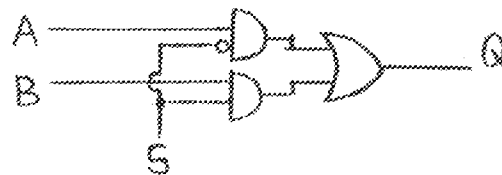
(a) block diagram



(b) The Truth Table

Select (S)	Input (A)	Input (B)	Output (Q)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

(c) The Logic circuit Diagram



when the select line is in logic 0, input A passes to the output Q. This produces the truth table shown in figure 2.15 (b). Therefore the output Q is given by

$$\begin{aligned}
 Q &= \bar{S}A\bar{B} + S\bar{A}B + S\bar{A}B + S\bar{A}B \\
 &= \bar{S}A(\bar{B}+B) + SB(\bar{A}+A) \\
 &= \bar{S}A + SB
 \end{aligned}$$

So the logic circuit implementation of the logic function is shown in figure 2.15

(c). Figure 2.14 shows how four of these multiplexers are inter connected to yield a 2 - input 4 bit multiplexer as contained in the 74157IC.

2.7 THE SOFTWARE

In this project, the peripheral Hardware is sending and receiving digital information (data) to and from the IBM PC respectively. This is done through the aid of a program which is able to send, accept and interpret these data. The software used for the project is called CAMD. The program was written in C++ PROGRAMMING Language. C++ is a development of C and C came out of the development of the UNIX operating system by Bell Labs. C++ is the language of choice for most commercial and scientific applications, because it is sufficiently low level to provide a very good speed of execution, it puts detailed control of the machine into the programmer's hands and it is transportable.

The flow chart of the program is shown in figure 2.16. In the program five screen displays were made in order to make the program user friendly. The screens are shown in figure 2.17. Screen (1) enables the user to decide whether to go on with the operation or to quit. Screen (2) enables the user to select the quantity to be measured, whether; dc voltage, maximum of 5V dc voltage, temperature, resistance, or to quit the program. In screen (3) the user selects the range if the quantity to be measured, as selected in screen (2), is dc voltage. Screen (4) enables the user to confirm the choices made and decide whether to go on displaying the output, or to make another choice. After the choice to display on output is made, the computer runs a subroutine (function) called Hardware input. This subroutine does the input output operations making use of input output instructions. Screen (5), displays the value measured which is the returned value from the subroutine, Hardware input.

THE INPUT/OUTPUT SUBROUTINE

In the Hardware input subroutine the following operations were done:

- Outputting of series of bit patterns through the data output port, to select the correct channel depending on the quantity selected on screen (2) to be measured. Then analogue - to- digital conversion commences.
- After a delay, on completion of conversion, the 8-bit byte output of the ADC is read in four by four bits starting from the most significant Bit (MSB) to the least significant Bit, through the status-port of the printer interface.

- The bits are rearranged back into 8-bit byte as outputed from theADC. Then, the 8-bit binary data is converted back to its equivalent analogue representation and store in a variable which is returned to main program. This variable is called anavalue.
- The variable,anavalue,is then displayed on the screen as the measured value.

The entire program is shown in appendix 1

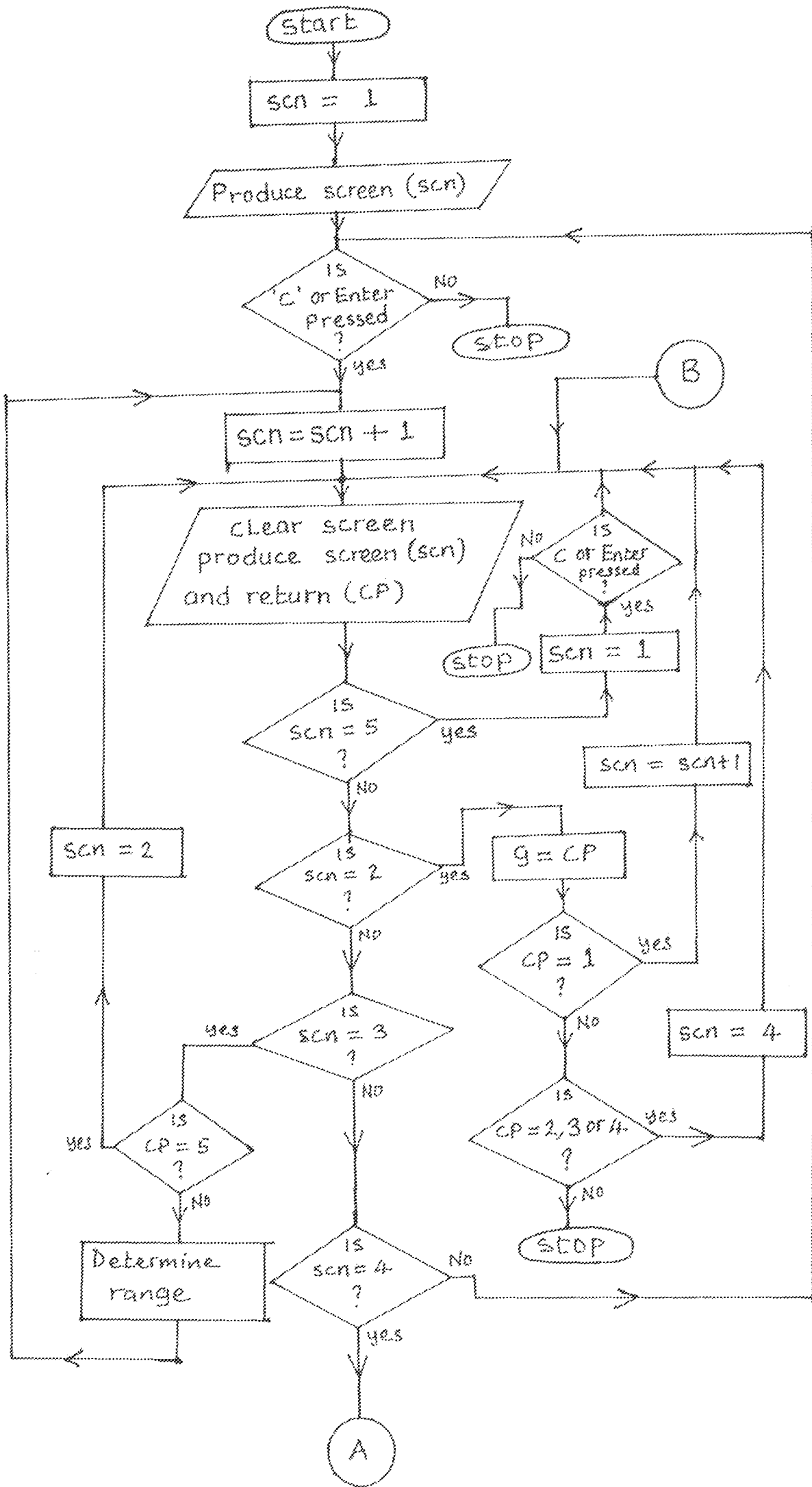


FIGURE 2.16 : THE FLOWCHART .

