

PROJECT REPORT

On

POWERLINE CARRIER COMMUNICATION BASED SPEED CONTROL OF SINGLE PHASE INDUCTION MOTOR

P-918



Estd. 1984

2002 – 2003



Submitted By

Arumugavel C,
Ganapathy Subramanian N,
Senthil Kumar K,
Suriya Prakash M,

Guided By

Mr.V.Kandasamy M.Tech.,
Lecturer, Dept. of EEE.

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

ELECTRICAL AND ELECTRONICS ENGINEERING

Bharathiar University, Coimbatore.

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY

COIMBATORE – 641 006.



Estd. 1984

DEPARTMENT OF ELECTRICAL AND ELECTRONICS
ENGINEERING
KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE – 641 006



ISO 9001:2000

CERTIFICATE

This is to certify that the Project Report entitled

**“POWERLINE CARRIER COMMUNICATION BASED SPEED
CONTROL OF SINGLE PHASE INDUCTION MOTOR”**

has been submitted by

ARUMUGAVEL C,	99EEE06
GANAPATHY SUBRAMANIAN N,	99EEE14
SENTHIL KUMAR K,	99EEE50
SURIYA PRAKASH M,	99EEE56

*in partial fulfillment of the requirements for the award of the degree of Bachelor
of Engineering in Electrical and Electronics branch of Bharathiar University,
Coimbatore during the academic year 2002 – 2003.*


(V. Kandasamy)
Project Guide


(Dr. T.M. Kameswaran)
Head of Department

*Certified that the candidate with University Register No. _____
was examined in project work Viva – Voce Examination on _____*


Internal Examiner


External Examiner

*DEDICATED TO OUR
BELOVED PARENTS*

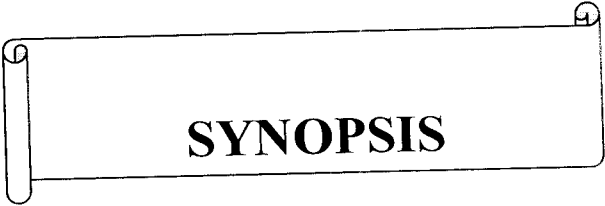
ACKNOWLEDGEMENT

We express our heartfelt gratitude to our principal Dr.K.K.Padmanabhan, B.Sc.,(Engg), M.Tech., Ph.D., F.I.E., for providing the necessary facilities for the successful completion of the project.

We are highly grateful to Dr.T.M.Kameswaran, B.E., M.Sc(Engg)., Ph.D., F.I.E., Head of the Department, Electrical and Electronics Engineering, for his enthusiasm and encouragement which has been instrumental in the success of the project.

We wish to express our deep sense of gratitude and indebtedness to our project guide Mr.V.Kandasamy, M.Tech., Lecturer, Department of Electrical and Electronics Engineering, for his invaluable guidance and constructive suggestions, the keen and constant inspiration and interest evinced by his at all stages right from the inception to the completion.

We are also indebted to the staff members of Electrical and Electronics Department for their co-operation. We are thankful to our student friends for their encouragement and help during the course of the project.



SYNOPSIS

SYNOPSIS

Various methods are adopted to control the speed of a motor. The control signals are generated by the power electronic devices that are triggered appropriately to achieve the necessary control.

The control over the motor was achieved, by accessing these control circuit at the place of their establishment. This project aims at achieving the same control from a remote place. To obtain the control from a distant place communication means such as radio communication (wireless or wired) can be had. This project aims at achieving the same with the concept of power line carrier communication (PLCC).

PLCC is a highly reliable communication scheme in which the data signal is modulated on carrier frequency and transmitted on a power line .the basic idea is couple the low voltage and high frequency carrier set to the high voltage and low frequency power line.

The ATMEL 89C51 microcontroller provides the actual control signal to the modulation circuit. In the carrier modulation circuit, the modulated signal is transmitted to the power line with use of appropriate coupling network.

The receiver side is also provided with the coupling network that possess strong band rejection properties, blocking the 240V, 50Hz main signal, but tapping the high frequency intelligent signal. The received signal is demodulated to obtain the original control signal. The microcontroller, with this signal as input, can be effectively be used to

any effective classical method of speed control of the motor considered, can be followed.

The range of communication depends on the strength of the signal transmitted.



CONTENTS

CONTENTS

	Pg. No
1. Introduction	1
2. Problem Statement	3
3. Methodology	5
3.1 Basic Block Diagram	6
3.1.1 Transmitter side Block Diagram	8
3.1.2 Receiver side Block Diagram	11
3.2 Description	
3.2.1 Concept of PLCC	15
3.2.2 Frequency Shift Keying	18
3.3 Advantages of PLCC	19
4. Implementation	20
4.1 Power Supply Unit	21
4.2 LCD Display	24
4.3 ATMEL 89C51 Microcontroller	25
ATMEL 89C2051 Microcontroller	30
4.4 PLCC Modem	33
4.5 PWM Technique	34
4.6 Proximity Sensor	37
5. Software Codings	38
6. Conclusion & Future Expansion	65
7. References	67
8. Appendix	69



INTRODUCTION

INTRODUCTION

The communication through wire system involves the use of a dedicated cables which increase the cost involved. The concept of PLCC enhances the mode of communication by having the power flow and the communication signal flow through the same network.

This project aims at realizing the concept of PLCC by applying it to a specific application, namely, the speed control of single phase induction motor. The project is divided in to three modules: Transmitter module, Receiver module and the speed control module.

The required speed of operation of the motor is set from a remote place using the transmitter module and also transmitted to the Powerlines using the same. The transmitted control signal can be tapped from the same Powerlines using the receiver module and the speed control can be achieved using the control circuit.



PROBLEM STATEMENT

PROBLEM STATEMENT

The main aim of the project is to realize the concept of Power Line Carrier Communication (PLCC) by controlling the speed of the single phase Induction motor. The control is achieved by applying the control signals from a remote place with the Power Mains as the communication medium.

In the conventional method, the user has to go near the motor to control the speed, which is tedious in case of motor at a distant place. To overcome this difficulty we have used Powerline Carrier Communication (PLCC) technology to send the control signals for controlling the speed of the motor from a convenient place. By this technique a long communication range can be established.

In the conventional method to control a motor at a distant place we require long cables and accessories. By the use of PLCC technique we make use of powerline cables to transmit the control signal hence laying of separate cables for the transmission of control signals are avoided, which in turn reduce the cost.



METHODOLOGY

3.1. BASIC BLOCK DIAGRAM

The Basic Block diagram of PLC based Motor Speed

Control is given below:

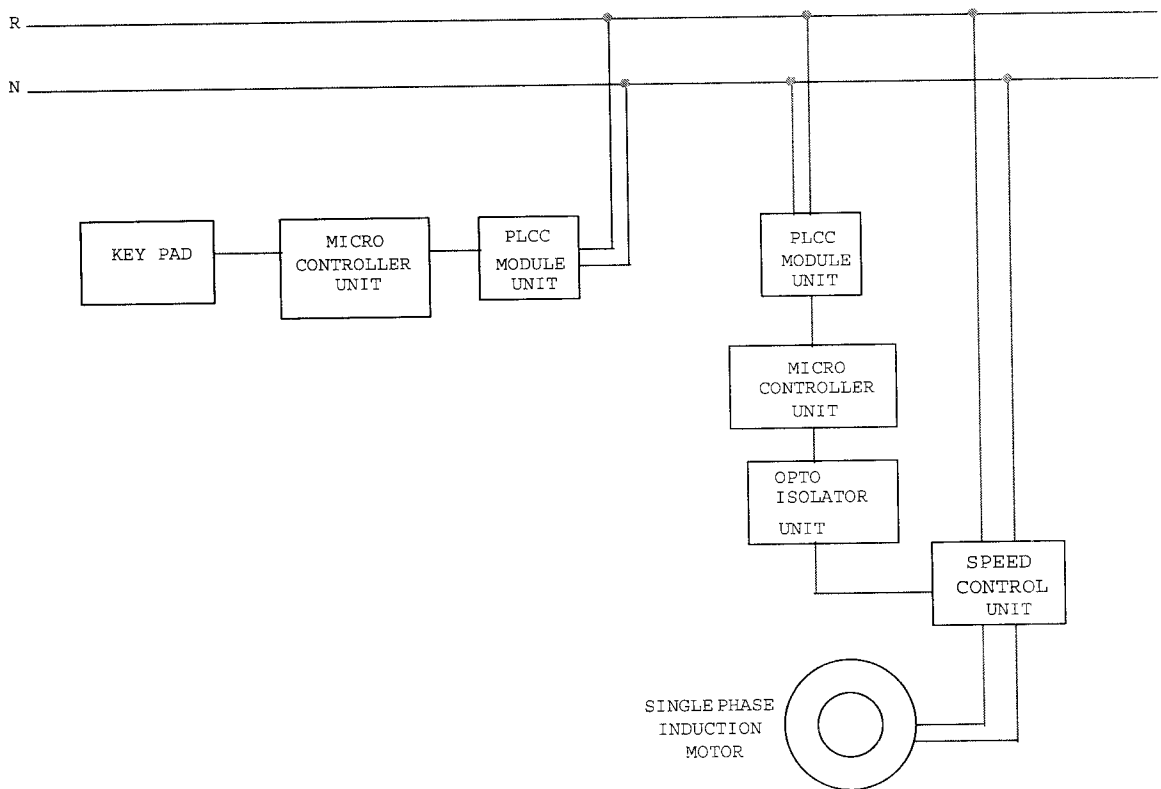


Figure 3.1 Basic Block Diagram of PLCC based speed control of single phase induction motor .

DESCRIPTION

The Figure 3.1 shows the basic block diagram of Powerline carrier communication based motor speed control of single-phase induction motor. The desired speed is sent to the microcontroller of the transmitter side through the keypad.

The data from the key pad is in the parallel form. This has to be converted into serial form before transmitting. The serialized data is then transmitted into the power line with the help of the power line carrier communication module.

A technique called “Frequency shift keying” is used for the PLC modulation. The transmitted data is received at the other end with the help of the power line carrier communication module. Provision is provided for displaying the speed transmitted and received by using seven segment displays.

The received data, which is obtained in the serial form, has to be converted into parallel form before being passed on to the triggering circuit. We are using pulse width modulation technique for controlling the speed of the Single phase induction motor. PWM is a technique in which the leading edge, the trailing edge or both may be varied as a function of the amplitude of the sampled signal.

