DEVELOPMENT AND STUDY OF A MICROPROCESSOR – BASED DIRECT CURRENT MOTOR CONTROLLER

BY

MWENDA PHYLIS MAKENA

A Thesis submitted in partial fulfillment for the requirements of the degree of Master of Science in Physics (Electronics) of Kenyatta University

OCTOBER 2007

DECLARATION

This Thesis is my original work and has not been presented for the award of a degree at any other University.

Mwenda M. Phylis
(156/7335/01)
Department of Physics
Kenyatta University
P.O Box 43844
Nairobi, Kenya

Signature _____ Date 17/12/2007

This Thesis has been submitted for examination with our approval as University Supervisors.

Dr. G.A. Ibitola
Department of Physics
Kenyatta University
P.O Box 43844
Nairobi, Kenya

Signature _____ Date 20/12/2007

Dr. A.S. Merenga
Department of Physics
Kenyatta University
P.O Box 43844
Nairobi, Kenya

Signature _____ Date 20/12/07

DEDICATION

This research work is dedicated to my husband Maxwell Mwenda and my son Newton Murithi. Thanks for your patience, support and encouragement during this study.

ACKNOWLEDGEMENTS

Let me take this opportunity to give my sincere gratitude to my supervisors Dr. G.A Ibitola, Dr.A.S.Merenga and Mr. D.K kiptui, for their guidance during the study. I am also grateful to all my lecturers in the physics department especially Dr. Merenga and Prof. Okumu for their concern and guidance in building the hardware system. Acknowledgements are made to the entire technical staff of department of physics for assisting me during the assembling of hardware. I would also like to thank the late Mr. Owade , Dr. Karimi and Mr. Musembi for guiding me in the software development of the system.

My colleagues Crucifixa, Emily, Rotich and Mapesa were always there to encourage me. Thanks to my parents, sisters and brothers for the support they have given me throughout the study. Finally, I would like to thank God for his unfailing love and grace during the study.

TABLE OF CONTENTS

Page

ABSTRACT

Control systems for electric motors have become so vital to the proper performance and protection of modern plant equipment that they are frequently the most essential links in complex industrial applications. Electric motors play a very important part in furnishing power for all types of domestic and industrial applications. Understanding the direct current drive is particularly important because it is so widely used as a yardstick by which other drives are measured. Users who develop a good grasp of the direct current drive will find their know-how invaluable in dealing with all other types, particularly if they can establish a firm grip on the philosophy of the control scheme. Analogue and digital motor control systems already exist. These control systems have inherent weaknesses such as lack of system's flexibility, poor responses, large power losses, several switches and relays, large complexity of wiring, and a large number of circuitry.
This thesis presents a control system based on a 8088 microprocessor for starting, stopping and changing the direction of DC motors. The hardware consists of the micro-controller and the switching circuit connected to the motor. The microprocessor with the help of a clock produces pulses which are output through the port B of the Programmable Peripheral Interface (8255). This signal was fed to the switching circuit through a non inverting buffer. To run a motor both forward and reverse, an H-bridge circuit has been set that can reverse its polarity. The average voltage applied to the motor depends on the amount of time when the switch is ON with respect to the time when the switch is OFF (duty cycle). The control software was developed using assembly language and coded in 8086/8088 assembler. Experimental results were achieved using Light Emitting Diodes (LEDs) and a PM 3384 Autoranging Combiscope. The system possesses some advantages over the analog and digital motor controllers.

CHAPTER 1

INTRODUCTION

1.1 Automated control systems

The purpose of an automatic control on a system is to produce a desired output when inputs to the system are changed. Inputs are in form of commands, which the outputs are expected to follow, and disturbances, which the automatic control is expected to minimize [1]. Control systems have been devised for a wide range of industrial applications. These include: voltage, power, temperature, level, flows, pressure, position and speed controls [2]. Many industrial processes now use electronics to produce and condition the required controlling signal due to the advantages of physical size, reliability and with the rapid introduction of microelectronics low costs.

1.2 Electric motor control systems

Control systems for electric motors have become so vital to the proper performance and protection of modern plant equipment that they are frequently the most essential links in complex industrial applications [3]. These systems may range in extent from the simple practice of merely starting and stopping an electric motor to that of directing the energy flow in a completely automated factory. More often, however, the arrangement may involve one or more of such functions as:

1. Rapid stopping (braking) and reversing of a motor.
2. Speed changing.
3. Travel limits of mechanical equipment, e.g., cranes, hoists and machine tools.
4. Timing of multimotor drives.
5. Regulating of parameters such as current, torque, speed, acceleration and deceleration.

1.2.1 DC Motor speed control

Electric motors are so much part of our every day life that we rarely give them a second thought. For instance, when an electric drill is switched on, it is expected to run rapidly up to the correct speed and we do not question how it knows what speed to run at. In the 1980s, the brushed (conventional) DC motor was the automatic choice where speed or torque control was central and this type still retains a large share of the market despite the growing challenge from the inverter-fed induction motor. Their applications range from steel- rolling mills, industrial drives for robotics, to printer and precision servos among others [4].

There are many varieties of motors but can be basically divided into three main categories: Direct Current (DC), Synchronous and Inductive. This division arises from the fact that if the drive current of a motor uses a direct current, then that motor is called a direct current motor. Under this category, they are divided into five different types [5] as briefly discussed below.

a) DC series motor;

b) Separately excited DC motor;

c) Shunt DC motor;

d) Compound DC motor;

e) Permanent Magnet DC motor.

The DC series motor has its excited wiring and the motor armature connected in series. The motor's special characteristic is its powerful starting torque. The separately excited DC motor has its exciting wiring armature separately connected in series. The motor's special characteristic is that it can rapidly change direction. The shunt DC motor has the exciting wiring and the armature hooked up in a shunt connection. Ordinarily, it is used both in fixed and variable speed functions. The compound DC motor has both separate exit wires and shunt wiring connected to the armature. This motor has the highest starting torque. Permanent Magnet DC motor has the magnetic field produced by a permanent magnet Iron core. Because of this, the curve of the speed/ torque relationship is inversely proportioned.

Each of the above motors must possess salient characteristics in order to meet the demands of control systems applications. These are [6, 7]:

a) Armature Inertia

b) Armature Inductance

c) Motor Peak-Current Capability

d) Linear Current-Torque Relationship

(a) Armature Inertia

For high performance control systems, fast acceleration and deceleration are important; therefore, the motor's contribution to the system's total inertia should be as close to being optimal as possible.

(b) Armature Inductance

A motor's inductance affects its current rise, therefore, in selecting a motor for fast response, it is important to place a maximum limit on the inductance of the armature.

(c) Motor Peak-Current Capability

Depending on the type of magnet used in the motor, the magnet may be magnetized if the applied current exceeds a certain value.

(d) Linear Current-Torque Relationship

Ideally, the torque constant of the motor should not be a function of the armature current. The motor should be so selected that it operates in the linear magnet region.

In controlling DC motors, it is possible to use a variable resistor, auto-transformer and transistors to carry out linear current control but these variables have very large power consumption and part of such power is thermally dissipated. Therefore, DC motors can use a switching type amplifier to act as a control device to implement the power consumption as discussed in this work [8]. The microprocessor that forms the heart of the motor speed controller has been discussed in the following subsections.

1.3 Microprocessors

Microprocessors are complete central processing units (CPUs) packaged as single large-scale integration (LSI) circuit chips. They are able to receive information in digital form, process this information according to a stored program and output information in the form of digital signals [9]. Basic operations of the microprocessor include:

a) Transfer of data to and from memory and I/O sections.

b) Response to external interrupt.

c) Provision of control signal for the entire system.

Various system elements are connected to the microprocessor via a group of wires called buses [10]. These are: the data bus, address bus and control bus.

The microprocessor uses the address bus to specify the desired memory address, the control bus to determine whether reading or writing is to take place and the data bus to send and receive digital signals.

1.3.1 Internal Registers

These are part of the Arithmetic Logic Unit (ALU). They provide the fastest level of data memory available to the system. Typically, the contents of internal registers can be accessed by the ALU in less than 100ns (versus 500ns or more for the main memory). They range from 8 to 64 bits in width depending on the type of the microprocessor.

1.3.2 Clock System

A computer system's clock is measured as a frequency, usually expressed as number of cycles per second. A crystal (oscillator) controls clock speed by use of a silver quartz in a small container. Inside the microprocessor, the clock signals are used to synchronize all the various operations of the control block. By ensuring that all the basic operations are instigated by rising and falling edges of the clock waveforms, the control circuitry can ensure that they occur at the correct time, and in the correct order.

1.4 Microprocessor –based control systems

The invention and wide applications of microprocessors have changed the philosophy of instrumentation designs, signal processing and control engineering fields [11, 12, 13, 14]. This is because microprocessors are cheap, small and consume little power. In addition, in recent years their performance has increased at a greater rate than that of some computers. These factors have led to an explosion in the applications of microprocessors.

Applications include:

1) Data collection, where microprocessors are used to monitor sensors and either record the collected information or communicate the information to some other computers.
2) Control, where microprocessors have largely replaced analog electronics for controlling everything from manufacturing robots to home appliances.

3) Computing, where microprocessors have transformed the concept of computer and made parallel processing possible.

There are many factors which influence the choice of particular type of microprocessor for a specific application. These include: Word size used by processor, Clock frequency, the Architecture and ease with which the processor can be interfaced to external devices.

1.5 8088 Microprocessor

The 8088 microprocessor represents a major breakthrough in Central Processing Unit (CPU) architecture. It combines the powerful resources of a 16-bit microprocessor internal architecture with an easy- to- use 8-bit bus interface [15,16,17,18]. The 16-bit internal architecture provides 16-bit registers and allows the use of 16-bit instructions identical to the ones found in 8086 microprocessor. The 8088 microprocessor is actually an 8086 that has been restricted to 8-bit external transfers.

The 8088 microprocessor executes programs using two separate processing units within the CPU.

a) Bus Interface Unit (BIU). This unit fetches instructions from the memory and passes data to and from the executing hardware to the outside world over the bus interface.

b) Execution Unit (EU). The unit executes the instructions, reads Operands and writes results.

Since the two units operate independently, the Bus interface unit fetches additional instructions while the Execution unit executes a previous instruction. This is made possible by the instruction pipeline (or queue) between the Bus interface unit and Execution unit. The BIU fills this pipeline with instruction awaiting execution, thus whenever the EU finishes executing a given instruction, the next instruction is usually ready for immediate execution without delays caused by instruction fetching.

This microprocessor processes data with the help of a program stored in the EPROM memory unit and outputs it in digital form in order to control the speed of a DC motor. The following parameters shall be considered when using the 8088 microprocessor-based motor speed controller, i.e., duty cycle, accuracy and time constants.

1.6 Computer Programming Languages

Computer software is generally divided into two broad categories, system software and user software. System software is the collection of programs, which are needed in the creation, preparation, and execution of other programs. User software consists of those programs generated by the various users of the system in their attempt to apply the computer to solving their problems.

There are three language levels of programming [19, 20]: Machine language, Assembly language and High level language.

1.6.1 Machine Language

When a program is written in binary 0 or 1 only, the program is said to be in Machine language. In the initial period of development of digital computers, when other more convenient programming languages were not developed, computer programs were written in machine language. The disadvantage of this method is that programs are hand-coded, which is a slow process and leads to errors.

1.6.2 Assembly Language

Here, standard binary codes for computer instructions are substituted by two or more letter codes so that memorizing of these codes is easier. These letter-codes are known as mnemonics meaning "aid to memory" and are usually initials or shortened forms of English language words.

Assembly language statements are usually written in a standard form that has four fields: Label field, Opcode field, Operand field and Comment field, e.g.:

Label field	Opcode field	Operand field	Comment field
START	MOV	A, B	; Move contents of
			; Register B register to
			;Accumulator

An Assembler converts the Assembler language program to make it suitable for computer processing. Using Assembly language allows the representation of all

binary instructions in symbolic form. Because each Assembly-level instruction will normally be translated into one Machine level (binary) instruction, this is the most efficient programming language from the standpoint of machine efficiency. It allows direct manipulation of the registers and bits within the machine. The disadvantage of Assembly language programming is that it is tedious and slow, since the programmer must write instructions for transfer at the registers and internal bus levels.

1.6.3 High-Level Language

Assembly language programming expects the programmer to be conversant with the hardware of the computer. To eliminate this requirement, High-level languages have been developed. These instructions are much closer to English language and are structured so that they naturally correspond to the way programmers think. Programs are written symbolically using variable names and other data structures. However, these programs must be converted into machine executable code.

A compiler translates each High-Level instruction into many binary instructions. Unfortunately translation is automatic, hence many assumptions are made by the computer that are less than ideal, and the resulting object code is not in its optional form. The result is wasted memory space and slower execution of a program. High-Level language, however, provides savings in the programmer's time, that is, it offers fast and efficient programming. Examples are: Basic, C, Pascal and FORTRAN.

1.7 Main Memory Subsystem

There are two types of memories used in a microprocessor system [21,22]. These are:

a) Random Access Memory (RAM)

b) Read Only Memory (ROM).

1.7.1 Random Access Memory (RAM)

This is a read/write memory. It is very volatile and looses memory whenever power disappears. There are two types of RAM memories, Static and Dynamic.

A static RAM stores a bit of information within a flip-flop. It is asynchronous and does not require a clock. A dynamic RAM stores a bit of information as a charge and it uses the gate- substrate capacitance of a MOS transistor as an elementary cell. The advantage of a dynamic RAM is that its elementary cell is smaller than that of a static RAM resulting in a much higher density.

1.7.2 Read Only Memory (ROM)

It is non-volatile thus necessary for long-term storage of data information. The information can be read but no new contents may be written. There are two types of ROM memories:

a) Masked-programmed. A masked-programmed ROM has the information written into it at the time of manufacture.

b) User programmable. These are two types, Programmable ROM and Erasable programmable ROM. Programmable ROM (PROM) is a read only memory that can be programmed directly by the user using a special PROM programmer. Erasable programmable ROM (EPROM) is a read only memory that can be programmed, erased and reprogrammed by the user a number of times.

1.8 Objectives

The general objective of this research is to design and construct a microprocessor-based DC motor controller using pulse width modulation (PWM).

The use of the 8088 microprocessor for control is supposed to achieve the following specific objectives:

1) To switch the motor on in a series of pulses.
2) To develop and implement software for starting, stopping and changing the direction of a DC motor.
3) To control more than one DC motor simultaneously.

1.9 Rationale/ Justification of the Study

The theories and definitions of automatic control have been developed to aid the designer to meet primarily three specifications, namely: Stability, Accuracy and Speed of response. For electric motors, speed designates the number of revolutions of the shaft with respect to time. The speed of rotation of a motor varies considerably depending on the load, which it has to turn.

A motor must be able to accelerate or decelerate (brake) its load to the desired operating speed when required without damage or adverse effect to itself, the load or the power source. Occasionally, it may be desired to reverse the rotation of the motor and at times rapidly and frequently. The motor will perform these functions when properly designed and operated with proper control. A motor can achieve this via the use of a microprocessor as a controller as attempted in this research work. A microprocessor is cheap, has simple architecture and adequate instruction set facilitating easy programming concepts.

The project undertaken is to produce a prototype that demonstrates the effectiveness of a microprocessor- based DC motor controller.

CHAPTER 2

LITERATURE REVIEW

2.1 Motor Speed Controllers

Many pieces of research work have been done in the field of automated control and monitoring [23, 24, 25]. Real world applications often call for controlling small – to medium - sized Direct Current motors from digital circuits. Many modern speed controllers for these motors use a technique called Pulse Width Modulation (PWM) to control the speed of the motor. Using a switching device such as a power transistor or enhancement mode MOSFET, power is switched on or off rapidly before it is supplied to the motor. The natural inductance and resistance of the motor acts as a low pass filter and makes the effective voltage seen by the motor to be the average value of the voltage over time. By varying the duty cycle (one time width) of this switched voltage, the effective average voltage can be lowered (narrow width pulse) or raised (wide pulse). The ratio of ON time to OFF time is what determines the speed of the motor. Another way to express this is having a duty cycle of 10%. There are many Integrated Circuit (IC) chips available that will do this. But these two methods only allow the motor to go in one direction.