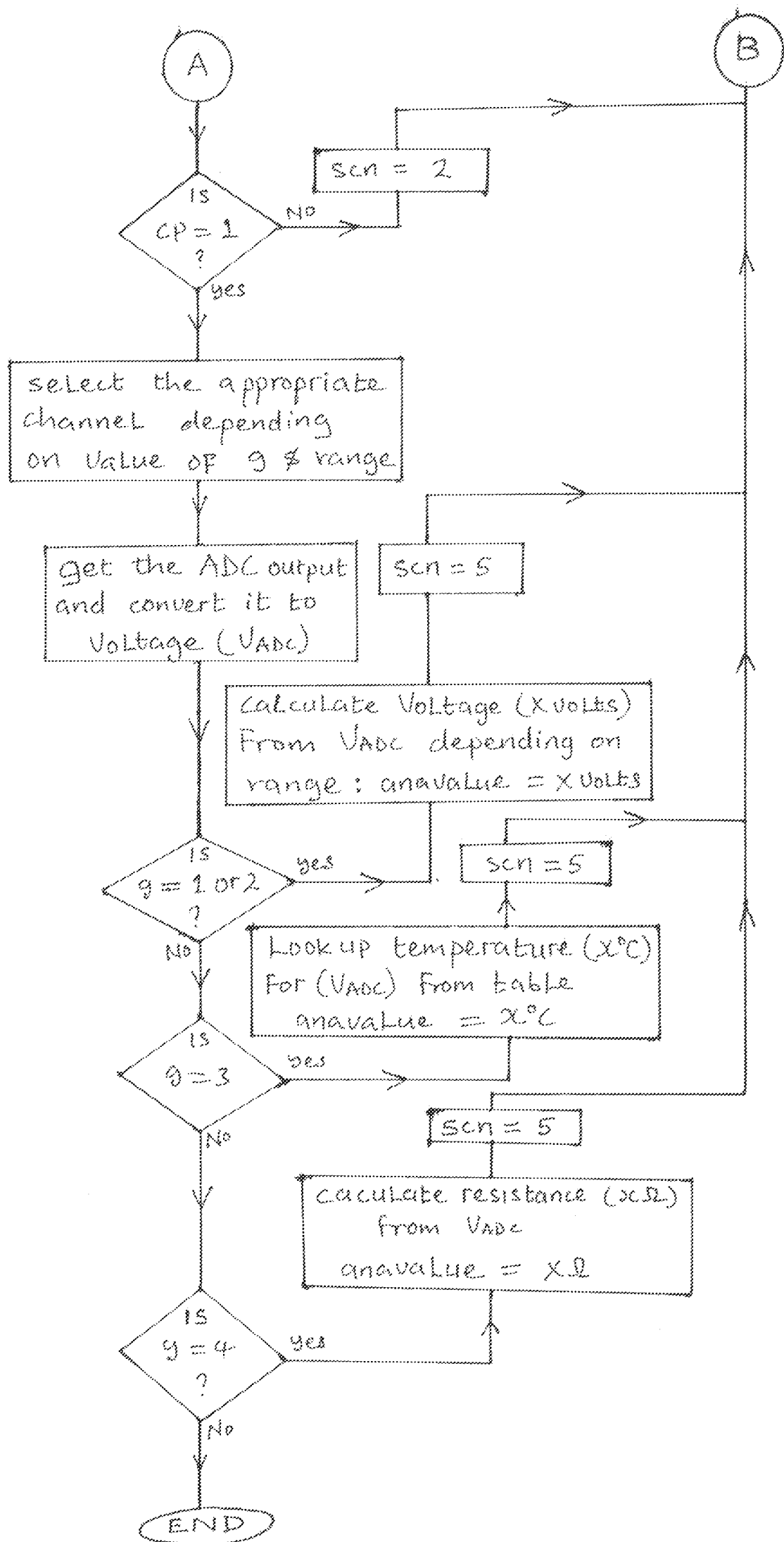


FIGURE 2.16 CONTINUES

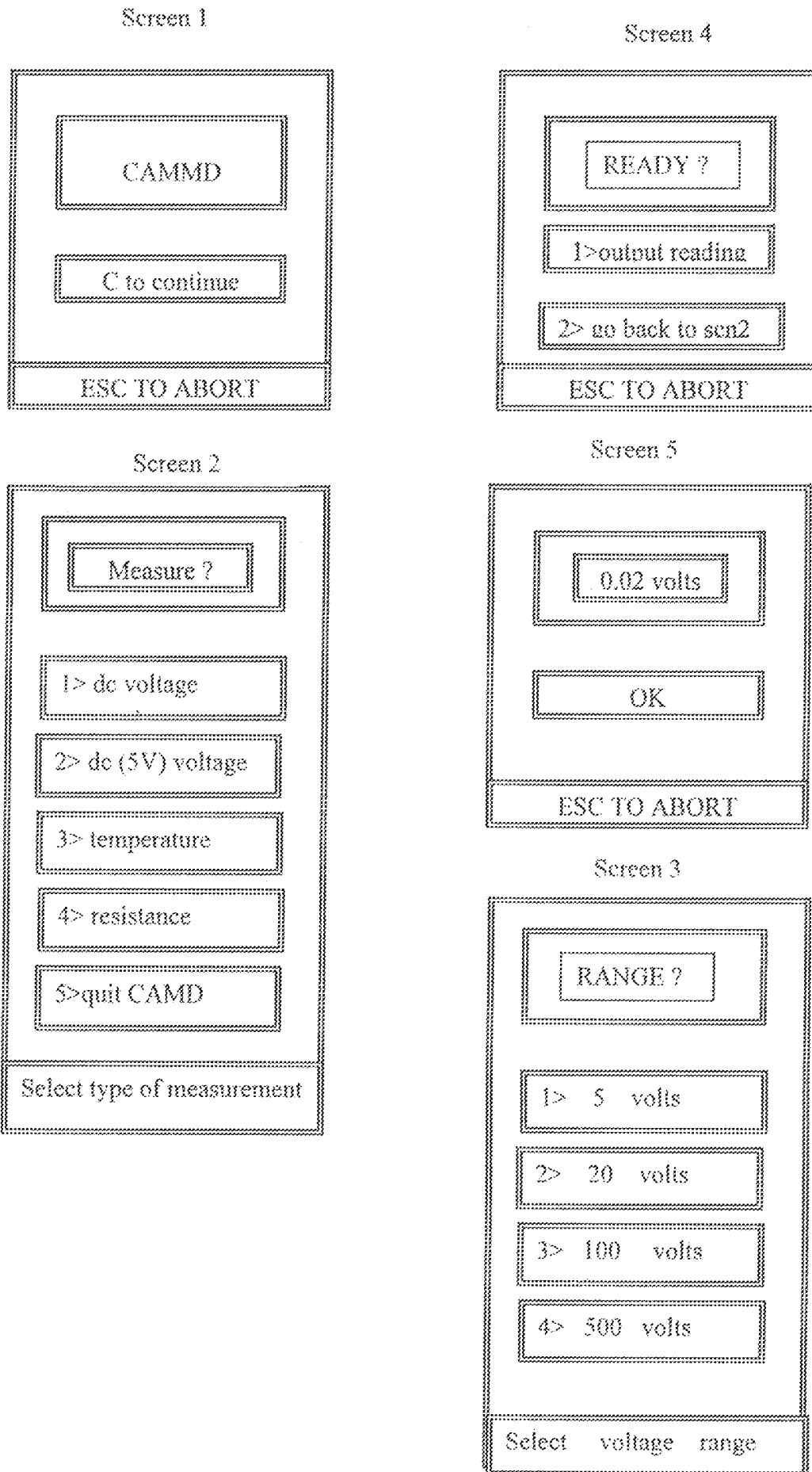


FIGURE 2-17 CAMMD SCREENS

2.8 POWER SUPPLY

The circuit of the project requires two different dc voltage supplies of +5v and +12V. It is necessary to supply the appropriate d.c. voltages to the device from a mains electricity supply at 240V AC. The block diagram for this power supply is shown in figure 2.18 it is composed of the following stages: stepping down with transformer, full wave rectification, smoothing circuit, and Regulation. The complete circuit diagram is shown in figure2.19.

To calculate the voltage rating of the transformer used in stepping down the mains and the peak Inverse voltage (PIV) of the diodes used in the full wave rectifier circuit: Let the output of the transformer, $V_s(t)$, which is an a.c. Voltage as shown in figure 2.1 be given by

$$V_s(t) = V_m \sin \omega t,$$