3.1.1 TRANSMITTER SIDE BLOCK DIAGRAM

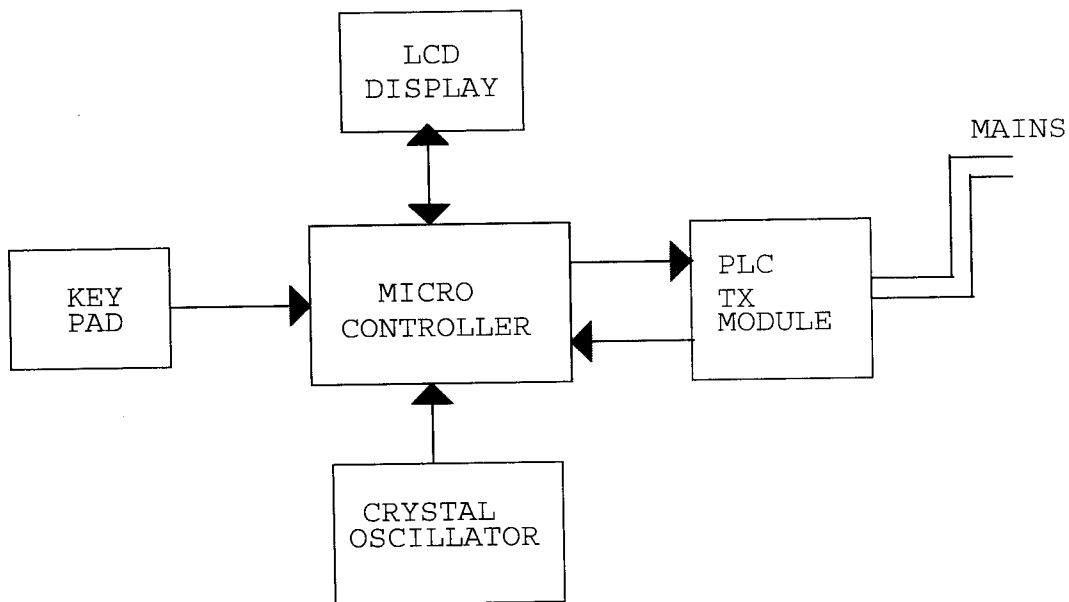


Figure 3.1.1(a) Block Diagram of the Transmitter side

DESCRIPTION

Figure 3.1.1(a) shows the block diagram of the transmitter side. The keypad and the LCD are interfaced with 89C51 microcontroller and the PLCC module is also interfaced with the same. A crystal oscillator is used to provide the microcontroller with the required clock pulses.

Keypad consists of three switches: MOV, INCR, SET. The 'MOV' switch is used to position the digit that is to be incremented. The 'INCR' switch is used to increment the positioned digit by one. The 'SET' switch is used to set the value of the speed required.

The required speed is set using the keypad which can be seen in the display. The microcontroller is programmed so as to provide the PLCC modem with the required digital data. PLCC modem in turn transmits the data to the power mains.

Figure 3.1.1(b) shows the circuit diagram of the above discussed transmitter side.

TRANSMITTER SIDE CIRCUIT DIAGRAM

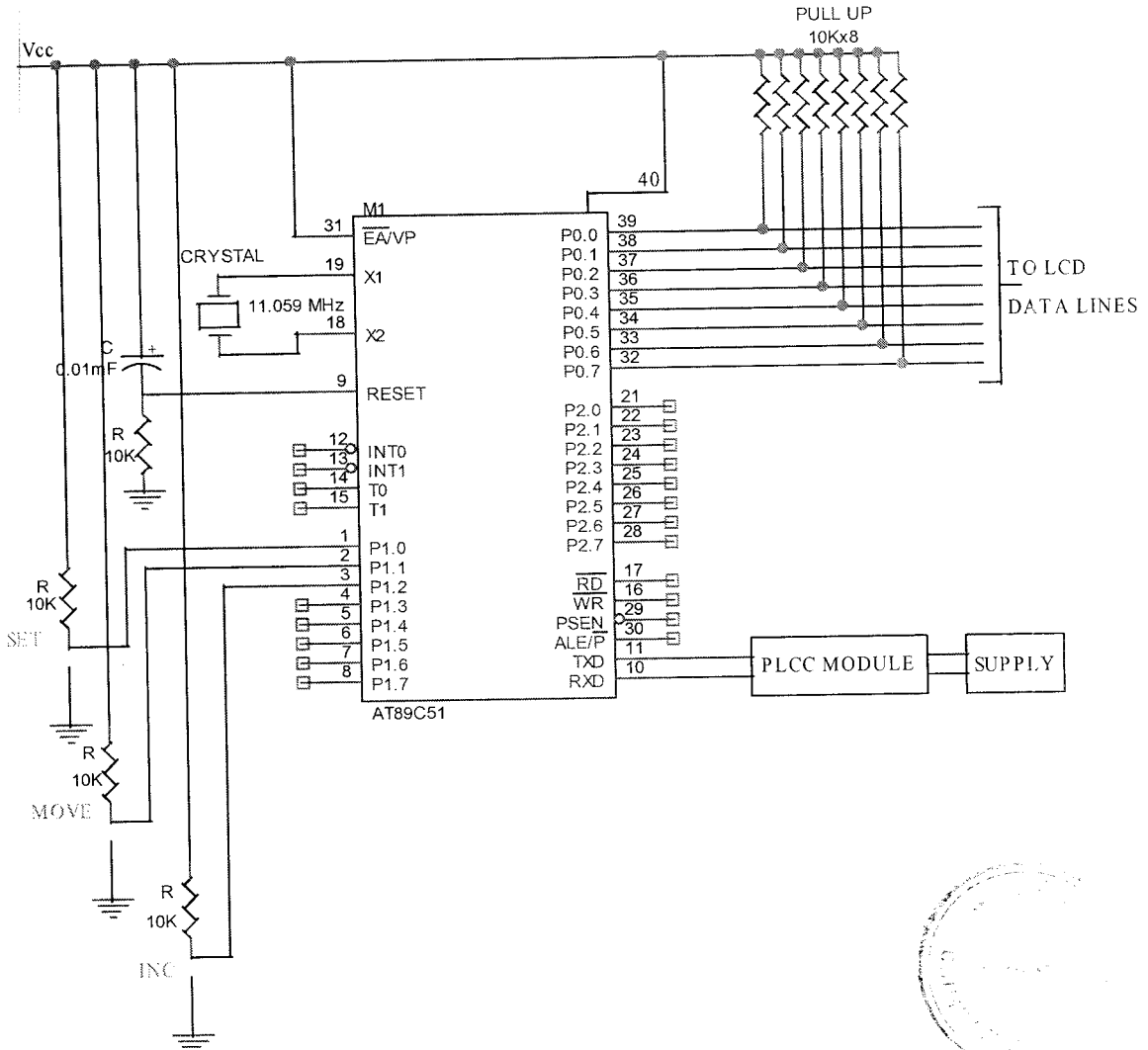


Figure 3.1.1(b) Circuit Diagram of Transmitter side

3.1.2 RECEIVER SIDE BLOCK DIAGRAM

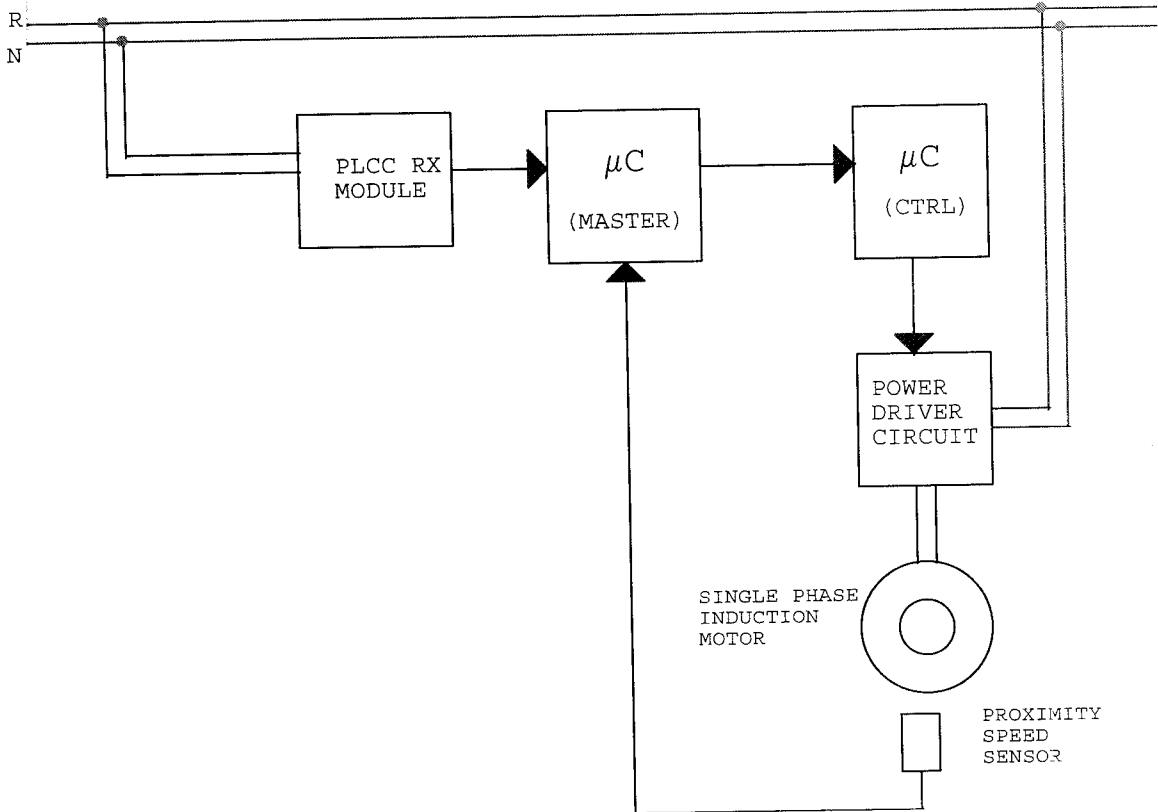


Figure 3.1.2(a) Block Diagram of the Receiver side

DESCRIPTION

Figure 3.1.2(a) shows the block diagram of the transmitter side. The PLCC modem receives the transmitted digital data from the Powerline and gives it to the 89C51 microcontroller. The PLCC modem also isolates the control circuit unit from the power mains.

Two microcontrollers are used in the receiver side. The Master microcontroller is interfaced with the PLCC modem so as to receive the data from the modem and send appropriate acknowledgment signals. An LCD display is also interfaced with the same microcontroller. The LCD display has two lines: the first line displays the value of speed entered in the transmitter side: the second line displays the value of the speed at which the motor runs.

Another 20 pin microcontroller is used which is fully dedicated for handling the speed control circuit. The control circuit and the proximity sensor are interfaced with this microcontroller. This microcontroller receives the set speed from the Master microcontroller and it also receives the actual speed from the proximity sensor. The microcontroller is programmed to send appropriate signals to the control circuit shown in Figure 3.1.3, using the inputs received. Figure 3.1.2(b) shows the circuit diagram of the receiver side.

