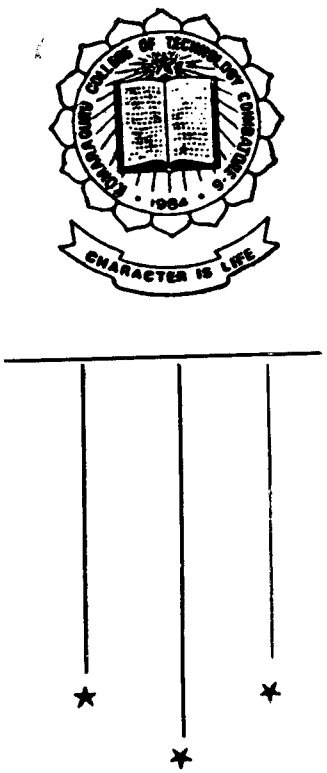
PC Based Controller for Stepper Motor

Project Work



Submitted by

A. LAKSHMANAN

SUMESH CHANDRAN

N. VIJAYA KUMAR

D. VISVANATHAN

Under the Guidance of

Dr. K. A. PALANISWAMY, Ph.D.,

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING

IN ELECTRICAL AND ELECTRONICS ENGINEERING

OF THE BHARATHIAR UNIVERSITY, COIMBATORE

1993-94

Department of Electrical and Electronics Engineering

Kumaraguru College of Technology

Coimbatore - 641 006

Department of Electrical and Electronics Engineering

Kumaraguru College of Technology

Coimbatore - 641 006

CERTIFICATE

Name..... *Roll No* .....

University Register No......

This is to certify that the Project work

"PC BASED CONTROLLER FOR STEPPER MOTOR"

is a bonafide work carried out by

Mr. A. LAKSHMANAN, SUMESH CHANDRAN, N. VIJAYA KUMAR, D. VISVANATHA

In partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering in Electrical and Electronics Engineering Branch of the Bharathiar University Coimbatore during the academic year 1993–94

Station :

Date : 12/4/94

Dr. K. A. PALANISWAMY, B.E.,M.Sc.(Engg).,Ph.D

Guide

Professor and Head

Department of Electrical and Electronics Engineering,

Kumaraguru College of Technology,

Coimbatore 641 006

Dr. K. A. PALANISWAMY, B.E.,M.Sc.(Engg).,Ph.D

H. O. D.

Professor and Head

Department of Electrical and Electronics Engineering,

Kumaraguru College of Technology,

Coimbatore 641 006

Submitted for the University Examination held on .....

.....

Internal Examiner

.....

External Examiner

ACKNOWLEDGEMENT

We are forever grateful to Dr. K.A. PALANISWAMY, B.E., M.Sc., (ENGG), C.ENG(I), Ph.D, MISTE, FIE, Professor and Head of the Department of Electrical and Electronics Engineering for his valuable guidance during the course of our project.

We have great pleasure in expressing our gratitute to the principal Dr. S.SUBRAMANIAN, Ph.D, for his encouragement and support for carrying out this project work.

We also thank all who have helped us directly and indirectly in completing this project work successfully.

SYNOPSIS

The explosive growth of the computer industry in recent years has meant an enormous growth too for stepper motors because these motors provide the driving force in many computer peripheral devices. They can be found, for example, driving the paper feed mechanism in the line printers and printing terminals. Stepper motors are also use exclusively in floppy disk drives where they provide precise positioning of the head on the disks. The x- and y- coordinate pens in expensive plotters are driven by the stepper motors. They offer good dynamic performance while eliminating the need for a heavy maintenance schedule associated with drive schemes that employ gears, slide wires and brushes. The stepper motor is especially suited for such aplication because it is an essential device which serves to convert input information in digital form to an output that is mechanical. It thereby provides a natural interface with the digital computer.

The present project deals with the design of the 8255A Programmable Peripheral Interface card with driver circuit of the stepper motor and an interactive software in QUICKBASIC language is developed to handle speed control, forward and backward direction rotation of motor with on-line stimulated graphical display facility. The above interface card can be

used to control 12 motors at a time which can control in such a way to form a ROBO for a particular work and also for other programmable logic control (PLC) applications.

CONTENTS

Page No.

CERTIFICATE

ACKNOWLEDGEMENT (ii)

SYNOPSIS (iii)

I. STEPPER MOTOR

1.1 Introduction 1

1.2 Constructional Features 2

1.3 Principle and Operation of Stepper motor 3

1.4 Speed-Torque Characteristics of Stepper motor 6

1.5 Half and Full step Sequence and Switching 8

1.6 Rating of Stepper motor 10

1.7 Applications of Stepper motor 11

II. 8255A PROGRAMMABLE PERIPHERAL INTERFACE (PPI)

2.1 Introduction 22

2.2 Methods of Parallel Data Transfer 22

2.3 Block Digramme of 8255A 27

2.4 Functions of 8255A 28

2.5 8255A Modes and Initialization 30

III. DIGITAL INPUT / OUTPUT CARD FOR PC

3.1 Introduction 42

Page No.

3.2 Installation 42

3.3 Address Mapping 42

IV. INTERFACING A MICROCOMPUTER TO A STEPPER MOTOR 47

V. PCB FABRICATION OF THE DRIVER CIRCUIT

5.1 PCB Drawing in Smart Work 54

5.2 Fabrication 55

5.3 Testing 55

5.4 Cabinet design and Making 56

VI. SOFTWARE AND TESTING

6.1 Flow Chart 60

6.2 Software 64

6.3 Testing 72

VII. CONCLUSION 73

REFERENCES 74

APPENDIX

CHAPTER I

STEPPER MOTOR

1.1 INTRODUCTION

The increasing trend toward digital control of machines and process functions has generated a demand for mechanical devices capable of delivering incremental motions of predictable accuracy. The stepper motor is often considered as a digital device which converts electrical pulses into proportionate mechanical movement. Each revolution of the stepper motor's shaft is made up of a series of discrete individual steps. The size of the step depends on the design of the motor and can be as small as 1.5 or as large as 30. The stepper motor shaft rotation is incremental. The basic feature of the stepper motor is that upon being energized it will move and come to rest after some number of steps in strict accordance wit the digital input commands provided. The stepping motor therefore allows control of the load's velocity, distance and direction. The motor usually provides for clockwise (CW) or counter clockwise (CCW) rotation. The repeatability (ability to position through the same pattern of movements a number of times) is very good. The only system error introduce by the stepper motor is its single - step error, which is small percentage of 1 step and is

generally less than 5% (0.09). Most significantly, this error is non-cumulative, regardless of distance travelled or the number of times repositioning takes place.

The stepper motor is an inherently reliable device with bearing being the only part subject to wear. In many applications, a stepper motor can replace shorter lived, more maintenance-intensive device such as brakes, clutches and gears with an overall improvement in reliability.

1.2 CONSTRUCTIONAL FEATURES :

The construction of the stepper motor is quiet simple. It consists of a slotted stator equipped with two or more individual coils and a rotor structure that carries no winding. The classification of the stepper motor is determined by how the rotor is designed. If it is provided with a permanent magnet attached to its shaft, it is called a permanent magnet (PM) stepper motor. If the permanent magnet is not included, it is classified as a reluctance type stepper motor. Attention here is directed first to the PM stepper motor. Of course, the presence of the permanent magnet furnishes the motor with the equivalent of a constant dc excitation. Thus, when one or more of the stator coils are energized, the machine behaves as a synchronous motor.

The basic construction details of the PM stepper motor is illustrated in Fig(1.1) for a four pole stator and a five pole rotor structure. Observe that the action of the permanent magnet on the particular orientation of the rotor structure is to magnetize each of the poles (or slots) at one end of the rotor with a N-pole polarity and each of the poles at the other end with S-pole polarity. Moreover, the N-pole set of rotor poles is arranged to be displaced from the S-pole set of rotor poles by one-half of a pole-pitch and is readily evident by a comparison of Fig (1.1.1 and 1.1.2). The symmetry that exists between the stator poles and each set of the rotor poles makes it apparent that each rotor pole set behaves in an identical fashion. For example, if coil A - A' is assumed to be energized to yield a N-pole polarity at A and a S-pole polarity at A', then the relationship between N of the stator and S1, S2 and S5 of the rotor in Fig (1.1.2) corresponds exactly to that of S of the stator and N1, N2 and N5 of the rotor in Fig (1.1.2). A similar statement can be made for the remaining sets of poles.

1.3 PRINCIPLE AND OPERATION OF STEPPER MOTOR

Stepper motors are divided in to three principle types of classes, each with distinct construction and performance characteristic: Variable - Reluctance (VR), Permanent Magnet (PM) and pm - hybrid.

A stepper motor operation is based on the basic magnetic principle that like magnetic poles repel and unlike poles attract. If the stator winding in Fig (1.2.1) energized so that stator A is the north pole, stator B is the south pole and the permanent magnet (PM) rotor is positioned as shown in Fig (1.2.2), a torque will be developed to position the rotor 180 from its indicated position. However it would be impossible to determine the direction of rotation and in fact the rotor may not move at all if the forces are perfectly balanced if as indicated in Fig (1.2.2) two additional stator poles C and D are added and energized as shown. We are able to predict the direction of rotation of the rotor. As indicated in Fig (1.2.2) the rotor's direction of rotation would be counter - clockwise with the rotor aligning itself between the "average" south pole and the "average" north pole as shown in Fig (1.2.3).

The distinguished feature of the PM stepper motor is the incorporation of a permanent magnet, usually in the rotor assembly. To allow better step resolution, four more stator poles are added, and teeth are machined on each stator pole and also on the rotor. The number of teeth on the rotor and stator determines the step angle that will be obtained each time the polarity of one winding is changed. The stepper motor shaft responds with a specific angular increment each

1.4 SPEED - TORQUE CHARACTERISTICS OF STEPPER MOTOR

A typical torque / speed characteristic for a stepper motor is shown in Fig (1.4). The graph shows speed along the horizontal axis (the characteristics for other types of motor showed speed along the vertical axis). This is the normal way to plot the stepper motor characteristics because it is the torque that depends on the speed and not vice-versa; the stepper motor either runs at the required speed or doesn't run at all. In other types of motor a load (torque) is applied and the characteristics indicates how fast the motor will run with that load. In speed torque characteristics, as the speed is increased, the torque falls to a very low value. Actually the situation is not quite as bad it appears, because it is possible to compensate for the drop in torque by increasing the supply voltage and is function of time.

Stepper motors are popular because they can be used in an open loop mode while still offering many of the desirable features of the feed back-type system: feed them a defined number of pulses and they will position within their step accuracy. The replacement of mechanical part (which are susceptible to wear), such as clutches and brakes, is eliminated because stepper motors provide a greater reliability and consistency, resonable cost, and consistent performance. The stepper motor is an excellent position

device. On the otherhand, stepper motors are not very energy-efficient, but that is the price one must pay to obtain the unique characteristics of the steper motor. Its limitations include the following: available torque is inversely proportional to speed: speed must increase gradually (it commended to go from stop to full speed immediately, it will stall): and the stepper motor exhibits a low speed resonance point, where torque is reduced drastically.

Some stepper motors achieve torque amplification by means of a gear system integral with the motor. Methods to achieve torque amplification are the use of planetary gears and the use of a flexing mechanical spline. Both methods allow a reduction in speed and increases in output torque by gearing down. For instance, a stepper that delivers 50 oz.in of torque at 72 rpm will, with a 4:1 step-down gearing, deliver 200 oz.in of output torque at a speed of 18 rpm. Although the stepper motor is driven by a properly sequenced set of digital signals, the motor iitself exhibits the characteristics of a synchronous motor. In one of its most popular forms, it acts as a doubly excited machine, which in turn means it produces a steady-state torque at one speed. Compensation circuits are available which can double the useful speed range but this is at the cost of increased

circuit complexity for the driver amplifier. However, one of the major drawbacks of the stepper motors is that its maximum torque capacity falls rapidly at higher stepping rates.

1.5 HALF - STEP AND FULL - STEP SEQUENCE AND SWITCHING

To understand the stepper motor characteristics thoroughly, we must examine the stepping motor logic sequencing. A simplified representation of a stepping motor is shown in Fig (1.5). Initially pole A and B are both energized with their north poles up, attracting the south pole to the position as shown in Fig (1.5) . Reversing the polarity of pole A Fig (1.5.1) draws the rotor clockwise 90 to its new position: this is known as a full step. If pole A had been turned off, instead of being reversed, the rotor would rotate 45x (clockwise) to line up with the field of pole B; this is known as a half step. The simple stepper motor in Fig (1.5) would only have four full steps (90 x) per revolution, or eight (45 x) half steps. Actual stepper motor obtains small angle increments by using large number of poles.

The most common stepper stator windings are center-tapped dual windings known as bifilar windings. Bifilar winding eliminates transformer coupling to adjacent windings. The use of bifilar windings on stepper motors also simlifies the required drive circuitary. Fig (1.6) shows a bifilar - wound

stepper motor, its power supply and the switching points. Only a single - polarity power supplys needed, whereas the motor of Fig (1.6) would require a dual power supply for reversal of the poles. Only a single power supply is needed with the center - tapped windings.

The switching sequence shown Fig (1.6.1) is called a four step sequence (full step). To reverse direction, read the sequencing chart upward from the bottom. Since current is maintained on the motor windings when the motor is not being stepped, a high holding torque results. As a rule of thumb you will find that the power supply is about five times the voltage rating of the motor. Series resistors are in the common leads to limit the current and improve the inductive / resistive (L/R) time contant, for better performance.

Fig (1.6.2) illustrates the eight - step switching sequence, often called electronic half stepping. (Half Step) under this condition, the rotor moves half its normal distance per step. For example: a 1.8x, 200 step per revolution motor would become a 0.9x, 400 step per revolution motor. Likewise a 0.72x, 500 step per revolution model moves in increment of 0.36x for 1000 steps plr revolution. The advantages of operating in this mode include finer resolution and greater speed capability, but with less available torque.

The switching sequence for this motor was originally achieved by mechanical switches, which has maintenance headache. Electronic switching resolved this problem quite easily and efficiently. The circuits use solid state devices to drive (switch) the stepper motor windings. It should be noted that the full - step control sequence is two on - time periods followed by two half - time periods. The rate of stepping is determined by the frequency of the applied clock, with each input pulse causing one step (or half step) in the stepper motor shaft. The half - step sequence pattern is three on - time periods followed by five off time periods: like the full step control it is very easily obtained by solid state circuitry.

1.6 RATINGS OF STEPPER MOTORS

A motor's output power is dependent upon the extent to which the copper is worked electrically and the iron is worked magnetically. The stepper motor ranks low on both counts. To achieve the characteristics which make the stepper motor unique, it is necessary that during each complete switching cycle each of the phase coils spend time in a de-energized state and that some part of the magnetic flux paths be essentially free of field flux. Consequently, stepper motors are found to range in output power from about 1 W to a maximum size of 3 hp. In terms of physical

dimensions, the largest machines are built with diameters up to seven inches while the smallest ones are constructed in a pancakes form with diameters as small as 1 inch.

Many step sizes are also available with stepper motors. They can be purchased with step sizes as small as 0.72 or as large as 90 . The most common step sizes are 1.8 , 7.5 , and 15 . To procure a step size of 1.8 in a PM type, it is necessary to design the stator for 40 poles and the rotor for 50 poles.

NOMINAL CHARACTERISTIC OF STEPPER MOTOR

1. Nominal Voltage = 12 V
2. Resistance / Phase = 84 ohms
3. Inductance / Phase = 12.7 mH
4. Mass = 145 g
5. Step Angle = 18 + 1°
6. Number of Steps / rev = 20
7. Power absobed / Phase = 1.7 W
8. Current / Phase = 140 mA
9. Max Starting F at no load = 75 steps 1

1.7 APPLICATIONS OF STEPPER MOTOR

The stepper motor is ideally suited for a wide variety of control and positioning applications in the industry. With

the rapid growth of solid states electronic and digital techniques, the stepper motor finds applications in peripherals, robotics, instrumentation control and machine tools. The stepper motor, however, can be found performing countless tasks outside the computer industry. In many commercial military, and medical applications, the stepper motors perform such functions as mixing, cutting, metering, stirring, blending and purging. They provide many supporting roles in the manufacture of packaged food stuffs, commercial and products, even the production science fiction movies. The advantages offered by the stepper motor in this application can be exploited with the use of two pieces of auxillary equipment, the microprocessor and controlled switches in the form of transistors.

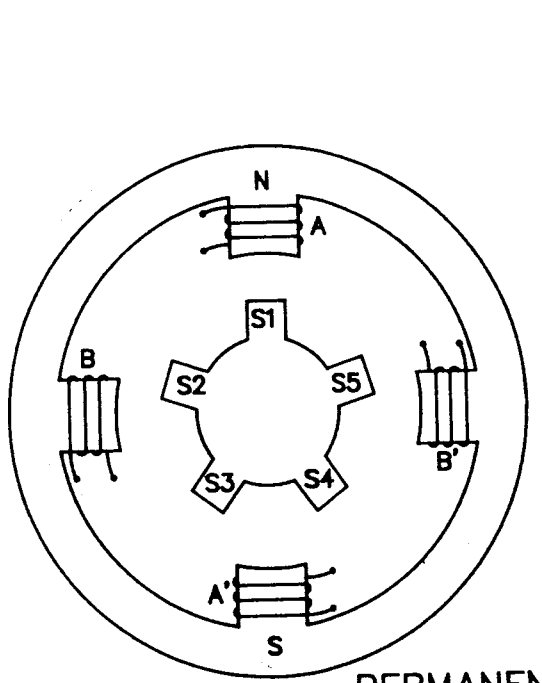


FIG.(1.1)

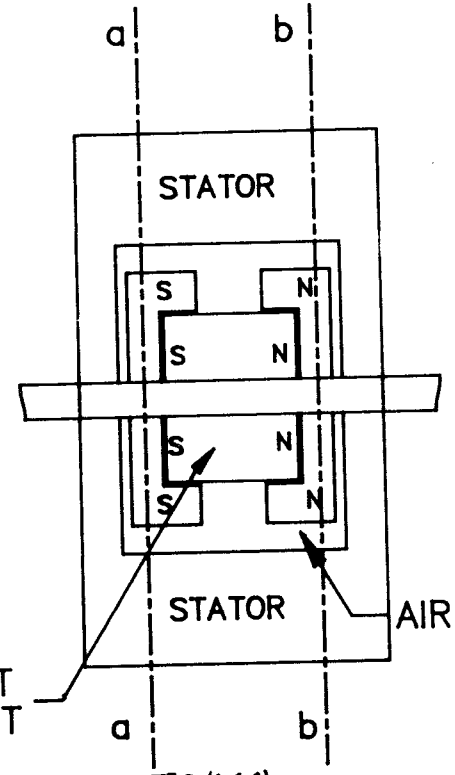


FIG (1.1.1)

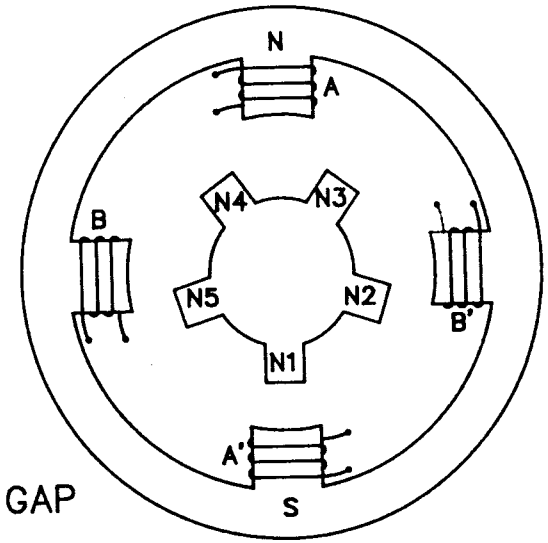


FIG (1.1.2)