Hardistyri R. designed and developed a motor speed controller using an IBM PC parallel port, via Universal Interface (UI), to control two DC motors of any current or voltage rating depending on the rating of the relays. The circuit used

provided two shaft encoders for positional feedback to the computer. There were two separate motor driver circuits and each comprises two transistors and relays. Each motor is controlled by two bits, giving four possible control actions for each motor [26, 27, 28]. With the use of a digital computer as a control element in some real-time control applications, it has been observed that the computer and other circuitry components make the system to be complex and expensive. Owade M. designed and developed a programmable Laboratory Interface system with an illustrative use in resistivity temperature measurement using Intel 8085 microprocessor [29]. A microprocessor-based programmable waveform generator (Karimi P.M) has been used to generate square, sine, triangular and random waveforms. Here, Intel 8088 microprocessor and other support chips have been used [30]. This multifunction generator can be used to control the amplitude of the fundamental output voltage by modulating the width of the pulses, which make up the output waveform of motors.

Thus microprocessors have been hailed as new technology and, as with all new technologies, at the early stages of the introduction; there is inevitably a scarcity of skills and expertise. It is therefore important to realize that skills do already exist in very similar forms in both computing (designers and programmers) and in electronic hardware (designers and engineers) [31,32,33]. Considering the foregoing discussion, speed of the motor has been controlled using either Analog or Digital circuits. These are presented in the following sections and their limitations discussed.

2.2 Digital Motor Speed Controllers

Fig 2.1 illustrates the block diagram of a digital motor speed controller [34,35].

In this system, the angular velocity of a motor is sensed by a variable reluctance tachogenerator and the measured speed in revolutions per minute (r.p.m) displayed on a cathode ray tube (CRT display). The tachogenerator gives an a.c output voltage whose frequency is proportional to motor angular velocity. This signal is input to the frequency to digital converter (Schmitt trigger) followed by 8-bit binary counter, which gives an 8-bit parallel digital output signal. This is connected to the parallel input interface of the microcomputer. The output digital signal to the CRT display must be in serial form and in American Standard Code For Information Interchange (ASCII) code; this is provided by the serial output interface. The parallel output interface provides the STOP COUNT (LSB) and RESET logic control signals for the counter whenever required.

The frequency F Hz of the tachogenerator output voltage is given by:

$$F = \frac{MW_r}{2\Pi} \quad \text{.....(2.1)}$$

Where W_r (radians/sec) is the angular velocity of the wheel and M is the number of teeth. If V is the angular speed in revolutions per minute (r.p.m), then:

$$\frac{V}{60} = \frac{W_r}{2\Pi} \quad \text{.....(2.2)}$$

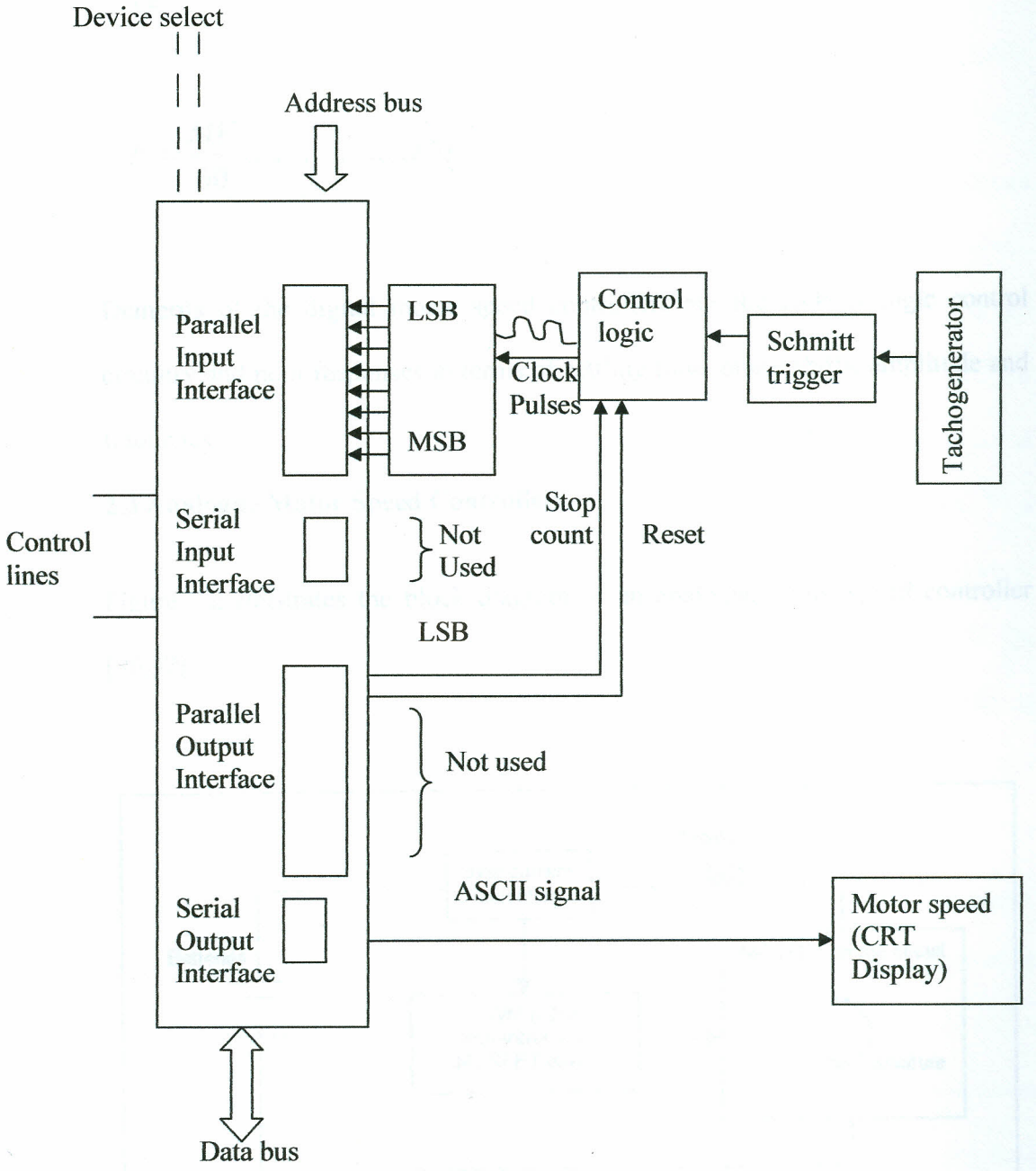


Fig. 2.1: Block diagram of a Digital motor speed controller.

i.e.

$$F = \frac{MV}{60} \text{.....(2.3)}$$

Demerits of the digital motor speed controllers are the lack of logic control circuitry and poor responses in terms of settling time, phase shifts, amplitude and frequency.

2.3 Analogue Motor Speed Controllers

Figure 2.2 illustrates the block diagram of an analogue motor speed controller [36,37].

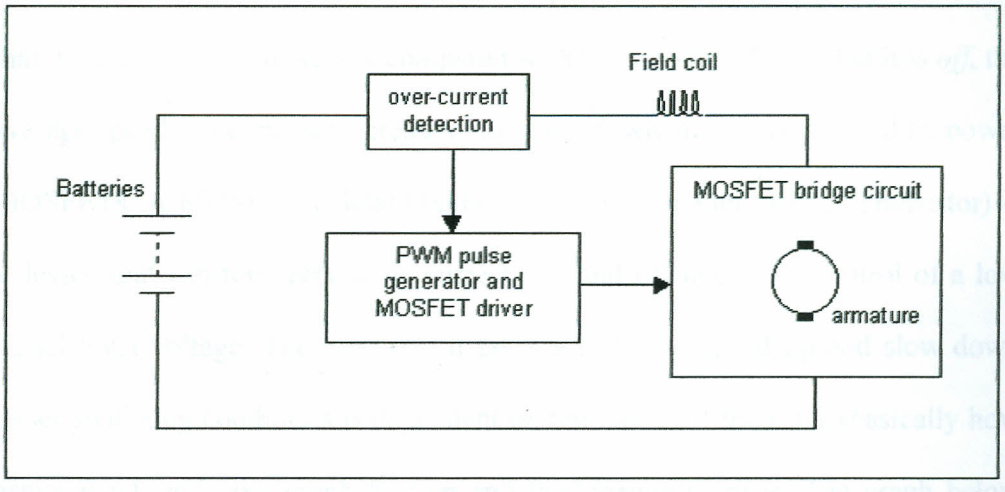


Fig: 2.2 Analogue motor speed controller.

The speed controller works by varying the average voltage sent to the motor. It could do this by simply adjusting the voltage sent to the motor, but this is quite inefficient to do. A better way is to switch the motor's supply on and off very quickly. If the switching is fast enough, the motor doesn't notice it, it only notices the average effect. Imagine a light bulb with a switch. When you close the switch, the bulb goes on and is at full brightness, say 100 Watts. When you open the switch it goes off (0 Watts). Now if you close the switch for a fraction of a second, then open it for the same amount of time, the filament won't have time to cool down and heat up, and you will just get an average glow of 50 Watts. This same principle is used by the speed controller to drive the motor. When the switch is closed, the motor sees 12 Volts, and when it is open it sees 0 Volts. If the switch is open for the same amount of time as it is closed, the motor will see an average of 6 Volts, and will run more slowly accordingly. As the amount of time that the voltage is *on* increases compared with the amount of time that it is *off*, the average speed of the motor increases.This *on-off* switching is performed by power MOSFETs. A MOSFET (Metal-Oxide-Semiconductor Field Effect Transistor) is a device that can turn very large currents on and off under the control of a low signal level voltage. The time that it takes a motor to speed up and slow down under switching conditions is dependent on the inertia of the rotor (basically how heavy it is), and how much friction and load torque there is. The graph below shows the speed of a motor that is being turned on and off fairly slowly.

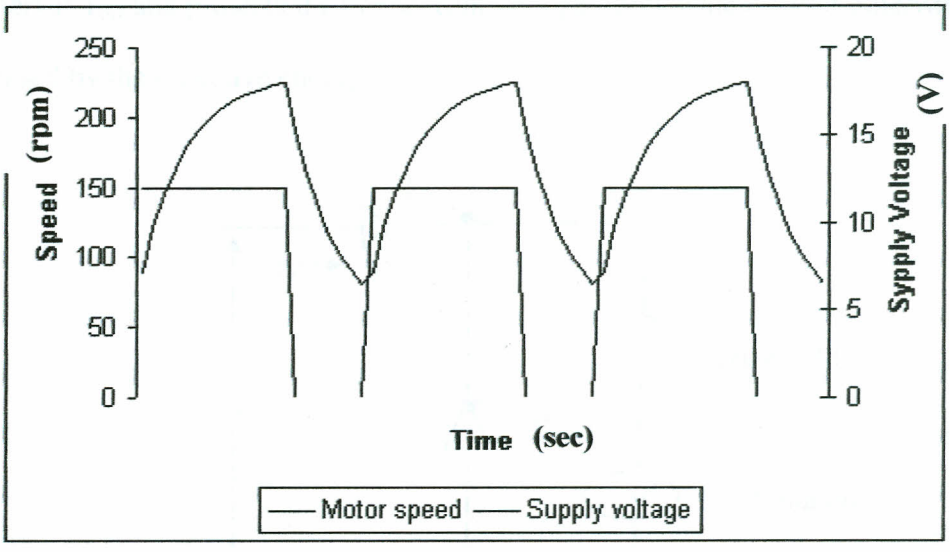


Fig.2.3: Graph showing the relationship between the speed of motor against time.

You can see that the average speed is around 150 rpm, although it varies quite a bit. If the supply voltage is switched fast enough, it won't have time to change speed much, and the speed will be quite steady. This is the principle of switch mode speed control. Thus the speed is set by PWM – Pulse Width Modulation.

2.4 Other Types Of Motor Controllers

2.4.1 Chopper Control of DC Motors

The Chopper circuit is assumed to be working on the principle of time ratio control. During the ON period T_{on}, SCR (T) remains conducting and the supply voltage V is applied to the motor. Motor current I_a flows through the turn ON time of SCR. During the OFF period T_{off}, SCR is blocked and hence no voltage is applied to the motor. Diode D provides the conducting path during the time

interval T_{off} and protects the motor from voltage peaks which would otherwise be caused by the inductive energy.

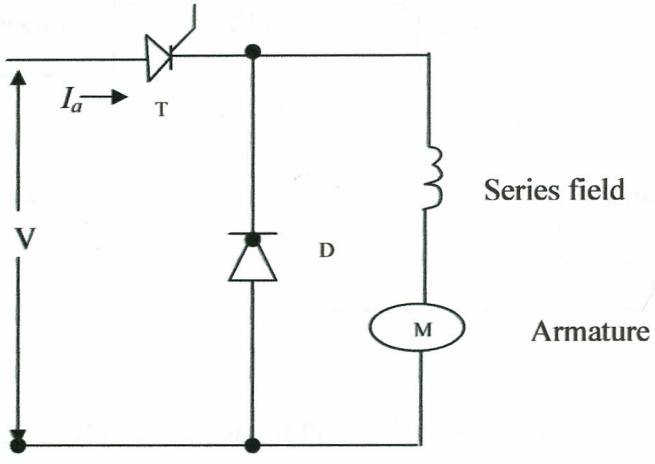


Fig.2.4:Chopper control of DC motor

2.4.2 Phase-Locked-Loop (PLL) DC Control

The tachogenerator (speed encoder) produces a pulse train whose frequency is proportional to the motor speed. This is fed back to the phase detector and compared with frequency of the reference signal. The phase detector provides an output voltage proportional to the difference in phase and frequency of the reference and feedback signal. The high frequency components of this voltage are filtered out by low pass filter. This output voltage is the control for the drive control circuit. When the motor is running at the same speed as set by the reference signal, the shaft speed encoder also provides a signal of the same speed. Then the two signals will be locked together with a phase difference. Hence the output of the phase detector will be a constant voltage which is proportional to this phase difference.

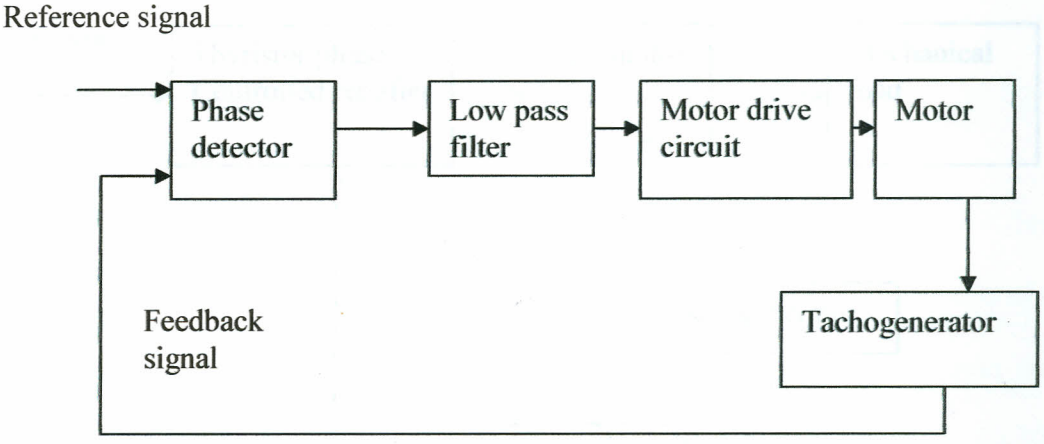


Fig.2.5: Block diagram of Phase -locked –loop (PLL) DC motor control system

2.4.3 Phase control of DC motors

The Thyristor phase- controlled rectifier is widely used to obtain a variable dc voltage from a constant ac voltage and frequency source. A closed- loop system senses the motor output and uses this information to correct the drive to the motor to eliminate error.