where V_m = peak of the alternating voltage. With full wave rectification employed, the output voltage of the rectifier, $v_o(t)$ as shown in figure 2.1 is given by:

$$V_o(t) = \begin{cases} V_m \sin \omega t, & 0 \leq t \leq \pi \\ -V_m \sin \omega t, & \pi < t \leq 2\pi \end{cases}$$

Therefore the average output d.c voltage, V_{dc} , is given by:

$$\begin{aligned} V_{dc} &= \frac{1}{T} \int_0^T V_o(t) dt \\ &= \frac{1}{\pi} \int_0^{\pi} V_m \sin \omega t d(\omega t) \quad - \quad - \quad - \quad 2.1 \end{aligned}$$

Integrating equation 2.1 and substituting the limits gives

$$V_{dc} = 2 \frac{V_m}{\pi}$$

with $V_{dc} = 12$ V for this project

$$V_m = \frac{V_{dc} \pi}{2} = \frac{12(3.14)}{2} = 18.8v$$

$$\text{but } V_{rms} = \frac{V_m}{\sqrt{2}} = 13.3v$$

Therefore a 240v/13.3v rated transformer was required. The Piv rating for a full wave bridge rectifier is given by

$$\begin{aligned} PIV &= 2 \times V_m \\ &= 2 \times 18.8 = 37.6V \end{aligned}$$