CONSTRUCTIONAL FEATURES OF A STEPPER MOTOR

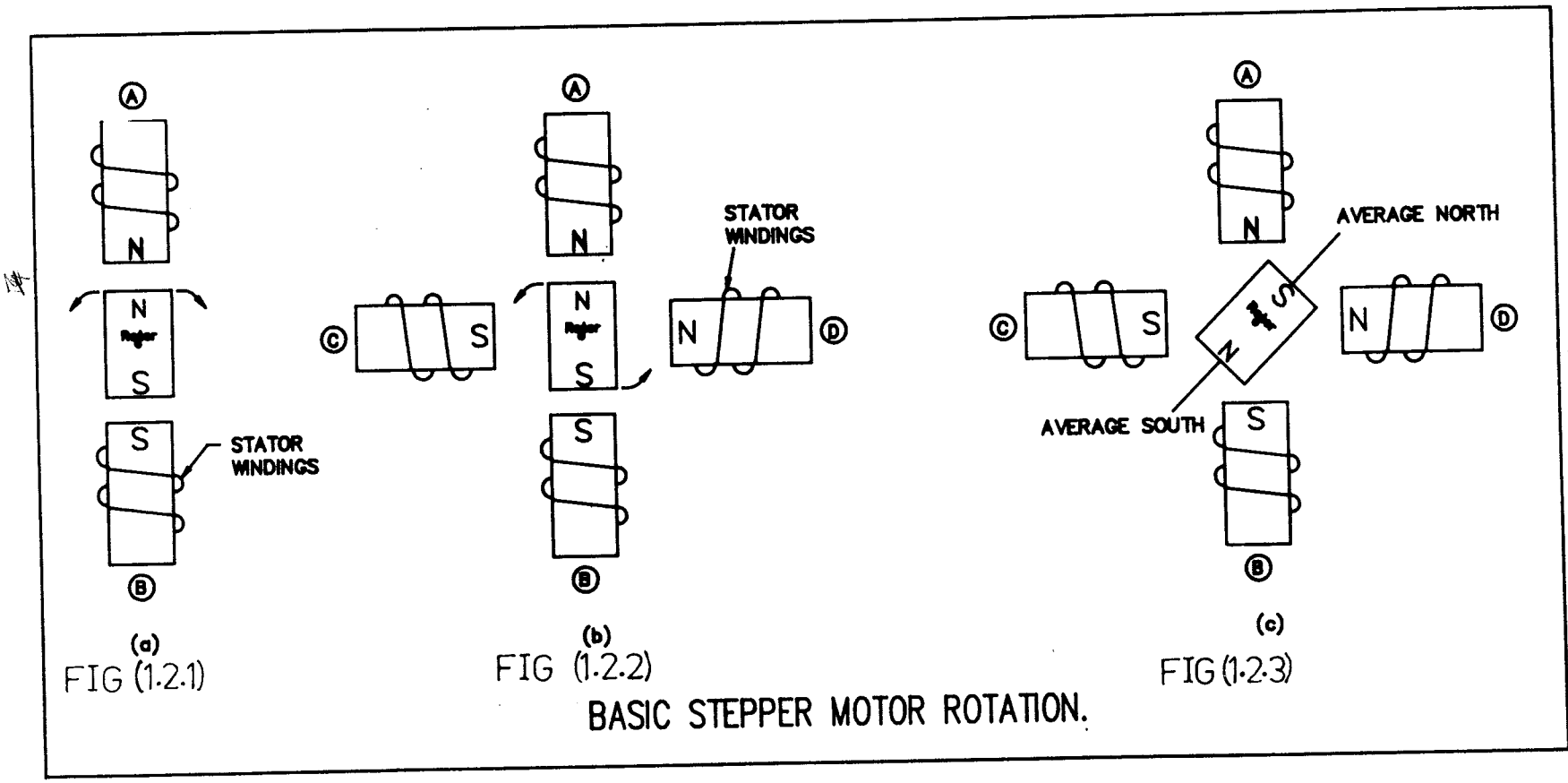


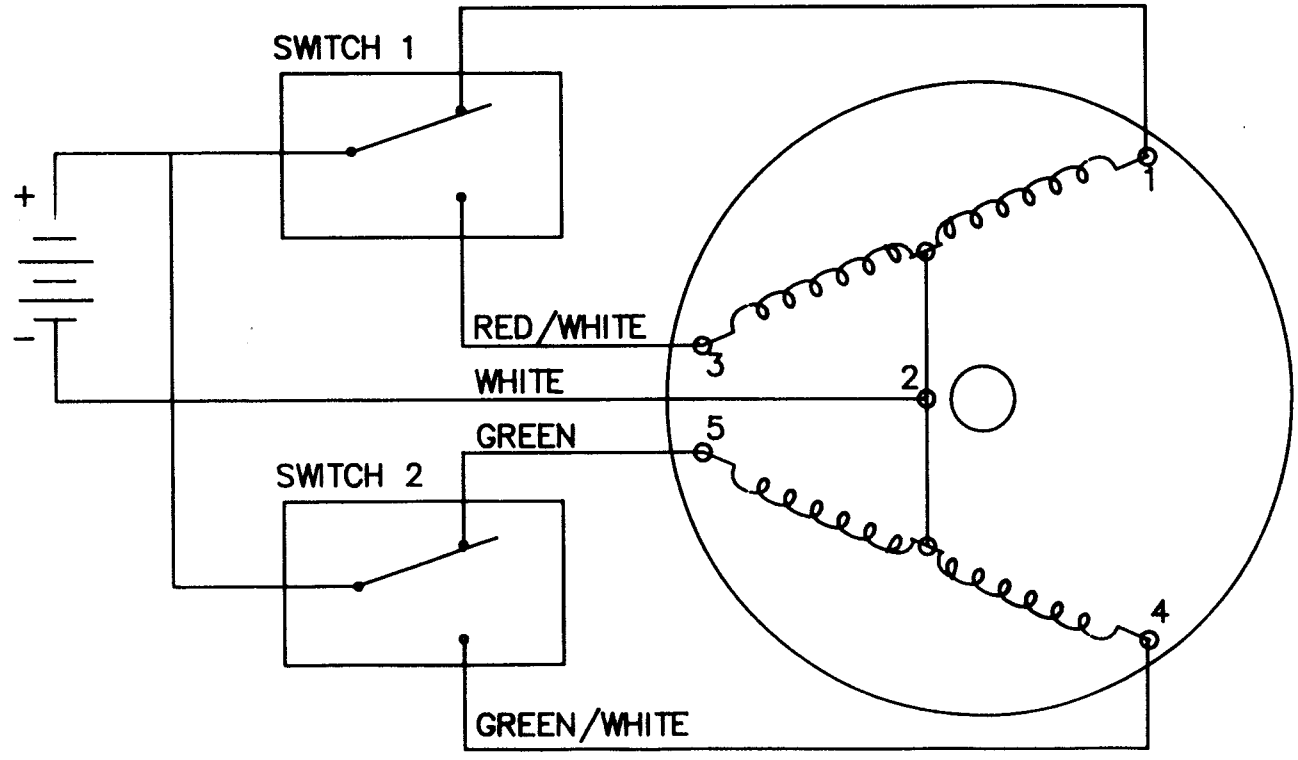
FIG (1.2.1)

FIG (1.2.2)

FIG (1.2.3)

BASIC STEPPER MOTOR ROTATION.

FIG.( ) FOUR-STEP SEQUENCE OF A PERMANENT MAGNET STEPPER MOTOR.



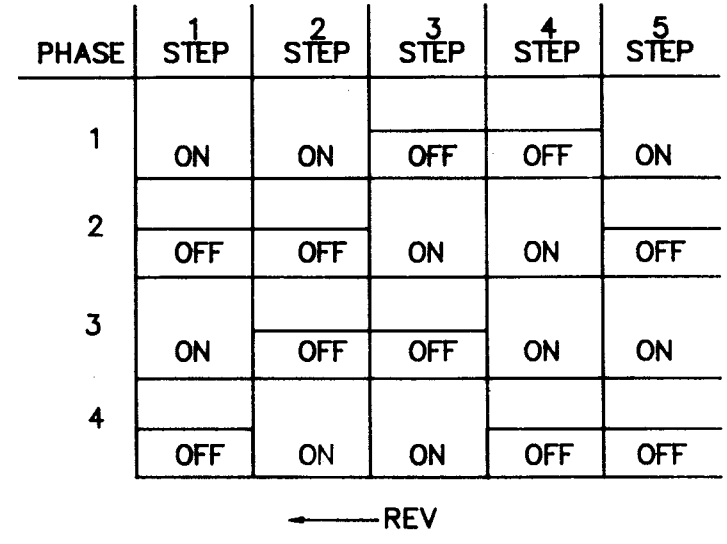
(a) SCHEMATIC DIAGRAM

FIG(1.3)

STEP	SWITCH #1	SWITCH #2
1	1	5
2	1	4
3	3	4
4	3	5
1	1	5

*TO REVERS DIRECTION, READ CHART UP FROM BOTTOM

(b) SWITCHING SEQUENCE.



(c) WAVEFORMS

FIG(1.3.1)

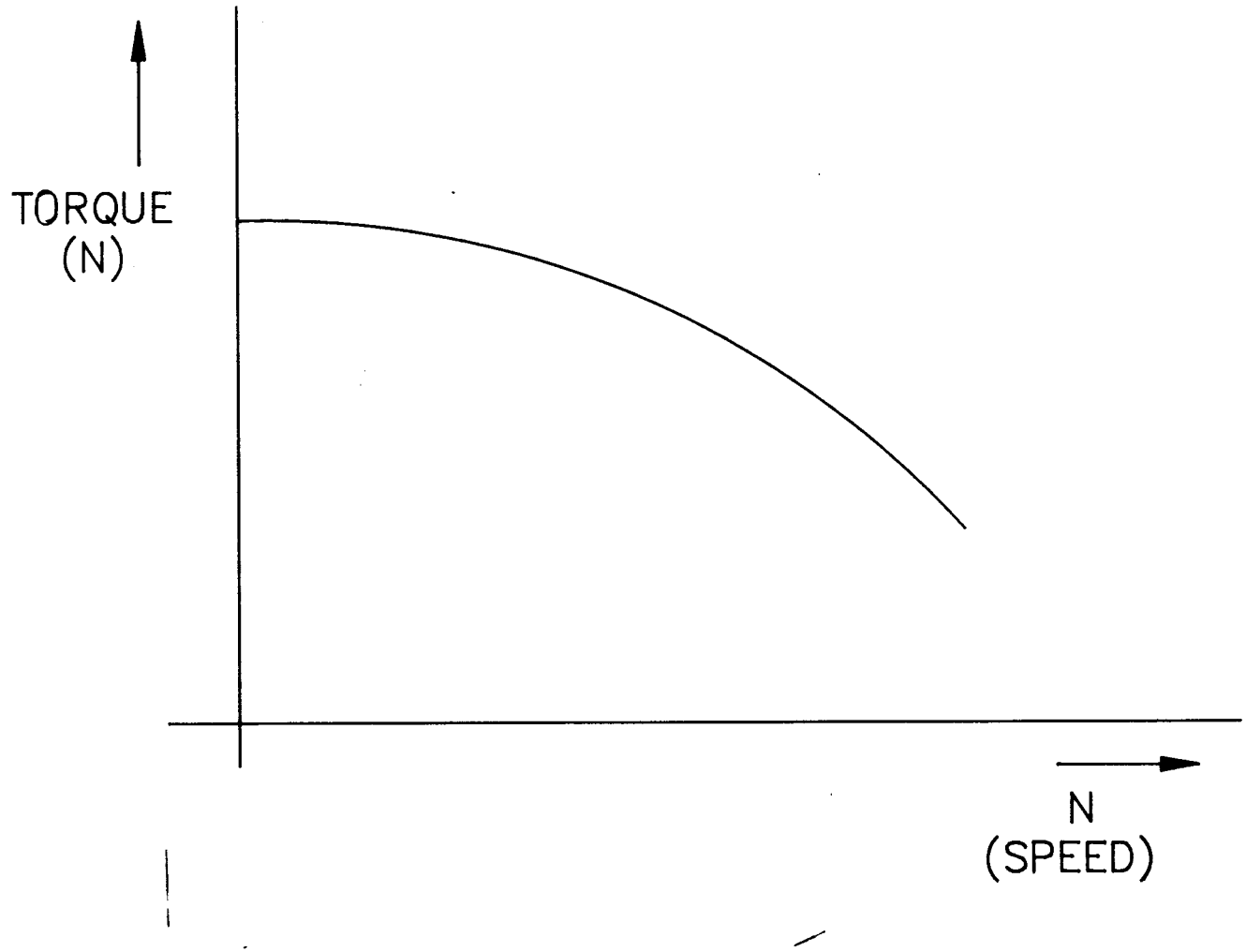


FIG.(1.4) SPEED–TORQUE CHARACTERISTICS OF A STEPPER MOTOR

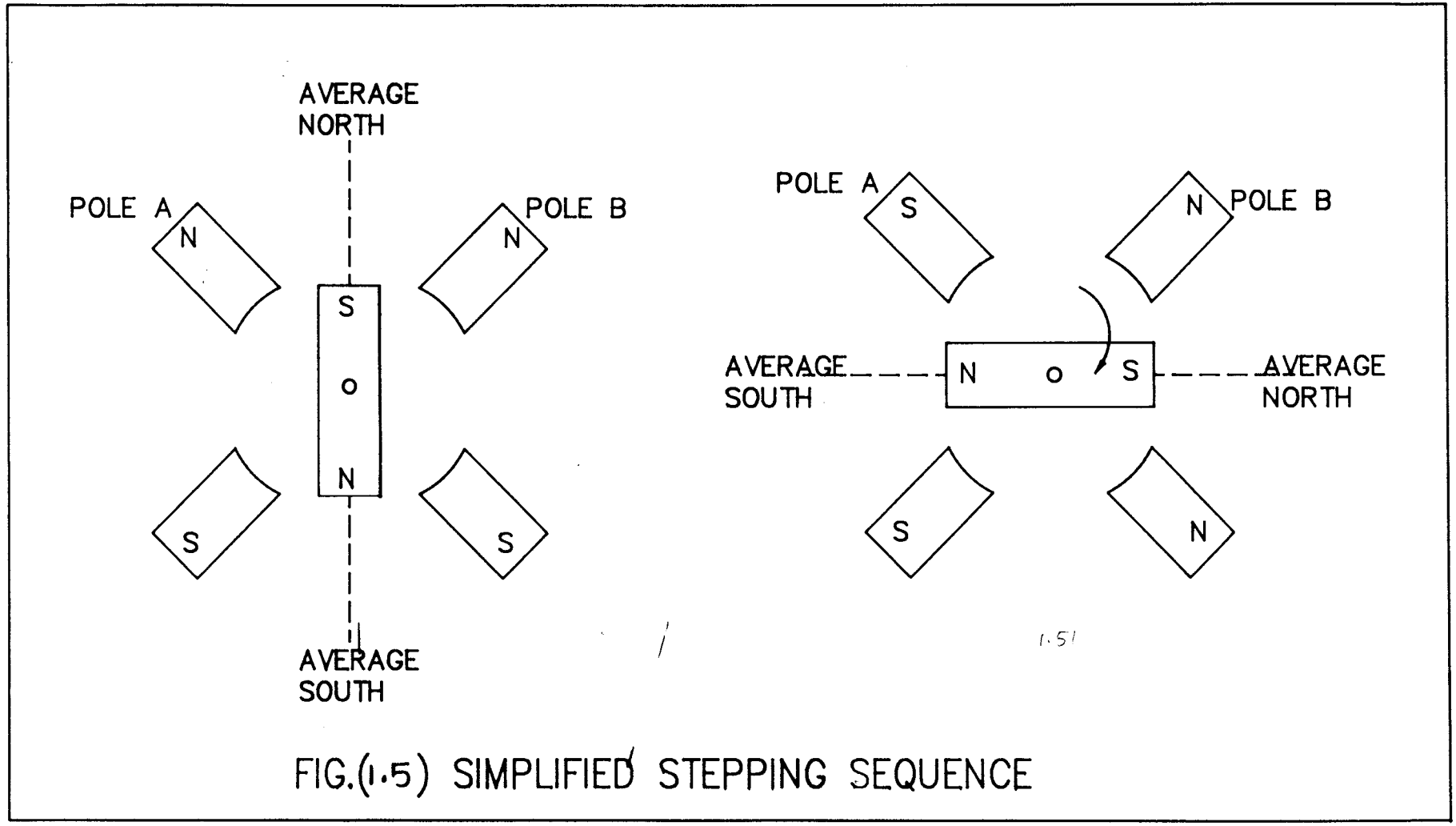
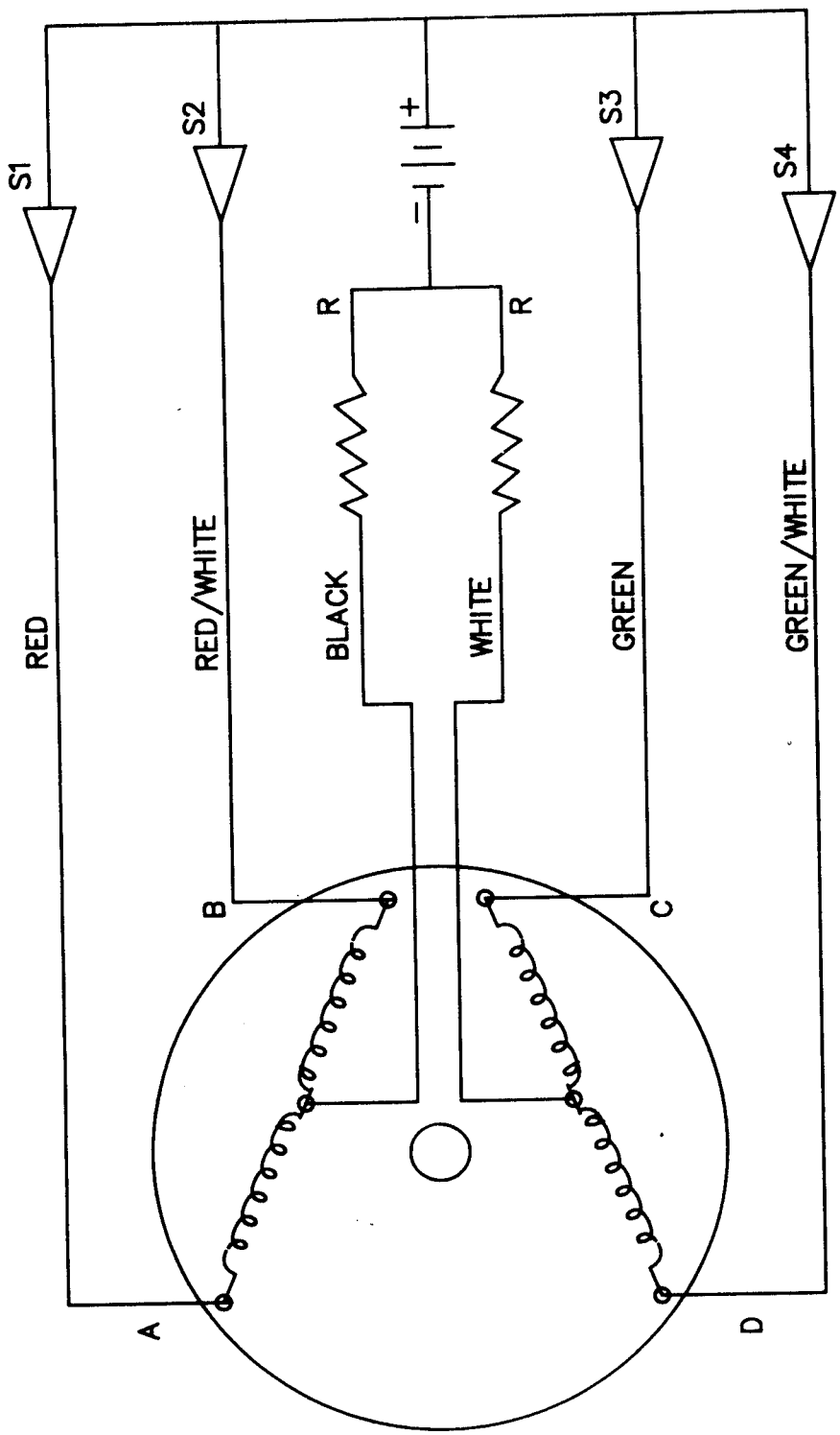


FIG.(1.5) SIMPLIFIED STEPPING SEQUENCE

FIG.( ) BIFILAR STEPPER MOTOR.