RECEIVER SIDE CIRCUIT DIAGRAM

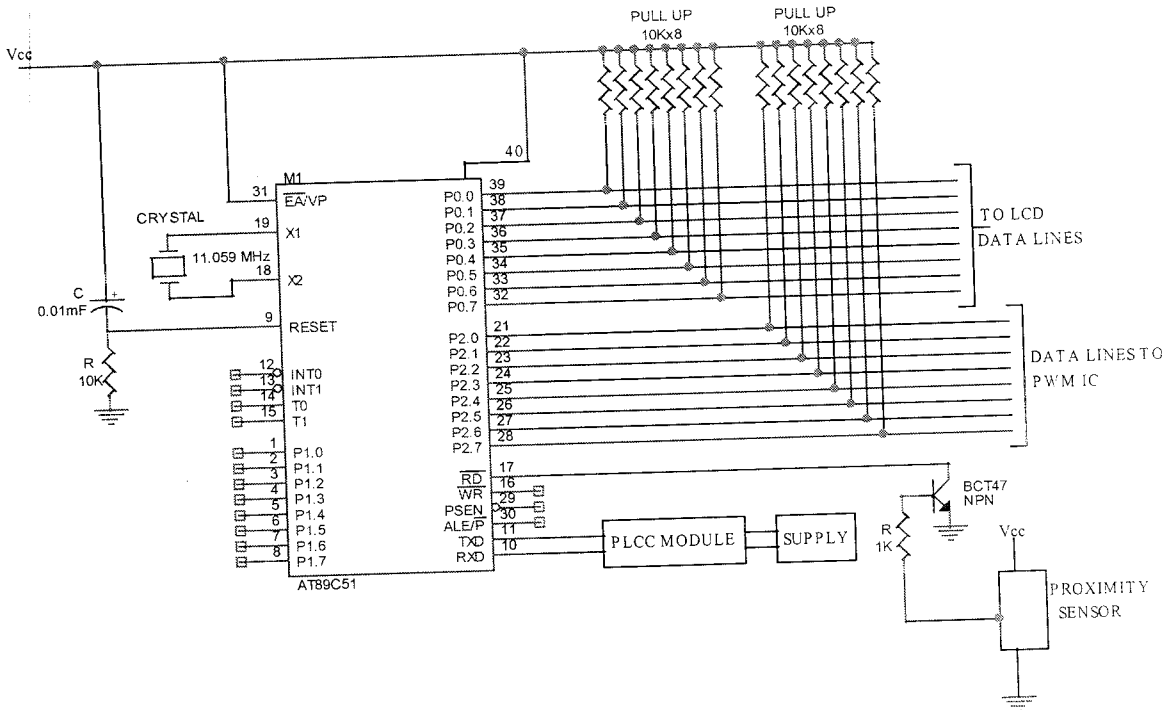


Figure 3.1.2(b) Circuit Diagram of the Receiver side

SPEED CONTROL CIRCUIT

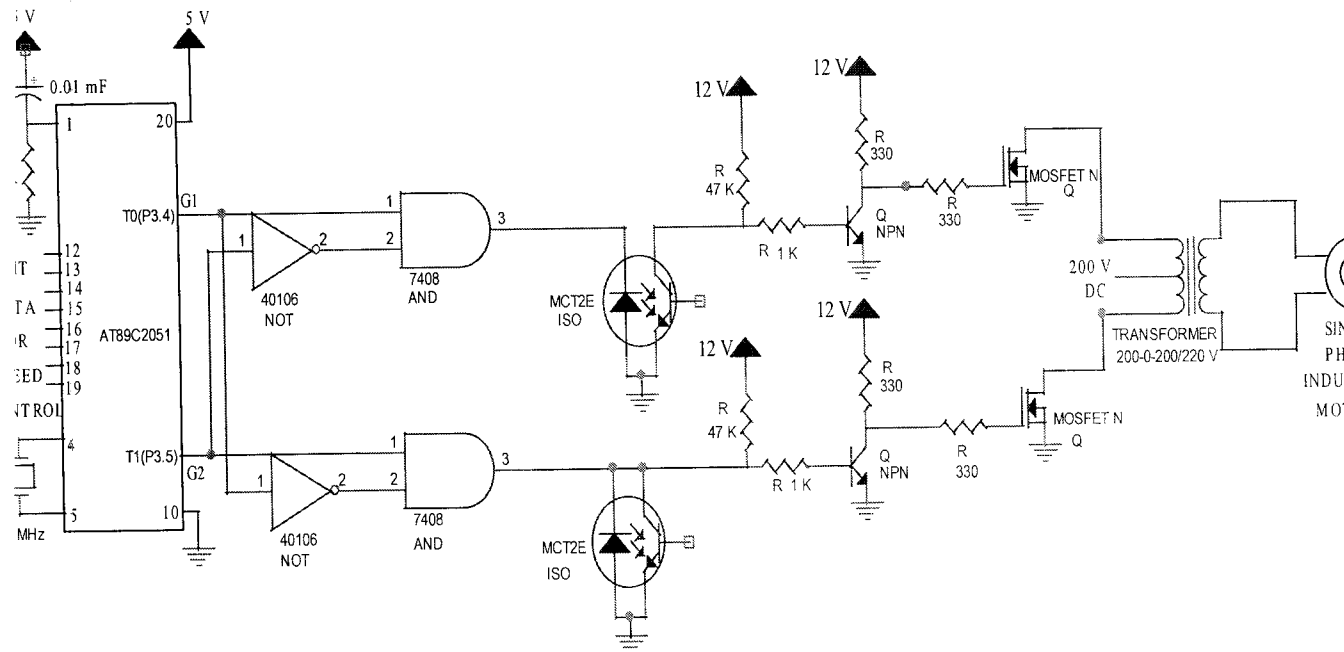


Figure 3.1.3 Circuit diagram of motor speed control

3.2.1 CONCEPT OF POWERLINE CARRIER COMMUNICATION:

Power lines and their associated networks are not designed for communications use. They are hostile environments that make the accurate propagation of communication signals difficult. Respectively, the problems are the numbers of users per transformer and the size and shape of light poles. Two of the biggest problems faced in using power lines for communications are excessive noise levels and cable attenuation. Noise levels are often excessive, and cable attenuation at the frequencies of interest is often very large. In home networking privacy and security of the data are important and is discussed in another section.

The most common causes of excessive noise in a domestic situation are the various household devices and office equipment connected to the network. Noise and disturbances on the power network include over voltages, under voltages, frequency variations and so on. However, the most harmful noise for PLCC applications is that superimposed on a power line. Switching devices such as light dimmers, induction motors in many common appliances and high-frequency noise caused by computer monitors and televisions often causes such superimposed noise.

For a power line carrier communications system to perform reliably it must be able to avoid, or cope with, the different types of noise encountered on its communications channel. These different types of noise exist at different frequencies, and occur at unpredictable times. Thus, it is not sufficient to design a system that simply avoids using certain parts of the available bandwidth. Rather, a technique called

agile", or to avoid noise when encountered. When a frequency-hopping communications system encounters noise at a certain bandwidth, it skips to a different bandwidth, moving away from the original interference. Frequency hopping is a spread-spectrum communications technique, meaning that the total available spectrum is split up into smaller sections so as to be better utilized effectively.

Signal attenuation in the power line environment is often great, and unpredictable. Attenuation has been measured at up to 100dB/km. Furthermore, it is very difficult to obtain a meaningful model of the PLCC channel, due to the time-variance of this channel. As devices are connected and disconnected from the power network, network characteristics change drastically. Unfortunately this makes coupling a signal to the power network difficult. Any coupling device must be adaptable to obtain reasonable performance in all expected situations.

A unique challenge of power line carrier communication systems is the method used to couple the communications signal onto the power network. In the receive direction we wish this coupling network to possess strong band-reject properties, blocking the 240V, 50 Hz mains signal, but passing the high-frequency communications signal unattenuated. In the transmit direction we wish the coupling network to have wide-pass properties, passing the transmitted signal unattenuated. For a single device to possess such properties, some compromise between the two is necessary. Also, for low attenuation we wish the coupling network to be approximately impedance-matched to the power line. The general approach is to use a coupling transformer to isolate the mains from the communications system. The impedance of this

"tank circuit" of suitable frequency response. Coupling network performance can be substantially improved by using a double-secondary transformer. With two secondary, two separate circuits can be designed, one with the required transmit response and the other the required receive response.

3.2.2 FREQUENCY SHIFT KEYING

In digital data communication, binary code is transmitted by shifting a carrier frequency between two preset frequencies. This type of transmission is called frequency shift keying technique. A 555 timer in astable mode can be used to generate FSK signals. When the input is high, transistor Q is cutoff and 555 timer works in the normal astable mode of operation. The frequency of the output waveform is given by

$$F_0 = 1.45 / (R_A + 2R_B) C$$

When the input is low Q goes on and connects the resistance R_C across R_A. The output frequency is given by

$$F_0 = 1.45 / ((R_A \parallel R_C) + 2R_B) C$$

The resistance is used to get the desired output frequency.

Frequency shift keying (FSK) is the mostly used method for transmitting digital data over telecommunications links. In order to use FSK a modulator-demodulator (modem) is needed to translate digital 1's and 0's into their respective frequencies and back again.

In FSK modulation, the carrier frequency is shifted in steps (or) levels corresponding to the levels of the digital modulating signal. In case of binary signal, two carrier frequencies are used, one was corresponding to binary '0' and another to binary '1'.

3.3.ADVANTAGES

Communication through power lines is having following advantages.

- The electric grids are modern, well maintained, and far superior to any of the wired communications networks.
- There are more electrical customers than telephone, cable, and other wired communications customers together.
- The backbone of the Internet as well as the long distance telephone lines are all fiber optic cables which can carry a signal for only 20 miles, without regeneration, coaxial cable 15 miles and copper telephone wire 5 miles. Signals over the power grid can travel more than 2000 miles without regeneration.
- The analog spread spectrum waves have much greater bandwidths or carrying capacity than the digital switched systems.
- Another inherent advantage to the Digital Power line model is the fact that it works well over the existing electric power infrastructure.

IMPLEMENTATION

4.1 POWER SUPPLY UNIT

The project was done in three modules, namely, the transmission module, the receiving module and the speed control module. Each module requires a power supply circuit for its operation. The required power supply circuits were designed and they are discussed below:

Figure 4.1.(a) & 4.1(b) shows the Power Supply Diagrams of the Transmitter and Receiver side respectively. Power Supply unit Comprises of two circuits, one providing a +5V supply and the other providing +12V and -12V supply. This unit consists of transformers, rectifiers, filters and regulators. The input A.C. voltage, typically 230V Rms is connected to a transformer to obtain the desired step- down AC voltage. A diode rectifier then provides a full-wave rectified voltage that is initially filtered by a simple capacitor filter to produce a DC voltage. This resulting DC voltage usually has some ripple or AC voltage variations. A regulator circuit can use this DC input to provide DC voltage that not only has much less ripple voltage but also remains the same DC value even if the DC voltage varies, or when the load connected to the output DC voltage changes.

RECTIFIER

The DC level obtained from a sinusoidal input can be improved 100% using a process called full-wave rectification. It uses 4 diodes in a bridge configuration. From the basic bridge configuration we see that two diodes (say D2 & D3) are conducting while the other two diodes (D1 & D4) are in “off” state during the period $t = 0$ to $T/2$.