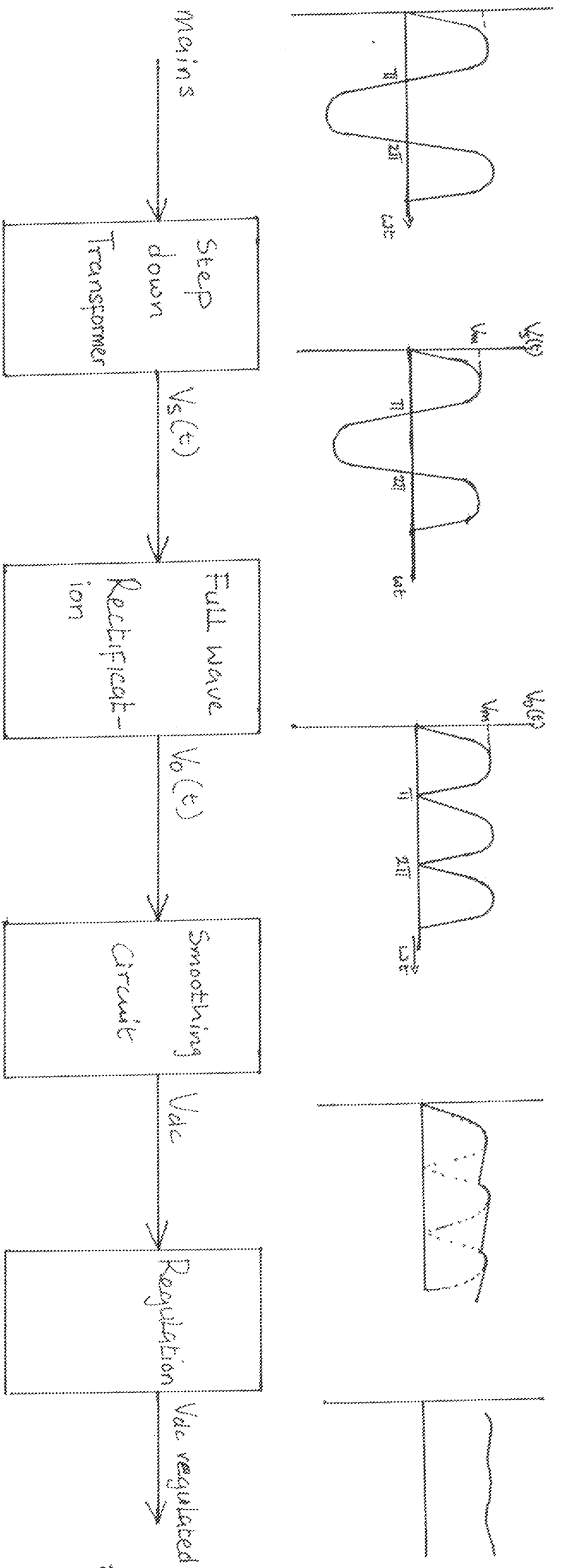


FIGURE 2.18 BLOCK DIAGRAM OF A POWER SUPPLY

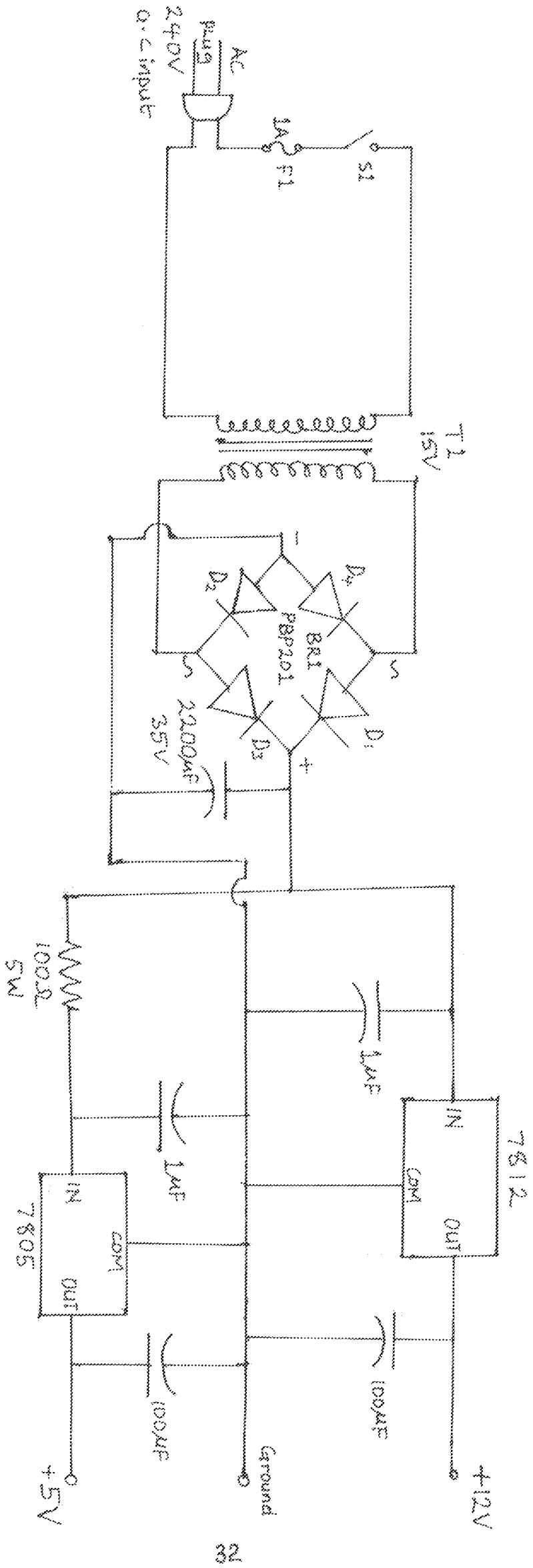


FIGURE 2.19 THE POWER SUPPLY CIRCUIT

Power supply circuits form an interesting and quite important branch of electronics in themselves, and the design of a good power supply used to be very difficult. Fortunately, we now have cheap and extremely reliable power-supply regulator ICs which greatly simplify the job. In this project regulator ICs 7805 and 7812 were used to provide the regulation.

CHAPTER THREE

3.0 CONSTRUCTION PROCEDURE

On completion of the design of this project, components, needed for its construction were bought and assembled together. The construction started with the implementation of the design on a breadboard. This was done in modules proceeding from one module to another after due testing. The modules are as follows

- The power supply circuit
- A/D conversion circuit
- Digital multiplexer circuit
- Analogue multiplexer circuit
- Signal conditioning circuits

various aspect of the design was then modified to obtain desirable working circuit before it was constructed permanently on a veroboard.

With the use of instruments such as soldering iron, soldering Lead, solder sucker, veroboard, connecting wire, cutler, screw driver, Analog and digital multimeter and long nose pliers, the following steps were taken for the construction of each module on a veroboard.

- i. The Layout design plan of the components position on the veroboard was made. Unnecessary distance between components were avoided to reduce the length of wire used. Components were also not placed too close to each other to avoid shorts.
- ii. IC sockets were soldered onto the veroboard in place allocated to them in the layout.
- iii. The discrete components:- resistors, capacitors and diodes were soldered directly to the board.
- iv. Other components of each module were soldered.
- v. And Analog multimeter was then used to check the contacts and continuity where necessary.
- vi. With the ICs plugged into their sockets, each module was tested before

proceeding to the next.

On the other hand the software was written and debugged on a Turbo C++ compiler. When both software and hardware were ready testing and debugging were carried out on them separately. Then the software was interfaced with the hardware with appropriate timing made.

3.1 TESTING

The testing of the hardware started from the construction stage, as each of the modules on completion was tested before moving on to the next one. The circuit was in each case checked for continuity using a digital multimeter. Also the required behaviour of each component after soldering was tested.