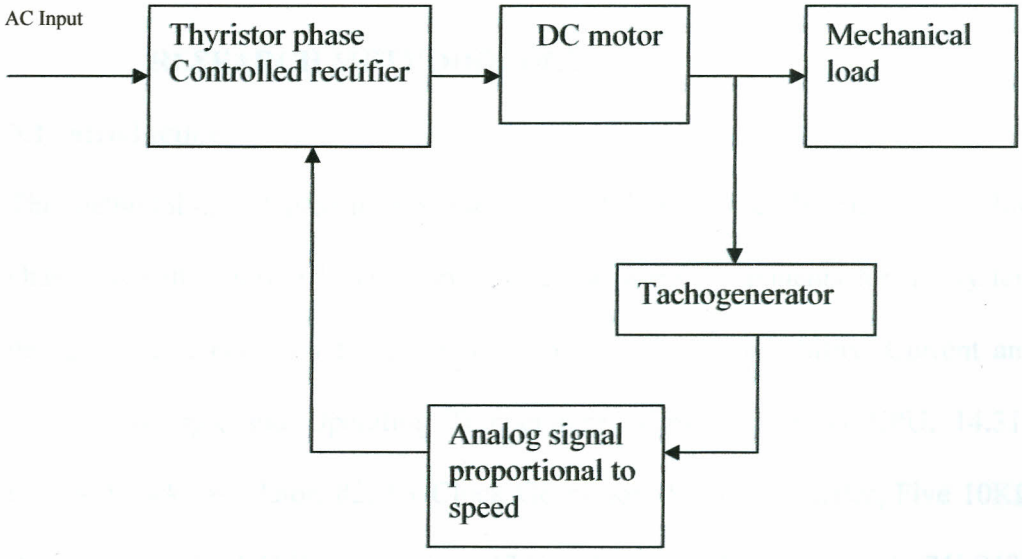


Fig.2.6: Block diagram of closed loop dc control scheme

Analogue motor speed controllers have been observed to possess the following drawbacks: large power losses, large complexity of wiring, several switches, relays and large number of wiring.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Introduction

The methodology adopted in this research work is illustrated by means of a flow chart given in Figure 3.1. The appropriate hardware components for the system design were selected on the basis of Logic Family Compatibility, Current and Voltage Ratings, and Operating Temperature. These are: 8088 CPU, 14.318 Crystal Clock Oscillator, 8284A Clock Generator, 2.2 μF Capacitor, Five 10KΩ Resistors, Six 4.7 KΩ Resistors, Two 47 KΩ Resistors, 74LS373 Latch, 74LS138 Decoder, 74LS367 Buffer, 7404 quad Inverter Chip, 2716 (2KB) EPROM, 6116 (2KB) RAM, 8255A Programmable Peripheral Interface, Two 2N3906 and Four 2N3904 Transistors, Two IRFI9630 and Two IRFZ34 MOSFETs, Reset Switch and 9V DC Motor.

The control system circuit was simulated using Electronics workbench software. The design and implementation of the software involved problem analysis, program design and coding, testing and debugging [38,39]. To integrate the system software with the hardware, the program was burnt into the erasable programmable read only memory.

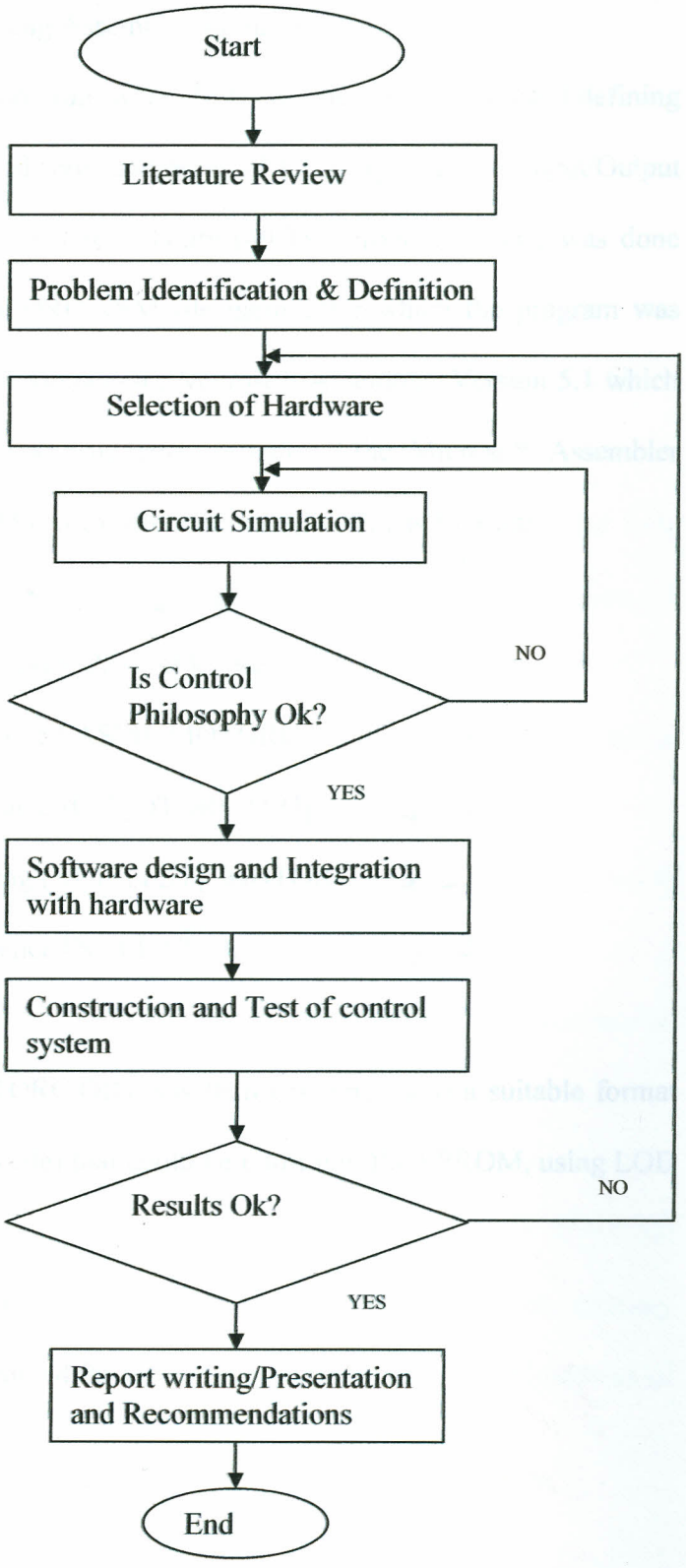


Fig.3.1: Research Methodology Flow Chart.

3.2 Software development

The software was developed using Assembly language.

First, the requirements of the program were clearly defined. These included defining the segment registers to be used, port addresses of the Programmable Input/Output port (PIO) and the memory map nomenclature (40,41). Program coding was done with an editor and given MOTORC.ASM file name after which the program was assembled using 8086/8088-Microprocessor Microsoft Assembler Version 5.1 which was performed as shown below. The correct path of the Microsoft Assembler (MASM) was entered into the Microsoft disk operating system as follows:

C: \ 8086 > MASM ↵

Microsoft ® Macro Assembler Version 5.1

Source filename [.ASM]: MOTORC ↵

Object filename [C: MOTORC.OBJ]: ↵

Source listing [NULL.LST]: MOTORC ↵

Cross reference [NULL.CRF]: ↵

C:\8086>

The assembled program MOTORC.OBJ was then converted into a suitable format (Intel Hex format or Hex ABS file) that could be burnt into the EPROM, using LOD 186 Loader as follows:

C:\ 8086> LOD 186 ↵

Paragon LOD 186 Loader Version 4.0h

Object/ Command File [.OBJ]: MOTORC ←┘

Output Object File [C: MOTORC.ABS]: ←┘

Map Filename [C: NULL.MAP]: ←┘

**LOAD COMPLETE

3.3 Programming the Erasable Programmable Read Only Memory [EPROM 2716]

The MOTORC.ABS File was loaded into the Erasable programmable read only memory [EPROM 2716] with the help of EPROM programmer Leaper Version 5.1 as follows:

1. The pins of the EPROM were checked for proper contact with the socket in the programmer.
2. The EPROM was checked whether it was inserted in the correct orientation in the programmer.
3. The EPROM was checked whether it was blank, i.e., by checking whether all bits are set to FFH.
4. Incase of blank failure, the EPROM was erased using EPROM eraser, which exposes UV light through a glass window of the EPROM.
5. After blank check, the appropriate address of the random access memory (RAM) of Leaper-10 programmer corresponding to the physical address of

the fabricated hardware of the 8088 microprocessor motor speed controller was given and data (MOTORC.ABS file) written to it from a PC.

6. The program was brought into the program mode and all data stored in the RAM locations were transferred in the EPROM.
7. After the chip was programmed, it was brought in the verify mode whereby all the programmed contents in the EPROM were compared with the RAM contents. Verify signal was displayed if the comparison was correct.
8. The program was then integrated with the hardware for the purpose of testing.

3.4 Testing and Evaluation

3.4.1 System testing and troubleshooting

A test program was developed for troubleshooting the system (Fig: 3.2). Here, all the three ports of the PPI were configured as outputs (control byte 80H) and then the display was observed using LEDs. This enabled us to test the working of the ICs of the micro-controller. It was after testing the ability of the micro-controller that a program was developed to control the speed of a DC motor. This involved program design and coding, testing and debugging where necessary.

3.4.2 Integration of the system hardware and software

This is where the hardware and the software had to work together in order to perform the intended task. Integration of the system software with the hardware involved the burning of the program into the EPROM using the procedure discussed above.

The software performance was evaluated from the behaviour of the motor and the display on the autoranging combiscope.

```
                TITLE: MOTORCONTROL
                  CODE SEGMENT
ASSUME      CS: CODE, DS: CODE, ES: CODE
                                        Comments
            ORG 0000H                   ;
START:      MOV AX, 0FF00H              ; load AX with 0FF00H
            MOV SS, AX                  ;
            MOV SP, 0100H               ; Initialise stack pointer
            MOV AL, 80H                 ; Configure 8225A ports as outputs
            OUT 03H, AL                 ;
BACK:       MOV AL, 55H                 ; Load Accumulator with 55A and
            OUT 00H, AL                 ; Output
            OUT 01H,AL                  ;
            OUT 02H,AL                  ;
            CALL DELAY                  ; Call delay subroutine
            MOV AL, 0AAH                ; Load Accumulator with AAH and
            OUT 00H,AL                  ;Output
            OUT 01H,AL                  ;
            OUT 02H,AL                  ;
            CALL DELAY                  ;
            JMP BACK                    ;
DELAY:      MOV CX, FFFFH               ;
AGAIN:      NOP                         ;
            NOP                         ;
            LOOP AGAIN                  ;
            RET                         ; otherwise return
                                        ;
```

Fig. 3.2: Program for troubleshooting the micro controller.

3.5 Control Philosophy of DC Motors

To understand the control mechanism of the system, we need to look at the control philosophy of DC motors (40,41,42,43).

The DC motor consists of stationary field winding and a rotating armature winding. The winding is supplied with a dc current to produce a static magnetic field within the machine. This magnetic field interacts with the current in the armature conductor to produce a torque. In order to sustain this torque, the armature current distribution must be maintained relative to the field, irrespective of the actual rotor position. This is achieved by the use of a commutator, which reverses the current in the armature conductors as they pass from one field pole to the next. As the armature conductors move through the magnetic field produced by the field winding, a back emf V_b is induced which appears at the commutator, Figure.3.3.

The field winding may be either from a separate dc source or may be connected as a part of the armature circuit.

The steady state equations are:

$$V_b = V - I_a R_a \quad \text{.....(3.1)}$$

$$V_b = k_a \varphi N_b \quad \text{.....(3.2)}$$

Where V_b is the back emf, V is the applied Voltage,

φ is the flux per pole and

N_b the rotational speed in radians per second.

Internal mechanical power is given by

$$TN_b = V_b I_b \quad \text{.....(3.3)}$$

Hence, Torque

$$T = \frac{V_b I_b}{N_b} = \frac{K_a \varphi N_b I_a}{N_b} \quad \text{.....(3.4)}$$

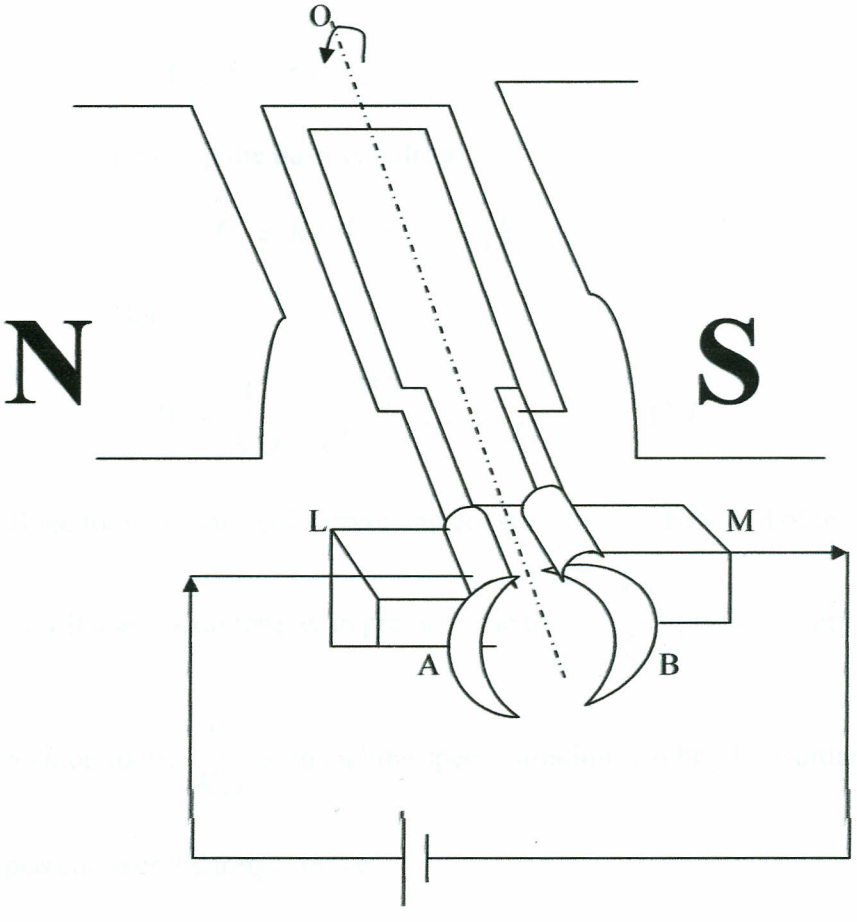


Fig: 3.3: Action of an electric motor.

A and B- Commutators or commutator segments

L and M-Carbon brushes

O-Horizontal axis of rotation.

3.5.1 DC motor operation

The torque- speed characteristics for shunt and separately excited motors are identical and may be developed by reference to the torque and voltage as follows:

$$E_a = K\varphi W \quad \text{.....(3.5)}$$

But $T = K\varphi I_a$

Ignoring the bush volt drop,

$$E_a = K\varphi W = V - I_a R_a \quad \text{.....(3.6)}$$

Hence,

$$W = \frac{V}{K\varphi} - \frac{TR_a}{\langle K\varphi \rangle^2} \quad \text{.....(3.7)}$$

Thus, for constant field current and constant flux φ, the speed of the machine will vary linearly with torque. In practice, the term $\frac{TR_a}{\langle K\varphi \rangle^2}$ will be very small in relation to the $\frac{V}{K\varphi}$ term and the speed variation will be of the order of a few percent over the torque range.

3.5.2 DC motor braking

In variable speed applications, a DC motor will be required to both accelerate and decelerate. Deceleration performance can be assisted by the use of the following ways.

a) Resistive (dynamic) braking

In the presence of a field, the back emf (E_a) produced will drive a current through the resistor, dissipating the energy stored in the rotating inertia of the machine as shown in Figure 3.3. The braking effect achieved is a function of speed.

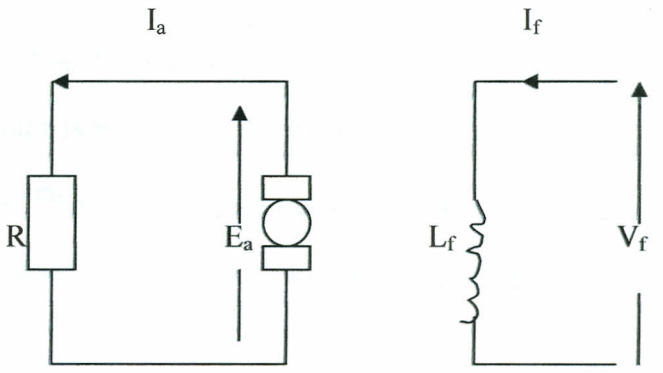


Fig. 3.4: Dynamic braking.

b) Regenerative braking

The field of a machine is adjusted so as to make $E_a > V$, when the machine acts as a generator returning energy to the supply and bringing the machine to rest. As speed increases, the field must be increased in order to maintain the condition $E_a > V$. The maximum field is limited by the field voltage supply and this will set a limit on the speed range over which regenerative braking can be used.