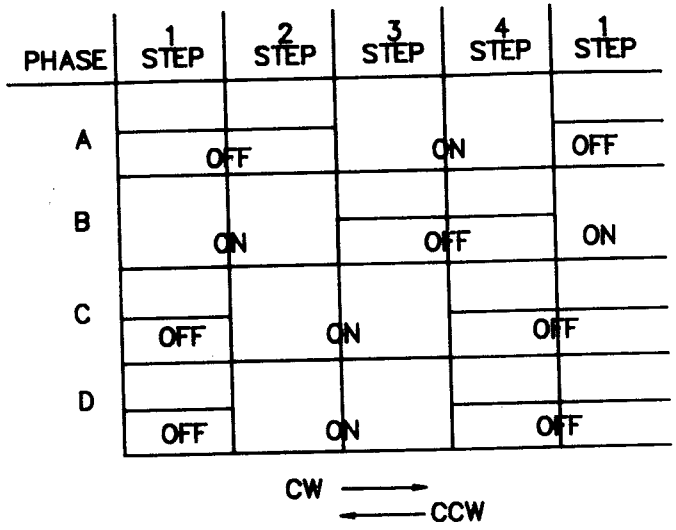
FIG(1.6)(a) CIRCUIT DIAGRAM

CW

STEP	S1	S2	S3	S4
1	OFF	ON	OFF	ON
2	OFF	ON	ON	OFF
3	ON	OFF	ON	OFF
4	ON	OFF	OFF	ON
1	OFF	ON	OFF	ON

CCW

*TO REVERS DIRECTION, READ CHART UP FROM BOTTOM.

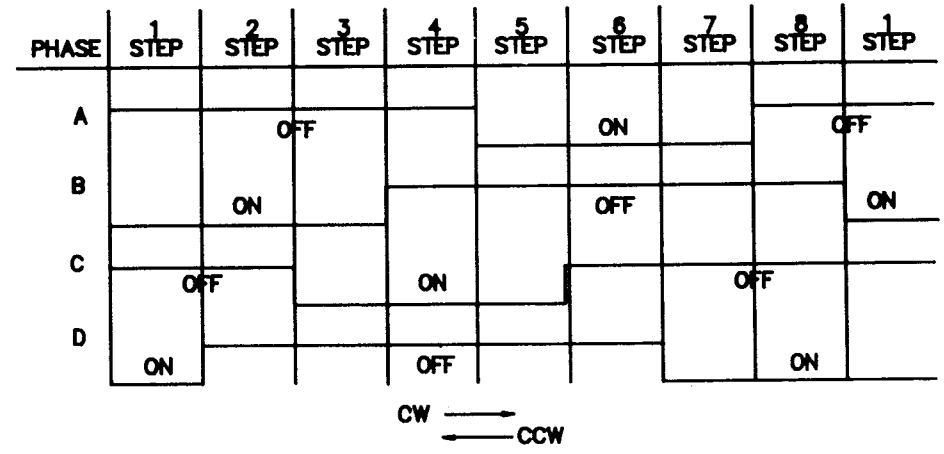


FIG(1.6.1)(b) FULL-STEP SEQUENCE AND WAVEFORMS.

STEP	S1	S2	S3	S4
1	OFF	ON	OFF	ON
2	OFF	ON	OFF	OFF
3	OFF	ON	ON	OFF
4	OFF	OFF	ON	OFF
5	ON	OFF	ON	OFF
6	ON	OFF	OFF	OFF
7	ON	OFF	OFF	ON
8	OFF	OFF	OFF	ON
1	OFF	ON	OFF	ON

CW ↓

↑ CCW



FIG(1.6.2)(c) HALF-STEP SEQUENCE AND WAVEFORMS.

CHAPTER II

8255A PROGRAMMABLE PERIPHERAL INTERFACE (PPI)

2.1 INTRODUCTION

The 8355A is used as general purpose programmable device. It is compatible with any microprocessor. The 8255A includes three programmable ports, one of which can be used for bi-directional data transfer. This is an important additional feature in comparision with the 8155A I/O ports. The 8255A is a widely used, programmable, parallel I/O device. It can be programmed to transfer data under various condition, from I/O to interrupt I/O. It is flexible, versatile and economical when multiple I/O ports are required.

2.2 METHODS OF PARALLEL DATA TRANSFER

Parallel communication is accomplished by simultaneously transfering bits over seperate lines. Its advantages over serial communication is that for lines with a given maximum bit rate, higher information rates can be attained due to the use of several lines. The disadvantage is the loss of the extra lines. This loss increases with distance. Parallel communication is used over long distances only if high data rates are required.

Parallel transfer may be performed by simply putting data in the interface's data in buffer register or taking data out of the interface's data-out buffer register, or they may be conducted under the control of timing signals. Normally one character is transfered at a time and there is no problem in determining the end of the character or in finding the begining of synchoronous or asynchronous transmissions, but is a timing signal is to co-ordinate the activity between the device and the interface, the transmission would be considered synchronous.

An interface may be designed for only outputting, only inputting, inputting and outputting over separate sets of the lines or performing I/O over a signal set of bi-directional lines.

1. SIMPLE INPUT AND OUTPUT

In this method when you need to get a data from a device, connect the device to an input port line and read the port and the device has data always present and ready so that it could be read it at any time. Likewise when you need to output data to a display device, all you have to do is connect the input of the display buffer on an output port pin. The display is always ready so that you can send data at anytime. The timing waveform is shown in Fig (2.1.1)

represents this situation. The crossed lines on the waveform represents the time at which a new data byte becomes valid on the output lines of the port. The absense of the other waveforms indicates that this output operation is not directly dependent on only other signals.

2. SIMPLE STROBE I/O

In many applications valid data is only present on an external device at a certain time and it must be read in at that time. An example of this is the ASCII encoded keyboard. When a key pressed on the keyboard, circuitry on the key board sents out the ASCII code for the pressed key on eight parallel data lines. The keyboard circuitry then sends out a strobe signal on another line to indicate that valid data is present on the eight data lines. The point here is that this transfer is time dependent. You can only read in data when a stroke pulse tells you that the data is valid. The Fig (2.1.2) shows the timing waveforms which represents this type of operation. The sending device, such as a keyboard, outputs parallel data on the data lines and then outputs an STB signal to let you know that valid data is present.

For low rates of data transfer, such as from a keyboard to a microprocessor, a simple strobe transfer works well. However, for high-speed data transfer this method does not

work because there is no signal which tells the sending device when it is safe to send the next data byte. In other words the sending system might send data bytes faster than the receiving system could read them. To prevent this problem a hand shake data transfer scheme is used.

3. SINGLE HANDSHAKE I/O

The Fig (2.1.3) shows some example timing waveforms for a handshaker data transfer from a peripheral device to a microprocessor. The peripheral outputs some parallel data and sends an STB signal to the microprocessor. The microprocessor detects the asserted STB signal on a polled or interrupt basis and reads in the byte of data. The microprocessor then sends an acknowledge signal ACK, to the peripheral byte of data from the viewpoint of the microprocessor, this operation is referred to as a handshake or strobed input.

The point of this handshake scheme is that the sending device or system cannot send the next data byte until the receiving device or system indicates with an ACK signal that it is ready to receive the next byte.

4. DOUBLE HANDSHAKE DATA TRANSFER

For data transfers where even more coordination is

required between the sending system and the receiving system, a double handshake is used Fig (2.1.4) shows some example waveforms for a double handshaker input from a peripheral to a microprocessor. Perhaps it will help, to follow these waveforms by thinking as conversation between two people. In these waveforms each signal edge has meaning. The sending device asserts its STB line low to ask "Are you ready?". The receiving system raises its ACK line high to say, "I'm ready". The peripheral device then sends the bytes of data and raises its STB line high to say "Here is some valid data for you". After it has read in the data the receiving system drops its ACK line low to say, "I have the data, thank you, and I await your request to send the next byte of data".

For a handshaker output of this type, from a microprocessor to a peripheral, the waveforms are the same but the microprocessor send the STB signal and the data, and the peripheral sends the ACK signal. For handshaker data transfer, a microprocessor can determine when it is time to send the next data byte on a polled or on as interrupt basis. We usually use the interrupt approach because it makes better use of the processor's time. The STB or ACK signals for these handshake transfers can be produced on a port pin by instruction in the program. The 8255A can be programmed to automatically receive a STB signal from a peripheral, send

an interrupt signal to the processor, and send the ACK signal to the peripheral at the proper times.

2.3 BLOCK DIAGRAM OF THE 8255A

The 8255A has 24 I/O pins that can be grouped primarily in two 8-bit parallel ports: A and B, with the remaining eight bits as port C. The eight bits of port C can be used as individual bits or grouped in teo 4 bitports: C upper (Cu) and C lower (Cl). Port A can be used as an 8 bit input port or as an 8 bit output port. Likewise port B can be used as an 8 bit input or as an 1 bit output port. The internal block diagram of 8255A programmable parallel port device is shown in Fig (2.2). In the left side of the diagram the usual signal lines used to connect the device to the system buses is shown and Eight data lines allow to write data bytes to a port or the control register and to read bytes from a port or the status register under the control of the RD and WR lines.

The RESET input of the 8255A is connected to the system reset line so that, when the system is reset, all of the port lines are initialized as input lines. This is done to prevent destruction of circuitry connected to port lines. If port lines were initialized as outputs after a power-up or reset, the port might try to output into the output of a

device connected to the port. The possible argument between the two outputs might destroy one or both of them. Therefore all of the programmable port devices initialize their port lines as input when reset.

2.4 FUNCTIONS OF THE 8255A

The Fig (2.3) shows all the functions of the 8255A, classified according to two modes, the Bit Set / Reset (BSR) mode and the I/O mode. The BSR mode is used to set or reset the bits in port C. The I/O mode is further divided into three modes: mode 0, mode 1 and mode 2. In mode 0, all ports function as simple I/O ports. Mode 1 is a handshake mode where by ports A and /or B use bits from port C as handshake signals. In the handshake mode, two types of I/O data transfer can be implemented: status check and interrupt. In mode 2, port A can be set up for bi-directional data transfer using handshake signals from port C, and port B can be set up either in mode 0 or mode 1.

2.4.1 CONTROL LOGIC

The control section has six lines. Their functions and connections are as follows

(a) RD (Read) : This control signal enables the Read operation. When the signal is low, it reads data from a selected I/O port of the 8255A.

(b) WR (Write) : This control signal enables the write operation when the signal goes low, the device writes into a selected I/O port or the control register.

(c) RESET (Reset) : This is an active high signal it clears the control register and sets all ports in the input mode.

(d) CS, Ao and Al : These are device select signal. CS is connected to a decoded address and Ao and Al are generally connected to device address lines Ao and Al, respectively.

The CS is the master chip select, and Ao and Al specify one of the I/O ports on the control register as given below

CS	Al	Ao	Selected
0	0	0	Port A
0	0	1	Port B
0	1	0	Port C
0	1	1	Control register
1	X	X	8255A is not selected

In Fig (2.4) shows a register called the control register. The contents of this register, called the control

word, specify an I/O function for each port. This register can be accessed to write a control word when Ao and Al are at logic 1. The register is not accessible for a Read operation.

Bit D7 of the control register specifies either the I/O function or the Bit Set / Reset function. If bit D7 = 1, bits D6-DO determine I/O functions in various modes. If bit D7=0, port C operates in the Bit Set / Reset mode. The BSR control word does not affect the functions of port A and B. To communicate with peripherals through the 8255A, three steps are necessary.

(a) Determine the addresses of ports A, B and C of the control register according to the chip select logic and address lines Ao and Al.

(b) Write a control word in the control register.

(c) Write I/O instructions to communicate with peripherals through ports A,B and C.

2.5 8255A MODES AND INITIALIZATION

The Fig (2.5) shown summarizes the different modes in which the ports of the 8255A can be initialized. The control word formats of 8255A in I/O mode and BSR mode is also shown in Fig (2.6).

MODE 0

To use a port for simple input or output without handshaking, initialize that port in MODE 0. If both port A and port B are initialized in MODE 0, then, the two halves of port c can be used together as an additional 8-bit port, or they can be used individually as two 4-bit ports. When used as outputs the port C lines can be individually set or reset by sending a special control word to the control register address. The two halves of port C are independent, so one half can be initialized as input, and the other half initialized as output.

MODE 1

To use port A or port B for a handshake (stobed) input or output operation, initialize that port in MODE 1. In this MODE some of the pins of port c function as handshake lines. Pins PC0, PC1 AND PC2 function as handshake lines for port B if it is intialized in MODE 1. If port A is initialized as a handshake (mode 1) input port, then pins PC3, PC4 AND PC5 function as handshake signals. Pins PC6 and PC7 are available for use as input lines or output lines. If port A is initialized as a handshake output port, then port C pins PC3, PC6 and PC7 function as handshake signals. Port C pins PC4 and PC5 are available for use as input or output lines. The 8255A is often used in this mode.

Any bits of port C which are programmed as inputs can be read by simply doing a read from the port C address. You can mask out any unwanted bits of the word read in. If port A and/or port B is programmed in a handshake mode, then some of the bits of a byte read in from port C represent status information about the handshake signals. The Fig (2.5) shows the meaning of the bits read from port C for port A and/or port B in MODE 1. If port B is initialized as a handshake (MODE 1) input port, then bits D0, D1 and D2 read from port C represent the status of the port B handshake signals. Bit D2 will be high if the port B interrupt request output has been enabled. Bit D2 is a copy of the level on the input buffer full (IBF) pin. Bit D3 is a copy of the interrupt request output, so it will be high if port B is requesting an interrupt.

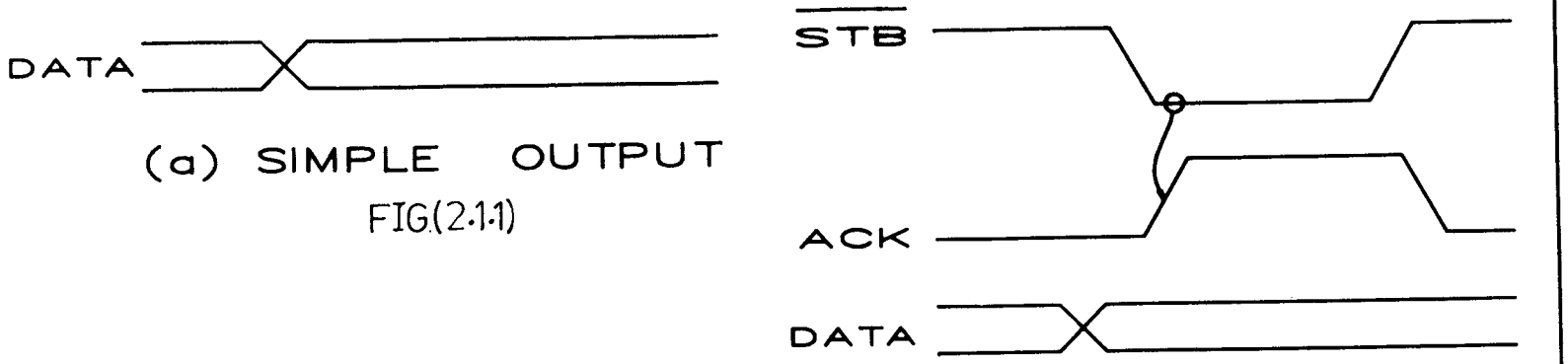
Port C bits not used foir handshake signals and programmed as output can be written to by sending bit set/reset (BSR) control words to the control register. Technically, bits PC0 - PC3 can also be written to directly at the port C address, but we have found it safer to just use the bit set/reset control word approach to write to all left over port C bits programmed as outputs.

MODE 2

Only port A can be initialized in MODE 2. The MODE 2, port A can be used for bi-directional handshake data transfer. This means that data can be output or input on the same eight lines. The 8255A might be used in this mode to extend the system bus to a slave or to transfer data bytes to and from a controller board. If port A is initialized in MODE 2, then pins PC3 - PC7 are used as handshake lines for port A. The other three pins of port C can be used for I/O if port B is in MODE 0. The three pins will used for port B handshake lines if port B is initialized in MODE 1.

BIT SET / RESET MODE (BSR)

The BSR mode is concerned only with the eight bits of port C, which can be set or reset by writing an appropriate control word in the control register. A control word with bit D7=0 is recognized as a BSR control word, and it does not alter any previously transmitted control word with bit D7=1; thus the I/O operations of port A and B are not affected by a BSR control word. In the BSR mode, individual bits of port C can be used for applications such as an on / off switch.