Finally, after interfacing both the software and the hardware on the Host micro computer, the IBM PC, the resultant measuring device was then tested. The voltage measurement was tested by measuring the voltage outputs of a variable power supply unit.

The resistance measurement was tested by measuring different resistance values of resistors. Finally the temperature measurement was tested by measuring the temperature of a hot water while comparing with a thermometer. All measuring systems were tested, proper and accurate functioning of the device was confirmed satisfactory.

3.2 DISCUSSION OF RESULT

The actual values (expected readings) of both voltage and resistance were compared with the actual readings on the multimeter after being tested. While that of temperature was compared with the readings on a thermometer. All tests were confirmed to have fallen within the expected or acceptable reading of accuracy limit.

CHAPTER FOUR

4.0 CONCLUSION AND RECOMMENDATION

4.1 CONCLUSION

The development of a Computer Aided Measuring / Monitoring Device Via the printer port interface was presented in this project.

The aim of the project was achieved after much technical difficulties, like some components were damaged and were replaced several times

4.2 RECOMMENDATION

Design and construction of Electronics project should be a regular practice to all students of electrical and computer engineering department right from their intermediate level, so as to get them familiar with handling project works.

Also, the Department should endeavor to assist students financially in carrying out mini-projects and the final year project that consume a lot of money. It is quite unfortunate that most of my work were done in a computer Business center were I was charged, because the Department denied me the access to the Departmental computer.

Therefore final year projects should be executed in the departmental Laboratory by the concerned students under the strict supervision of their supervisors.

REFERENCE

1. Barry Kanler, PC Architectue & Assembly Language, Third Edition, 1990.
2. Britannica Micropaedia, 15th Edition, 1998.
3. Douglas V. Hall, Microprocessors and Digital Systems, Second Edition, 1987.
4. Gordon Maccomb, The Robot Builders Bonanza, Mc Graw Hill Inc, 1987
5. Hayes, John Patric, Digital System Design and Microprocessors, 3rd Printing, 1987.
6. James W Coffron, IBM PC Connection, 1987.
7. John Watson, Meastering Electronics, Fourth Edition, 1996.
8. Lionel Warnes, Electronic and Electrical Engineering, Macmillan Press Ltd, Second Edition, 1998.
9. Lovelyday George, Microprocessors in Engineering Systems, Pitman Publishing, 1998
10. Robin Holland, Microcomputer Fault-finding and Design, Macmillan Press Ltd, Second Edition, 1995

APPENDIX I

THE PROGRAM

```
#include<stdio.h>
#include<stdlib.h>
#include<graphics.h>
#include<conio.h>
#include<dos.h>
#define KEYC 0x2e63
#define ESC 0x011b
#define ENTER 0x1c0d
#define UPARR 0x4800
#define DOWNARR 0x5000
#define ZF 0x40
#define TRUE 1
#define FALSE 0
#define VREF 5.04
#define TM 250
#define HD 20
typedef struct{
    int x1,y1,x2,y2;
}vertexpoints;
typedef struct{
    char *sdata;
}stringdata;
typedef struct{
    vertexpoints sent,roid;
    int fcolor,fpattern,blcolor,tcolor,thk,txtcolor,txtdir;
    int txtfont,txtsize,stringnum;
}animatedbar;
static char upattern[5][8]={{
    {0xdb,0x7f,0xdb,0xef,0xdb,0xfd,0xdb,0xfe},
    {0xaa,0xff,0xaa,0xff,0xaa,0xff,0xaa,0xff},
    {0xf,0x75,0xbe,0xd7,0xfa,0x5f,0xea,0x7d},
    };
char quant[5][8]={"VOLTS","DRG CEL","kOHMS","VOLTS"};
stringdata sd[33];
animatedbar ab[20];

static float v[60]={0.12,0.15,0.20,0.26,0.35,0.42,0.48,0.55,0.63,0.71,
0.79,0.86,0.95,1.02,1.09,1.16,1.24,1.32,1.40,1.48,1.56,1.65,1.71,1.78,1.88,
1.96,2.03,2.11,2.18,2.23,2.30,2.44,2.49,2.58,2.66,2.74,2.80,2.89,2.98,3.04,
3.12,3.20,3.29,3.37,3.46,3.53,3.60,3.68,3.70,3.77,3.80,3.90,4.00,4.10,4.30,
4.42,4.52};
static float t[60]={43.0,44.0,45.0,46.0,47.0,48.0,49.0,50.0,51.0,52.0,53.0,
54.0,55.0,56.0,57.0,58.0,59.0,60.0,61.0,62.0,63.0,64.0,65.0,66.0,67.0,68.0,
69.0,70.0,71.0,72.0,73.0,74.0,75.0,76.0,77.0,78.0,79.0,80.0,81.0,82.0,83.0,
84.0,85.0,86.0,87.0,88.0,89.0,90.0,91.0,92.0,93.0,94.0,95.0,96.0,97.0,98.0,
99.0};

struct viewporttype vp;
struct REGPACK reg;
void clearfill(vertexpoints vps,int fillcolor,int fpattern);
void writestringat (int p,int signum);
void blinktext(int p,int num, int times);
```

```

void designbar(int p);
void menuatob(int a,int b);
int makeachoice(int scn);
void drawscreen(int scn);
void writeonscreen(int scn);
int movecursor(int new,int old);
void initializegr(void);
void filldata(void);
int processscreen(int scn);
void body(void);
void frame(void);
void mscreen(void);
void status(void);
void monitorfun(void);
float Hardwareinput(void);
int firstmenu,lastmenu,cursorposn,rorange,screennum,maxx,maxy;
int gdriver,gmode,errorcode,waste,format=0;
float anavalue=00.00;
int datport,staport,nextscn = 1;
unsigned hb,lb,d0,d1,d2,a8,g,g1=0;
int main(){
    int i;
    initializegr();
    filldata();
    clrscr();
    do{
        nextscn = processscreen(nextscn);
    }while(nextscn != 0);
    closegraph();
    printf("GOODBYE");
    getch();
    return 0;
}
void initializegr(void){
    int i;
    gdriver = DETECT;
    initgraph(&gdriver,&gmode,"c:\tc\bgi");
    errorcode = graphresult();
    if(errorcode != grOk){
        printf("graph error : %\n",grapherrormsg(errorcode));
        printf("press any key to halt:");
        getch();
        exit(1);
    }
    maxx = getmaxx();
    maxy = getmaxy();
    for(i = 0 ; i <= 10 ; i++){
        ab[i].fcolor = LIGHTGRAY;
        ab[i].tcolor = WHITE;
        ab[i].bcolor = DARKGRAY;
        ab[i].fpattern = SOLID_FILL;
        ab[i].thk = 3;
        ab[i].tstcolor = BLUE;
        ab[i].txtdir = HORIZ_DIR;
    }
}