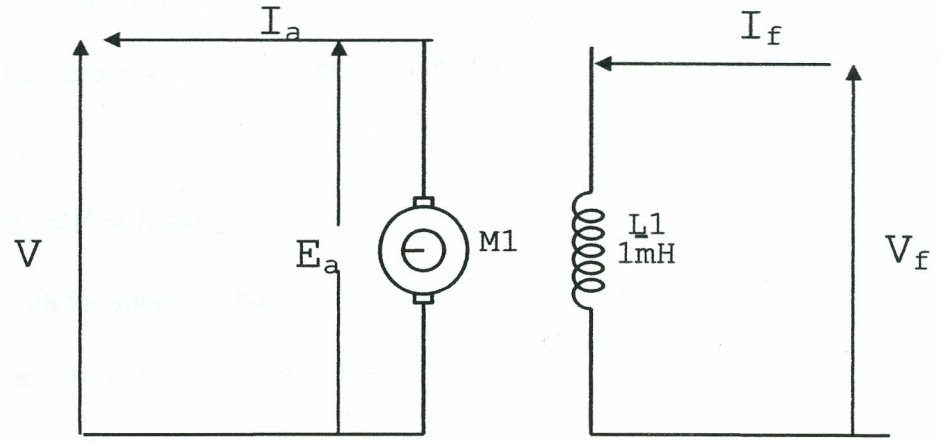


Fig. 3.5: Regenerative braking.

c) Reverse current braking (plugging)

By reversing the connection to the supply, the direction of current through the armature of the machine is reversed producing a reverse torque that rapidly decelerates the motor. This is a very severe condition and a current limiting resistor is normally included in the armature circuit to limit the current.

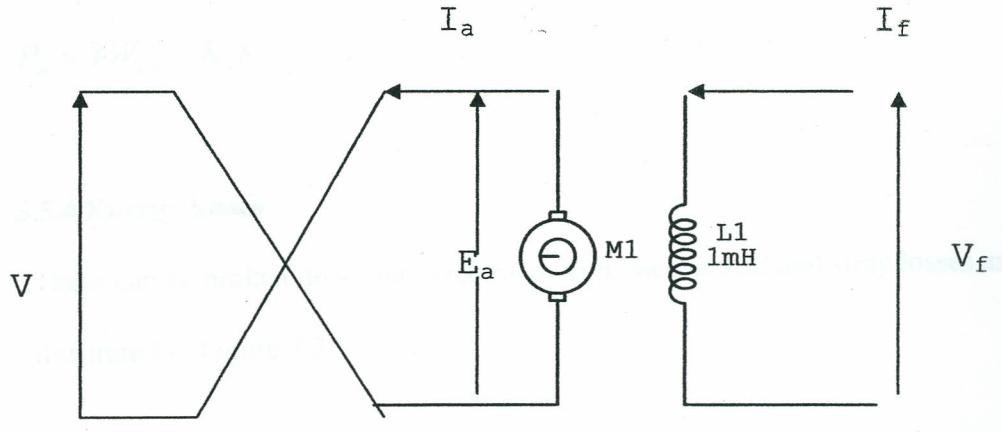


Fig. 3.6: Reverse current braking.

3.5.3 Speed control

If armature resistance is ignored, then the equation of the armature circuit reduces to:

$$E_a = V = K\varphi W_{rm} \quad \text{.....(3.8)}$$

With the supply voltage (V) constant the relationship between field current and the speed of the machine can be written as:

$$W_{rm} \alpha \frac{1}{\varphi} \quad \text{.....(3.9)}$$

Thus reducing the field φ of the DC motor by reducing the field current is associated with an increase in speed. Taking the rated value of armature current as the limiting condition, the maximum torque derived will be:

$$T_{\max} = K_t \varphi = \frac{K_t'}{W_{rm}} \quad \text{.....(3.10)}$$

And the mechanical power developed under these conditions is:

$$P_m = TW_{rm} = K_t K_t' \quad \text{.....(3.11)}$$

3.5.4 Energy losses

These can be broken down into electrical, iron, mechanical and stray losses as illustrated in Figure 3.7.

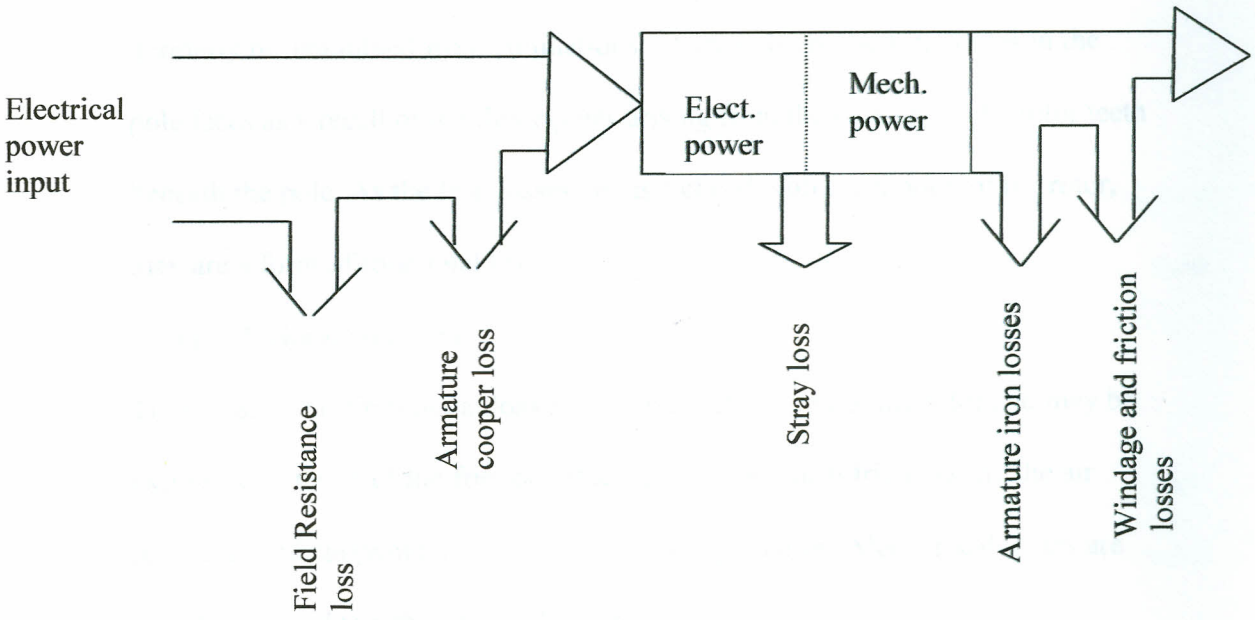


Fig. 3.7: Energy losses in a DC motor.

a) Electrical losses

The principal electrical losses are the I^2R or copper losses in the field and armature circuits. In addition, there is loss at the brushes, which is a function of the current through the brushes and the brush volt drop. For a typical carbon graphite brush, the effective brush resistance will decrease as the current through the brushes increases with the result that the brush volt drop remains approximately constant over the operating range of the machine.

b) Iron losses

As the rotor of the DC motor is moving through the field produced by the field windings, it will be subjected to both hysteresis and eddy current losses.

To reduce eddy current loss in particular, the rotor of a DC machine will normally be assembled from laminations. There will also be Iron losses in the pole faces as a result of the flux ripples arising from the passage of the rotor teeth beneath the pole. As the Iron losses are associated with the motion of the rotor, they are a form of rotational loss.

c) Mechanical losses

These losses are a rotational loss associated with motion of the rotor and may be expressed in terms of the friction effects (bearings, brush friction) and the air resistance effects (windage) experienced during rotation. Mechanical losses are usually grouped together as windage and friction.

d) **Stray Losses**

The term stray loss is used to group together a number of effects such as skin effect in armature conductors and additional Iron loss arising from leakage fluxes and is typically of the order of 1% of the rated power of the machine.

CHAPTER 4

THE SYSTEM DESIGNS

The system design consists of two parts namely: the Hardware and the Software designs. This chapter describes the various components of the designs, their performance and implementation [46-50].

4.1 The System Hardware Design

The microprocessor-based motor controller consists of 8088 Microprocessor, Clock Generator circuit, Main Memory (EPROM 2716 and RAM 6116 AP), Decoder Circuit, Address Latch (74LS373) Buffer (74LS367), Programmable Peripheral Interface (8225A), Switching Circuit and 9V DC motor.

4.1.1 Circuit Description, Construction and Operation

The system block diagram of the DC Motor Controller is illustrated in Figure 4.1.

On powering up the system, the reset circuit initialises the system. The motor control software is loaded from the memory subsystem (EPROM). Accessing of the memory subsystem or Input/Output chip (8255A) is determined by IO/$\overline{m}$ signal from the 8088 Microprocessor. This pin is connected to the decoder (74LS138) for memory selection and to 8255A PPI active low Chip-Enable pin through a NOT gate for inverting the IO signal. The 8255A chip has its ports configured as follows:

Ports A and C are unused; Port B is used as an Output port for outputting the Digital control Word corresponding to the desired waveforms/pulses. The digital word controls the motor speed and direction. This is achieved by switching the MOSFETs ON and OFF to modulate the current to the motor. The ratio of ON time to OFF time is what determines the speed of the motor.

The system components were assembled in their respective positions on the breadboard. Pin-Out diagrams of ICs (Integrated circuits) and terminal configurations of other parts are obtained using electronics data sheets. Components were inserted in the correct orientation on the breadboard with their joints soldered where necessary.

The complete circuit diagram of the motor controller is displayed in Figure 4.2. The support chips of the system are described below.

4.1.2 8284A Clock Generator

The 8088 microprocessor requires a clock signal with fast rise and fall times between low and high voltages. The Intel 8284A clock generator with the driver generates a train of pulses at constant frequency and synchronizes ready signals, which initialise the system [51, 52, 53].

The frequency source is a crystal connected across X_1 and X_2. The output frequency is one third of the input frequency. Therefore,

$$f_c = 3 \times F_{clk} \quad \text{.......................................}(4.1)$$

Where f_c is the crystal frequency and F_{clk} is the clock output frequency.

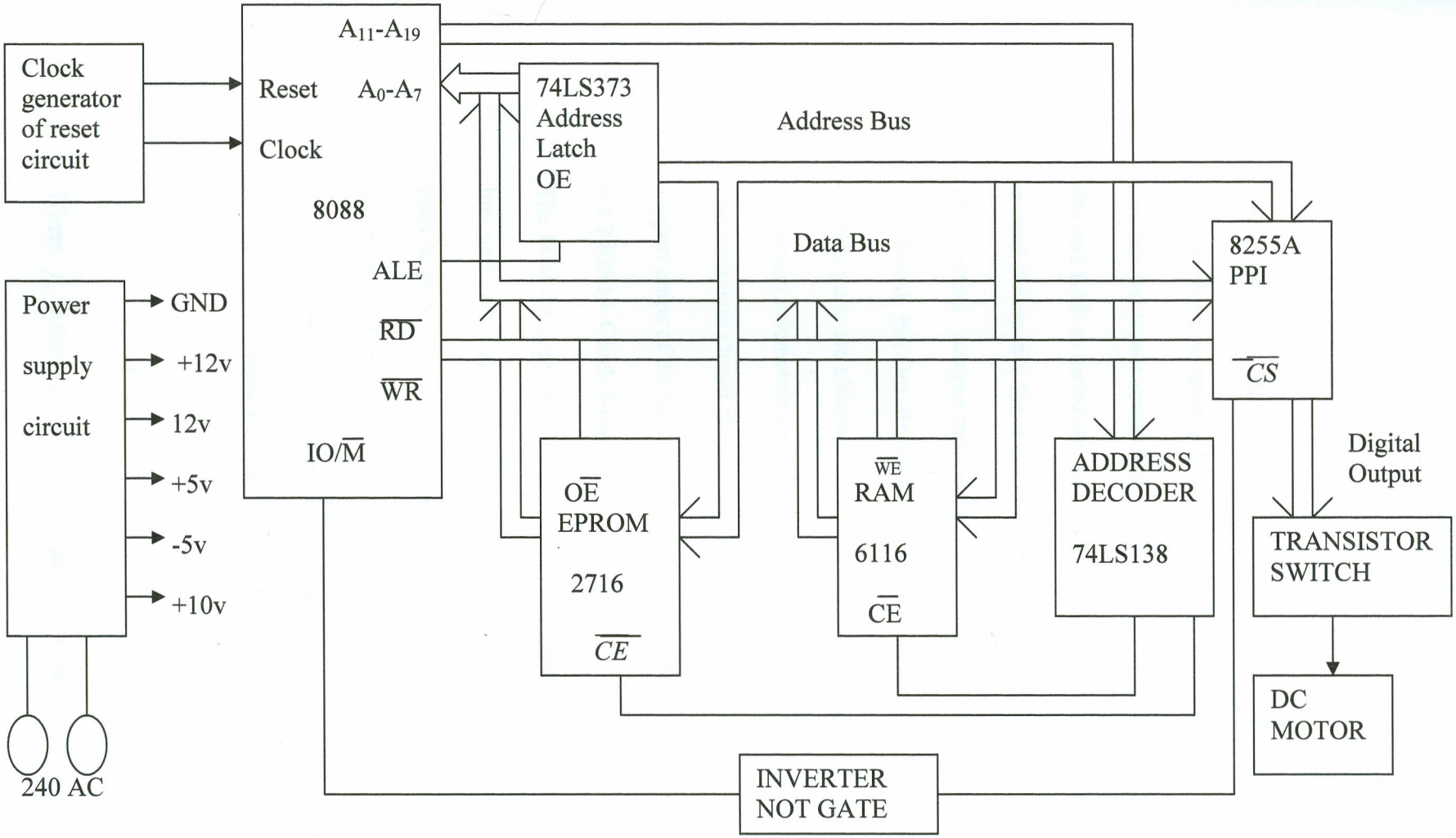


Fig 4.1: The System Block Diagram of the Microprocessor Based Motor Controller.

Ports A and C are unused; Port B is used as an Output port for outputting the Digital control Word corresponding to the desired waveforms/pulses. The digital word controls the motor speed and direction. This is achieved by switching the MOSFETs ON and OFF to modulate the current to the motor. The ratio of ON time to OFF time is what determines the speed of the motor.

The system components were assembled in their respective positions on the breadboard. Pin-Out diagrams of ICs (Integrated circuits) and terminal configurations of other parts are obtained using electronics data sheets. Components were inserted in the correct orientation on the breadboard with their joints soldered where necessary.

The complete circuit diagram of the motor controller is displayed in Figure 4.2. The support chips of the system are described below.

4.1.2 8284A Clock Generator

The 8088 microprocessor requires a clock signal with fast rise and fall times between low and high voltages. The Intel 8284A clock generator with the driver generates a train of pulses at constant frequency and synchronizes ready signals, which initialise the system [51, 52, 53].

The frequency source is a crystal connected across X_1 and X_2. The output frequency is one third of the input frequency. Therefore,

$$f_c = 3 \times F_{clk} \quad \text{.....(4.1)}$$

Where f_c is the crystal frequency and F_{clk} is the clock output frequency.