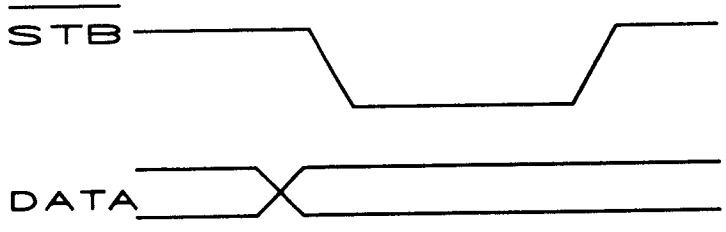


(a) SIMPLE OUTPUT

FIG.(2.1.1)

(c) SINGLE HANDSHAKE I/O

FIG.(2.1.3)



(b) SIMPLE STROBE I/O

FIG.(2.1.2)

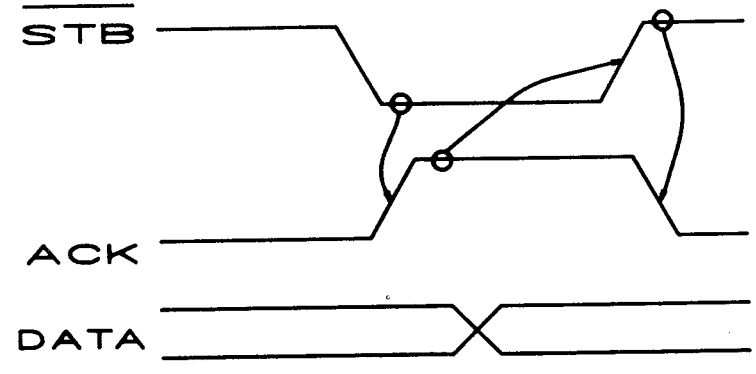


FIG.(. .)(d) DOUBLE HANDSHAKE I/O

PARALLEL DATA TRANSFER

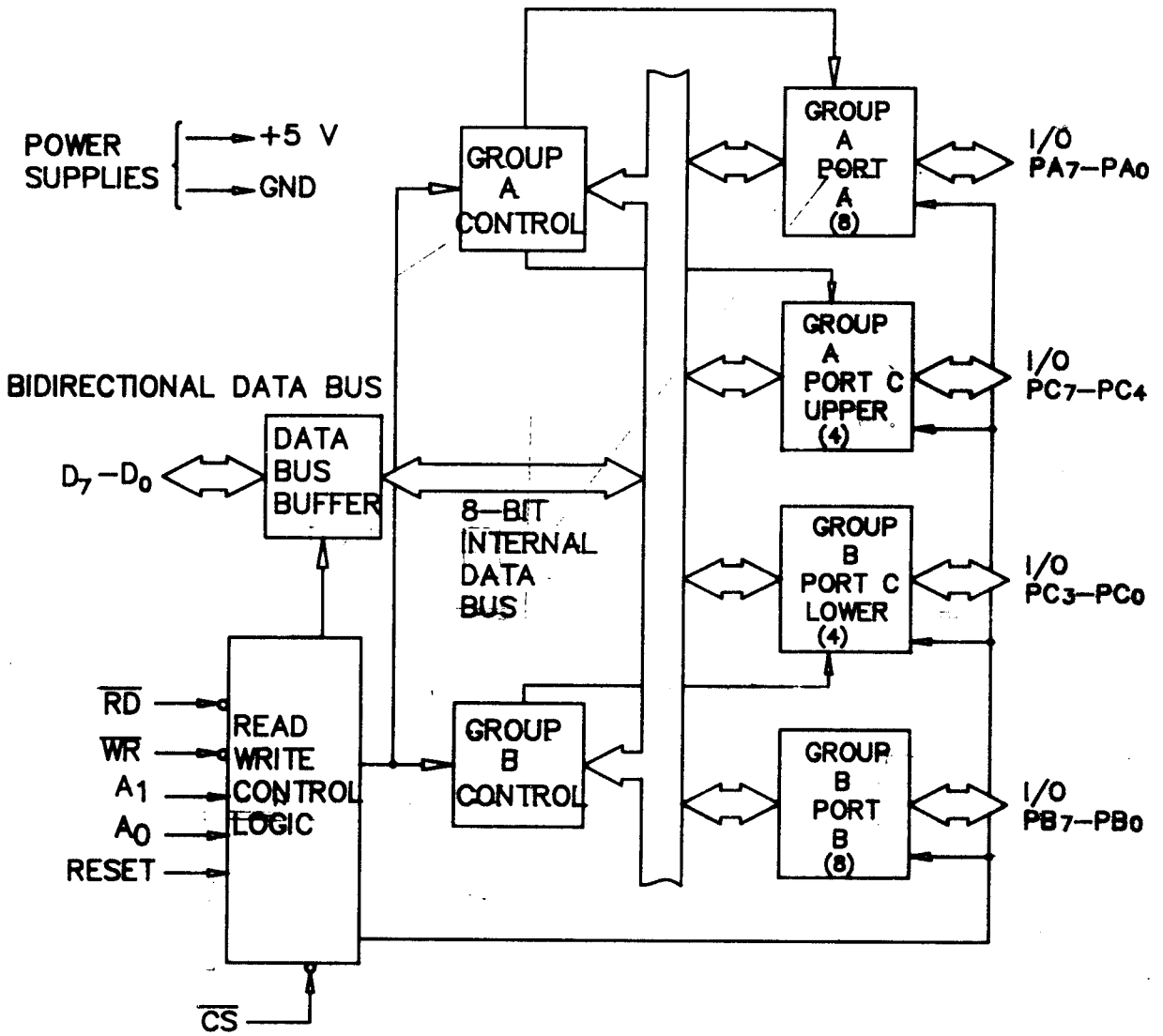
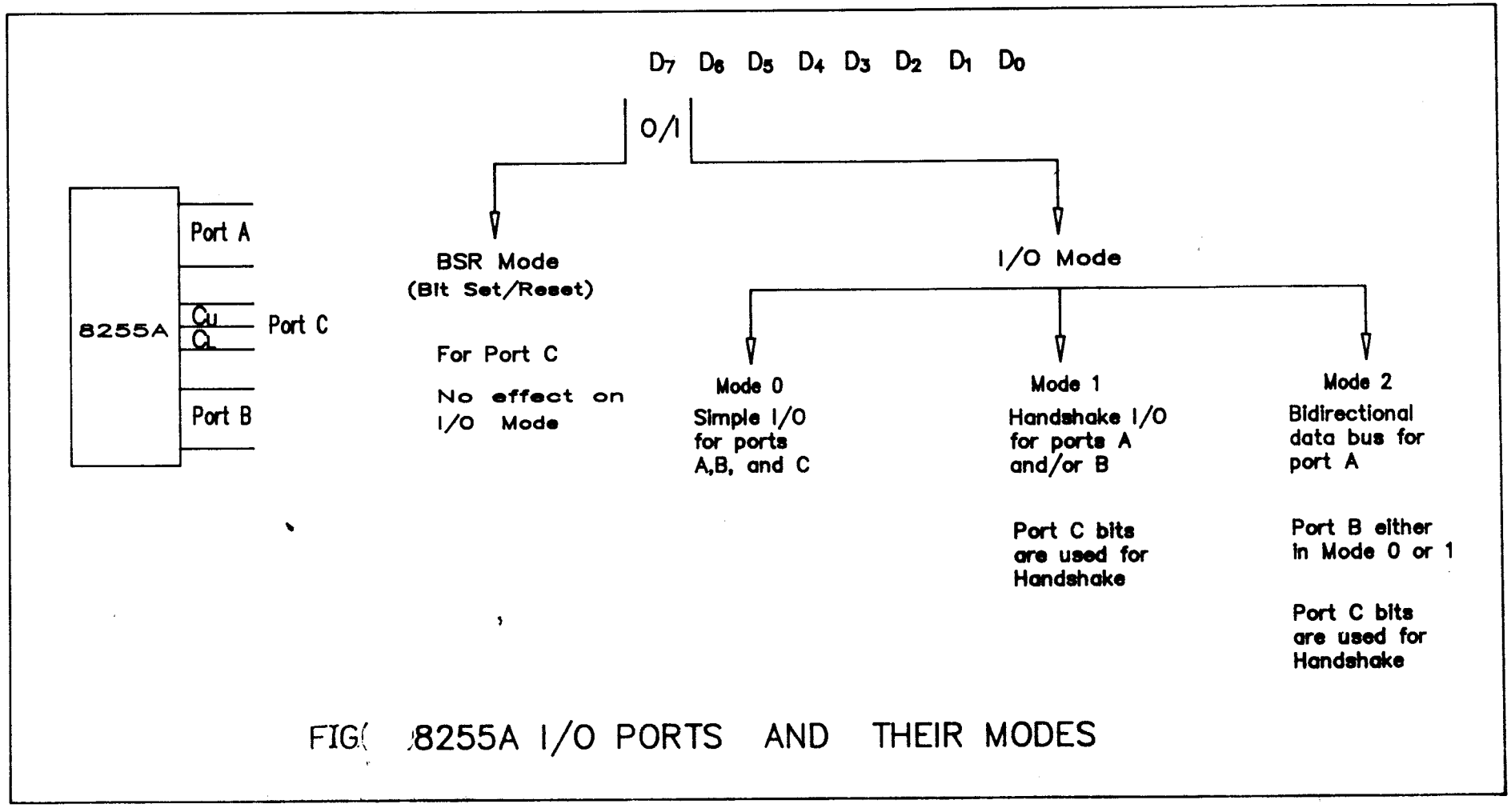


FIG.( ) BLOCK DIAGRAM OF 8255A AND EXPANDED VERSION OF THE CONTROL LOGIC



FIG(8255A I/O PORTS AND THEIR MODES

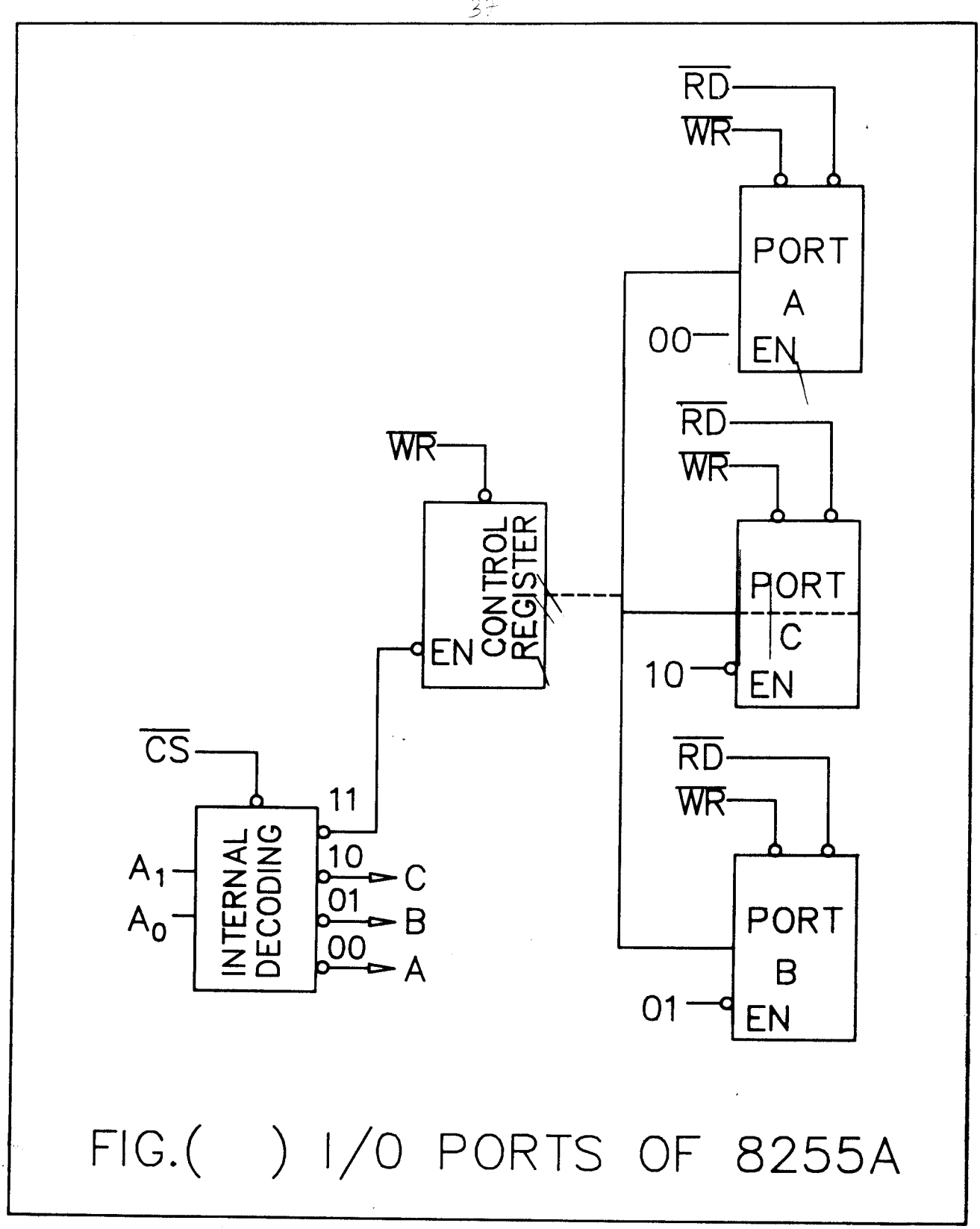


FIG.( ) I/O PORTS OF 8255A

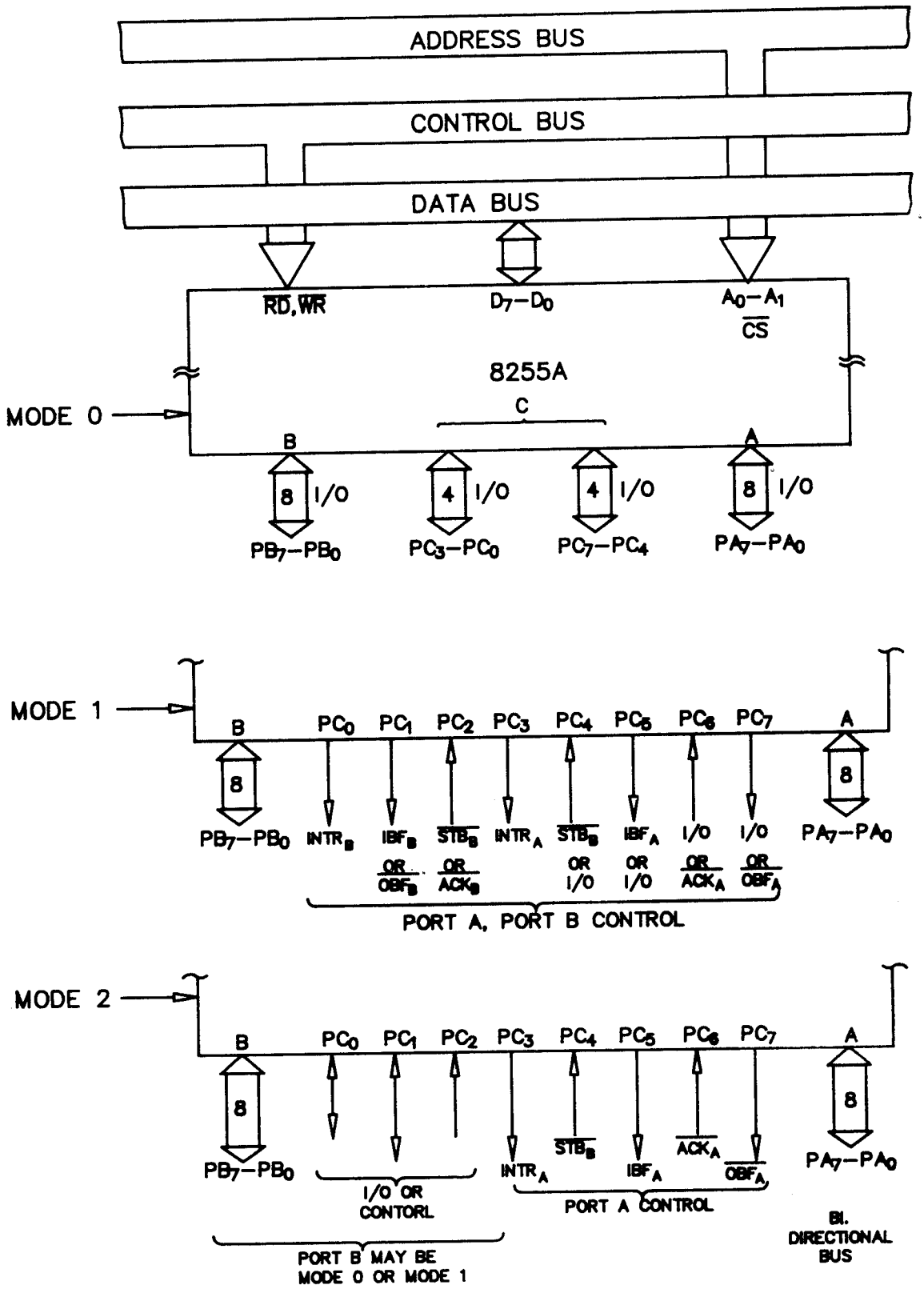
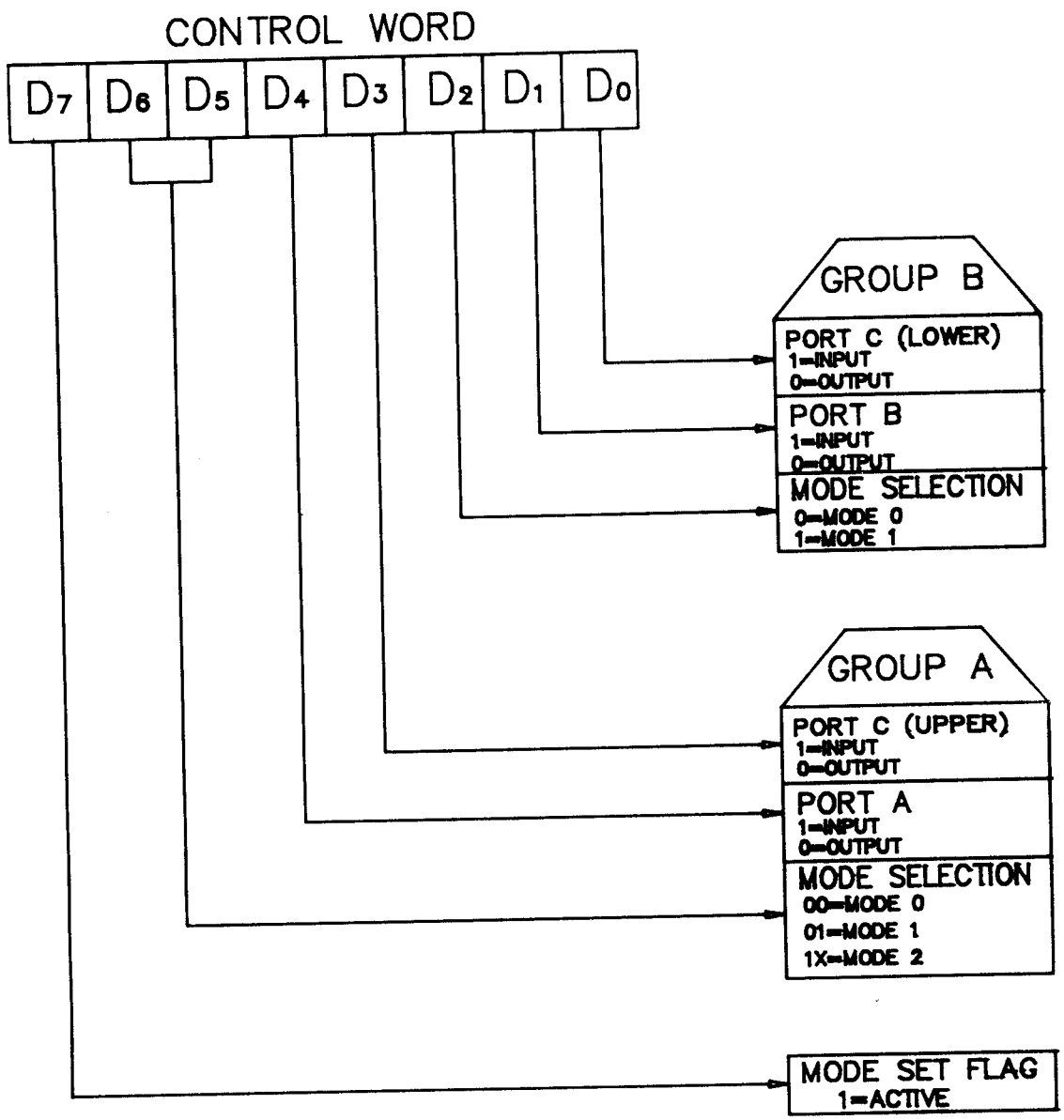


FIG.( ) SUMMARY OF 8255A OPERATING MODES.