TRANSMITTER SIDE POWER SUPPLY

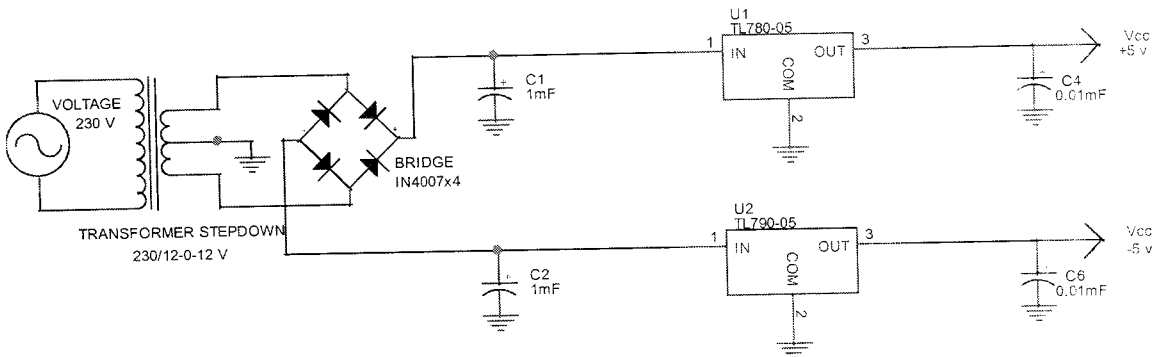


Figure 4.1.(a) Transmitter side Power Supply Diagram

RECEIVER SIDE POWER SUPPLY

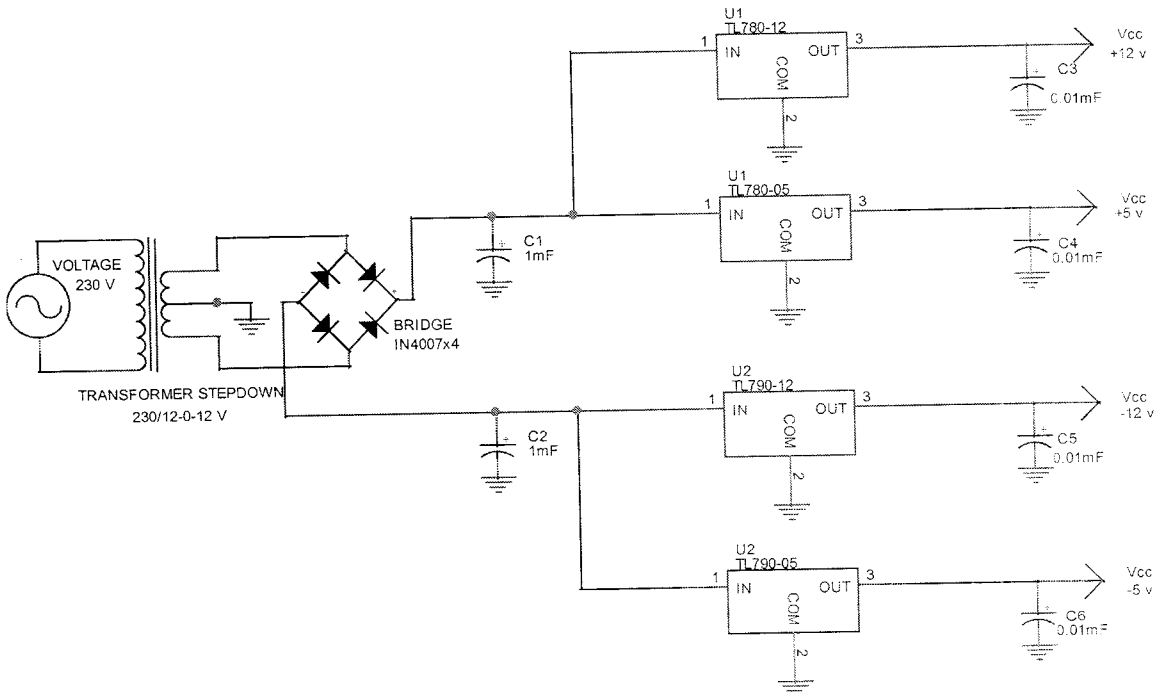


Figure 4.1.(b) Receiver side Power Supply Diagram

Accordingly for the negative half cycle of the input the conducting diodes are D1 & D4. Thus the polarity across the load is the same.

FILTER

The filter circuit used here is the capacitor filter circuit where a capacitor is connected at the rectifier output, and a DC is obtained across it. The filtered waveform is essentially a DC voltage with negligible ripples, which is ultimately fed to the load.

REGULATORS

The voltage regulator is a device, which maintains the output voltage constant irrespective of the change in supply variations, load variation and temperature changes. The fixed voltage regulator is a three terminal device which has an unregulated dc input voltage, V_i , applied to one input terminal, a regulated output dc voltage, V_o , from a second terminal, with the third terminal connected to ground. For a selected regulator, IC device specifications list a voltage range over which the input voltage can vary to maintain a regulated output voltage over a range of load current. The specifications also list the amount of output voltage change resulting from a change in load current or in input voltage.

4.2 LCD DISPLAY

In our project, the readings obtained are shown on an LCD display. An LCD consists of two glass panels, with the liquid crystal material sandwiched between them. The inner surface of the glass plates are coated with transparent electrodes which define the character, symbols or patterns to be displayed. Polymeric layers are present in between the electrodes and the liquid crystal, which makes the liquid crystal molecules to maintain a defined orientation angle.

Polarizers are pasted outside the two glass panels. These polarizers would rotate the light rays passing through them to a definite angle, in a particular direction. When the LCD is in the off state, light rays are rotated by the two polarizers and the liquid crystal, such that the light rays come out of the LCD without any orientation, and hence the LCD appears transparent.

When sufficient voltage is applied to the electrodes, the liquid crystal molecules would be aligned in a specific direction. The light rays passing through the LCD would be rotated by the polarizers, which would result in activating / highlighting the desired characters.

4.3 ATMEL 89C51

FEATURES:

- Compatible with MCS-51™ Products
- 4K Bytes of In-System Reprogrammable Flash Memory
 - Endurance: 1,000 Write/Erase Cycles
- Fully Static Operation: 0 Hz to 24 MHz
- Three-Level Program Memory Lock
- 128 x 8-Bit Internal RAM
- 32 Programmable I/O Lines
- Two 16-Bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial Channel
- Low Power Idle and Power Down Modes

DESCRIPTION:

The AT89C51 is a low-power, high-performance CMOS 8-bit microcomputer with 4K bytes of Flash Programmable and Erasable Read Only Memory (PEROM). The device is manufactured using Atmel's high density nonvolatile memory technology and is compatible with the industry standard MCS-51™ instruction set and pinout. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C51 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications.

AT89C51

The AT89C51 provides the following standard features: 4K bytes of Flash, 128 bytes of RAM, 32 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator and clock circuitry. In addition, the AT89C51 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The Power Down Mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

PIN DESCRIPTION:

VCC

Supply voltage.

GND

Ground.

Port 0

Port 0 is an 8-bit open drain bidirectional I/O port. As an output port each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high impedance inputs. Port 0 may also be configured to be the multiplexed low order address/data bus during accesses to external program and data memory. In this mode P0 has internal pullups. Port 0 also receives the code bytes during Flash programming, and outputs the code bytes during program verification. External pullups are required during program verification.

Port 1

Port 1 is an 8-bit bidirectional I/O port with internal pullups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (IIL) because of the internal pullups. Port 1 also receives the low-order address bytes during Flash programming and verification.

Port 2

Port 2 is an 8-bit bidirectional I/O port with internal pullups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (IIL) because of the internal pullups. Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @DPTR). In this application it uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3

Port 3 is an 8-bit bidirectional I/O port with internal pullups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (IIL) because of the pullups. Port 3 also serves the

Port 3 also receives some control signals for Flash programming and verification.

RST

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device.

ALE/PROG

Address Latch Enable output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input (PROG) during Flash programming. In normal operation ALE is emitted at a constant rate of 1/6 the oscillator frequency, and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external Data Memory. If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

PSEN

Program Store Enable is the read strobe to external program memory.

Port Pin Alternate Functions

P3.0 RXD (serial input port)

P3.1 TXD (serial output port)

P3.2 INT0 (external interrupt 0)

P3.3 INT1 (external interrupt 1)

P3.4 T0 (timer 0 external input)

P3.5 T1 (timer 1 external input)

P3.6 WR (external data memory write strobe)

When the AT89C51 is executing code from external program memory, PSEN is activated twice each machine cycle, except that two PSEN activations are skipped during each access to external data memory.

EA/VPP

External Access Enable. EA must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, EA will be internally latched on reset. EA should be strapped to VCC for internal program executions. This pin also receives the 12-volt programming enable voltage (VPP) during Flash programming, for parts that require 12-volt VPP.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier.

ATMEL 89C2051

FEATURES

- Compatible with MCS-51™ Products
- 2K Bytes of Reprogrammable Flash Memory
 - Endurance: 1,000 Write/Erase Cycles
- 2.7V to 6V Operating Range
- Fully Static Operation: 0 Hz to 24 MHz
- Two-level Program Memory Lock
- 128 x 8-bit Internal RAM
- 15 Programmable I/O Lines
- Two 16-bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial UART Channel
- Direct LED Drive Outputs
- On-chip Analog Comparator
- Low-power Idle and Power-down Modes

DESCRIPTION

The AT89C2051 is a low-voltage, high-performance CMOS 8-bit microcomputer with 2K bytes of Flash programmable and erasable read only memory (PEROM). The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard MCS-51 instruction set. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C2051 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications. The AT89C2051 provides the following standard features: 2K bytes of Flash, 128 bytes of RAM, 15 I/O lines, two 16-bit timer/counters, a five vector

analog comparator, on-chip oscillator and clock circuitry. In addition, the AT89C2051 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The power-down mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

PIN DESCRIPTION:

VCC

Supply voltage.

GND

Ground.

Port 1

Port 1 is an 8-bit bi-directional I/O port. Port pins P1.2 to P1.7 provide internal pullups. P1.0 and P1.1 require external pullups. P1.0 and P1.1 also serve as the positive input (AIN0) and the negative input (AIN1), respectively, of the on-chip precision analog comparator. The Port 1 output buffers can sink 20 mA and can drive LED displays directly. When 1s are written to Port 1 pins, they can be used as inputs. When pins P1.2 to P1.7 are used as inputs and are externally pulled low, they will source current (IIL) because of the internal pullups. Port 1 also receives code data during Flash programming and verification.

Port 3

Port 3 pins P3.0 to P3.5, P3.7 are seven bi-directional I/O pins with internal pullups. P3.6 is hard-wired as an input to the output of the on-chip comparator and is not accessible as a general purpose I/O pin. The Port 3 output buffers can sink 20 mA. When 1s are written to Port 3

inputs. As inputs, Port 3 pins that are externally being pulled low will source current (IIL) because of the pullups. Port 3 also serves the functions of various special features of the AT89C2051 as listed below:

Port Pin Alternate Functions

P3.0 RXD (serial input port)

P3.1 TXD (serial output port)

P3.2 INT0 (external interrupt 0)

P3.3 INT1 (external interrupt 1)

P3.4 T0 (timer 0 external input)

P3.5 T1 (timer 1 external input)

RST

Reset input. All I/O pins are reset to 1s as soon as RST goes high. Holding the RST pin high for two machine cycles while the oscillator is running resets the device. Each machine cycle takes 12 oscillator or clock cycles.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier.

4.4.PLCC MODEM

The new ATL90 series Embedded PLC modem series is based on the Direct Sequence Spread Spectrum Technology. The new technology in Power Line Carrier (PLC) communication is well known for its high immunity to electrical noise persistent in the power line. With the new solution, the form factor of the PLC modem is further reduced and its cost lowered.

The Embedded PLC Modem is in the form of a ready-to-go circuit module, which is capable of transferring data over the power cable at the low voltage end of the power transformer of a 3-phase/ 4-wire distribution network. A pair of Embedded PLC Modems connected on the power line can provide low speed bi-directional data communication at a baud rate of 300/600 bps. It is built in a small form factor that can be easily integrated into and become part of the user's power line data communication system.

The modules provide bi-directional half-duplex data communication over the mains of any voltage up to 250v a. c., and for frequency of 50 or 60 Hz. Data communication of the modules is transparent to user's data terminals and protocol independent; as a result, multiple units can be connected to the mains without affecting the operation of the others. The use of DSSS modulation technique ensures high noise immunity and reliable data communication. There is no hassle of building interface circuits. Interface to user's data devices is a simple data-in and data-out serial link. It has a built-in on board AC coupling circuit, which allows direct and simple connection to the mains.