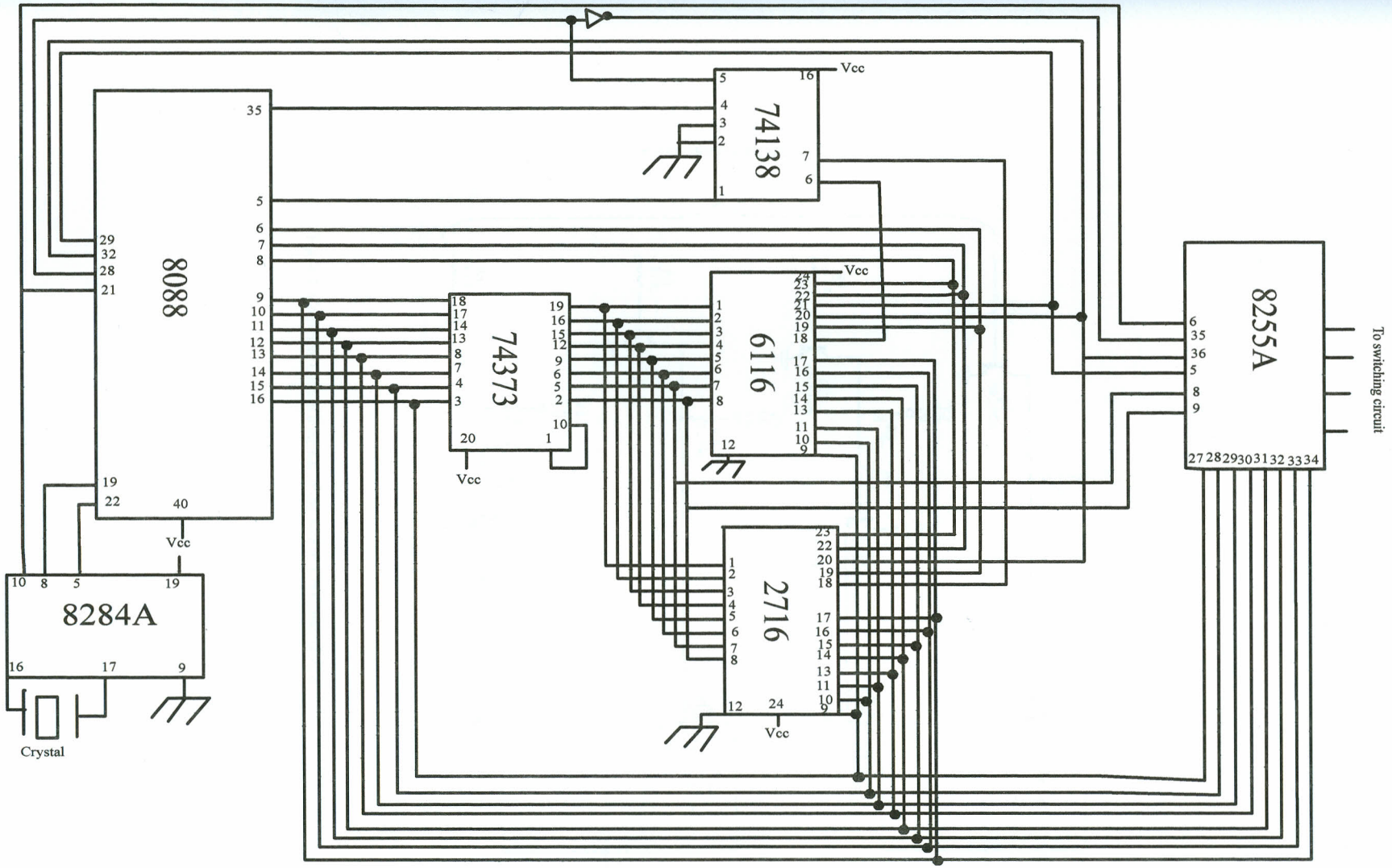


Fig.4.2: Complete Circuit Diagram of the Motor Controller

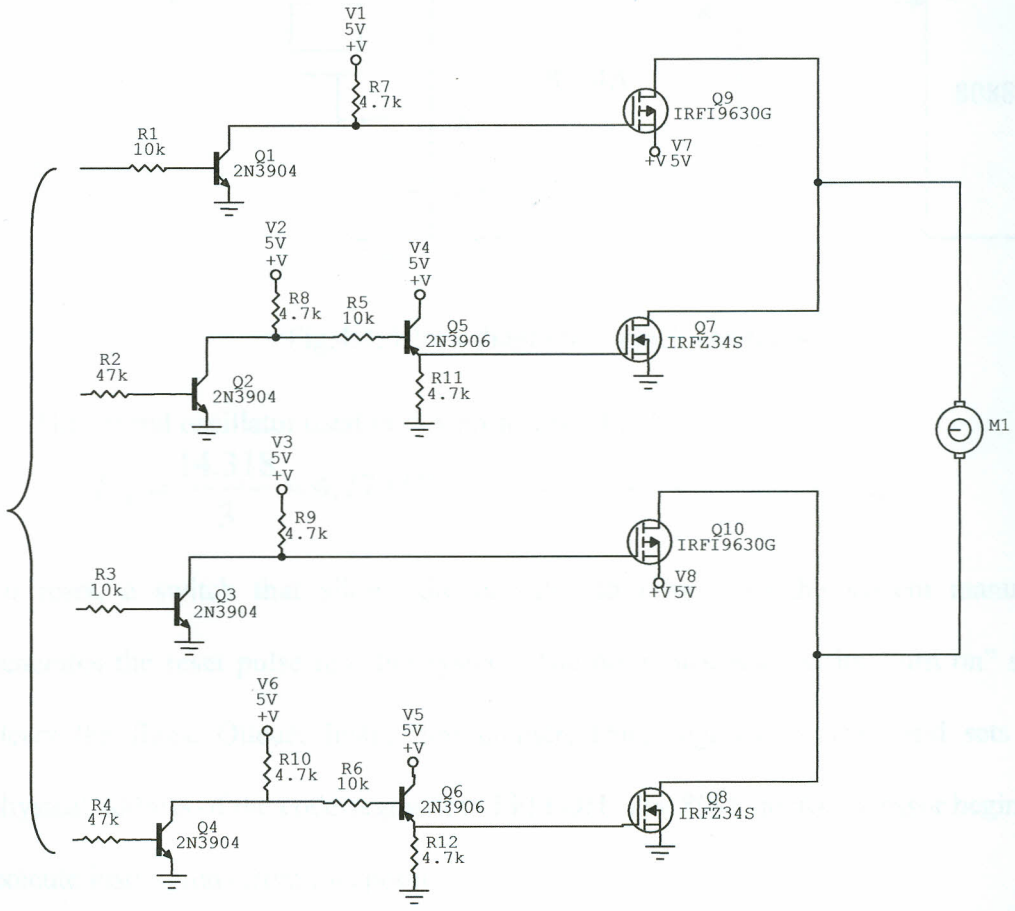


Fig. 4.2 Complete Circuit diagram of the Motor Controller cont'd

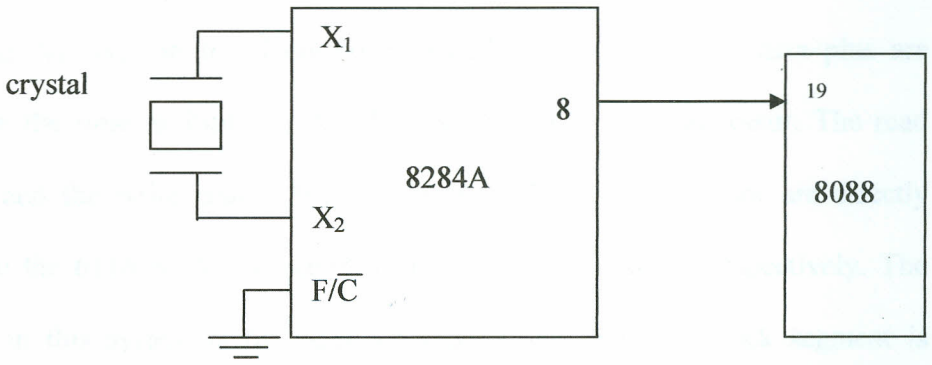


Fig.4.3: Block diagram of clock generator

The crystal oscillator used in this project is 14.318 MHz.

$$F_{clk} = \frac{14.318}{3} = 4.77MHz \text{ ----------(4.2)}$$

To reset, a switch that allows the operator to reinitialise the system manually generates the reset pulse into the system. The microprocessor in its "turn on" state clears the flags, Queue, Instruction pointer, Data segment register and sets the physical address of the code segment to FFFFOH. The 8088 microprocessor begins to execute instructions from this point.

4.1.3 Memory

The memory used in this system includes $2K \times 8$ EPROM IC and $2k \times 8$ RAM IC These two ICs are discussed below.

a) Random Access Memory (RAM 6116AP)

The RAM 6116AP is a static read/write memory, a low type complementary metal oxide semiconductor organised as 2048 words × 8 bits with access time 120ns and

voltage supply of +5V direct current. The address pins of the 6116AP are directly connected to A_0 –A_{10} on the demultiplexed address bus while the data pins are connected to the time multiplexed data bus on the 8088 microprocessor. The read enable $\overline{RD}$ and the write enable $\overline{WE}$ signal from the microprocessor are directly connected to the 6116 RAM to enable read and write operations respectively. The RAM used in this system is for stack operations and therefore stack segment is mapped onto it.

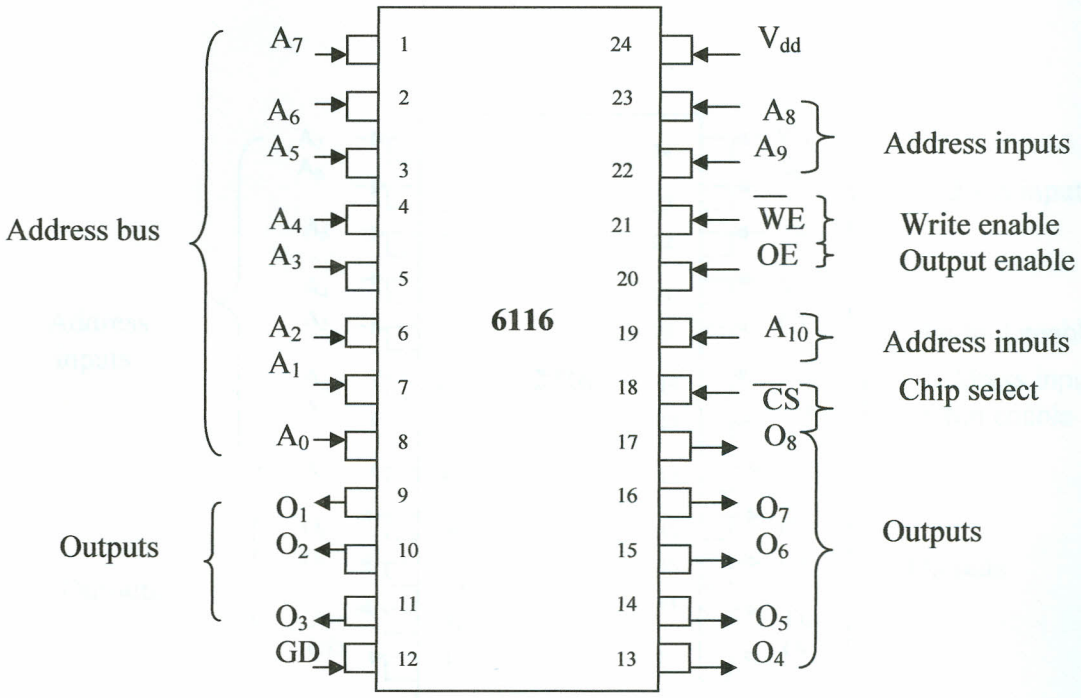


Fig.4.4: Pin out of 6116 (RAM)

b) Erasable Programmable Read Only Memory (EPROM 2716)

The 2716 EPROM, 2048 memory locations X 8 bits Erasable Programmable Read Only Memory is a metal oxide semiconductor with access time of 450ns. It is designed to work with microprocessor buses. Addresses A_0-A_{10} of the EPROM are connected to the demultiplexed Address bus A_0-A_{10} of the 8088 microprocessor. The time multiplexed address /data bus A_0-A_7 of the control line drives the output enable OE of the 2716 and enables the output data onto the data bus from it with the proper timing to prevent bus contention problem.

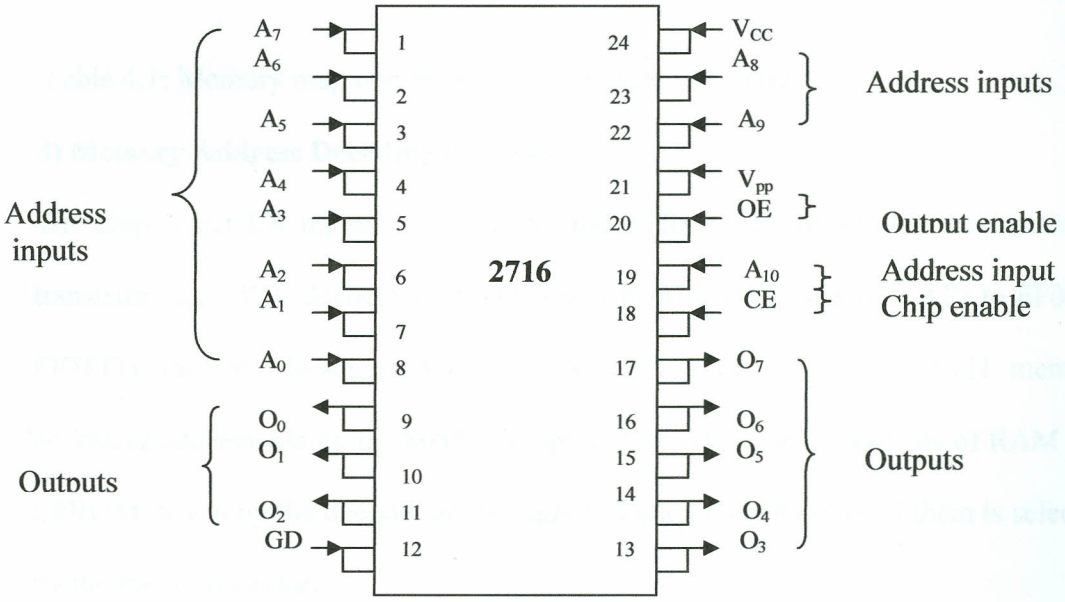


Fig.4.5:Pin out of 2716 EPROM

c) Memory Map

The 8088 microprocessor is directly mapped to location FFFFOH when power is first applied and the first instructions are kept in this location onwards. In this research work, the EPROM IC is mapped from location FF 800~FFFFFH and the RAM from location FFOOO~ FF7FF H.

ADDRESS	MEMORY	TYPE USED
00000-FEFFFH	Unused	-
FF000-FF7FFH	RAM	6116
FF800-FFFFFH	EPROM	2716

Table 4.1: Memory map nomenclature of the Motor Controller.

d) Memory Address Decoding Circuitry

The chip select CS inputs of the RAM and EPROM are driven by the transistor-transistor logic TTL decoder (74LS138) which maps the RAM 6116AP) to FF000~ FF7FFH memory locations and EPROM (2716) to FF800~ FFFFFH memory addresses addressable by the 8088 microprocessor. The chip select lines of RAM and EPROM driven by the decoder are brought to logic O when either of them is selected by the microprocessor.

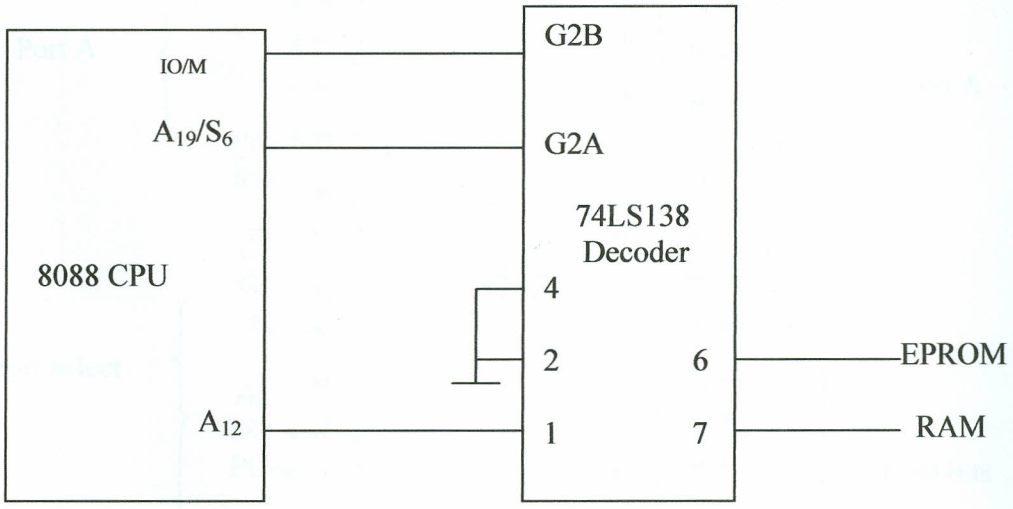


Fig. 4.6: The Decoder Configuration.