FIG ( ) 8255A CONTROL WORD FORMATS



(a) MODE SET CONTROL WORD

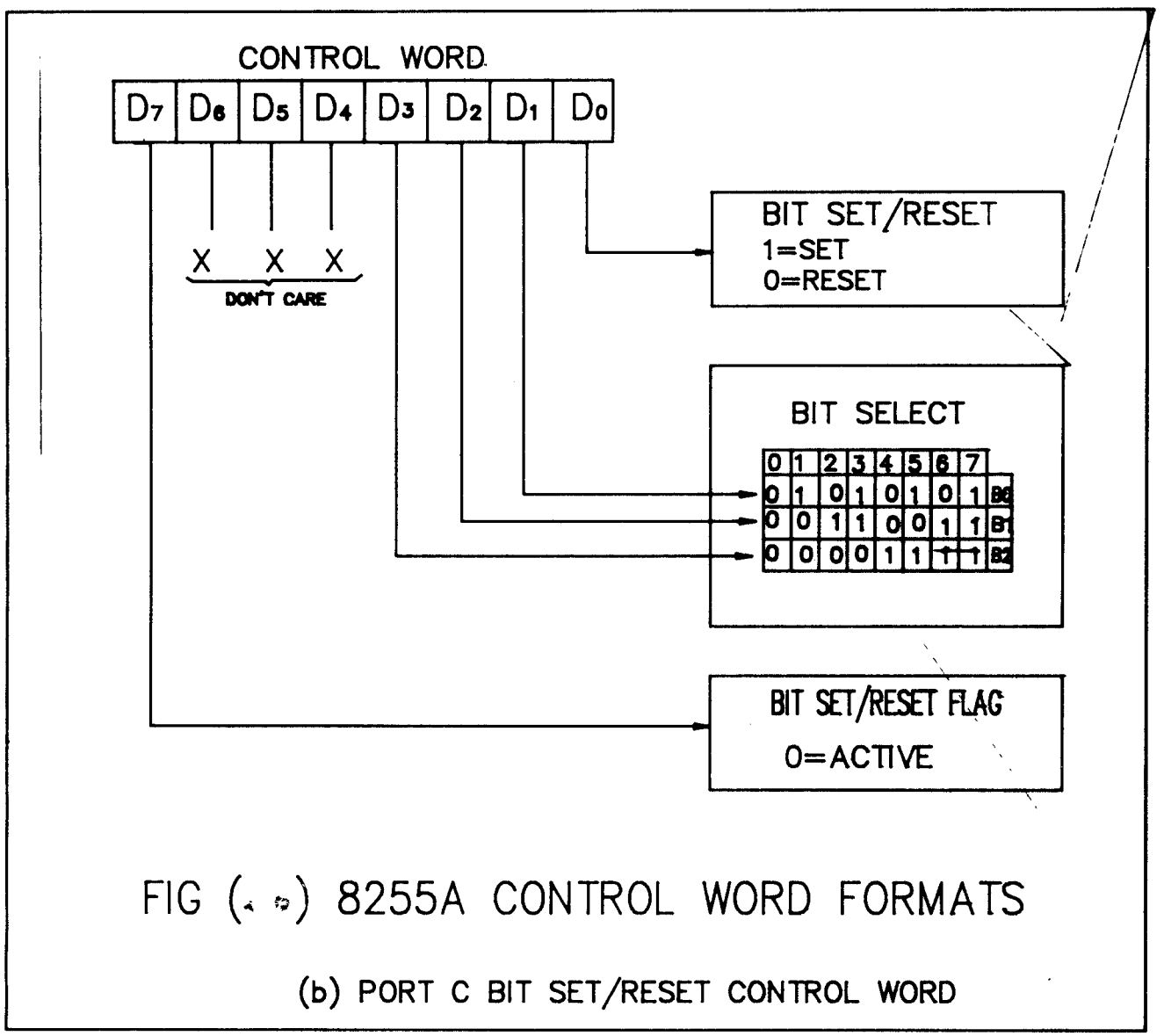
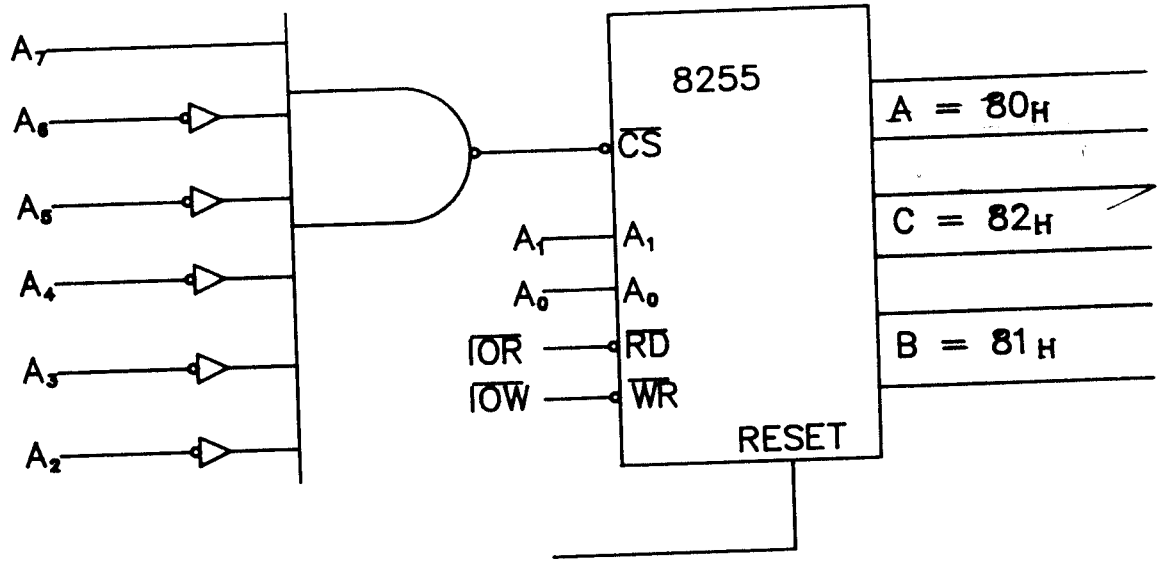


FIG (4.5) 8255A CONTROL WORD FORMATS

(b) PORT C BIT SET/RESET CONTROL WORD



(a)

$\overline{CS}$			HEX ADDRESS	PORT
A7 A6 A5 A4 A3 A2	A1	A0		
1 0 0 0 0 0	0	0	= 80H	A
	0	1	= 81H	B
	1	0	= 82H	C
	1	1	= 83H	CONTROL REGISTER

(b)

FIG.( ) 8255A CHIP SELECT LOGIC (a) and I/O PORT ADDRESS (b)

CHAPTER III

DIGITAL INPUT / OUTPUT CARD FOR PC

3.1 INTRODUCTION

The Digital I/O Card is IBM PC / XT / AT compitable with two 8255A PPI's which provide 48 programmable I/O lines. The 24 I/O lines of first 8255A are brought out through a 26 pin ribbon cable connector. The next 24 I/O lines of the second 8255A are brought out through another 26 pin ribbon cable connector. The ports of the 8255A's can be mapped at different addresses, using the 8 way DIP switch provided.

3.2 INSTALLATION

Install the card in any one of the available slots in the system. The diagram and pin details are shown in Fig (3.1).

3.3 ADDRESS MAPPING

The ports of the two 8255A's can be mapped addresses. The lower addresses refer to the first 8255A whose output are brought out on 26 pin ribbon cable connector and the higher addresses for the second 8255A whose outputs are brought out on another 26 pin ribbon cable connector. For example if the address range is specified as 0380H - 0380H refer to the

first 8255A and the addresses 0384 - 0387H refer to the second 8255A.

The address 0380H refers to port A of lower 8255A (J)

0381H to port

0382H to port C and

0383H to control port

similarly the address

0384H refers to port A of upper 8255A (J2)

0385H to port B

0386H to port C and

0387H to port control port

The I/O address can be selected by the 8 way DIP switch. All the possible address ranges are given the table below.

8 SELECTION DIP SWITCH SETTINGS

SW8	SW7	SW6	SW5	SW4	SW3	SW2	SW1	ADDRESS
ON	OFF	ON	OFF	ON	OFF	ON	OFF	0380H-0387H*
ON	OFF	ON	OFF	ON	OFF	ON	OFF	0388H-038FH
ON	OFF	ON	OFF	ON	OFF	ON	OFF	0390H-0397H
ON	OFF	ON	OFF	OFF	ON	OFF	ON	0398H-039FH
ON	OFF	OFF	ON	ON	OFF	ON	OFF	03AOH-03A7H
ON	OFF	OFF	ON	ON	OFF	OFF	ON	03A8H-03AFH
ON	OFF	OFF	ON	OFF	ON	ON	OFF	03BOH-03B7H
ON	OFF	OFF	ON	OFF	ON	OFF	ON	03B8H-03BFH
OFF	ON	ON	OFF	ON	OFF	ON	OFF	03COH-03C7H
OFF	ON	ON	OFF	ON	OFF	OFF	ON	03C8H-03CFH
OFF	ON	ON	OFF	OFF	ON	ON	OFF	03DOH-03D7H
OFF	ON	ON	OFF	OFF	ON	OFF	ON	03D8H-03DFH
OFF	ON	OFF	ON	ON	OFF	ON	OFF	03EOH-03E7H
OFF	ON	OFF	ON	ON	OFF	OFF	ON	03E8H-03EFH
OFF	ON	OFF	ON	OFF	ON	ON	OFF	03FOH-03F7H
OFF	ON	OFF	ON	OFF	ON	OFF	ON	03F8H-03FFH

* FACTORY SETTING

The following addresses cannot be made use of as they are used by the system for the followin||.

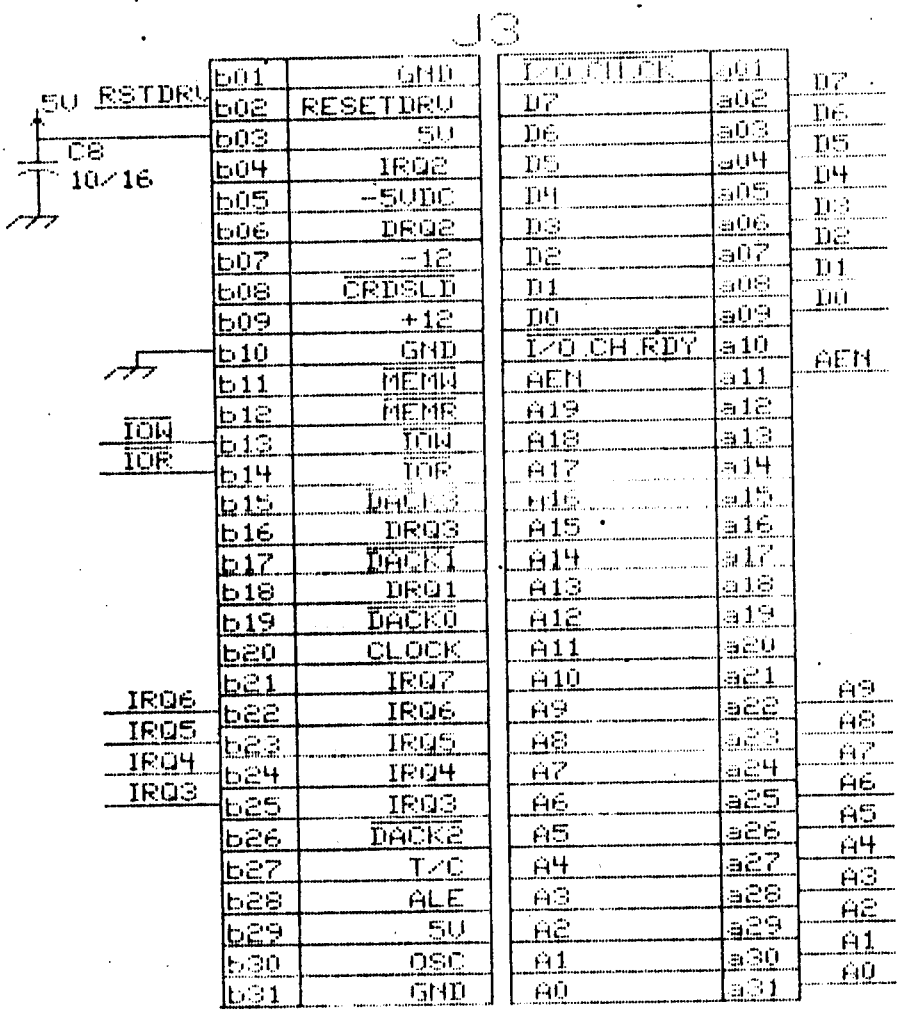
(1) 02D0H - 03DFH : colour graphics adapter

(2) 03FOH - 03F7H : diskette adapter

(3) 03F8H - 03FFH : serial port (COM 1)

CONNECTOR DETAILS

The 24 programmable I/O lines of the first 8255A are brought out on a 26 pin ribbon cable connector J1. For either case the pin connection are given. Care should be execised to refer to the correct type. The 24 programmable I/O lines of the second 8255A are brought out another 26 pin ribbon cable connector J2, the pin configuration of which is given in the diagram.



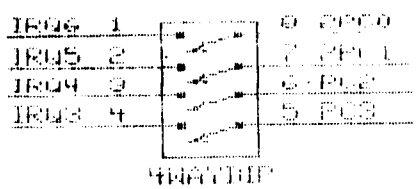


FIG 3.1 48 DIO CARD FOR IBM PC

CHAPTER IV

INTERFACING A MICROCOMPUTER TO A STEPPER MOTOR

A unique type of motor useful for moving things in small increments is the Stepper motor. Instead of rotating smoothly around or "step" from one fixed position to the next. Common step sizes range from 0.9 to 30. Stepper motor is stepped from one position to the next by changing the currents through the fields in the motor. The two common field connections are referred to as two - phase and four - phase.

Fig (4.1) shows a circuit can use to interface a small four - phase stepper motor. Since the IC 7406 buffers are inverting, a high on output - port pin turns on current to a winding. Fig (4.2) shows the switching sequence to step a motor such as this clockwise, as face the motor shaft, or counter clockwise. Here's how this works. Suppose that SW1 and SW2 are turned on. Turning off SW2 and turning on SW4 will cause the motor to rotate one step of 1.8 clockwise. Changing to SW4 and SW3 on will cause the motor to rotate another 1.8 clockwise. Changing to SW3 and SW2 on will cause another step. After that, changing to SW2 and SW1 on again will cause another step clockwise. You can repeat the sequence until the motor has rotated as many steps clockwise

as you want. To step the motor counterclockwise, you simply work through the switch sequence in the reverse direction. In either case the motor will be held in its last position by current through the coils. Fig (4.2) shows the switch sequence that can be used to rotate the motor half-steps of 0.9 clockwise or counterclockwise.

A close look at the switch sequence in Fig (4.2) shows an interesting pattern. To take the first step clockwise from SW2 and SW1 being on, the pattern of 1's and 0's is simply rotated one bit position around to the right. The 1 from SW1 is rotated around into bit 4. To take the next step the switch pattern is rotated one more bit position. To step counterclockwise the switch pattern is rotated one more bit position. To step counter-clockwise the switch pattern is rotated left one bit position for each step desired. Suppose that you initially load and output this to the switches. Duplicating the switch pattern in the upper half will make stepping easy. To step the motor clockwise, you just rotate this pattern right one bit position and output it to the switches. To counterclockwise you rotate the switch pattern left one bit position and output it. After you output one step code you must wait a few milli seconds before you output another step command, because the motor can only step so fast. Maximum stepping rates for different types of steppers

vary from a few hundred steps per second to several thousand steps per second. To achieve high stepping rates the stepping rate is slowly increased to maximum, then it is decreased as the desired number of steps is approached.

As a stepper motor steps to a new position it tends to oscillate around the new position before setting down. A common software technique to damp out this oscillation is to first send the pattern to step the motor toward the new position. When the motor has rotated part of the way to the new position, a word to step the motor backwards is output for a short time. This is like putting brakes on. The step forward word is then sent again to complete the step to the next position. The timing for the damping command must be determined experimentally for each motor load.

In case you want to add a stepper to your robot or some other project, first of all don't forget the clamp diodes across each winding to save the transistors from the current-limiting resistors , R1 and R2. The motor we used here has nominal voltage rating of 5.5 V. This means that we could have designed the circuit to operate with a voltage of about 6.5 V on the emitters of the driver transistors (5.5 V for the motor plus 1 V for the drop across the transistors). For low stepping rates, this would work fine. However, for higher stepping rates and more torque while stepping, we use

a higher supply voltage and current-limiting resistors as shown. The point of this is that by adding series resistance, we decrease the L/R time constant. This allows the current to change more rapidly in the windings. For the motor we used, the current per winding is 0.88A. Since only one winding on each resistor is ever on at a time, 6.5V/0.88A gives a resistor value of 6.25 ohms. To be conservative we used 8 ohms, 10 W resistors. The optional transistors switch and diode connection to the +5 V supply are used as follows. When not stepping, the switch to +12 V is off so the motor is held in position by the current from the +5 V supply. Before you send a step command, you turn on the transistors to +12 V to give the motor more current for stepping. When stepping is done you turn off the switch to +12 V, and drop back to the +5 V supply. This cuts the power dissipation. The home position sensor is shown in Fig (4.4).