4.5. PWM TECHNIQUE

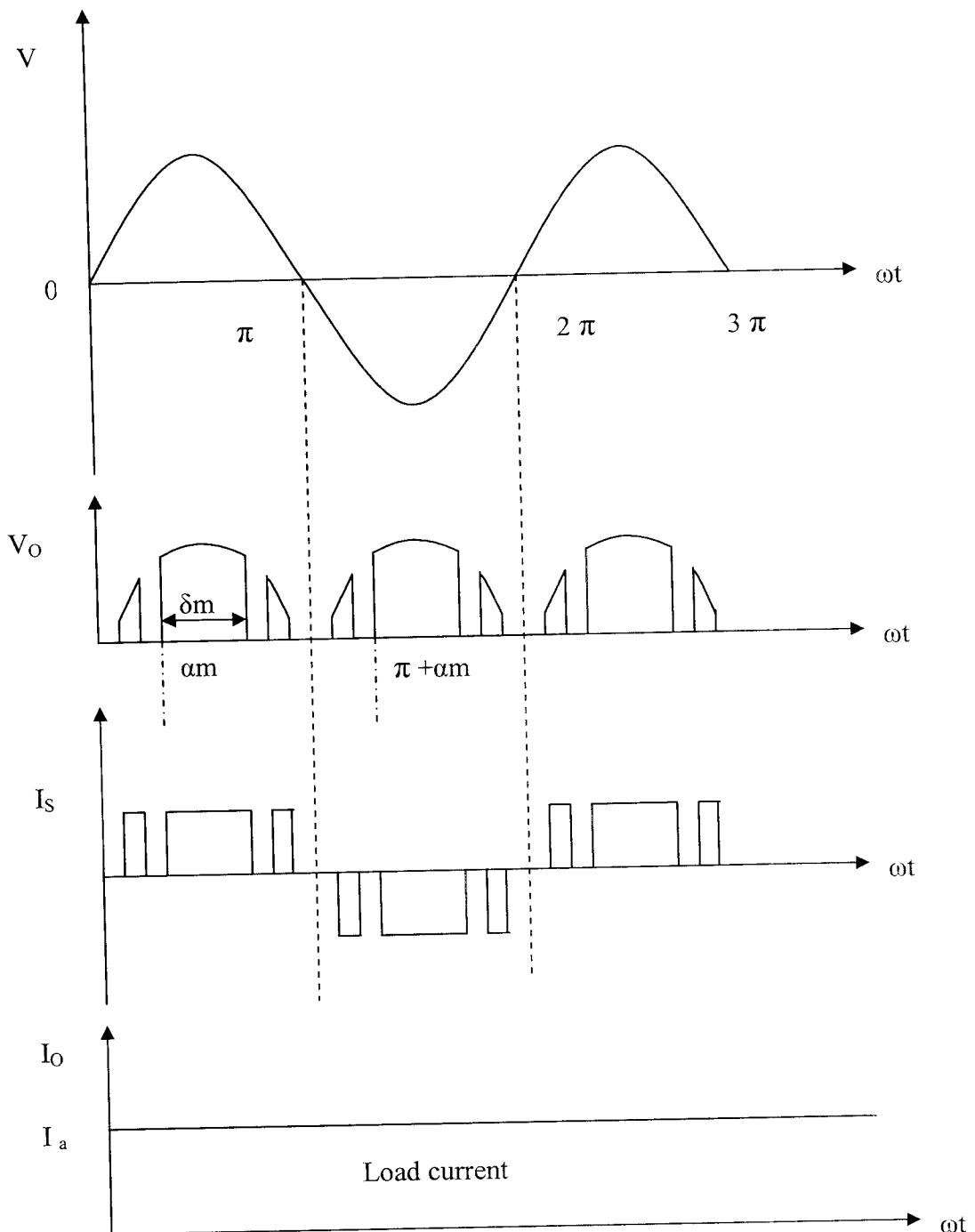
PWM is a technique in which the leading edge, the trailing edge or both may be varied as a function of the amplitude of the sampled signal. In general, PWM requires a greater average power than PAM (Pulse Amplitude Modulation) systems. Also the PWM system requires a large bandwidth than PAM.

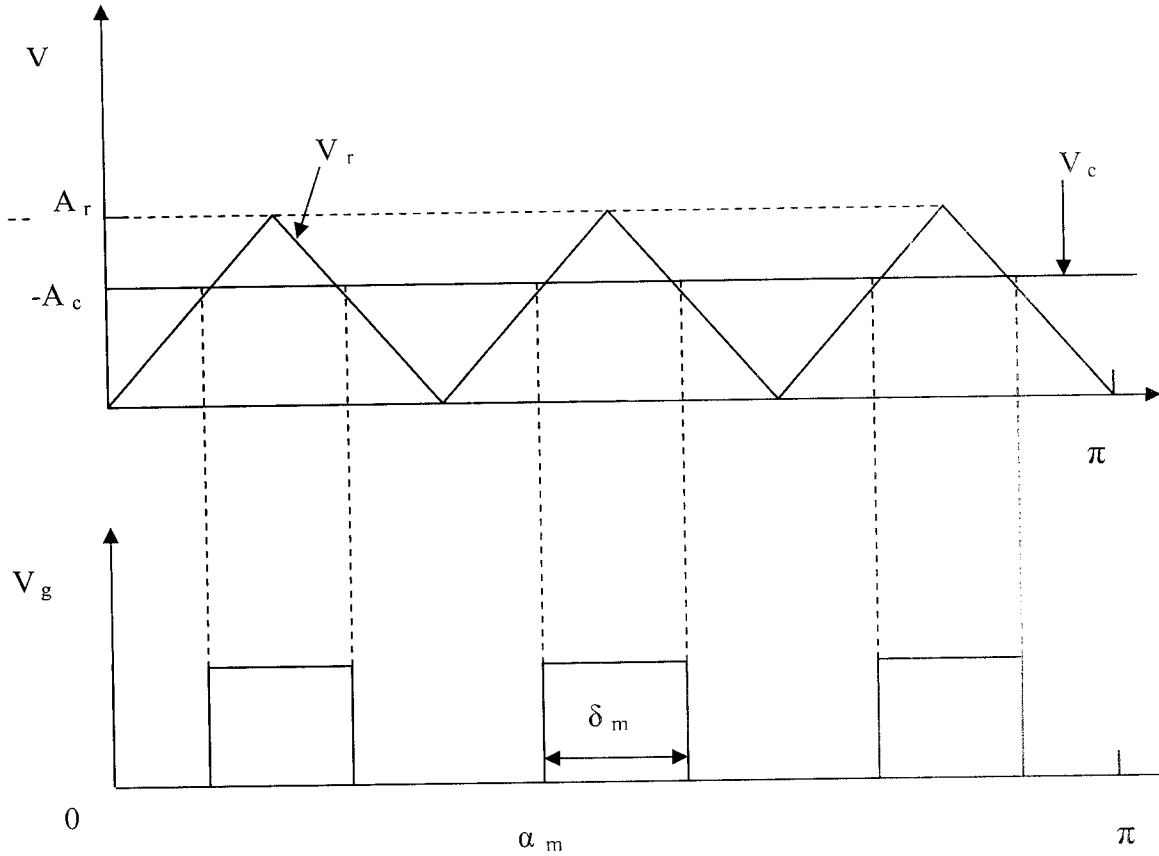
Voltage control in the square wave inverter has been external to the inverter, by means of a phase controlled rectifier on the line side. This posed some practical application problems on the drive by limiting the lowest operating frequency and introducing torque pulsations and harmonic heating. The problems can be reduced to a minimum if voltage control is obtained in the inverter itself.

The inverter is supplied with constant dc voltage and the inverter is controlled so that the average amplitude of the output voltage is variable. By this the operation of the inverter can be extended up to zero frequency, as the commutation is effective at all frequencies. By this control the output voltage is no longer a square wave but a pulsed wave, the average of which tends to be sinusoidal if sinusoidal modulation is used. The stator current also tends to be sinusoidal.

In PWM control, the converter switches are turned on and off several times during a half cycle and the output voltage is controlled by varying the width of the pulses. The gate signals are generated by comparing a triangular wave with a dc signal.

PULSE-WIDTH MODULATION CONTROL





The output voltage and performance parameters of the converter can be determined in two steps (1) by considering only one pair of pulses such that if one pulse starts at $\omega t = \alpha_1$ and ends at $\omega t = \alpha_1 + \delta_1$, the other pulse starts at $\omega t = \pi + \alpha_1$ and ends at $\omega t = (\pi + \alpha_1 + \delta_1)$, and (2) by combining the effects of all pairs. If the m th pulse starts at $\omega t = \alpha_m$ and its width is δ_m , the average output voltage due to p number of pulses is

$$V_{dc} = \Sigma [\cos \alpha_m - \cos(\alpha_m + \delta_m)]$$

4.6. PROXIMITY SENSOR

The proximity switches have been used as a transducer to convert the mechanical rotational energy into electrical energy. Here NPN type inductive proximity switch is used.

The proximity switch is a non-contact limit switch and is also called as eddy current oscillator.

ADVANTAGES:

- Moisture environment does not affect its operation.
- Since this project is used in the industries, dirty environment due to lubrication of electric drives does not affect its operation.
- Lighting condition in the environment does not affect its operation.

PULSE PRODUCTION:

Conversion of rotational energy to electrical energy is achieved by placing the pickup in nearly opposite to the rotating disk, which has two teeth. The two pulses are produced per one revolution of the shaft. Then the speed is calculated with the formula,

$$\text{SPEED} = \frac{(\text{NUMBER OF PULSES/SEC}) * 60}{\text{NUMBER OF TEETHS}} \text{ rpm}$$



**SOFTWARE
CODINGS**

Software coding were written in 'C' Programming language for all the three microcontrollers used and the 'C' Coding were converted into 89C51 Assembly language using the RIDE Software. The coding for the three modules are given below:

5.1. TRANSMITTER SIDE OF SPEED CONTROL

```
#include "reg51.h"
#include "sec-lcd.c"
#define TV0 0xfc//f6//TIMER 1 INTERRUPT CALL
#define TVL0 0x17 //3b
void delay();
void put_disp_msg();
void call_1sec();
void put_tra_msg();
void put_disp1_msg();
void call_unit();
void motor_on_sw();
void dly1();
void sepr();
void delay1();
void skey();
void bin_bcd(unsigned int g);
void bin_bcd3(unsigned int v);
void rate_conv();
void check_master();
void check_rate();
void conversion();
void incr();
void inc(unsigned char z);
double bcd_bin();
void delay3();
void set();
void move();
bit motor_flag;
bit fault_flag;
bit accept_flag;
bit on_flag;
bit tra_flag;
bit rate_flag;
bit reset_flag;
bit flagt;
bit flagr;
bit fini_flag;
bit timer_flag;
```



```

bit set_flag;
bit menu_flag;
bit inc_flag;
sbit menu_sw=P20;
sbit set_sw=P21;
sbit mov_sw=P22;
sbit inc_sw=P23;
sbit freq=P24;
unsigned char idata r,la,p8,digit5,digit1,digit2,digit3,digit4,rec_reg,f1,t1;
unsigned char l_year1,h_year1,l_rate1,h_rate1,month1,counter,fault_reg;
unsigned int idata
misc,units,l_units1,h_units1,dunit,check1,check2,check3,check4,rate1,year1;
unsigned int dum_reg2,t,set_rpm,y1,y2,y3,y4,y5,y6,y7,y8;
main()
{
    call_1sec();
    init_lcd();
    PCON=0x0; //intialise smod in pcon register to low
    stop_flag=0;
    IE=0x1a; //interrupt enable for serial and timer1
    IP=0x10; //interrupt priority for serial
    TMOD=0x21; //timer1 in auto reload mode
    SCON=0x50; //intialise scon as 8 bit uart mode with reception being enable
    TL1=0xd0; //
    TH1=0xd0; // timer1 value for serial interrupt @ 11.09mhz baud rate1200
    TCON=0x40; //enabling timer1
    TI=0; //clearing ti bit in scon register
    RI=0; //clearing ri bit in scon register
    flagt=0; //bit addressable flags
    flagr=0;
    rec_reg=0; //ram register
    tra_flag=0;
    rate_flag=0;
    fini_flag=0;
    TL0=TVL0;
    TH0=TV0;
    TR1=1;
    TR0=1;
    timer_flag=0;
    time_out_flag=0;
    set_flag=0;
    EA=1; // enables all interrupts
    fill_blank();
    put_lcd_download_1line();
    put_lcd_download_2line();
    put_disp_msg();
    put_lcd_download_1line();
    while(1)