```

```

        ab[i].txtfont = TRIPLEX_FONT;
        ab[i].txtsize = 1;
        ab[i].stringnum = 22;
        ab[i].sent.x1 = 179;
        ab[i].sent.y1 = 199;
        ab[i].sent.x2 = 459;
        ab[i].sent.y2 = 291;
        ab[i].retd.x1 = 183;
        ab[i].retd.y1 = 113;
        ab[i].retd.x2 = 455;
        ab[i].retd.y2 = 197;
    }
    setbkcolor(4);
    datport = 888;
    staperi = 889;
}
void filldata(void){
    sd[0].sdata = "VOLTMETER";
    sd[1].sdata = "PRESS C TO CONTINUE";
    sd[2].sdata = "ESC TO ABORT";
    sd[3].sdata = "MEASURE ?";
    sd[4].sdata = "DC VOLTAGE";
    sd[5].sdata = "LOW DC VOLTAGE";
    sd[6].sdata = "TEMPERATURE";
    sd[7].sdata = "RESISTANCE";
    sd[8].sdata = "QUIT VOLTMETER";
    sd[9].sdata = "SELECT TYPE OF MEASUREMENT";
    sd[10].sdata = "RANGE \?";
    sd[11].sdata = "5 VOLTS";
    sd[12].sdata = "20 VOLTS";
    sd[13].sdata = "100 VOLTS";
    sd[14].sdata = "500 VOLTS";
    sd[15].sdata = "BACK TO QUANTITY MENU";
    sd[16].sdata = "SELECT VOLTAGE RANGE";
    sd[17].sdata = "READY ?";
    sd[18].sdata = "OUTPUT READING";
    sd[19].sdata = sd[15].sdata;
    sd[22].sdata = "OK";
    sd[23].sdata = " ";
    sd[24].sdata = "C TO CONTINUE OR ESC TO END";
    sd[25].sdata = "GOING BEYOND NOT ALLOWED";
    sd[26].sdata = "MONITOR";
    sd[27].sdata = "CHOSE (1) IF READY FOR OUTPUT";
    sd[28].sdata = "PRESS ANY KEY TO STOP";
    sd[29].sdata = "WAITING..";
    sd[30].sdata = sd[6].sdata;
    sd[31].sdata = "MONITORING";
}
int movecursorio(int new,int old){
    int a;
    if ((new >= firstmenu)&&(new <= lastmenu)){
        if(old != 0){
            clearfill(ab[old].retd,ab[old].fcolor,ab[old].fpattern);
            writestringat (old,ab[old].stringnum);
        }
    }
}

```



```

    }
    ab[new].txtcolor = WHITE;
    clearfill(ab[new].retd,BLUE,ab[new].fpattern);
    writestringat(new,ab[new].stringnum);
    ab[new].txtcolor = BLUE;
    cursorpostn = new;
    a = TRUE;
    }
    else{
        sound(20);
        delay(TM);
        nosound();
        a = FALSE;
    }
    return(a);
}
int makeeachoice(int scn)
{
    unsigned int a;
    int b;
    b = movecursorto(cursorpostn,0);
    do{
        do{
            if((scn != 5)&&(scn != 1))
                blinktext(2,1,TM);
            else
                blinktext(8,1,TM);
                reg_r_ax = 0x1 << 8;
                intr(0x16,&reg);
        }while(reg_r_flags & ZF);
        a=reg_r_ax;
        waste = getch();
        if (a==UPARR){
            getch();
            if(scn==4)
                b=movecursorto(cursorpostn-2,cursorpostn);
            else
                b=movecursorto(cursorpostn-1,cursorpostn);
            if(b==0){
                clearfill(ab[8].retd,ab[8].fcolor,ab[8].fpattern);
                writestringat(8,25);
            }
        }
        else if(a==DOWNARR){
            getch();
            if(scn == 4)
                b = movecursorto(cursorpostn+2,cursorpostn);
            else
                b = movecursorto(cursorpostn+1,cursorpostn);
            if(b == 0){
                clearfill(ab[8].retd,ab[8].fcolor,ab[8].fpattern);
                writestringat(8,25);
            }
        }
        else if(a==KEYC){
            if ((scn == 1)||((scn==5))){

```

```

        if(sc == 5)
            cursorposn = 5;
            return (cursorposn);
        }
    }
    else if(a==ENTER){
        if(sc == 5)
            cursorposn = 5;
            return (cursorposn);
        }
    }
    else if(a==ESC){
        closegraph();
        printf("GOODBYE");
        getch();
        exit(1);
    }
    else {
        clearfill(ab[8].rctd,ab[8].fcolor,ab[8].fpattern);
        sprintf(sd[21].sdata,"READ INSTRUCTIONS OR USE ARROW KEYS OR PRESS ENTER KEY");
        writestringat(8,21);
    }
} while(TRUE);
}
void drawscreen(int sc){
    int a,b;
    if(sc ==1){
        a = 5;b = 5;
    }
    else if((sc ==2)||(sc == 3)){
        a = 3;b = 7;
    }
    else if((sc ==4)||(sc == 6)){
        a = 4;b = 6;
    }
    setviewport(0,0,maxx,maxy,1);
    clearviewport();
    if(sc != 1)
        body();
    if(sc !=5){
        menustob(a,b);
        if(sc ==4)
            clearfill(ab[5].rctd,ab[0].fcolor,ab[0].fpattern);
        }
        else
            designbar(9);
        frame();
        if(sc !=1)
            mscreen();
            status();
        }
}
void writeonscreen(int sc){
    int a,i;
    switch(sc){
        case 1:
            writestringat(1,0);