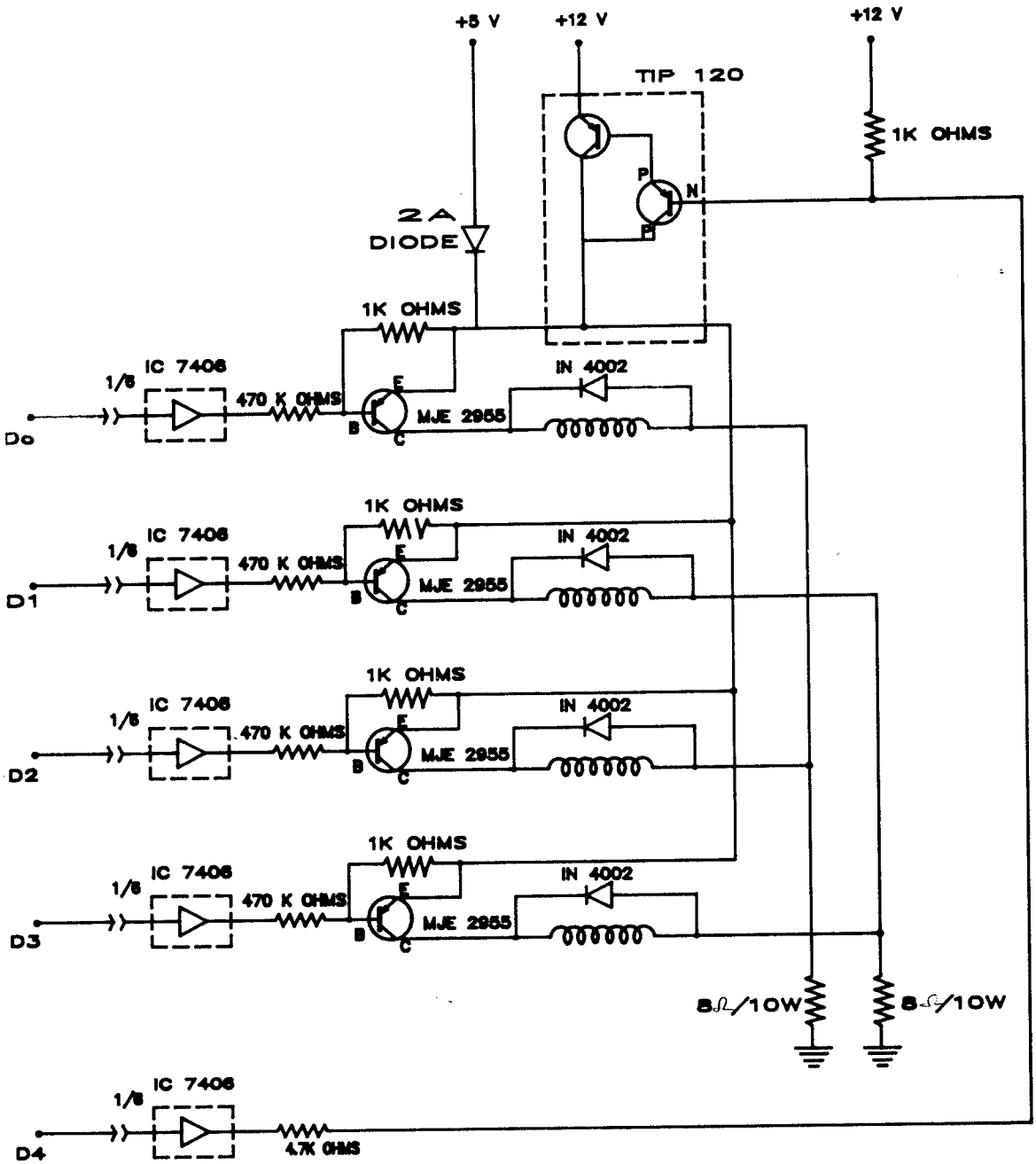
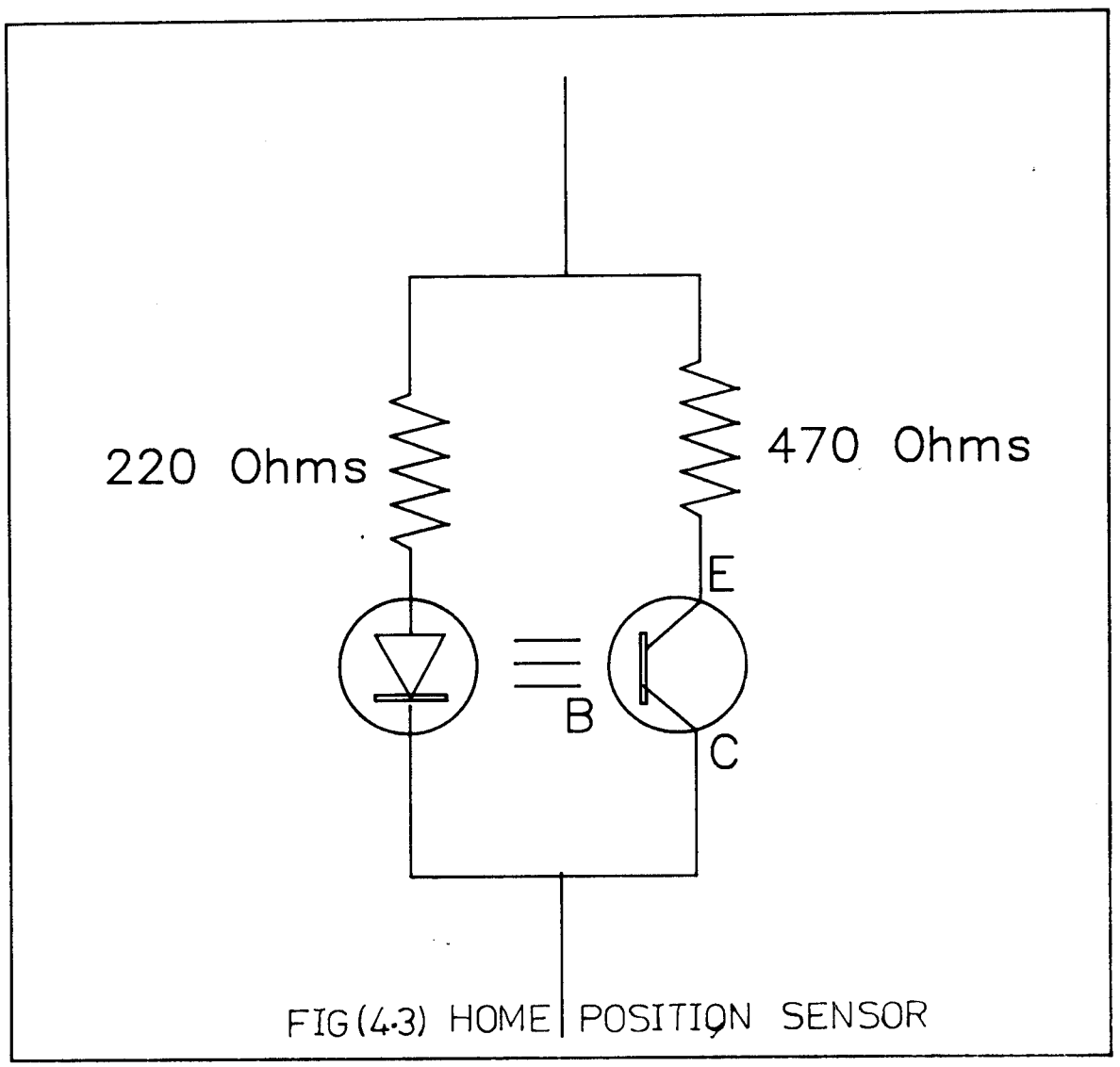


FIG.(4.1) DRIVE CIRCUIT FOR STEPPER MOTOR



FIG(4.3) HOME POSITION SENSOR

STEP	SWITCH				
	SW4	SW3	SW2	SW1	
1	0	0	1	1	CW
2	1	0	0	1	
3	1	1	0	0	
4	0	1	1	0	
1	0	0	1	1	CCW

1 SWITCH ON.

FULL STEP

EIGHT STEP INPUT SEQUENCE
(HALF STEP MODE)

STEP	SW4	SW3	SW2	SW1
1	OFF	OFF	ON	ON
2	OFF	OFF	OFF	ON
3	ON	OFF	OFF	ON
4	ON	OFF	OFF	OFF
5	ON	ON	OFF	OFF
6	OFF	ON	OFF	OFF
7	OFF	ON	ON	OFF
8	OFF	OFF	ON	OFF
1	OFF	OFF	ON	ON

CHAPTER V

PCB FABRICATION

5.1 PCB DRAWING IN SMART WORK

The Printed Circuit Board (PCB) making is the arrangement of components in a neat, compact way on a copper clad board with circuit connections.

The various components used in PCB assembly have standard dimensions. Based on this suitable spacing is to be provided while drawing a PCB. First the circuit diagram is throughly studied. The size of the different components are noted down. Approximate placing of the different components are taken in preparing a PCB layout. The layout should be a compact, arranged the components neatly, spacing is to be provided sufficiently according to the size of the components so that the leads do not break by bending or the components do not get crowded, and all the connecting lines are drawn in the board.

A rough layout for the circuit is prepared on a paper indicating the placing of the components and lines interconnecting them. This diagram is then converted to required PCB layout with the help of SMART WORK which is a software design producing a printout of the layout. The SMART WORK Commands are SAVE, LOAD, CLEAR, QUIT, DIP and

function keys are F1, F2, F3, F4, F5 and F6. The solder side and component side layout is shown in Fig (5.1, 5.2).

5.2 FABRICATION

The layout of the PCB drawn in graph sheet is transferred to the copper side of the board. This may be done with help of carbon paper. Then the lines and pads on the board are painted with acid resist ink using fine brush. Then the board is put in a solution of ferric chloride in water with little quantity of Hcl and the solution is slightly stirred. This process is called etching. After this only copper lines and pads are available for connections. Then the holes are drilled (1mm) for mounting the components.

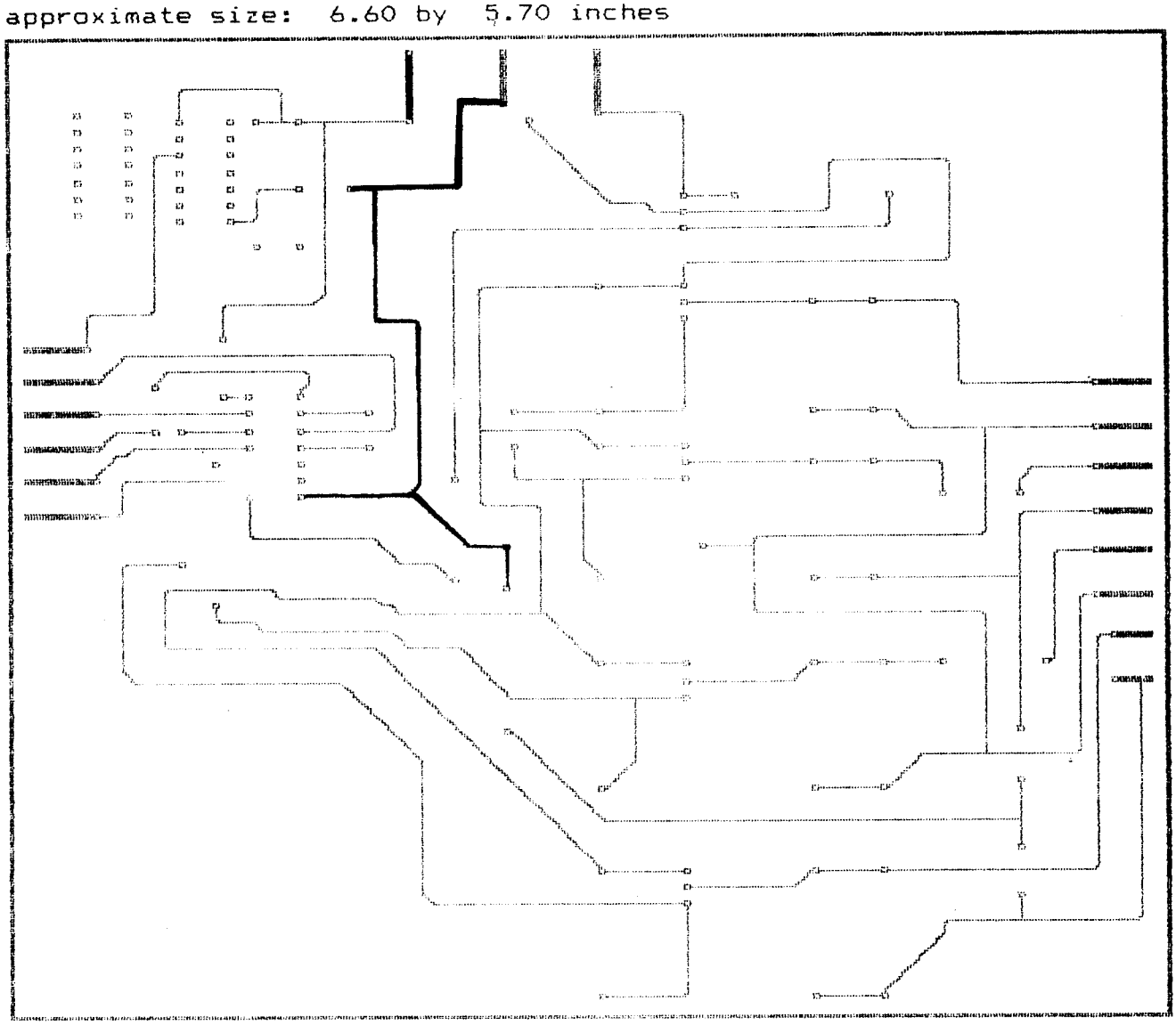
5.3 TESTING

The completed PCB is tested for the continuity wherever required by using multimeter. Shorts between lines are also checked and removed if found. Then assembly of components is done by inserting the leads of the components in to the holes and solder it using lead. Care is to be taken while bending otherwise the leads will break away from the body. Then the assembled PCB is tested and checked whether the required output is obtained.

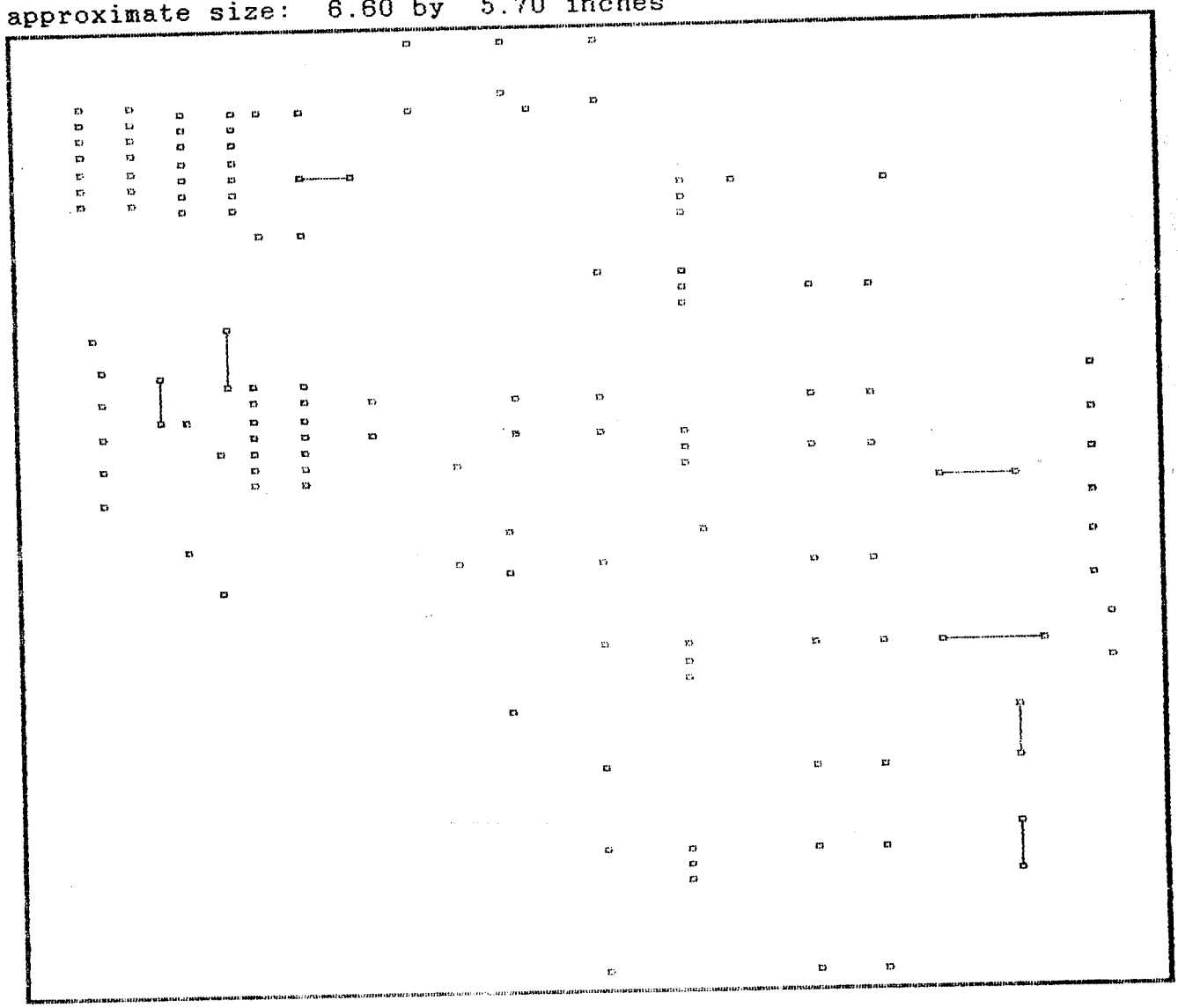
5.4 CABINET DESIGN AND MAKING

An Isometric drawing of the cabinet is drawn first. It should show the front panel and controls (if required) then a development drawing is drawn for the given size. The drawings are shown in Fig (5.3). A cabinet should accomodate easy acessability for the service. Finally the PCB is mounted in the cabinet.

1X checkplot 1 Jan 1987 11:11:10
a:\stepper
v1.2 r2 holes: 136 solder side
approximate size: 6.60 by 5.70 inches



1X checkplot 1 Jan 1987 11:24:14
a:\stepper
v1.2 r2 holes: 136 component side
approximate size: 6.60 by 5.70 inches