4.1.4 Programmable Peripheral Interface (PPI)

The 8255A is a Programmable Peripheral Interface (PPI) device designed for use in most microcomputer systems. It is fabricated using NMOS technology and contains 40 pins. Its function is that of a general-purpose I/0 component to interface peripheral equipment to the microprocessor system bus. It is a powerful tool with enough flexibility to interface almost any I/O device to the CPU bus without the need of additional external logic. Each peripheral device normally has a unique service routine.

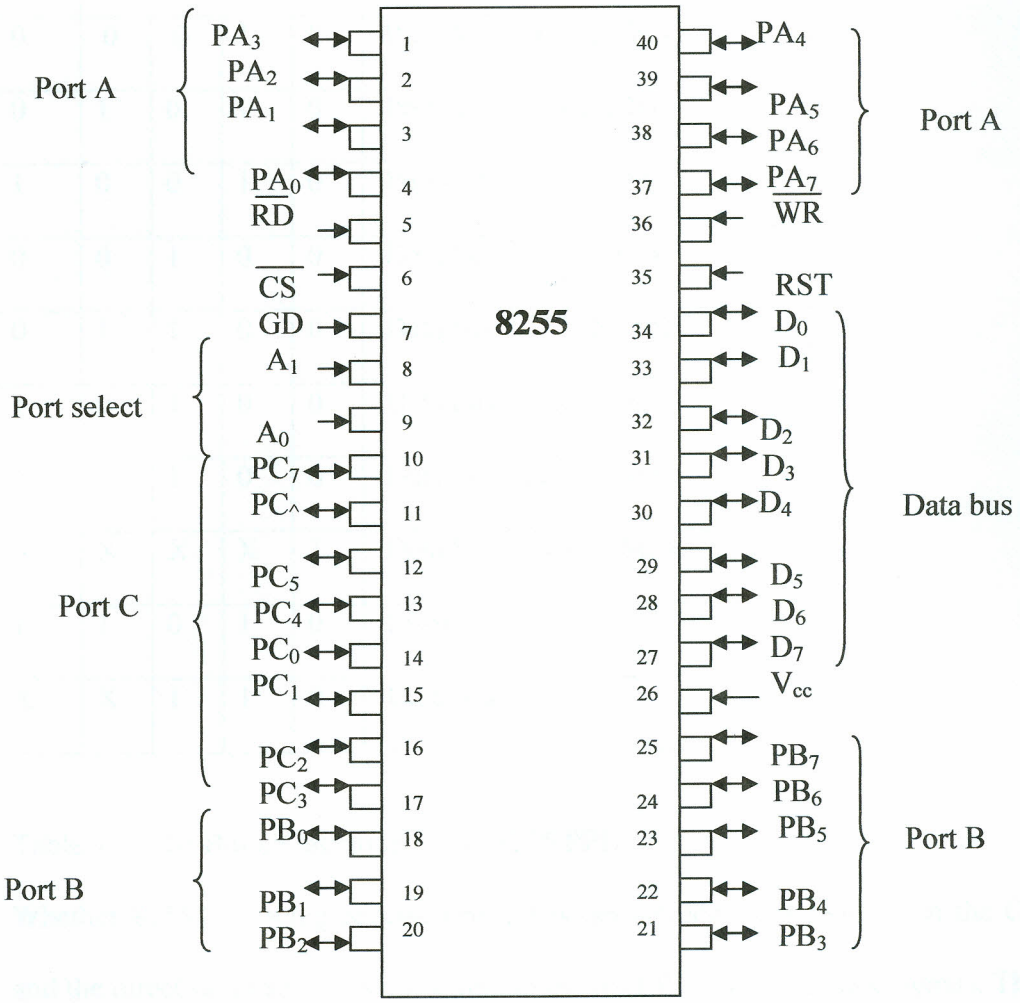


Fig.4.7: Pin out of 8255A PPI.

The PPI contains a control register and three separately 8-bit addressable input/output ports labelled A, B and C which are related to the basic operation of 8255 as shown in table 4.2 below.

A_1	A_0	RD	WR	CS	Operation
0	0	0	1	0	Port A ⟶ Data bus
0	1	0	1	0	Port B ⟶ Data bus
1	0	0	1	0	Port C ⟶ Data bus
0	0	1	0	0	Data bus ⟶ Port A
0	1	1	0	0	Data bus ⟶ Port B
1	0	1	0	0	Data bus ⟶ Port C
1	1	1	0	0	Data bus ⟶ Control
X	X	X	X	1	Data bus ⟶ 3 State
1	1	0	1	0	Error
X	X	1	1	0	Data bus

Table 4.2: The Basic Operation of the 8225 PPI.

Whether 8255A is being accessed or not is determined by the signal on the CS pin and the direction of access is according to the read $\overline{RD}$ and write $\overline{WR}$ signals. The bits in the three ports are attached to the pins that may be connected to the input/output device. The bits are divided into groups A and B.

Group A:

Consists of bits in port A (PA_7-PA_0) and the four most significant bits (MSBs) of port C (PC_7-PC_4).

Group B:

Consist of port B (PB_7-PB_0) and the four least significant bits (LSBs) of port C (PC_3-PC_0).

In this research project, the 8255 has been used in mode 0, a basic input/output mode in which ports A and B can be individually configured as input or output, port C upper or lower bits can also be individually configured as input and output.

MSF	**Mode selection**		**Port A (output)**	**Port C-uppe (output)**	**Mode**	**Port B (output)**	**Port C-lower (output)**	**HEX**
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	0	0	0	0	0	0	0	80H

Table 4.3: System control word.

The 24 I/O pins of the 8255 are divided as;

Port A: Pins 1-4, 37-40

Port B: Pins18-25

Port C: (Upper) Pins 10-13 ,(Lower) Pins 14-17

The system command word 80H therefore sets the ports as follows:

Port A : OUTPUT

Port B: OUTPUT

Port C: OUTPUT

The port addresses and functions performed by individual ports in the control of the motor speed is given as,

Address	Port
00H	Port A
01H	Port B
02H	Port C
03H	Command register

4.1.5 Switching circuit

This circuit is developed using six transistors (2N3904 and 2N3906), four high power MOSFETs (IRFZ34 and IRFI9630) and resistors. These act as state switches for controlling the speed of the DC motor [54-61]. Here, Pulse Width Modulation (PWM) method has been used where a switch is turned ON and OFF to modulate the current to the motor. To run a motor both forward and reverse, an H-bridge circuit has been set that can reverse its polarity.

If we close switches S_3 and S_2, the motor will spin in one direction. By modulating one of the switches, the speed of the motor can be controlled. Similarly, by closing switches S_4 and S_1, the motor spins in the other direction. The switching circuit of the system has applied this principle as shown in Figure 4.9.

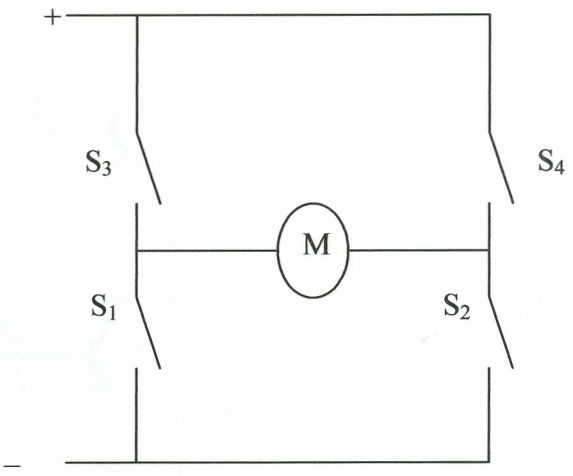


Fig. 4.8: H-bridge circuit

Operation

The pair of the MOSFETs to be turned on depends on the output from the PPI (8255A). Since port B is used as an output port, pins 25-22 (PB_7-PB_4) act as the inputs to the switching circuit where pin 25 (PB_7) is the most significant bit. When the input to the circuit is 90H, pin25 and pin 22 are high and the motor turns clockwise. When the input to the circuit is 60H, pin 24 and pin 23 are high and the motor turns anticlockwise.

These switches turn the power to the motor fully on or fully off at a given frequency. The average voltage (or current) applied to the motor can be varied by varying the amount of time the switch is ON with respect to the time when the switch is OFF (duty cycle). The choice of high power MOSFETs (IRF Z34 and IRFI 9630) as switching devices is as a result of their high current capacity without the need of heat sinks.

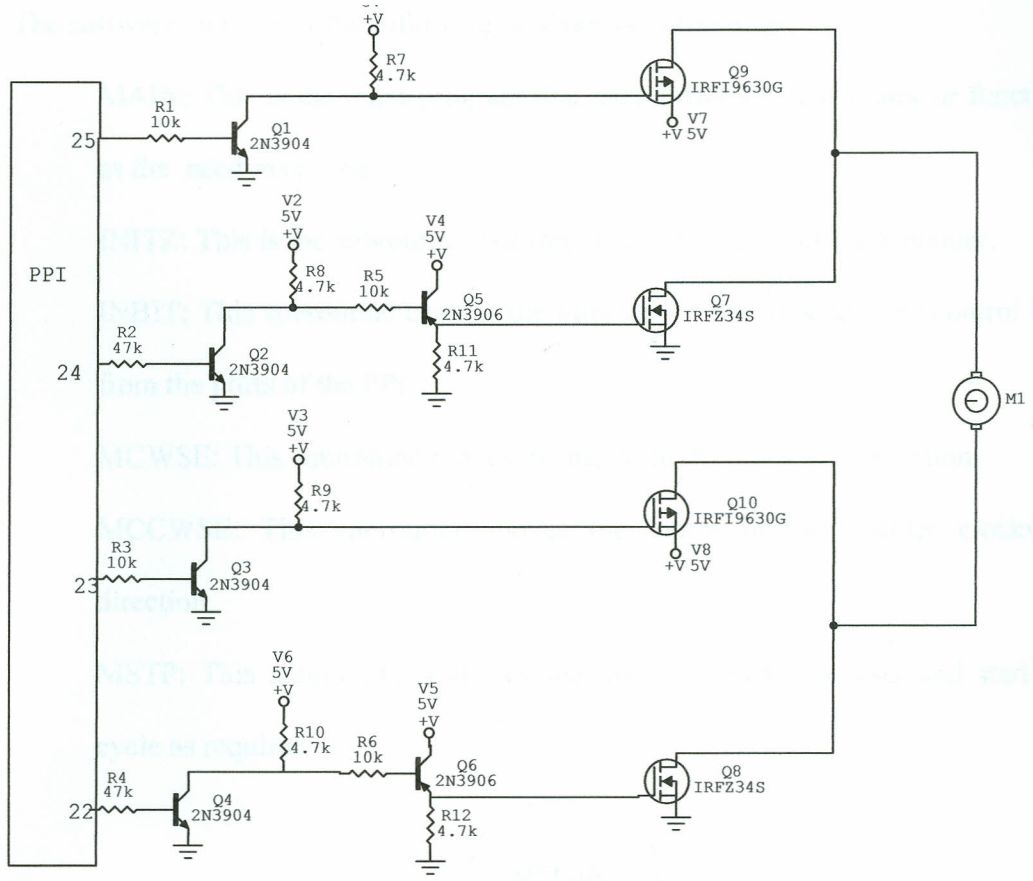


Fig.4.9: The Switching circuit being used

4.2 The System Software Design

The sequence of steps for the system software was developed using assembly language and coded in 8088 assembler. The control and monitoring program consists of several subroutines for execution. Its implementation involved program design and coding, testing and debugging. The block diagram of the control program is displayed in Figure 4.10.

The software consists of the following modules or subroutines:

MAIN: This is the main program that calls different subroutines or functions as the need may arise.

INITZ: This is the subroutine that initializes the ports and stack pointer.

INBIT: This subroutine enables the microcontroller to accept the control byte from the ports of the PPI.

MCWSE: This subroutine moves the motor in the clockwise direction.

MCCWSE: This subroutine moves the motor in the counter clockwise direction.

MSTP: This subroutine waits for the microcontroller to reset and start the cycle as required.

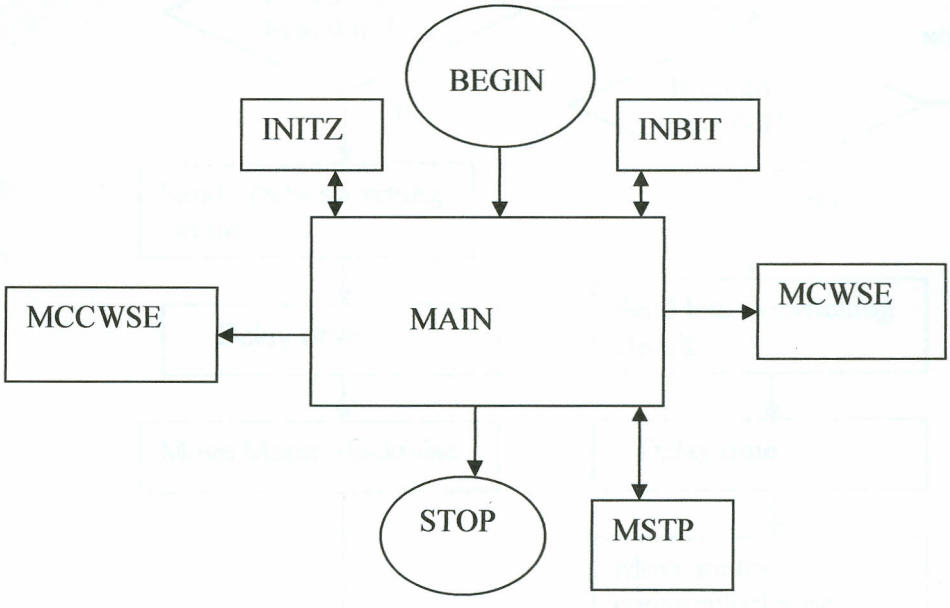


Fig.4.10: Block diagram of the system software design.

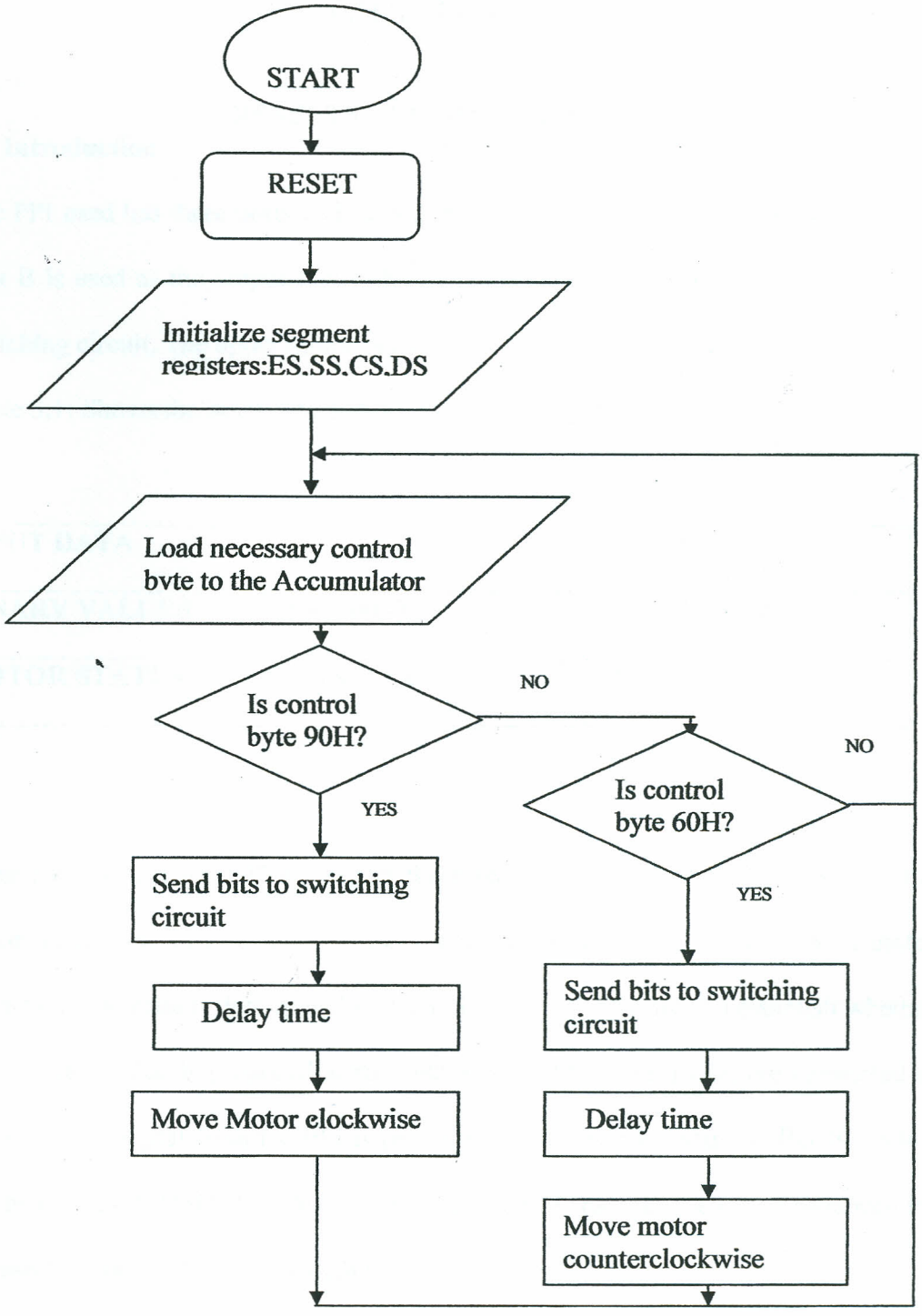


Fig.4.11: Software Design Flow Chart.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Introduction

The PPI used has three ports and each port has 8 bits. Ports A and C are unused and Port B is used as the output port.. The output from Port B becomes the input to the switching circuit. The upper nibble is used to switch a 9V DC motor.