```

```

        writestringat(8,2);
        cursorpostn = 5;
        break;
    case 2:
        writestringat(2,3);
        for(i = 4 ; i <= 8 ; i++)
            writestringat(i-1,i);
        writestringat(8,9);
        cursorpostn = 7;
        break;
    case 3:
        writestringat(2,10);
        for(i = 11 ; i <= 13 ; i++)
            writestringat(i-8,i);
        writestringat(8,16);
        cursorpostn = 7;
        break;
    case 4:
        writestringat(2,17);
        writestringat(4,18);
        writestringat(6,19);
        if(g == 0)
            sprintf(sd[20].sdata,"Range is %d",rrange);
        else
            sd[20].sdata = sd[27].sdata ;
        writestringat(8,20);
        cursorpostn = 6;
        break;
    case 5:
        if(g != 1){
            sprintf(sd[21].sdata,"%06.2f %s".anavalue,quant[g]);
            writestringat(2,21);
            writestringat(9,22);
            writestringat(8,24);
        }
        else{
            writestringat(2,29);
            writestringat(9,30);
            writestringat(8,28);
            monitorfun();
            g! = 0;
        }
        cursorpostn = 10;
        break;
    case 6:
        writestringat(2,6);
        writestringat(4,18);
        writestringat(5,26);
        writestringat(6,19);
        writestringat(8,27);
        cursorpostn = 6;
        break;
    default:
        printf("\nwriteonscreen default switch \n");

```

```

        getch();
    }
}

void monitorfun(void){
    while(!kbhit()){
        anavalue = Hardwareinput();
        sprintf(sd[21].sdata,"%06.2f %s",anavalue,quant[g]);
        clearfill(ab[2].retd,ab[2].fcolor,ab[2].fpattern);
        writestringat(2,21);
    }
    waste = getch();
    clearfill(ab[9].retd,ab[9].fcolor,ab[9].fpattern);
    writestringat(9,22);
    clearfill(ab[8].retd,ab[8].fcolor,ab[8].fpattern);
    writestringat(8,24);
}

void designbar(int p){
    int dx,dy,t;
    setlinestyle(SOLID_LINE,0,NORM_WIDTH);
    if(ab[p].fpattern >= 12)
        setfillpattern(upattern[ab[p].fpattern-12],ab[p].fcolor);
    else
        setfillstyle(ab[p].fpattern,ab[p].fcolor);
    bar(ab[p].sent.x1,ab[p].sent.y1,ab[p].sent.x2,ab[p].sent.y2);
    dx = ab[p].sent.x2-ab[p].sent.x1;
    dy = ab[p].sent.y2-ab[p].sent.y1;
    for( i = 0 ; i <= ab[p].thk ; i++){
        moveto(ab[p].sent.x1+i,ab[p].sent.y1+i);
        setcolor(ab[p].bcolor);
        lineref(0,dy-i*2);
        lineref(dx-i*2,0);
        setcolor(ab[p].trcolor);
        lineref(0,-(dy-i*2));
        lineref(-(dx-i*2),0);
    }
    ab[p].retd.x1 = ab[p].sent.x1 + ab[p].thk+1;
    ab[p].retd.y1 = ab[p].sent.y1 + ab[p].thk+1;
    ab[p].retd.x2 = ab[p].sent.x2 - (ab[p].thk+1);
    ab[p].retd.y2 = ab[p].sent.y2 - (ab[p].thk+1);
}

void writestringat (int p,int strgnum){
    int x,y;
    getviewsettings(&vp);
    setviewport(ab[p].retd.x1,ab[p].retd.y1,ab[p].retd.x2,ab[p].retd.y2,1);
    x = (ab[p].retd.x2-ab[p].retd.x1)/2;
    y = (ab[p].retd.y2-ab[p].retd.y1)/2;
    settxtstyle(ab[p].txtfont,ab[p].txtdir,ab[p].txtsize);
    setcolor(ab[p].txcolor);
    if(p != 9){
        settxtjustify(CENTER_TEXT,CENTER_TEXT);
        outtextxy(x,y,sd[strgnum].sdata);
    }
    else{
        settxtjustify(CENTER_TEXT,BOTTOM_TEXT);
        outtextxy(x,y,sd[strgnum].sdata);
    }
}

```

```

    settextrjustfy(CENTER_TEXT, TOP_TEXT);
    outtextxy(x,y,cd[strnum+1].sdata);
}
ab[p].stringnum = strnum;
setviewport(vp.left, vp.top, vp.right, vp.bottom, 1);
}
void blinktext(int p, int num, int timed){
    int i;
    if(num == 0){
        while(!kbhit()){
            clearfill(ab[p].retd, ab[p].fcolor, ab[p].fpattern);
            delay(timed);
            writestringat(p, ab[p].stringnum);
            delay(timed);
        }
    }
    else
        for(j=1; j<=num; j++){
            clearfill(ab[p].retd, ab[p].fcolor, ab[p].fpattern);
            delay(timed);
            writestringat(p, ab[p].stringnum);
            delay(timed);
        }
}
void clearfill(vertexpoints vps, int fillcolor, int fpattern){
    getviewsettings(&vp);
    setviewport(vps.x1, vps.y1, vps.x2, vps.y2, 1);
    clearviewport();
    setviewport(vp.left, vp.top, vp.right, vp.bottom, 1);
    if(fpattern >= 12)
        setfillpattern(upattern[fpattern-12], fillcolor);
    else
        setfillstyle(fpattern, fillcolor);
    bar(vps.x1, vps.y1, vps.x2, vps.y2);
}
int processscreen(int scn){
    int ps=scn, crsrpos;
    drawscreen(scn);
    writeonscreen(scn);
    crsrpos = makeachoice(scn);
    ++ps;
    if(scn == 2){
        switch(crsrpos){
            case 3: ps=3; g=0;
                break;
            case 4: d0=79; d1=76; d2=77; ps=4; g=3;
                break;
            case 5: d0=199; d1=196; d2=197; ps=6; g=1;
                break;
            case 6: d0=131; d1=128; d2=129; ps=4; g=2;
                break;
            case 7: ps=0;
                break;
            default: clrscr();
                printf("\n\nWrong measurement choice\n");
                getch();
        }
    }
}