```

```

}
if(fini_flag==0)
{
    if(tra_flag==0)
    {
        check_master();
    }
    if(tra_flag==1)
    {
        tra_flag=0;
        fini_flag=0;
        tra_flag=0;
        put_tra_msg();
        put_lcd_download_1line();
        delay();
        delay();
        delay();
        delay();
        delay();
        delay();
        delay();
        delay();
        delay();
        fill_blank();
        put_disp_msg();
        put_lcd_download_1line();
    }
}
}
void put_disp_msg()
{
    disp[1]='S';
    disp[2]='P';
    disp[3]='E';
    disp[4]='E';
    disp[5]='D';
    disp[6]=' ';
    disp[7]='C';
    disp[8]='O';
    disp[9]='N';
    disp[10]='T';
    disp[11]='R';
    disp[12]='O';
    disp[13]='L';
    disp[14]=' ';
    disp[15]=' ';
    disp[16]=' ';
}
void put_tra_msg()

```

```

disp[2]='r';
disp[3]='a';
disp[4]='n';
disp[5]='s';
disp[6]='m';
disp[7]='i';
disp[8]='t';
disp[9]='t';
disp[10]='e';
disp[11]='d';
disp[12]=' ';
disp[13]=' ';
disp[14]=' ';
disp[15]=' ';
disp[16]=' ';
}
void put_disp1_msg()
{
disp[1]='N';
disp[2]='O';
disp[3]=' ';
disp[4]='O';
disp[5]='F';
disp[6]=' ';
disp[7]='U';
disp[8]='N';
disp[9]='T';
disp[10]='T';
disp[11]='=';
}
void dly1()
{ //unsigned char fl;
for (fl=1;fl<=3;)
{fl++;}
}
serint ()interrupt 4 //serial vector address
{
switch (TI)//checks whether trasmission is over
{
case 1:
{TI=0; //clears ti flag
flagt=1;//sets flagt
break;
}
case 0:
{
switch (RI)//checks whether reception is over
{

```

```

    case 1:
        {rec_reg=SBUF; //received data is fed to rec_reg from sbuf
          RI=0; //clears ri flag
          flagr=1;/
); //hour should not be greater than 23hrs if(digit5 >9)
  //in case of digit 6 is 2 digit5=0;
disp[z]=(digit5+0x30);
break;}
}
inc_flag=0;
send_cursor();
}
}
// SET SWITCH FUNCTION
void set()
{
set_flag=0;
if(set_sw==0) // check for key press
{ delay3();
  delay3();
  if(set_sw==0) //verify for key press
  { do{delay3();}
    while(set_sw==0); //wait until key is released
    set_flag=1; // set the flag
  }
}
}
void menu()
{
menu_flag=0;
if(menu_sw==0) // check for key press
se if(cursor_pos==9)
cursor_pos=5;
send_cursor();
}
}

```

5.2. RECEVIER SIDE CODING

```

#include <stdio.h>
#include <reg51.h>
#include "sec-lcd.c"
#define TV0 0xfc //TIMER O INTERRUPT CALL
#define TVL0 0x17
#define PULSE_WIDTH P1

```

```

void bin_bcd(unsigned int g):

```

```
void dly1sec1();
void comm_fun();
void ok_msg();
void error_msg();
void save_data();
void delay6();
void skey();
void set();
void start();
void inc();
void move();
void delays();
void call_eeprom();
void delay();
void fuc1();
void call_1sec();
void esc_sw();
void rate_fun();
void compare();
void rpmspeed();
void put_set_msg();
void ldelay();
```

```
bit pm1;
bit set_flag;
bit inc_flag;
bit start_flag;
bit timer_flag;
bit time_out_flag;
bit flagt;
bit flagr;
bit stop_flag;
bit esc_flag;
bit trans_flag;
bit rpm_flag;
bit count_flag;
bit flg_ovr;
bit freq_flag;
```

```
sbit set_sw=P24;
sbit mov_sw=P23;
sbit freq=P24;
sbit comp_led=P22;
```

```
sbit led=P35;
sbit mode_sw=P36;
sbit pulse =P37;
```

```

unsigned char idata digit1,digit2,digit3,digit4,digit5,rec_reg;

unsigned char n1,m1,fault1,h_units1,l_units1,counter,dump_reg,l_rate1,h_rate1;
unsigned int jy,t,misec,misec1,dum_reg,r,initial_speed,set_rpm,pulse_count;
unsigned int y1,y2,y3,y4,y5,y6,y7,y8,tw;
main()

{
init_lcd();
PCON=0x0; //initialise smod in pcon register to low

IE=0x1a;//interrupt enable for serial and timer1
IP=0x10; //interrupt priority for serial
TMOD=0x21;//timer1 in auto reload mode
SCON=0x50;//initialise scon as 8 bit uart mode with reception being enable
TL1=0xd0; //
TH1=0xd0; // timer1 value for serial interrupt @ 11.09mhz baud rate1200
TCON=0x40;//enabling timer1
TI=0;//clearing ti bit in scon register
RI=0;//clearing ri bit in scon register
flagt=0; //bit addressable flags
flagr=0;
rec_reg=0;//ram register
TL0=TVL0;
TH0=TV0;
TR1=1;
TR0=1;
timer_flag=0;
time_out_flag=0;
trans_flag=0;
rpm_flag=0;
count_flag=0;
freq_flag=0;

EA=1; // enables all interrupts
set_flag=0;
start_flag=0;
timer_flag=0;
time_out_flag=0;
esc_flag=0;
rpm_flag=0;
initial_speed=175;

PULSE_WIDTH=initial_speed;

while(1)
{
if(start_flag==0)

```

```

}
if(trans_flag==0)
{
    comm_fun();
}
if(rpm_flag==1)
{
    rpmspeed();
    bin_bcd3(pulse_count);
    put_mass_msg1();
    put_lcd_download_1line();
    compare();
    rpm_flag=0;
}
}
}
void dly1sec1()
{
    for(jy=0;jy<=3000;)
    {
        jy++;
    }
}
void timer0() interrupt 1 // Interrupt for real time clock
{
    TR0=0;
    TH0=TV0;//0xfe; // Timer 0 16-bit timer
    TL0=TVL0;//0x0b;
    if(timer_flag==1)
    {
        msec++;
        if(msec>=4500)
        {
            msec=0;
            time_out_flag=1;
        }
    }
    if(rpm_flag==1)
    {
        if(freq_flag==1)
        {
            msec1=msec1+1;
            if(msec1 >=1000) // 1000 millisecond makes 1 second
            {
                count_flag=1;
                msec1=0;
            }
        }
    }
}

```

```

}
void set()
{
    if(set_sw==0) // check for key press
    { delay();
      delay();
      if(set_sw==0) //verify for key press
      { do{delay();}
        while(set_sw==0); //wait until key is released
        set_flag=1; // set the flag
      }
    }
}
void esc_sw()
{
    if(mov_sw==0) // check for key press
    { delay();
      delay();
      if(mov_sw==0) //verify for key press
      { do{delay();}
        while(mov_sw==0); //wait until key is released
        esc_flag=1; // set the flag
      }
    }
}

```

//BCD TO BINARY CONVERSION

```

void delay() //delay for 100millisec
{
    int ia;
    for(ia=1;ia<=800;)//800
    {ia++;}
}

void delay6() //delay for 100millisec
{
    int ip;
    for(ip=1;ip<=50;)//800
    {ip++;}
}

void delay1()
{for(t=1;t<=600;)
 {t++;}
}

void ldelay()
{for(tw=1;tw<=15000;)

```



```

}

void delays()
{
    int c;
    for(c=0;c<=200;)
        {c++;
        }
}

void comm_fun()
{delay();
delay();
delay();
delay();
delay();
fuc1();

}

void fuc1()
{ led=0;
  SBUF=0x80;
  do {};while(flagt==0);
  flagt=0;
  call_1sec();
  do{
      timer_flag=1;
      if(time_out_flag==1)
          {
              delay();
              SBUF=0x80;
              do {};while(flagt==0);
              flagt=0;
              time_out_flag=0;
              msec=0;
              counter++;
              if(counter>=3)
                  {
                      flagr=1;
                      stop_flag=1;
                      break;
                  }
          }
      }while(flagr==0);
  timer_flag=0;
  counter=0;
  time_out_flag=0;
  msec=0;

```

```

if(stop_flag==0)
{
    SBUF=0x81;
    do{;}while(flagt==0);
    flagt=0;
    call_1sec();
do{
    timer_flag=1;
    if(time_out_flag==1)
    {
        delay();
        SBUF=0x81;
        do{;}while(flagt==0);
        flagt=0;
        time_out_flag=0;
        msec=0;
        counter++;
        if(counter>=3)
        {
            flagr=1;
            stop_flag=1;
            break;
        }
    }
}while(flagr==0);
flagr=0;
counter=0;
timer_flag=0;
time_out_flag=0;
msec=0;
l_units1=rec_reg;
}

```

```

if(stop_flag==0)
{
    SBUF=0x82;
    do{;}while(flagt==0);
    flagt=0;
    call_1sec();
do{
    timer_flag=1;
    if(time_out_flag==1)
    {
        delay();
        SBUF=0x82;
        do{;}while(flagt==0);
        flagt=0;
        time_out_flag=0;

```

```

        if(counter>=3)
        {
            flagr=1;
            stop_flag=1;
            break;
        }
    }
} while(flagr==0);
flagr=0;
counter=0;
timer_flag=0;
time_out_flag=0;
misc=0;
h_units1=rec_reg;
}
SBUF=0x83;
do {}; while(flagt==0);
flagt=0;
counter=0;
led=1;
start_flag=0;
if(stop_flag==1)
{
    trans_flag=0;
}
else
{
    dum_reg=h_units1<<8;
    dum_reg=dum_reg|units1;
    set_rpm=dum_reg;
    fill_blank();
    EA=0;
    bin_bcd3( set_rpm);
    EA=1;
    put_set_msg();
    put_lcd_download_2line();
    fill_blank();
    rpm_flag=1;
}
if(stop_flag==1)
{
    stop_flag=0;
    rpm_flag=0;
}
}
void bin_bcd( unsigned int g)
{
    unsigned int y1,y2,y3,y4,y5,y6,y7,y8;

```

```

y3=y2/1000;
y4=y2 % 1000;
y5= y4/100;
y6=y4 % 100;
y7=y6/10;
y8=y6%10;
digit1=y1;
digit2=y3;
digit3=y5;
digit4=y7;
digit5=y8;
}
void call_1sec()
{
for(r=1;r<=40;)
{
delay1();
r++;
}
}
serint ()interrupt 4 //serial vector address
{
switch (TI)//checks whether trasmission is over
{
case 1:
{TI=0; //clears ti flag
flagt=1;//sets flagt
break;
}
case 0:
{
switch (RI)//checks whether reception is over
{
case 0:
break;
case 1:
{rec_reg=SBUF; //received data is fed to rec_reg from sbuf
RI=0; //clears ri flag
flagr=1;//sets flagr
break;}
}
}
}
}
void rate_fun()
{
SBUF=0xaa;
do {}; while(flagt==0);
}