Table 5.1: Shows the input data and the corresponding decimal values.

INPUT DATA	90H	60H
BINARY VALUES	1001 0000	0110 0000
MOTOR STATUS	Clockwise	Anticlockwise

If the input is at logic "0" we expect the motor to stop. The MOSFETs switch the motor on and off in a series of pulses as stated in the software. The time a motor remains in one state is determined by the delays in the software. To establish whether the expected value is produced at the output port of the PPI, LEDs are connected at this port. The signal from the PPI is passed through a non inverting buffer IC 74367 and then to the MOSFETs. This is to protect the PPI from the back emf produced by the motor when the power is switched off.

Table 5.2: Buffer (IC 74367) pin connections

PORT B AND BUFFER PIN CONNECTIONS		
	IN	**OUT**
P25	2	3
P24	4	5
P23	6	7
P22	14	13

5.2 System Control

The voltage at the output was measured to be + 4.25V DC. This voltage was capable of lighting a 5V LED but needed to be stepped up in order to drive a 9V DC motor. This was achieved by passing it through a solid state switch which could be turned on by a voltage as low as 3V. The effectiveness of the control process depended entirely on the delays in the control software. This is because the motor needed enough time during the transitions from clockwise to counter clockwise and vice versa. The pulses generated by the microprocessor to control the motor were displayed on a combiscope as shown in Figure 5.1.

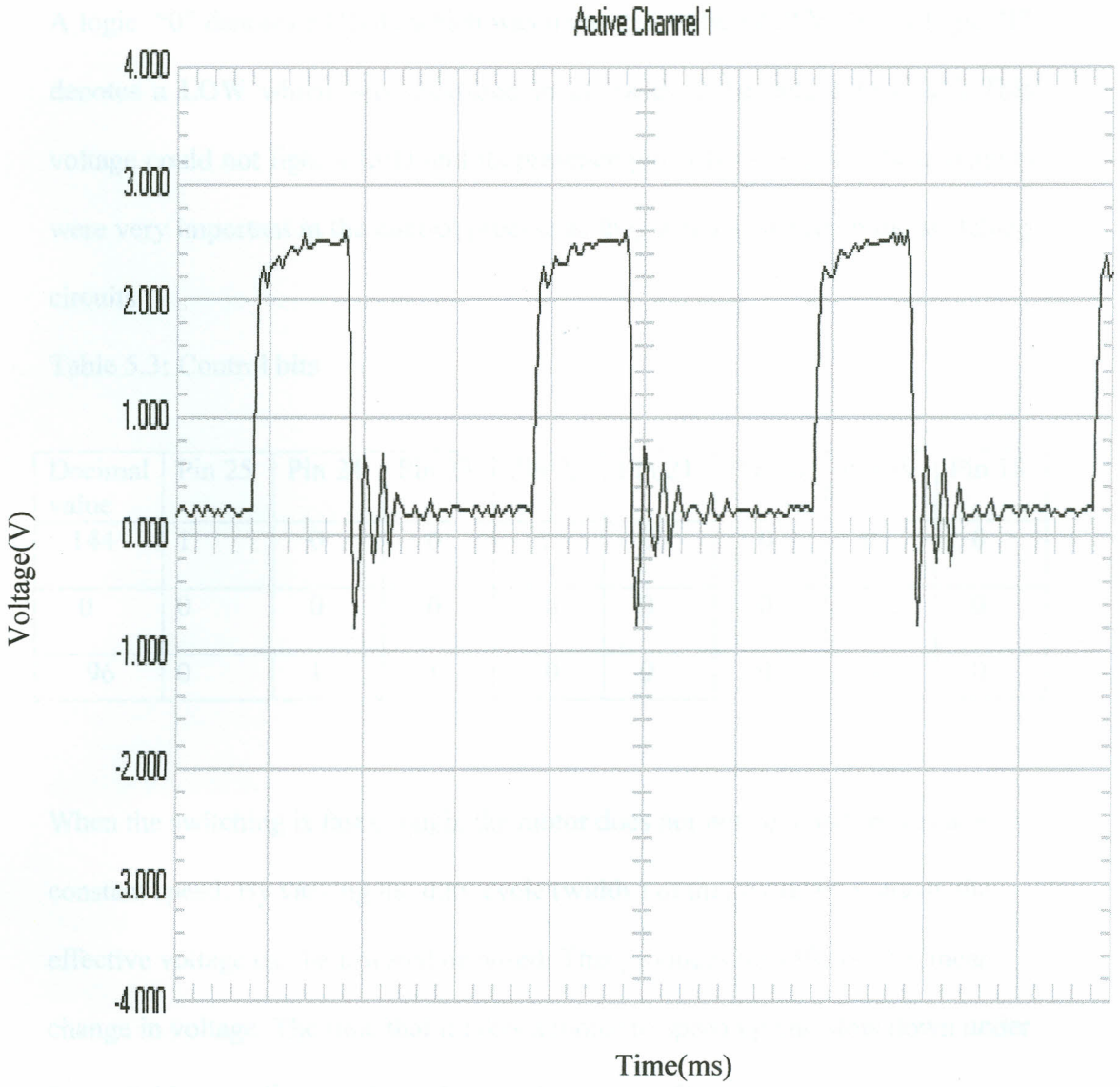


Fig.5.1: Signal waveform for controlling the DC motor as displayed on the screen of a Combiscope

A logic "0" denotes a HIGH which was measured to be +2.25V DC. A logic "1" denotes a LOW which was measured in all cases to be below +1V DC. This voltage could not light a LED and its presence puts off the diode. These outputs were very important in the control process as they acted as inputs to the switching circuits.

Table 5.3: Control bits

Decimal value	Pin 25	Pin 24	Pin 23	Pin 22	Pin 21	Pin 20	Pin 19	Pin 18
144	1	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0
96	0	1	1	0	0	0	0	0

When the switching is fast enough, the motor does not notice it and moves at a constant speed. By varying the duty cycle (width) of this switched voltage, the effective voltage can be lowered or raised. This produces the effects of a linear change in voltage. The time that it takes a motor to speed up and slow down under the switching condition is dependent on the inertia of the rotor.

5.3 Motor Control

The following factors were put into consideration when controlling the 9V motor.

a) Effect of varying load

When the motor is set to operate at a specified speed of 1000 rpm for a particular load, increase in load will tend to slow down the motor. The reduced speed results in a lower value of the back emf E_b. Similarly, decrease in load increases the speed to its preset value.

b) Effect of varying supply voltage

An increase in supply voltage tends to speed up the motor. This tendency is partially counteracted by the fact that increased voltage across field windings increases the field current. An increase in field current will tend to slow down the motor. Therefore there is a balancing process between the armature and field circuits.

c) Effects of pulsating supply voltage

At the moment of starting the motor, there is no back emf in the armature winding. Since the armature resistance is very small, the motor will tend to take an excess current which may burn out the armature winding if the full steady state voltage is applied at the time of start. It may be necessary to connect a resistor in series with the armature circuit at the time of start and then cut it off as the motor speeds up. This is minimized by applying a pulsating dc voltage to the armature through a solid state switch. When the switching circuit is conducting, a back emf is induced across the armature owing to the pulsating nature of the applied voltage. Hence no excess current flows through the armature.

d) Effect of reversing armature terminals

To reverse the direction of rotation of the motor, the connections to the armature terminals alone, or the connections to the field windings alone, must be reversed. If both are reversed, the direction remains unchanged. The connections to the armature terminals were reversed by sending either a logic "0" (0V) or logic "1" (5V). This made the motor to run in the opposite direction.

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

6.1 Conclusion

The microprocessor-based motor controller was successfully designed and constructed. The prototype allows greater flexibility since output controls are generated through the software rather than the hardware. The system constructed and described uses Pulse Width Modulation method to control the current to the motor as desired hence achieving the research objectives. It is capable of switching DC motors and also changing their directions of rotation. From the results obtained, the system can be improved further and developed to be used in industries.

6.2 Recommendations

Recommendations for further research are:

a) Develop software that is capable of running the motor at variable speeds depending upon the requirements.

b) Addition of an overload relay assembly which provides the motor with an overload protection.

c) Use Printed Circuit Board (PCB) to minimize power losses

REFERENCES

(1) Frenzel, L.E., *Crash Course in Electronics Technology*, Newnes, U.S.A, 1996.

(2) Mazza, L., *Micro controller-based motor speed control*, Webelectric magazine, 2001.

(3) Torrens, R., *PWM speed control*, June 2002.

W ww.4qdtec.comm/pwm-02.html

(4) Sinclair, I., *Practical Electronics Handbook*, Newnes, Great Britain, 1991.

(5) R&D Division of K&H Experimental book of MTS-88.C: *Microcomputer Teaching System*, MFG. CO. Ltd, 1991.

(6) Houpis, H.C., and Lamout, B.G., *Digital Control Systems: Theory and Hardware, Software*, McGraw-Hill Inc., Singapore, 1992.

(7) Wollard, B.G., *Practical Electronics*, McGraw-Hill Book Company, London, 1999.

(8) Hughes, A., *Electric Motors and Drives*, Newnes, Great Britain, 1993.

(9) Subbarrao, W.V., *The 8086/8088 Family of Microprocessors Software and System Applications*, Delmar Publishers Inc. U.S.A, 1992.

(10) Gaokar, R.S., *Microprocessor Architecture, Programming and Applications with 8085, 3rd edition*, Prentice Hall Inc., New Delhi, 1996.

(11) Bansal,R., *Digital Logic and Microprocessor Technology*, Galgata publication Ltd, New Delhi, 1999.

(12) Tseng, V., *Microprocessor Development and Development Systems*, Granada, Great Britain, 1982.

(13) Dorf, R., *Electrical Engineering Handbook*, CRC Press, U.S.A, 1993.

(14) *Simple microprocessor- based motor speed controller*, The model electronics company, Project article No. 1.0, November 1999.

(15) Comer, D. J., *state machine verses microprocessor controller: Digital System Design*, Vol. E-30 No.2, May 1987.

(16) *Intel Corporation, iapx 8088 manual*, Santa Clara, 1981.

(17) Ciarcia, S., *Speed up your IBM PC with 16-bit co processing power*, Byte, Vol. 9 No. 5, May 1984.

(18) Meyerle, G., *16-Bit Microprocessor Technology, Computers and Electronics*, Vol.21 No.3 ,March 1983.

(19) Yu-Cheng and Gibson, A.G., *Microcomputer Systems: The 8086/8088 Family*, Prentice-Hall, New Delhi, 1997.

(20) Clements, A., *The Principles of Computer Hardware* ,Oxford University Press, New York, 1991.

(21) Downtown, A.C., *Computers and Microprocessors; Components and Systems*, Chapman & Hall, Hong Kong, 1992.

(22) Gottrieb,I.M., *Electronic power control*, TATA- McGraw-Hill publishers, New Delhi, 1999.

(23) Tiernan, C., *Realizing benefits from IT*, IMIS vol 11, No.5, October 2001.

(24) Reaves, E.A., *Newnes Electrical Pocket Book*, Newnes, Great Britain, 1995.

(25) Kelly,D.O., *Performance and Control of Electrical Machines*, McGraw-Hill Company, Great Britain,1991.

(26) Gieras, J.F., *Linear Induction Drives*, Oxford University Press,U.S.A, 1994.

(27) Hardistyri, R., Computer Controller for DC Motors, Electronics Australia, page 51, 1994.

(28) Orla, E., and Tedson, E., *Direct Current Fundamentals*, Delmar publishers Inc. U.S.A, 1991.

(29) Owade, M., *Design and Development of a Programmable Laboratory interface System with an illustrative use in Resistivity Experiment*, Msc. Thesis, Kenyatta University, 1998.

(30) Karimi, P., *Microprocessor Controlled Multifunction Generator*, Msc. Thesis, Kenyatta University, 2000.

(31) Bourne, V., *Networking-the sum of parts*, IMIS Vol. 11 No. 3, June 2000.

(32) Comer D., *State machine verses microprocessor controller: Digital System Design*, Vol. E-30 No.2, May 1987.

(33) David et al, *Using the 8051 Microcontroller with Resonant Transducers, IEEE Transaction on Industrial Electronics*, Vol. IE-32 No. 2, November 1985.

(34) Bently, J.P., *Measurement Systems*, Addison Wesley Longman Ltd, UK, 2000.

(35) Fendt. W., *Direct current electrical motor*, January 15, 1999. http://www.home.a-city.de/walter.fendt/physeng/electricmotor.html

(36) Rudolf, F.G., *Encyclopedia of Electronics Circuits Vol. 3*, TAB Books, U.S.A, 1991.

(37) Rudolf, F.G., and Sheets, W., *Encyclopedia of Electronics Circuits Vol. 4*, TAB Books, U.S.A. 1992.

(38) Savitch, W., *Problem solving with C^{++} the object of programming 2^{nd} Edition*, Addison-Wesley Longman Inc. U.S.A. 1999.

(39) Holland, R., *Microcomputer Fault- Finding and Design*, Macmillan Press LTD, London, 1995.

(40) Hall, D.V., *Digital circuits and systems*, McGraw-Hill Inc., U.S.A ,1990.

(41) Uffenbeck.J., *Digital Electronics: A Modern Approach*, Prentice Hall, Eaglewood cliffs- New Jersey,1994.

(42) Bradly,D., *Basic electronic power and machines*, Chapman and Hall, Great Britain, 1994.

(43) Schwarz, K., *Design of industrial electrical motor drives*, Butterworth-Heinmann Ltd, Great Britain, 1991.

(44) Mehta,V.K., and Mehta, R., *Principles of electrical machines*, CHAND and Company Ltd., New Delhi, 2002.

(45) Mcmanis,C., *DC motor speed controller.*

http:/www.ozitronics.com

(46) Cook, G.F., *Pulse Width Motor Speed Controller/ DC Light Dimmer*, Home Power Magazine, No. 75, 1999.

(47) Ronald.J.T., and Neal.S.W, *Digital systems; Principles and Applications 8th edition*, Prentice Hall of India, New Delhi ,2004.

(48) Nave.R, DC motor operation, June 2000.

http://www.hyperphysics.phy.ast.gsu.educ/hbase/magnetic/motdc.html

(49) *Control Tutorials for Matlab/DC motor modeling*, University of Michigan, August 1997.

http://www.engin.umich.educ/group/etm/examples/motors.html

(50) Sawhney.A.K, *A course in Electrical and Electronic measurements and Instrumentation*, Educational and Technical Publishers, New Delhi, 2003.