```

```

    }
    else if(scu == 3){
        if(crsrpos == 3){
            d0=17;d1=68;d2=69;
            mrange=5;
        }
        if(crsrpos == 4){
            d0=255;d1=252;d2=253;
            mrange=20;
        }
        if(crsrpos == 5){
            d0=223; d1=220;d2=221;
            mrange=100;
        }
        if(crsrpos == 6){
            d0=232;d1=236;d2=237;
            mrange=500;
        }
        if(crsrpos == 7)
            ps=2;
    }
    else if(scu ==4){
        if(crsrpos == 4){
            anavalue = Hardwareinput();
        }
        else
            ps=2;
    }
    else if(scu == 5)
        ps=1;
    else if(scu ==6){
        if(crsrpos == 4){
            anavalue = Hardwareinput();
            ps=5;
        }
        else if (crsrpos == 5){
            gl=1;
            ps=5;
        }
        else
            ps=2;
    }
    return (ps);
}
float Hardwareinput(void){
    int i;
    float anaval;
    outputb(datport,d0);
    delay(HD);
    outputb(datport,d1);
    delay(HD);
    outputb(datport,d2);
    delay(HD);
    hb = inputb(staport);
}

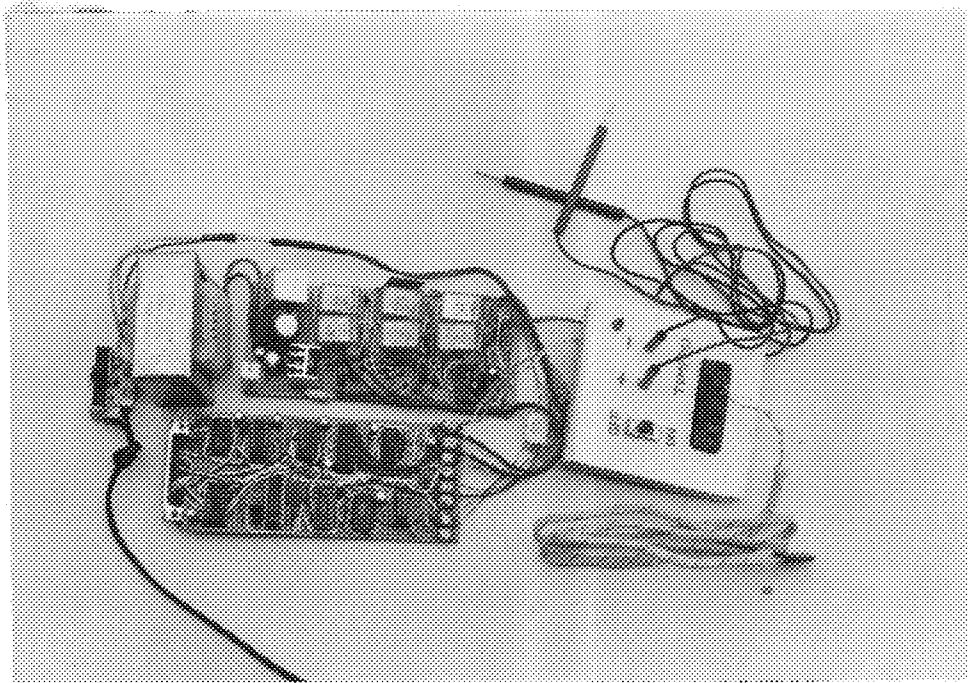
```

```

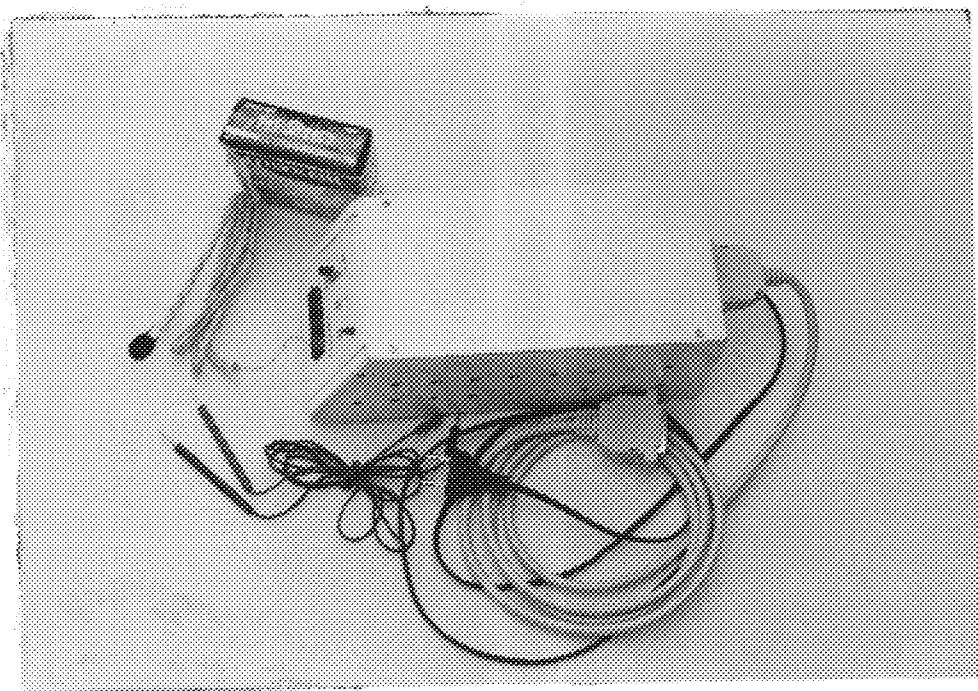
lb = ~hb;
hb &= 120;
outportb(datport,255);
delay(HD);
lb = inportb(staport);
lb = ~lb;
hb &= 120;
hb <<= 1;
lb >>= 3;
a8 = hb/lb;
anaval = a8/255.0*VREF;
if (g==2)
anaval = anaval*2.1/(4.8-anaval);
if (g == 1){
    i=0;
    while(anaval > v[i])
        i++;
    anaval = (v[i-1]+(v[i]-v[i-1])*(anaval-v[i-1]))/(v[i]-v[i-1]);
}
return(anaval);
}

```

APPENDIX 2



PICTORIAL VIEW OF THE CONSTRUCTION



THE FINISHED PRODUCT