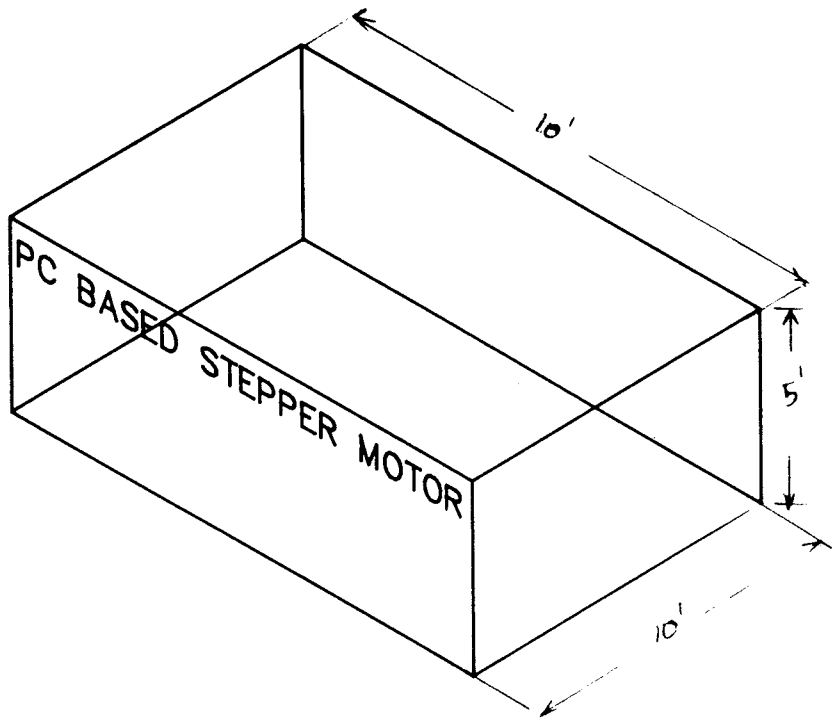


FIG.(5.3) CABINET DESIGN

CHAPTER VI

SOFTWARE AND TESTING

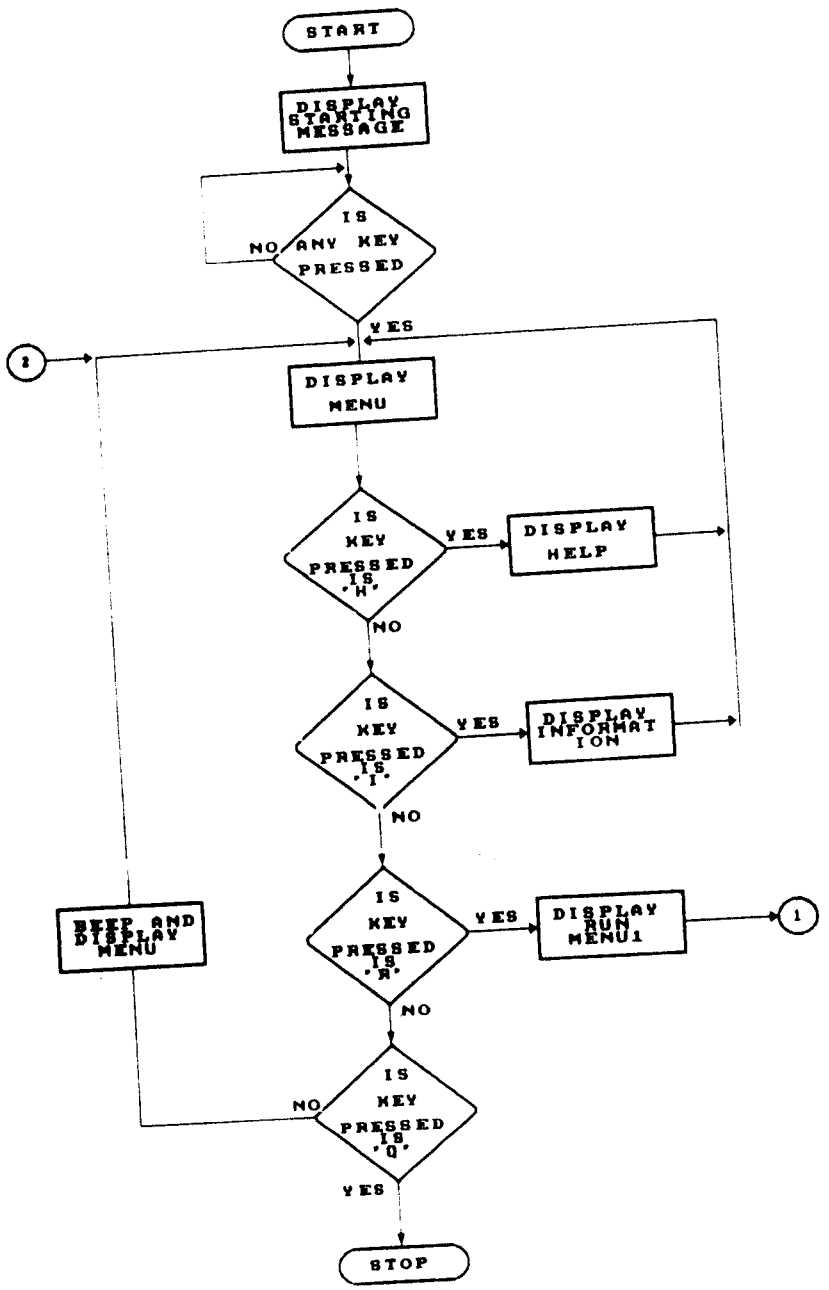
6.1 FLOW CHART

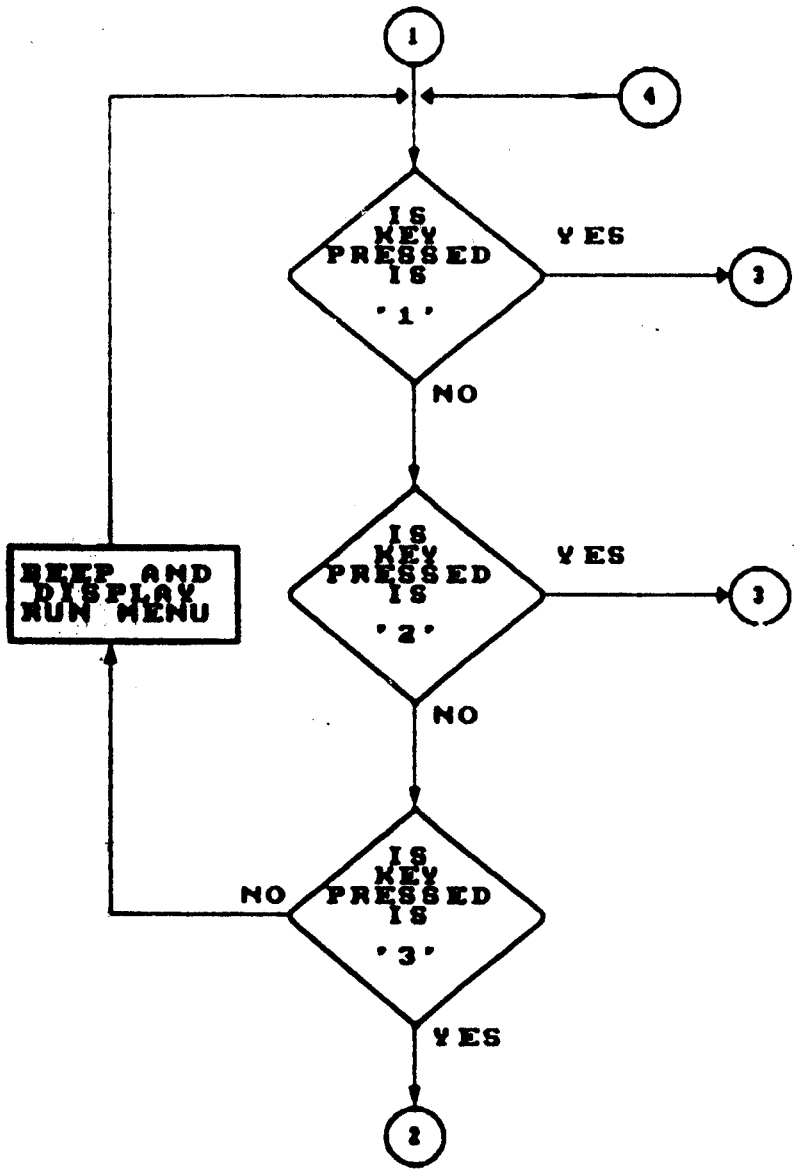
In this chapter, the flow chart and the software program are given. The various steps for speed control of a stepper motor using a personal computer are shown in the flow chart Fig (6.1).

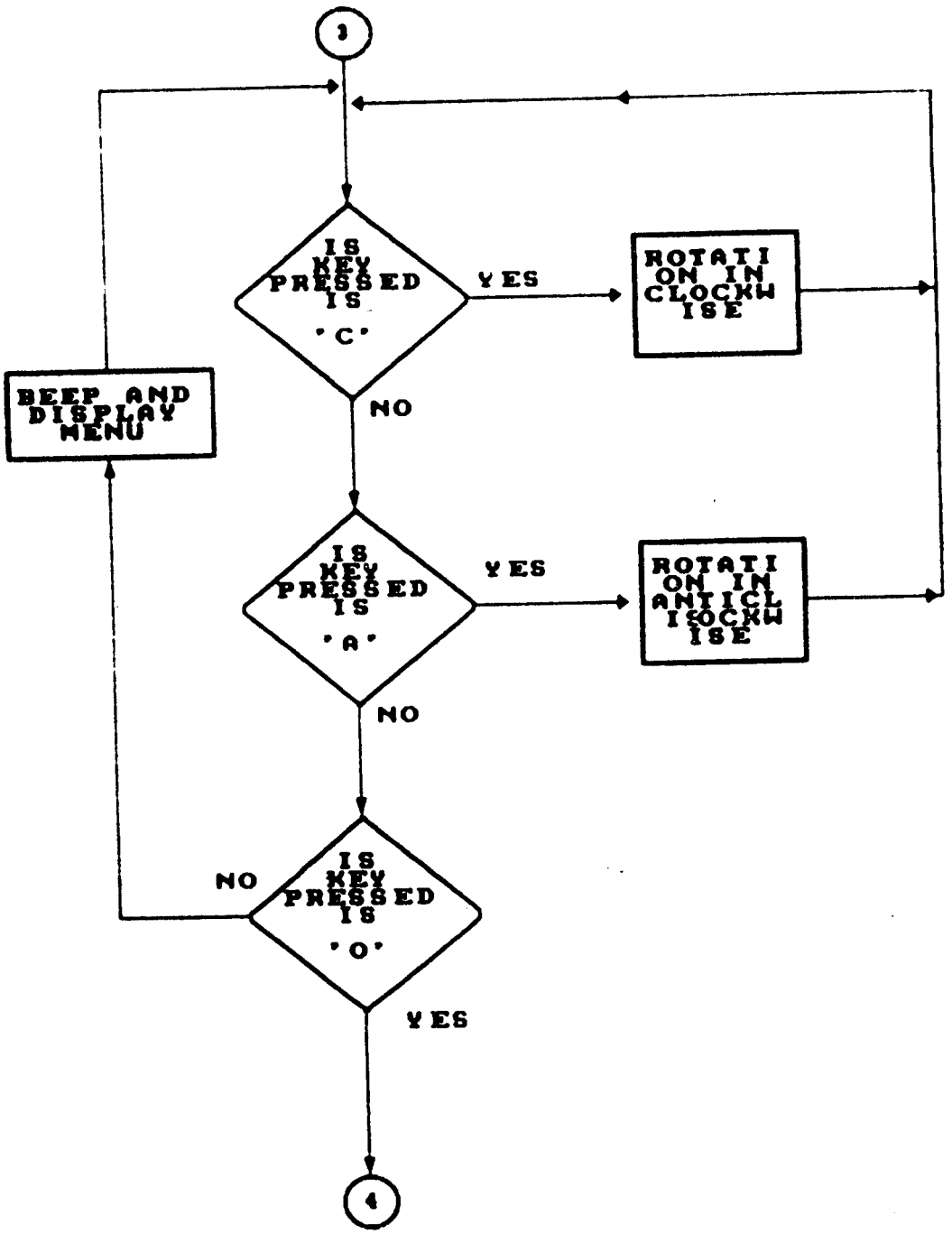
ALGORITHM

1. Initialize the 8255A ports by using the control word 80H
2. Select one of the port.
3. Use another port for home.
4. Give suitable addresses through the port.
5. Repeat the steps for various addresses for the Half step or full step rotation.
6. For anti-clockwise read the address from down to up.

FIG (6.1) FLOW CHART




1
4
IS KEY PRESSED IS '1'
YES
3
NO
IS KEY PRESSED IS '2'
YES
3
NO
BEEP AND DISPLAY MAIN MENU
IS KEY PRESSED IS '3'
NO
YES
2


3
IS KEY PRESSED IS 'C'
YES
ROTATION IN CLOCKWISE
NO
BEEP AND DISPLAY MENU
IS KEY PRESSED IS 'A'
YES
ROTATION IN ANTICLOCKWISE
NO
IS KEY PRESSED IS 'O'
NO
YES
4

6.2 SOFTWARE

The software used is QUICK BASIC language which has menu for the control of speed of the motor. This menu consists of selection of sequence and direction with Help menu. The input given are speed and Number of Rotation. The function of this program is to communicate both with the user and the PC through the I/O card. This software also contains a simulated graphic display routine to show the rotation and necessary message. The source code of the program is given in Fig (6.2).

```
'*************************************************
'* SOFTWARE FOR THE SPEED CONTROL OF STEPPER MOTOR *
'*************************************************
DECLARE SUB hccw ()
DECLARE SUB hc ()
DECLARE SUB m1 ()
DECLARE SUB runn ()
DECLARE SUB menu (keychoice$)
DECLARE SUB help1 ()
DECLARE SUB info ()
DECLARE SUB menu1 (keychoice$)
DECLARE SUB menu2 (keychoice$)
DECLARE SUB motorcw ()
DECLARE SUB motorccw ()
DECLARE SUB commh ()
DECLARE SUB delay ()
DECLARE SUB fcw ()
DECLARE SUB fccw ()
DECLARE SUB commf ()

          ***************************
            REM   MAIN    PROGRAM
          ***************************
 DO
 CALL m1
 CALL menu(keychoice$)
 SELECT CASE keychoice$
 CASE "H"
 CALL help1
 CASE "I"
 CALL info
 CASE "R"
 CALL runn
 CASE "Q"
 EXIT DO
 END SELECT

 LOCATE 25, 5
 COLOR 2
 PRINT "press any key return to    MENU     "
 waitkey$ = INPUT$(1)
 LOOP
END


SUB commf STATIC
CLS
SCREEN 1
DO

CALL menu2(keychoice$)
SELECT CASE keychoice$
CASE "C"
LOCATE 2, 2
PRINT "enter number of revolution N=";
INPUT N
LOCATE 4, 2
PRINT "enter the speed (delay ) S=";
INPUT S
```

```
CALL motorcw
CALL fcw
NEXT rot
CASE "A"
LOCATE 2, 2
PRINT "enter number of revolution N=";
INPUT N
LOCATE 4, 2
PRINT " enter the speed (delay) S=";
INPUT S
CLS
FOR rot = 1 TO N
CALL motorccw
CALL fccw
NEXT rot
CASE "O"
EXIT DO
END SELECT
LOOP
END SUB

SUB commh STATIC
DO
CALL menu2(keychoice$)
SELECT CASE keychoice$
CASE "C"
LOCATE 2, 2
PRINT "enter number of revolution N=";
INPUT N
LOCATE 4, 2
PRINT "enter speed (delay) S=";
INPUT S
CLS
FOR rot = 1 TO N
CALL motorcw
CALL hc
NEXT rot
CASE "A"
LOCATE 2, 2
PRINT "enter number of revolution N=";
INPUT N
LOCATE 4, 2
PRINT "enter speed (delay) S=";
INPUT S
CLS
FOR rot = 1 TO N
CALL motorccw
CALL hccw
NEXT rot
CASE "O"
EXIT DO
END SELECT
LOOP
END SUB

SUB delay STATIC
FOR i = 1 TO S
NEXT i
```

```
END SUB

SUB fccw STATIC
OUT &H380, 22
CALL delay
OUT &H380, 66
CALL delay
OUT &H380, 44
CALL delay
OUT &H380, cc
CALL delay
OUT &H380, 88
CALL delay
OUT &H380, 99
CALL delay
OUT &H380, 11
CALL delay
OUT &H380, 33
CALL delay
END SUB

SUB fcw STATIC
OUT &H380, 33
CALL delay
OUT &H380, 11
CALL delay
OUT &H380, 99
CALL delay
OUT &H380, 88
CALL delay
OUT &H380, cc
CALL delay
OUT &H380, 44
CALL delay
OUT &H380, 66
CALL delay
OUT &H380, 22
CALL delay
END SUB

SUB hc STATIC
CLS
OUT &H380, 33
CALL delay
OUT &H380, 99
CALL delay
OUT &H380, cc
CALL delay
OUT &H380, 66
CALL delay
END SUB

SUB hccw STATIC
CLS
OUT &H380, 66
CALL delay
OUT &H380, cc
CALL delay
OUT &H380, 99
CALL delay
```

```
OUT &H380, 33
CALL delay
END SUB

SUB help1 STATIC
CLS
SCREEN 0
LOCATE 3, 25
PRINT "H E L P   M E N U"
LOCATE 4, 25
PRINT "****************"
LOCATE 8, 5
PRINT "Check the connections of the circuits and give supply to the motor.
LOCATE 11, 5
PRINT "Select the proper letters from the MENU "
LOCATE 14, 5
PRINT "Choose proper step of rotation from the menu"
LOCATE 17, 5
PRINT "choose proper direction of running of the motor from sub menu"
LOCATE 20, 5
PRINT "Give  the number of rotation and delay "
LOCATE 23, 5
PRINT "Run the program by pressing F5"
END SUB

SUB info STATIC
CLS
SCREEN 0
LOCATE 5, 15: PRINT "  PC BASED CONTROLLER FOR STEPPER MOTOR "
LOCATE 5.5, 15
PRINT " ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^"
LOCATE 9, 20
PRINT " PROJECT   WORK    DONE   BY "
LOCATE 11.5, 25
PRINT "  A.LAKSHMANAN   "
LOCATE 13.5, 25
PRINT "  SUMESH  CHANDRAN  "
LOCATE 15.5, 25
PRINT "  N.VIJAYAKUMAR   "
LOCATE 17.5, 25
PRINT "  D.VISVANATHAN "
LOCATE 23, 17
PRINT " GUIDED    BY    Dr.K.A.PALANISWAMY Ph.D   "
LOCATE 24, 17
PRINT "****************************************"
END SUB

SUB m1 STATIC

SCREEN 1
COLOR 2, 0, 12

LINE (0, 0)-(319, 0)
LINE (319, 0)-(319, 190)
LINE (319, 190)-(0, 190)
LINE (0, 190)-(0, 0)

LINE (5, 5)-(314, 5)
LINE (314, 5)-(314, 185)
LINE (314, 185)-(5, 185)
```

```
LINE (5, 185)-(5, 5)

LINE (80, 60)-(230, 60)
LINE (230, 60)-(230, 130)
 LINE (230, 130)-(80, 130)
LINE (80, 130)-(80, 60)

LINE (115, 19)-(193, 19)
LINE (193, 19)-(193, 35)
LINE (193, 35)-(115, 35)
LINE (115, 35)-(115, 19)

COLOR 4
LOCATE 3.5, 15.5
PRINT " M E N U "
END SUB

SUB menu (keychoice$) STATIC

SCREEN 1
COLOR 2, 0, 12

COLOR 9: LOCATE 9, 13: PRINT "("; : COLOR 4: PRINT "H";
COLOR 9: PRINT ")elp"

LOCATE 14.6, 13: COLOR 9: PRINT "("; : COLOR 4: PRINT "Q";
COLOR 9: PRINT ")uit"

COLOR 9: LOCATE 10.6, 13: PRINT "("; : COLOR 4: PRINT "I";
COLOR 9: PRINT ")nformation"

COLOR 9: LOCATE 12.6, 13: PRINT "("; : COLOR 4: PRINT "R";
COLOR 9: PRINT ")unning"

COLOR 25, 2
LOCATE 25, 5.4: PRINT CHR$(206);
COLOR 10: PRINT "  PRESS A LETTER (H,I,R or Q) ";
COLOR 27: PRINT CHR$(206);

COLOR 7, 0
DO
BEEP: keychoice$ = UCASE$(INPUT$(1))
LOOP WHILE INSTR("HIRQ", keychoice$) = 0
END SUB

SUB menu1 (keychoice$) STATIC

CLS
SCREEN 0
COLOR 4, 0, 12

COLOR 9: LOCATE 10, 30: PRINT "("; : COLOR 4: PRINT "1";
COLOR 9: PRINT ") HALF STEP ROTATION"

LOCATE 13, 30: COLOR 9: PRINT "("; : COLOR 4: PRINT "2";
COLOR 9: PRINT ") FULL STEP ROTATION"

LOCATE 16, 30: COLOR 9: PRINT "("; : COLOR 4: PRINT "3";
COLOR 9: PRINT ") QUIT"

COLOR 25, 2
COLOR 12, 27: PRINT CHR$(1);
```

```
PRINT "  YOUR  CHOICE (1,2 OR 3 )  ";
COLOR 27: PRINT CHR$(1);

COLOR 7, 0
DO
BEEP: keychoice$ = UCASE$(INPUT$(1))
LOOP WHILE INSTR("123", keychoice$) = 0
END SUB

SUB menu2 (keychoice$) STATIC

CLS
SCREEN 0
COLOR 4, ( , 12

COLOR 9: LOCATE 10, 30: PRINT "("; : COLOR 4: PRINT "C";
COLOR 9: PRINT ")lockwise "

LOCATE 12, 30: PRINT "("; : COLOR 4: PRINT "A";
COLOR 9: PRINT ")nticlockwise"

COLOR 4: LOCATE 14, 30: PRINT "("; : COLOR 4: PRINT "O";
COLOR 9: PRINT ")ut"

COLOR 27, 1
LOCATE 21, 24: PRINT CHR$(2);
COLOR 10: PRINT " CHOOSE YOURS LETTER  ( C, A, OR O )  ";
COLOR 27: PRINT CHR$(2);


DO
BEEP: keychoice$ = UCASE$(INPUT$(1))
LOOP WHILE INSTR("CAO", keychoice$) = 0
END SUB

SUB motorccw STATIC
CLS
KEY OFF
SCREEN 1
CIRCLE (150, 100), 55
PAINT (150, 100), 2, 0
p$ = " bm 150,100;u40"
FOR i = 1 TO 360
DRAW " bm 150,100; c0 u40"
DRAW "c1 ta=" + VARPTR$(i) + p$
NEXT i
END SUB

SUB motorcw STATIC
CLS
SCREEN 1
KEY OFF
CIRCLE (150, 100), 55
PAINT (150, 100), 2, 0
p$ = "bm 150,100;u40"
FOR i = 360 TO 1 STEP -1
DRAW "bm 150,100;c0u40"
DRAW "c1ta=" + VARPTR$(i) + p$
NEXT i
END SUB
```

```
SUB runn STATIC
CLS
DO
CALL menu1(keychoice$)
SELECT CASE keychoice$
CASE "1"
CALL commh
CLS
CASE "2"
CALL commf
CLS
CASE "3"
EXIT DO
END SELECT
LOOP
CLS
END SUB
```

6.3 TESTING

The whole circuit is assembled and the Digital I/O card is placed in any one of the slots of the computer and the software is executed. The outputs are checked at the various points in the circuit and the port used is A and its address is 0387H. Then the inputs are given as speed and rotaion and output is checked. The number of steps of rotation are found to be same as given in the program.

CHAPTER VII

CONCLUSION

A software has been developed in QUICK BASIC for controlling the speed of a stepper motor using a PC. The software has been tested on a 12 V, 4 phase, 10 Kgm stepper motor using a PC in the Department of Computer Science and Engineering at the college. It has been found to contol the speed successfully. The above project can be used in any off process control by including more number of motors. The digital I/O card is capable of accessing 12 motors at a time and it can control sequentially as a Robot. In robot application, the pulses needed no work the stepper motor are produced by a computer. A controller is needed to generate the correct pattern of signals at the correct level of power needed to operate the coils within the motor. As far as the computer is concerned, sending 1 pulse to the controller causes the motor to move one step. The direction of rotation is controlled by a GATE. As long as the computer maintains a voltage at the GATE, the motor will rotate in a clockwise direction. The limitation is that this software is not suitable for high invertion motors.