(51) *Understanding D.C. Motor Characteristics, copyright 1999 center for innovation in product management*, January 20,1999.

http://www.oddparts.com/ocsi/motorut.html

(52) French.C.S, *Computer Science , 5th edition,* Great Britain, 1996.

(53) *CMOS Clock Generator*, Intersil Corporation, March 1997.

(54) Herbert, S., *Load current sensing in switch mode bridge motor driving circuits*, Stmicroelectronics, December 2003.

(55) Chuck, M., *DC Motor Controllers*, Wikipedia, the Free Encyclopedia, October 2007.

(56) Dr.Bimbha, P.S., *Power Electronics*, KHANNA Publishers, New Delhi,2003.

(57) Schuler, A., *Electronics Principles and Applications*, McGraw-Hill,U.S.A.,1999.

(58) H-bridge Motor Controls

http:/www.4qd.co.uk/fac/index.html

(59) Rutkowski, B.G., and Olesky, E.J., *Solid State Electronics*, McGraw-Hill, U.S.A., 1992.

(60) Kuphaldt,T.R., *Electric Motors*, Industrial Electronics, Vol. 2,December 2006

(61) Thomson,S.G., *Speed Controllers*, Digital Systems, Vol. 3, November 2005.

APPENDIX A

PROGRAM LISTING

The complete control software for the DC motors is listed as follows:

Microsoft ® Macro Assembler Version 5.10

```
;PROGRAMMER    : Mwenda M. Phylis
;PROGRAM TITLE: DC Motor Controller (MOTORC.asm)
;ABSTRACT      : This program generates pulses that switches on DC motors and
               : also change their direction of rotation. The user loads the
               : Accumulator with the control bytes 90H and 60H. These bytes
               :  switches a DC Motor through an H-bridge circuit.
;REGISTERS USED: AL,CX,SP,CS,IP,SS,DS,ES
; PORTS         :Port A (00H) Unused
               : Port B (01H) Output port
               : Port C (10H) Unused
               :
;MEMORY        : 00000H~FEFFFH (Unused)
               : FF000H~FF7FFH (RAM-SS)
               : FF800H~FFFFFH (EPROM-DS,CS and ES)

                CODE SEGMENT
                      ASSUME CS:CODE,ES:CODE, DS,CODE

                 ORG 0000H

                CWISE   EQU  0FF00H
                CCWISE  EQU  0FF01H
                RESET   EQU  07F0H
                        ;
        INIT:           NOP
                        NOP
                        NOP
                        NOP
                        NOP
                        ;
```

```
        MOV  AX,0FF70H
        MOV  SS,AX     ;Initialise stack segment to start at RAM's first
                       ; Memory location
        MOV  AX,FF800H
        MOV  DS,0100H  ;Initialize data segment to start at ROM'S first
                       ;memory location
        MOV  AL,80H
        OUT 03H,AL     ;Configure port B as an output port
        ;
        MOV  AL,90H    ;Load Accumulator with 90H
        MOV  BX, CWISE
        MOV  [BX],AL
        MOV  AL,60H    ;Load Accumulator with 60H
        MOV  BX,CCWISE
        MOV  [BX],AL
        ;
MORE:   MOV  SI,OFFSET CWISE
        MOV  AL, BYTE PTR CS:[SI]
        OUT  01H,AL  ;Output Clockwise bits at port B
        CALL DELAY
        ;
        MOV  SI,OFFSET CCWISE
```

```
        MOV  AL, BYTE PTR CS:[SI]
        OUT  01H,AL   ;Output counterclockwise bits at port B
        CALL DELAY
        ;
DELAY:  MOV  AL,[BX]
        MOV  CL, AL
        INC  BX
        MOV  AL,[BX]
        MOV  DL,AL
AGAIN:  NOP
        NOP
        NOP
        NOP
        LOOP AGAIN
        RET
        ;
        ORG RESET
        NOP
        NOP
        NOP
        CODE ENDS
        END
```

APPENDIX B

8086/8088 INSTRUCTION SETS

	GENERAL PURPOSE
MOV	Move byte or word
PUSH	Push word onto stack
POP	Pop word off stack
XCHG	Exchange byte or word
XLAT	Translate byte
	INPUT/OUTPUT
IN	Input byte or word
OUT	Output byte or word
	ADDRESS OBJECT
LEA	Load effective address
LDS	Load pointer using DS
LES	Load pointer using ES
	FLAG TRANSFER
LAHF	Load AH register from flags
SAHF	Store AH register in flags
PUSHF	Push flags onto stack
POPF	Pop flags off stack
	LOGICALS
NOT	"Not" byte or word
AND	"And" byte or word
OR	"Inclusive or" byte or word
XOR	"Exclusive or" byte or word
TEST	"Test" byte or word
	SHIFTS
SHL/SAL	Shift logical/arithmetic left byte or word
SHR	Shift logical right byte or word
SAR	Shift arithmetic right byte or word
	ROTATES
ROL	Rotate left byte or word
ROR	Rotate right byte or word

RCL	Rotate through carry left byte or word

RCR	Rotate through carry right byte or word

	ADDITION
ADD	Add byte or word
ADC	Add byte or word with carry
INC	Increment byte or word with carry
AAA	ASCII adjust for addition
DAA	Decimal adjust for addition
	SUBTRACTION
SUB	Subtract byte or word
SBB	Subtract byte or word with borrow
DEC	Decrement byte or word by 1
NEG	Negate byte or word
CMP	Compare byte or word
AAS	ASCII adjust for subtraction
DAS	Decimal adjust for subtraction
	MULTIPLICATION
MUL	Multiply byte or word unsigned
IMUL	Integer multiply byte or word
AAM	ASCII adjust for multiply
	DIVISION
DIV	Divide byte or word unsigned
IDIV	Integer divide byte or word
AAD	ASCII adjust for division
CBW	Convert word to double word
	STRING OPERATIONS
MOVS	Move byte or word string
MOVSB/MOVSW	Move byte or word string
CMPS	Compare byte or word string

SCAS	Scan byte word string
LODS	Load byte or word string
STOS	Store byte or word string
REPE/REPZ	Repeat while equal/zero
	FLAG OPERATIONS
STC	Set carry flag
CLC	Clear carry flag
CMC	Complement carry flag
STD	Set direction flag
CLD	Clear direction flag
STI	Set interrupt enable flag
CLI	Clear interrupt enable flag

	EXTERNAL SYNCHRONIZATION
HLT	Halt until interrupt or reset
WAIT	Wait for TEST pin active
ESC	Escape to external processor
LOCK	Lock bus during next instruction
	NO OPERATION
NOP	No operation

	ITERATION CONTROLS
LOOP	Loop
LOOPE/LOOPZ	Loop if equal/ zero
LOOPNE/LOOPNZ	Loop if not equal/ not zero
JCXZ	Jump if register CX=0
	INTERRUPT
INT	Interrupt
INTO	Interrupt if overflow
IRET	Interrupt return

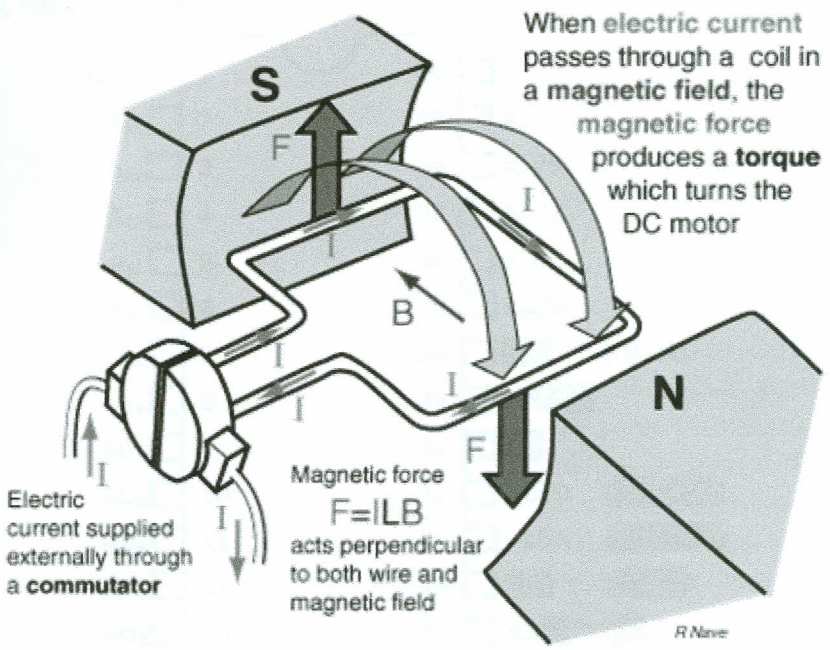
	UNCONDITIONAL TRANSFERS
CALL	Call procedure
RET	Return from procedure
JMP	Jump

	CONDITIONAL TRANSFERS
JA/JNBE	Jump if above/ not below nor equal
JAE/JNB	Jump if above or equal/not below
JB/JNAE	Jump if below/not above nor equal
JBE/JNA	Jump if below or equal/not above

JC	Jump if carry
JE/JZ	Jump if equal / zero
JG/JNLE	Jump if greater/ not less nor equal
JGE/JNL	Jump if greater or equal/ not less
JLE/JNG	Jump if less or equal/not greater
JNC	Jump if not carry
JNP/JPO	Jump if not parity/parity odd
JNS	Jump if not sign
JO	Jump if overflow
JP/JPE	Jump if parity/parity even
JS	Jump if sign
JL/JNGE	Jump if less/not greater nor equal
JNE/JNZ	Jump if not equal/ not zero

APPENDIX C

DC MOTOR OPERATION



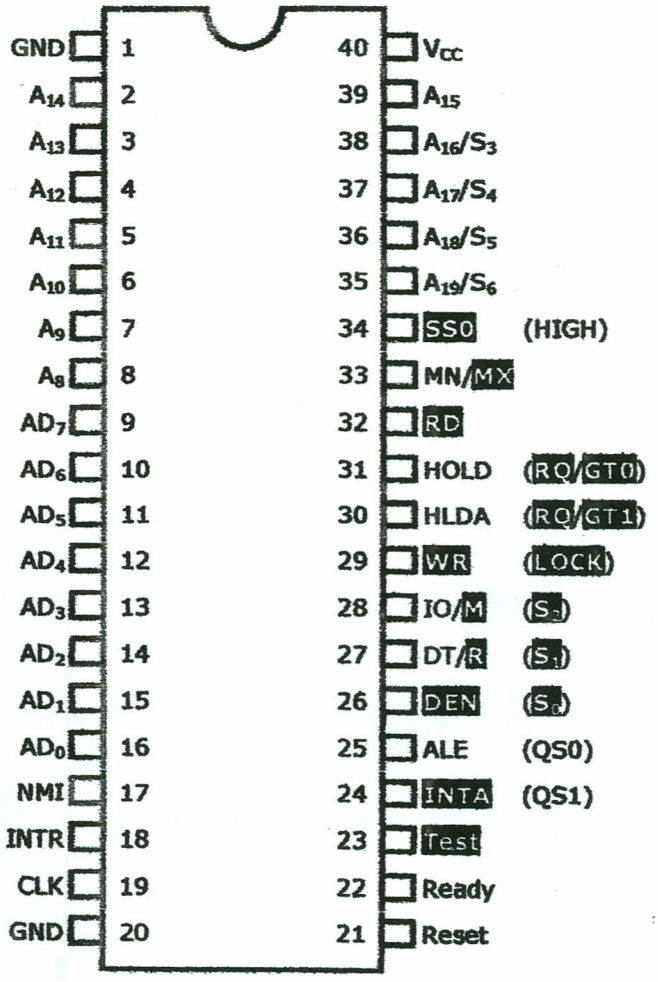
Magnetic interactions with charge

APPENDIX D

8088 MICROPROCESSOR PIN CONFIGURATION

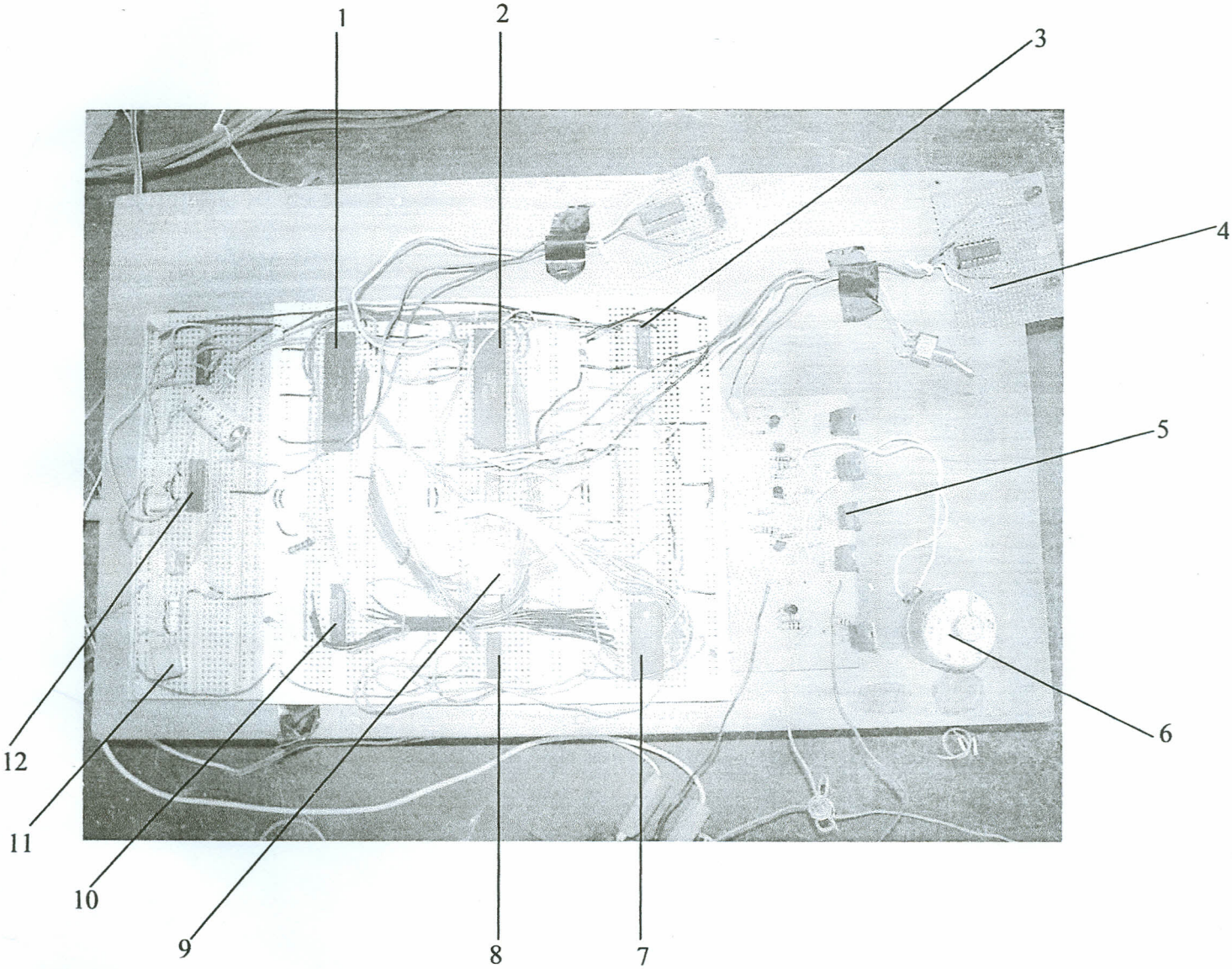
8088, 80C88 microprocessor - DIP 40 package

Manufacturers: AMD, Intel, NEC, Siemens



APPENDIX E

PHOTOGRAPH OF THE MICROPROCESSOR-BASED DC MOTOR CONTROLLER



1. 8088 Microprocessor
2. 8255A PPI
3. NOT gate
4. LEDs
5. MOSFETs
6. DC Motor
7. 6116 RAM
8. 74LS138 Decoder
9. 2716 EPROM
10. 74LS373 Lacth
11. Reset Switch
12. 8284A Clock Generator