```

```

timer_flag=1;
if(time_out_flag==1)
{
delay();
SBUF=0xaa;
do {};while(flagt==0);
flagt=0;
time_out_flag=0;
misc=0;
}
}while(flagr==0);
time_out_flag=0;
timer_flag=0;
misc=0;
flagr=0;
reg==0xab)
{
P35=1;
SBUF=l_rate1;
do {};while(flagt==0);
flagt=0;
do{
timer_flag=1;
if(time_out_flag==1)
{
delay();
SBUF=l_rate1;
do {};while(flagt==0);
flagt=0;
time_out_flag=0;
misc=0;
}
}while(flagr==0);
flagr=0;
time_out_flag=0;
timer_flag=0;
misc=0;
SBUF=h_rate1;
do {};while(flagt==0);
flagt=0;
do{
timer_flag=1;
if(time_out_flag==1)
{
delay();
SBUF=h_rate1;
do {};while(flagt==0);
flagt=0;

```

```

    }
    }while(flagr==0);
    flagr=0;
    timer_flag=0;
    time_out_flag=0;
    msec=0;
}
}
void rpmspeed()
{EA=0;
TH0=0xfc;//0xfb;    // Timer 0 16-bit timer
TL0=0x17;//0x0b;
pulse_count=0;
count_flag=0;
msec=0;
do{;}while(freq==1);
EA=1;
TR1=0;
TR0=1;
freq_flag=1;
do{
do{
if(count_flag==1)
    break;
}while(freq==0);
do{if(count_flag==1)
    break;
}while(freq==1);
pulse_count=pulse_count+1;
}while(count_flag==0);
count_flag=0;
freq_flag=0;
EA=0;
TR0=0;
EA=0;
pulse_count=pulse_count;
pulse_count=(pulse_count*60);
}
void compare()
{
do{
comp_led=0;
if(set_rpm>pulse_count)
{
initial_speed=initial_speed-1;
if(initial_speed==1)
initial_speed=2;
PULSE_WIDTH=initial_speed;
}
}
}

```

```

{
initial_speed=initial_speed+1;
  if(initial_speed==240)
    initial_speed=240;
PULSE_WIDTH=initial_speed;
}
ldelay();
ldelay();
rpmspeed();
fill_blank();
bin_bcd3(pulse_count);
put_mass_msg1();
put_lcd_download_1line();
ldelay();
ldelay();
ldelay();
ldelay();
ldelay();
ldelay();
ldelay();
  if(set_rpm==0)
    flg_ovr=0;
}
if (pulse_count<=(set_rpm+240) && pulse_count>=(set_rpm-240))
  { flg_ovr=0;
  }
}while(flg_ovr==1);
flg_ovr=1;
EA=1;
TR0=1;
TR1=1;
comp_led=1;
}
void bin_bcd3(unsigned int v)
{
  y1=v/10000;
  y2=v-y1*10000;
  y3=y2/1000;
  y4=y2%1000;
  y5=y4/100;
  y6=y4%100;
  y7=y6/10;
  y8=y6%10;
  digit1=y1;
  digit2=y3;
  digit3=y5;
  digit4=y7;
  digit5=y8;
  disp[5]=(digit1+0x30);
}

```

```

        disp[8]=(digit4+0x30);
        disp[9]=(digit5+0x30);
    }
void put_mass_msg1()
{
    disp[1]='R';
    disp[2]='P';
    disp[3]='M';
    disp[4]=': ';
    disp[10]='r';
    disp[11]='p';
    disp[12]='m';
    disp[13]=' ';
    disp[14]=' ';
    disp[15]=' ';
    disp[16]=' ';
}
void put_set_msg()
{
    disp[1]='s';
    disp[2]='e';
    disp[3]='t';
    disp[4]=': ';
    disp[10]='r';
    disp[11]='p';
    disp[12]='m';
    disp[13]=' ';
    disp[14]=' ';
    disp[15]=' ';
    disp[16]=' ';
}

```

```

void fill_blank()

```

```

{
    disp[1]=' ';
    disp[2]=' ';
    disp[3]=' ';
    disp[4]=' ';
    disp[5]=' ';
    disp[6]=' ';
    disp[7]=' ';
    disp[8]=' ';
    disp[9]=' ';
    disp[10]=' ';
}

```



```

disp[13]=' ';
disp[14]=' ';
disp[15]=' ';
disp[16]=' ');}
}

```

5.3. PROGRAM FOR DISPLAY

```

#include "reg51.h"
#include "stdio.h"
unsigned char disp[17],lcd_location,cursor_pos;
sbit lcd_e =P27;//4-pin P02
sbit lcd_rs=P26; //5-pin P01.
sbit lcd_rw=P25; //6-pin P03
#define LCD_PORT P0
#define FUNCTION_SET 56
#define CURSOR_DISPLAY_SHIFT 16
#define ENTRY_MODE 6
#define CLEAR_DISPLAY 1
#define DISPLAY_ON 12
#define CURSOR_ON 13
void init_lcd();
void dly_250m();
void lcd_pulse();
void lcd_busy();
void setup_read_busy();
void setup_lcd_cmd();
void set_lcd_addr();
void set_lcd_data();
void lcd_com();
void dly1sec();
void put_mass_msg();
void put_mass_msg1();
void put_lcd_download_1line();
void put_lcd_download_2line();
void enable_cursor();
void disable_cursor();
void send_cursor();

void init_lcd()
{
    setup_lcd_cmd();
    dly_250m();
    LCD_PORT = FUNCTION_SET;
    lcd_pulse();
    dly_250m();
    LCD_PORT = FUNCTION_SET;
    lcd_pulse();
}

```

```

lcd_pulse();
dly_250m();
LCD_PORT = DISPLAY_ON;
lcd_pulse();
lcd_busy();
LCD_PORT = ENTRY_MODE;
lcd_pulse();
lcd_busy();
LCD_PORT = CLEAR_DISPLAY;
lcd_pulse();
lcd_busy();
}
void dly_250m()
{
    unsigned int i;
    for(i=0;i<=4500;i++)
    {}
}
void lcd_pulse()
{
    unsigned int i;
    lcd_e=1;
    for(i=0;i<=100;i++)
    {}
    lcd_e=0;
}
void lcd_busy()
{
    unsigned int i,d;
    setup_read_busy();
    for(i=0;i<=10;i++)
    {}
    do
    {
        lcd_e=1;
        for(i=0;i<=10;i++)
        {}
        d=LCD_PORT;
        lcd_e=0;
        LCD_PORT=d;
    }
    while(P07==1);
    for(i=0;i<=10;i++)
    {}
}
void setup_lcd_cmd()
{
    lcd_rs=0;

```

```

void setup_read_busy()
{
    lcd_rw=1;
    lcd_rs=0;
}
void put_lcd_download_1line()
{
    lcd_location=0x01;
    set_lcd_addr();
    lcd_com();
}
void set_lcd_addr()
{
    unsigned int i,d,b;
    lcd_busy();
    setup_lcd_cmd();
    for(i=0;i<=10;i++)
        {;}
    d=0x80;
    b=lcd_location;
    if(lcd_location > 16)
    {
        d=0xc0;
        b=lcd_location-16;
    }
    b=b-1;
    LCD_PORT=(d+b);
    lcd_pulse();
    lcd_busy();
}

void setup_lcd_data()
{
    lcd_rw=0;
    lcd_rs=1;
}
void lcd_com()
{
    unsigned int i,j;
    setup_lcd_data();
    for(i=1;i<=16;i++)
    {
        lcd_busy();
        setup_lcd_data();
        for(j=0;j<10;j++)
            {;}
        LCD_PORT=disp[i];
    }
}

```

```

    lcd_location=lcd_location+1;
    if(lcd_location > 33)
    {
        lcd_location=0;
    }
    lcd_location=lcd_location;
}
}
void put_lcd_download_2line()
{
    lcd_location=17;
    set_lcd_addr();
    lcd_com();
}

void dly1sec()
{
    unsigned int i;
    for(i=0;i<=30000;i++)
        {;}
}
void enable_cursor()
{
    lcd_busy();
    setup_lcd_cmd();
    LCD_PORT=CURSOR_ON;
    lcd_pulse();
    lcd_busy();
}
void disable_cursor()
{
    lcd_busy();
    setup_lcd_cmd();
    LCD_PORT=DISPLAY_ON;
    lcd_pulse();
    lcd_busy();
}
void send_cursor()
{ lcd_location =cursor_pos;
  set_lcd_addr();
}

```

PROGRAM FOR SPEED CONTROL

```
#include<reg51.h>
#define port P1
sbit t1 = P34;
sbit t2 = P35;
sbit fr = P31;
int count,count1,count2,count3,count4,f_count,value,flag=0;
void main()
{
    fr =0;
    t1 =1;
    t2 =1;
    IE = 0x82;
    TMOD = 0x02;
    TL0=0xeb;           // load timer0 for 50 micro second delay
    TH0=0xeb;           // count =0x190;
    do{
        value = port;
        value = value & 0x80;
    } while(value != 0x00);
    count = 0x1bc;
    count4 = count/4; // T=400 * 50 MICRO SECOND, TON/2
= T/4 ,      400/4 =100
    count1 = count4/10;           //count3 = 100/10 =10
    count2 = count1*2;           //count2 = 30*2 = 20
    count3 = count1*3;           //count1 = 15*3 = 30*/
    while(1)
    {
        count4 = count/4 ;           // T =400 * 50 MICRO SECOND,
// TON/2 =T/4 , 400/2 =100
        count1 = count4/10;           //count3 = 100/10 =10
        count2 = count1*2;           //count2 = 30*2 = 20

        count3 = count1*3;           //count1 = 15*3 = 30
// T1 PWM

        TR0 = 1;
        f_count = 0;
        fr = 0;
        t1 = 1;
        t2 = 0;           // ON T1 for 10
        do{;
        }while(f_count <= count1);
        f_count = 0;
        t1 =0;
        t2 =0;           // OFF T1 for 10
        do{;
        }while(f_count <= count1);
        f_count = 0;
    }
}
```

```

do{
}while(f_count <= count1);
  f_count = 0;
  t1 =0;    t2 =0;
  do{
  }while(f_count <= count1);
  f_count = 0;
  t1 = 1;
  t2 = 0;
  do{
  }while(f_count <= count2);
  f_count = 0;
  t1 = 0;
  t2 = 0;
  do{
}while(f_count <= count1);

  f_count = 0;
  t1 = 1;
  t2 = 0;
  do{
  }while(f_count <= count3);
f_count = 0;
  t1 =1;
  t2 =0;
  do{
  }while (f_count <= count3);
  f_count = 0;
  t1 =0;
  t2 =0;
  do{
  }while(f_count <= count1);
  f_count = 0;
  t1 = 1;
  t2 = 0;
  do{
  }while(f_count <= count2);
  f_count = 0;
  t1 =0;
  t2 =0;
  do{
  }while(f_count <= count1);
  f_count = 0;
  t1 = 1;
  t2 = 0;
  do{
  }while(f_count <= count1);