REFERENCES

1. GAONKAR, " Microprocessor Architecture, programming and applications with the 8085/8080 A", Wiley Eastern Limited, New Delhi, 1992.
2. YU-CHENG LIU AND GLENN A. GIBSON, " Microcomputer systems: The 8086/8088 family Architecture, programming and Design", Prentice-Hall of India, New Delhi, 1988.
3. DOUGLAS V. HALL, " Microprocessors and Interfacing Programming and Hardware", McGraw Hill International Editions, New York, 1988.
4. VINCENT DEL TORO, " Electric Machines and power systems", Prentice-Hall of India, New Delhi, 1988.
5. CHARLES A. SCHULER AND WILLIAM L. MCNAMEE, "Industrial Electronics and Robotics", McGraw Hill International Editions, New York, 1988.
6. STEVEN AND NAMER OFF, " QUICK BASIC the complete Reference", McGraw Hill International Editions, New York, 1989.
7. Journal of "The Institution of Engineers (India)" Volume 72, October 1991.

Logic Products

7404, LS04, S04
Inverters

·Hex Inverter
Product Specification

TYPE	TYPICAL PROPAGATION DELAY	TYPICAL SUPPLY CURRENT (TOTAL)
7404	10ns	12mA
74LS04	9.5ns	2.4mA
74S04	3ns	22mA

ORDERING CODE

PACKAGES	COMMERCIAL RANGE V_{CC} = 5V ±5%; T_A = 0°C to +70°C
Plastic DIP	N7404N, N74LS04N, N74S04N
Plastic SO	N74LS04D, N74S04D

NOTE:
For information regarding devices processed to Military Specifications, see the Signetics Military Products Data Manual.

FUNCTION TABLE

INPUT	OUTPUT
A	Y
L	H
H	L

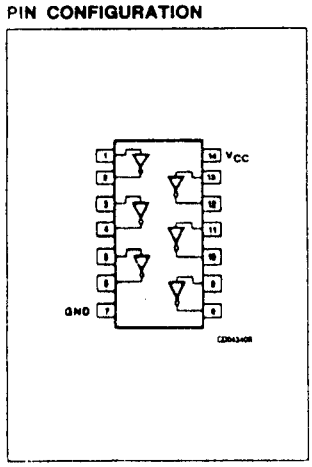
H = HIGH voltage level
L = LOW voltage level

INPUT AND OUTPUT LOADING AND FAN-OUT TABLE

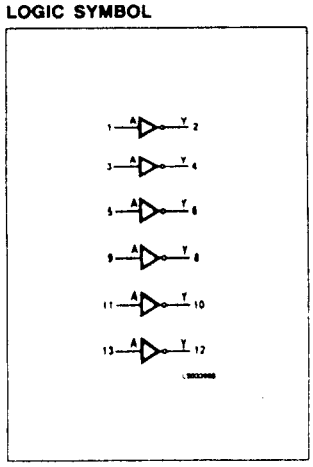
PINS	DESCRIPTION	74	74S	74LS
A	Input	1ul	1Sul	1LSul
Y	Output	10ul	10Sul	10LSul

NOTE:
Where a 74 unit load (ul) is understood to be 40μA I_{IH} and −1.6mA I_{IL}, a 74S unit load (Sul) is 50μA I_{IH} and −2.0mA I_{IL}, and 74LS unit load (LSul) is 20μA I_{IH} and −0.4mA I_{IL}.

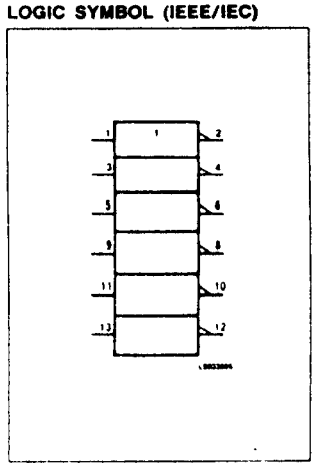
PIN CONFIGURATION



LOGIC SYMBOL



LOGIC SYMBOL (IEEE/IEC)



7406, 07
Inverter/Buffer/Drivers

'06 Hex Inverter Buffer/Driver (Open Collector)
'07 Hex Buffer/Driver (Open Collector)
Product Specification

TYPE	TYPICAL PROPAGATION DELAY	TYPICAL SUPPLY CURRENT (TOTAL)
7406	10ns (t_{PLH}) 15ns (t_{PHL})	31mA
7407	6ns (t_{PLH}) 20ns (t_{PHL})	25mA

ORDERING CODE

PACKAGES	COMMERCIAL RANGE V_{CC} = 5V ±5%; T_A = 0°C to +70°C
Plastic DIP	N7406N, N7407N
Plastic SO	N7406D, N7407D

NOTE:
For information regarding devices processed to Military Specifications, see the Signetics Military Products Data Manual.

FUNCTION TABLE

'06		'07	
INPUT	OUTPUT	INPUT	OUTPUT
A	Y	A	Y
H	L	H	H
L	H	L	L

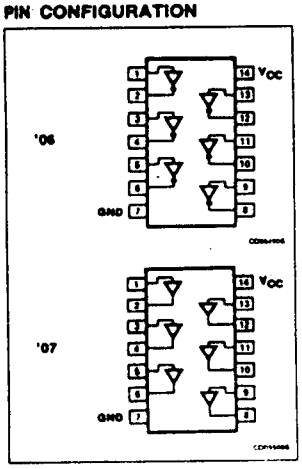
H = HIGH voltage level
L = LOW voltage level

INPUT AND OUTPUT LOADING AND FAN-OUT TABLE

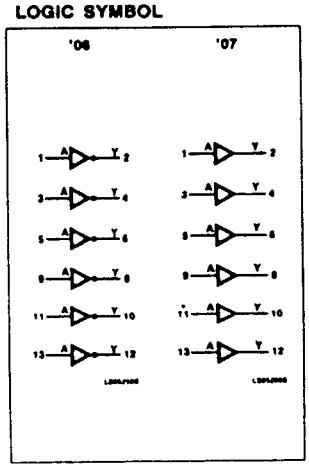
PINS	DESCRIPTION	74
A	Input	1ul
Y	Output	10ul

NOTE:
Where a 74 unit load (ul) is understood to be 40μA I_{IH} and −1.6mA I_{IL}.

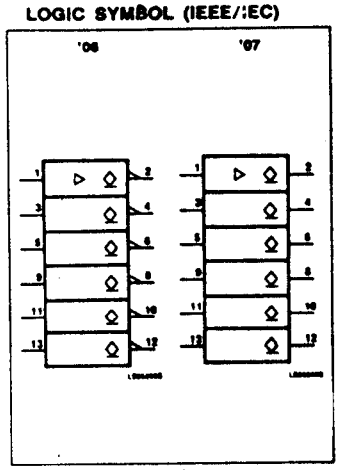
PIN CONFIGURATION



LOGIC SYMBOL



LOGIC SYMBOL (IEEE/IEC)



7430, LS30
Gates

Eight-Input NAND Gate
Product Specification

Logic Products

TYPE	TYPICAL PROPAGATION DELAY	TYPICAL SUPPLY CURRENT (TOTAL)
7430	11ns	2mA
74LS30	11ns	0.5mA

ORDERING CODE

PACKAGES	COMMERCIAL RANGE V_{CC} = 5V ±5%; T_A = 0°C to +70°C
Plastic DIP	N7430N, N74LS30N
Plastic SO	N74LS30D

NOTE:
For information regarding devices processed to Military Specifications, see the Signetics Military Products Data Manual.

FUNCTION TABLE

INPUTS								OUTPUT
A	B	C	D	E	F	G	H	Y
L	X	X	X	X	X	X	X	H
X	L	X	X	X	X	X	X	H
X	X	L	X	X	X	X	X	H
X	X	X	L	X	X	X	X	H
X	X	X	X	L	X	X	X	H
X	X	X	X	X	L	X	X	H
X	X	X	X	X	X	L	X	H
X	X	X	X	X	X	X	L	H
H	H	H	H	H	H	H	H	L

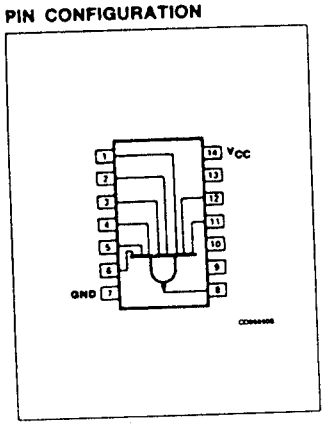
H = HIGH voltage level
L = LOW voltage level
X = Don't care

INPUT AND OUTPUT LOADING AND FAN-OUT TABLE

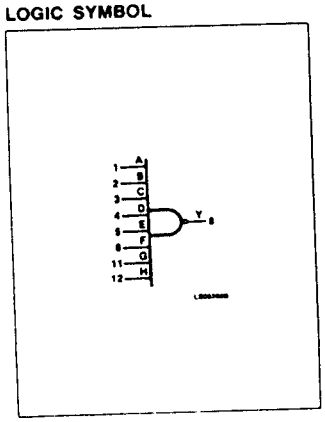
PINS	DESCRIPTION	74	74LS
A – H	Inputs	1ul	1LSul
Y	Output	10ul	10LSul

NOTE:
Where a 74 unit load (ul) is understood to be 40μA I_{IH} and −1.6mA I_{IL}, and a 74LS unit load (LSul) is 20μA I_{IH} and −0.4mA I_{IL}.

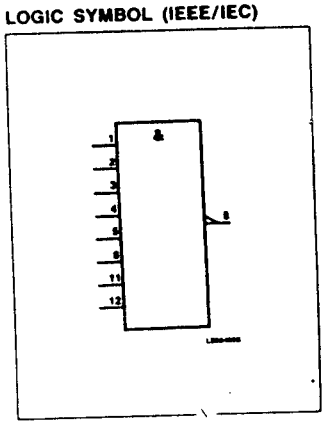
PIN CONFIGURATION



LOGIC SYMBOL



LOGIC SYMBOL (IEEE/IEC)



ogic Products

7432, LS32, S32
Gates

Quad Two-Input OR Gate
Product Specification

TYPE	TYPICAL PROPAGATION DELAY	TYPICAL SUPPLY CURRENT (TOTAL)
7432	12ns	19mA
74LS32	14ns	4.0mA
74S32	4ns	28mA

ORDERING CODE

PACKAGES	COMMERCIAL RANGE V_{CC} = 5V ±5%; T_A = 0°C to +70°C
Plastic DIP	N7432N, N74LS32N, N74S32N
Plastic SO-14	N74LS32D, N74S32D

NOTE:
For information regarding devices processed to Military Specifications, see the Signetics Military Products Data Manual.

UNCTION TABLE

INPUTS		OUTPUT
A	B	Y
L	L	L
L	H	H
H	L	H
H	H	H

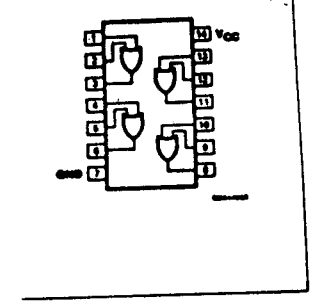
= HIGH voltage level
= LOW voltage level

INPUT AND OUTPUT LOADING AND FAN-OUT TABLE

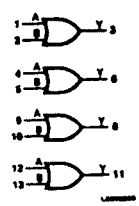
PINS	DESCRIPTION	74	74S	74LS
A, B	Inputs	1ul	1Sul	1LSul
Y	Output	10ul	10Sul	10LSul

NOTE:
Where a 74 unit load (ul) is understood to be 40µA I_{IH} and −1.6mA I_{IL}, and a 74S unit load (Sul) is 50µA I_{IH} and −2.0mA I_{IL}, and a 74LS unit load (LSul) is 20µA I_{IH} and −0.4mA I_{IL}.

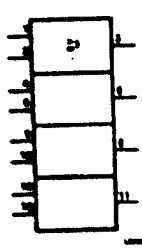
IN CONFIGURATION



LOGIC SYMBOL



LOGIC SYMBOL (IEEE/IEC)



74LS245
Transceiver

Octal Transceiver (3-State)
Product Specification

›gic Products

EATURES

- Octal bidirectional bus interface
- 3-State buffer outputs
- PNP inputs for reduced loading
- Hysteresis on all Data inputs

DESCRIPTION

The 'LS245 is an octal transceiver featuring non-inverting 3-State bus compatible outputs in both send and receive directions. The outputs are all capable of sinking 24mA and sourcing up to 15mA, producing very good capacitive drive characteristics. The device features a Chip Enable (CE) input for easy cascading and a Send/Receive (S/R) input for direction control. All data inputs have hysteresis built in to minimize AC noise effects.

TYPE	TYPICAL PROPAGATION DELAY	TYPICAL SUPPLY CURRENT (TOTAL)
74LS245	8ns	58mA

ORDERING CODE

PACKAGES	COMMERCIAL RANGE V_{CC} = 5V ±5%; T_A = 0°C to +70°C
Plastic DIP	N74LS245N
Plastic SOL-20	N74LS245D

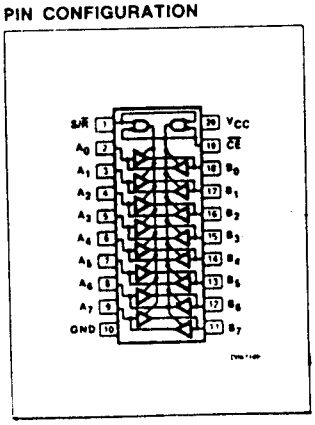
NOTE:
For information regarding devices processed to Military Specifications, see the Signetics Military Products Data Manual.

INPUT AND OUTPUT LOADING AND FAN-OUT TABLE

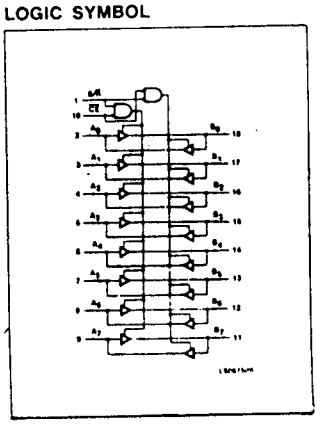
PINS	DESCRIPTION	74LS
All	Inputs	1LSul
All	Outputs	30LSul

NOTE:
Where a 74LS unit load (LSul) is 20µA I_{IH} and −0.4mA I_{IL}

PIN CONFIGURATION



LOGIC SYMBOL



LOGIC SYMBOL (IEEE/IEC)

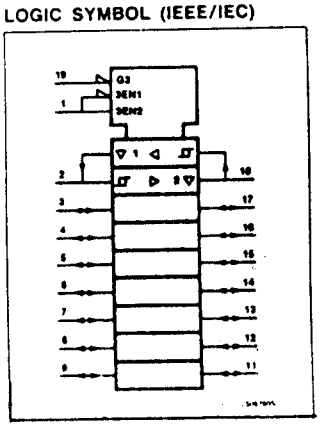
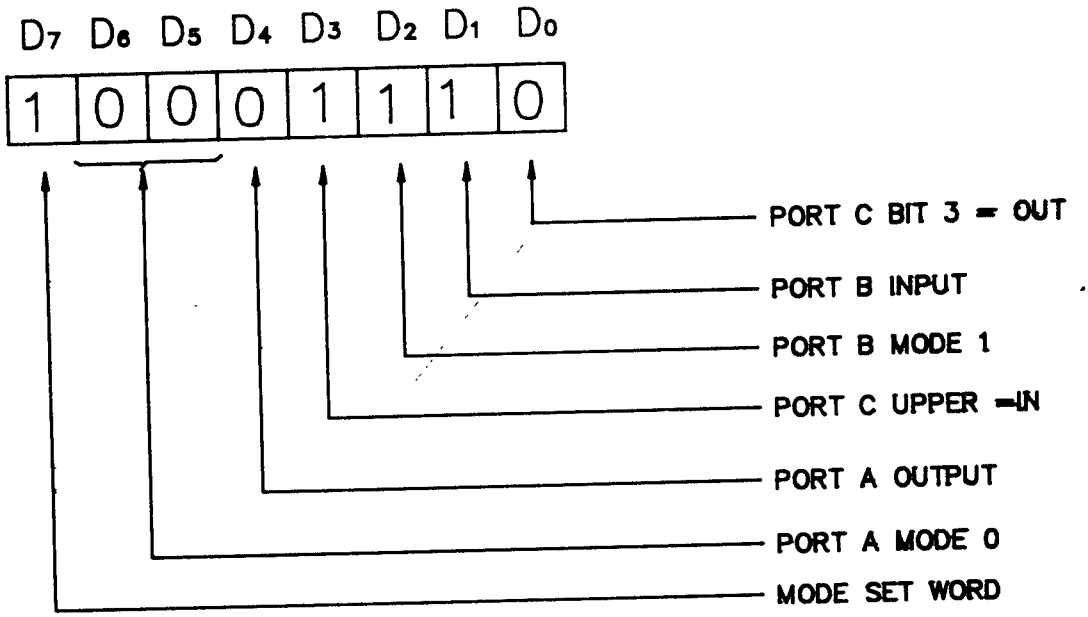
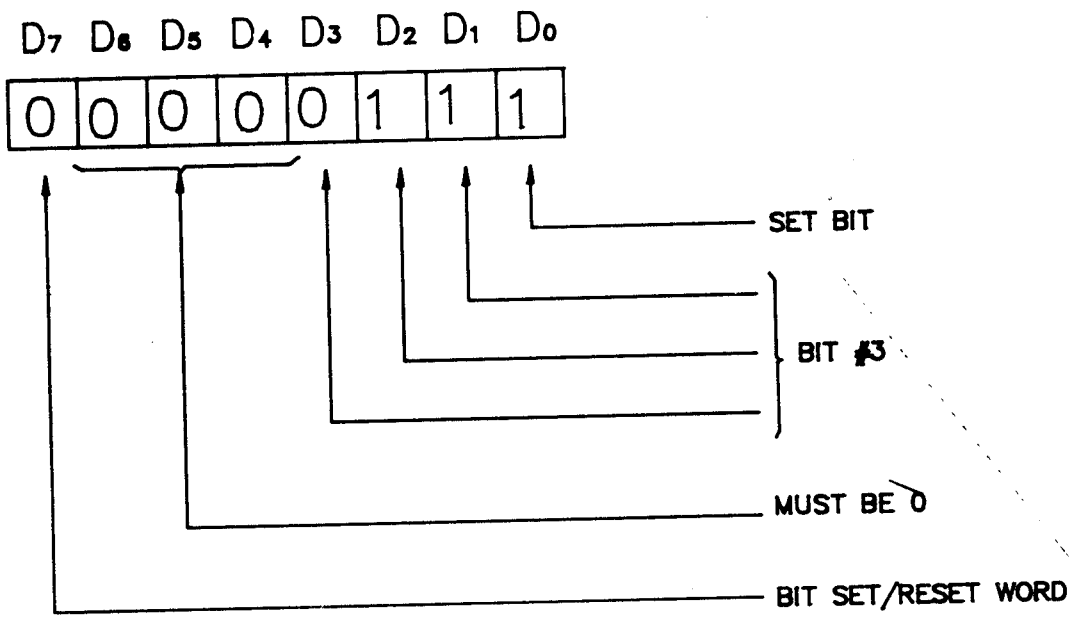


FIG.( ) CONTROL WORD EXAMPLES FOR 8255A.



(a) MODE SET CONTROL WORD



(b) PORT C BIT SET/RESET CONTROL WORD TO SET BIT 3.