```

// OFF T1 for 10

// ON T1 for 20

// OFF T1 for 10

// ON T1 for 30

// ON T1 for 30

// OFF T1 for 10

// ON T1 for 20

// OFF T1 for 10

// ON T1 for 10

```

t1 =0;
t2 =0;
do{; // OFF T1 for 10
}while(f_count <= count1);
f_count = 0;
t1 = 1;
t2 = 0; // ON T1 for 10
do{;
}while(f_count <= count1);
// DELAY
f_count = 0;
t1 = 0;
t2 = 0; // 200 microsecond delay bt/. ON
and OFF
do{;
}while(f_count <= 4);

//T2 PWM
TR0 = 1;
f_count = 0;
fr = 1;
t1 = 0;
t2 = 1; // ON T1 for 10
do{;
}while (f_count <= count1);
f_count = 0;
t1 =0;
t2 =0; // OFF T1 for 10
do{;
}while(f_count <= count1);
f_count = 0;
t1 =0; t2 =0;
do{; // OFF T1 for 10
}while(f_count <= count1);
f_count = 0;
t1 = 0;
t2 = 1; // ON T1 for 20
do{;
}while(f_count <= count2);
f_count = 0;
t1 = 0;
t2 = 0; // OFF T1 for 10

```

```

f_count = 0;
t1 = 0;
t2 = 1; // ON T1 for 30
do {;
} while(f_count <= count3);
f_count = 0;
t1 = 0;
t2 = 1; // ON T1 for 30
do {;
} while (f_count <= count3);
f_count = 0;
t1 = 0;
t2 = 0; // OFF T1 for 10
do {;
} while(f_count <= count1);
f_count = 0;
t1 = 0;

t2 = 1; // ON T1 for 20
do {;
} while(f_count <= count2);
f_count = 0;
t1 = 0;
t2 = 0; // OFF T1 for 10
do {;
} while(f_count <= count1);
f_count = 0;
t1 = 0;
t2 = 1; // ON T1 for 10
do {;
} while(f_count <= count1);
f_count = 0;
t1 = 0;
t2 = 0; // OFF T1 for 10
do {;
} while(f_count <= count1);
f_count = 0;
t1 = 0;
t2 = 1; // ON T1 for 10
do {;
} while(f_count <= count1);
// DELAY
f_count = 0;
t1 = 0;
t2 = 0; // 200 microsecond delay bt/. ON
and OFF
do {;
} while(f_count <= 4);

```



```

        count =0x1bc;                // 225
// port read
    TR0 = 0;
    do {
        value = port;
        value = value & 0x80;
    } while(value !=0x00);
    count = count-port;
    if(count < 0xfa)                // 225 - port value
        count =0xfa;
    }
}
void timer0()interrupt 1
{
    f_count = f_count + 0x01;
}

```



CONCLUSION

CONCLUSION

Powerline carrier communication technology is definitely an exciting alternative to the communication methods used. The project work done realized the concept of PLCC with the speed control of a single phase induction motor. The concept promises to be an economical means for the control of any type of drive where a dedicated cable for communication should be avoided .

This technology promises to be a viable concept for household automation, multiple drive speed control, etc. Though this technology is not commercially available yet, it should be available before other broadband technologies due to the relatively low cost of its local loop.



REFERENCES

7.1. WEB SITES

- <http://www.atmel.com>
- <http://www.efy.com>
- http://plugtek.com/xyz/NEW_ARTICLES.shtml
- http://www.electricsmarts.com/content/res_hardwired.asp
- <http://www.lughnetworks.com/powerlinetutorial.html>
- <http://www.homeplugandplay.com/index.shtml>

7.2. BOOKS

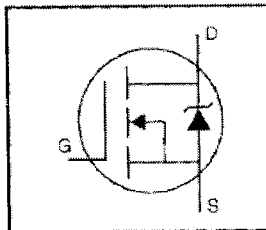
- Linear Integrated Circuits by D. Roy Choudhury & Shail Jain
- Power Electronics by Muhammad H. Rashid
- Electric Drives by Vedam Subramanyam
- Power Electronics by Bhimbra
- Communication Engineering by Anokh Singh
- Let us C by Yashwant Kanitkar



APPENDIX

HEXFET® Power MOSFET

- Dynamic dv/dt Rating
- Repetitive Avalanche Rated
- Fast Switching
- Ease of Paralleling
- Simple Drive Requirements



$V_{DSS} = 500V$

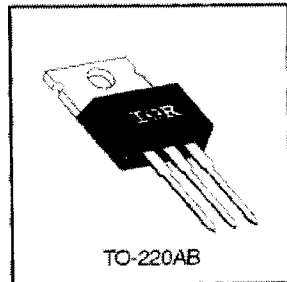
$R_{DS(on)} = 0.85\Omega$

$I_D = 8.0A$

Description

Third Generation HEXFETs from International Rectifier provide the designer with the best combination of fast switching, ruggedized device design, low on-resistance and cost-effectiveness.

The TO-220 package is universally preferred for all commercial-industrial applications at power dissipation levels to approximately 50 watts. The low thermal resistance and low package cost of the TO-220 contribute to its wide acceptance throughout the industry.



DATA SHEETS

Absolute Maximum Ratings

	Parameter	Max.	Units
I_D @ $T_C = 25^\circ C$	Continuous Drain Current, $V_{GS} @ 10 V$	8.0	A
I_D @ $T_C = 100^\circ C$	Continuous Drain Current, $V_{GS} @ 10 V$	5.1	
I_{DM}	Pulsed Drain Current ①	32	
P_D @ $T_C = 25^\circ C$	Power Dissipation	125	W
	Linear Derating Factor	1.0	W/°C
V_{GS}	Gate-to-Source Voltage	±20	V
E_{AS}	Single Pulse Avalanche Energy ②	510	mJ
I_{AR}	Avalanche Current ③	8.0	A
E_{AR}	Repetitive Avalanche Energy ④	13	mJ
dv/dt	Peak Diode Recovery dv/dt ⑤	3.5	V/ns
T_J T_{STG}	Operating Junction and Storage Temperature Range	-55 to +150	°C
	Soldering Temperature, for 10 seconds	300 (1.6mm from case)	
	Mounting Torque, 6-32 or M3 screw	10 lbf-in (1.1 N·m)	

+5V-Powered, Multichannel RS-232 Drivers/Receivers

General Description

The MAX220–MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly applications where $\pm 12V$ is not available.

These parts are especially useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than 5 μ W. The MAX225, MAX233, MAX235, and MAX245/MAX246/MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

Applications

Portable Computers
 Low-Power Modems
 Interface Translation
 Battery-Powered RS-232 Systems
 Multidrop RS-232 Networks

Features

Superior to Bipolar

- ◆ Operate from Single +5V Power Supply (+5V and +12V—MAX231/MAX239)
- ◆ Low-Power Receive Mode in Shutdown (MAX223/MAX242)
- ◆ Meet All EIA/TIA-232E and V.28 Specifications
- ◆ Multiple Drivers and Receivers
- ◆ 3-State Driver and Receiver Outputs
- ◆ Open-Line Detection (MAX243)

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220CID	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX220/222/232A/233A/242/243

Supply Voltage (V _{CC})	-0.3V to +6V
Input Voltages	
T _{IN}	-0.3V to (V _{CC} - 0.3V)
R _{IN} (Except MAX220)	$\pm 30V$
R _{IN} (MAX220)	$\pm 25V$
T _{OUT} (Except MAX220) (Note 1)	$\pm 15V$
T _{OUT} (MAX220)	$\pm 13.2V$
Output Voltages	
T _{OUT}	$\pm 15V$
R _{OUT}	-0.3V to (V _{CC} + 0.3V)
Driver/Receiver Output Short Circuited to GND	Continuous
Continuous Power Dissipation (T _A = +70°C)	
16-Pin Plastic DIP (derate 10.63mW/°C above +70°C)	842mW
18-Pin Plastic DIP (derate 11.11mW/°C above +70°C)	889mW

20-Pin Plastic DIP (derate 9.00mW/°C above +70°C)	440mW
16-Pin Narrow SO (derate 8.70mW/°C above +70°C)	593mW
16-Pin Wide SO (derate 9.62mW/°C above +70°C)	762mW
18-Pin Wide SO (derate 9.62mW/°C above +70°C)	762mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C)	800mW
20-Pin SSOP (derate 8.00mW/°C above +70°C)	640mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C)	800mW
18-Pin CERDIP (derate 10.63mW/°C above +70°C)	842mW
Operating Temperature Ranges	
MAX2_ _AC_ _ , MAX2_ _C_	0°C to +70°C
MAX2_ _AE_ _ , MAX2_ _E_	-40°C to +85°C
MAX2_ _AM_ _ , MAX2_ _M_	-55°C to +125°C
Storage Temperature Range	-65°C to +160°C
Lead Temperature (soldering, 10sec)	+300°C

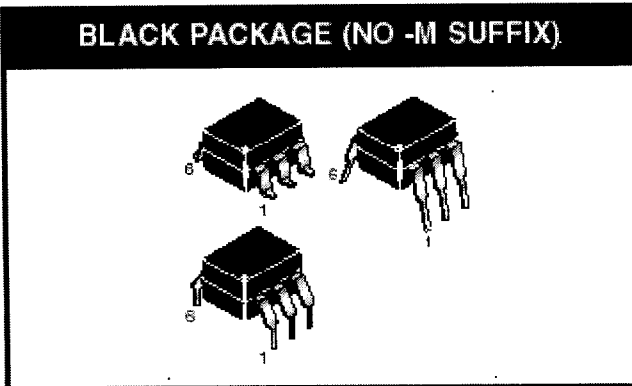
Note 1: Input voltage measured with T_{OUT} in high-impedance state, \overline{SHDN} or V_{CC} = 0V.

Note 2: For the MAX220, V₊ and V₋ can have a maximum magnitude of 7V, but their absolute difference cannot exceed 13V.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

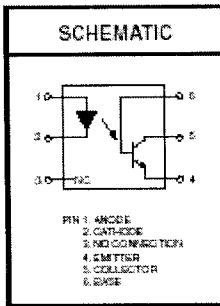
PHOTOTRANSISTOR OPTOCOUPLERS

BLACK PACKAGE (NO -M SUFFIX)



DESCRIPTION

The MCT2XXX series optoisolators consist of a gallium arsenide infrared emitting diode driving a silicon phototransistor in a 6-pin dual in-line package.



FEATURES

- UL recognized (File # E90700)
- VDE recognized (File # 94766)
 - Add option V for white package (e.g., MCT2V-M)
 - Add option 300 for black package (e.g., MCT2.300)
- MCT2 and MCT2E are also available in white package by specifying -M suffix, eg. MCT2-M.

APPLICATIONS

- Power supply regulators
- Digital logic inputs
- Microprocessor inputs

Features

- Compatible with MCS-51™ Products
- 2K Bytes of Reprogrammable Flash Memory
 - Endurance: 1,000 Write/Erase Cycles
- 2.7V to 6V Operating Range
- Fully Static Operation: 0 Hz to 24 MHz
- Two-level Program Memory Lock
- 128 x 8-bit Internal RAM
- 15 Programmable I/O Lines
- Two 16-bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial UART Channel
- Direct LED Drive Outputs
- On-chip Analog Comparator
- Low-power Idle and Power-down Modes

Description

The AT89C2051 is a low-voltage, high-performance CMOS 8-bit microcomputer with 2K bytes of Flash programmable and erasable read only memory (PEROM). The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard MCS-51 instruction set. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C2051 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications.

The AT89C2051 provides the following standard features: 2K bytes of Flash, 128 bytes of RAM, 15 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, a precision analog comparator, on-chip oscillator and clock circuitry. In addition, the AT89C2051 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The power-down mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.



8-bit
Microcontroller
with 2K Bytes
Flash

AT89C2051

Pin Configuration

PDIP/SOIC

