

OBSTACLE SENSING AUTONOMOUS MOBILE ROBOT

By

AZRIHAN BIN ABDULLAH

submitted in partial fulfillment of
the requirements for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)
JUNE 2010

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Obstacle Sensing Autonomous Mobile Robot

By

Azrihan Bin Abdullah

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
BACHELOR OF ENGINEERING (Hons)
(ELECTRICAL & ELECTRONICS ENGINEERING)

Approved by,



(Dr. Noohul Basheer Zain Ali)

Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

JUNE 2010

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

A handwritten signature in dark ink, consisting of a stylized 'A' followed by a series of loops and a horizontal stroke at the end.

Azrihan Bin Abdullah

ABSTRACT

The aim of this project is to design an autonomous mobile robot that have the ability to sense and avoid obstacles with considerable amount of cost. The robot is requires to have its own movement based on sensor and programmed microcontroller. The scope of the study for this project is covering several areas such as electronic circuit design, programming based software and mechanical design. In order to fulfil the design requirements, a process flow chart has been constructed to assist the author throughout this project. The discussions parts are focus on the development process during the prototype fabrication period. Finally, this project is about fulfilling its objective.

ACKNOWLEDGEMENT

Final Project has been such a challenging yet satisfying experience for me as the author myself. First of all, I am grateful for Al-Mighty for giving me the opportunity and strength to finish this course. I would like to show my sincere appreciation to the project supervisor, Dr. Noohul Basheer Zain Ali for helping me throughout this project. I would also like to express my gratitude to my colleagues who had helped me giving ideas and support me accomplish this project. I would also love to mention my deep appreciation towards the lab technicians from the Department of Electrical & Electronic Engineering who have been assisting me through this pass two semesters. Last but not least, I would like to thank my family for their support in helping me to keep focus and being responsible in order to finish this project.

TABLE OF CONTENTS

CERTIFICATION OF APPROVAL.....	ii
CERTIFICATION OF ORIGINALITY.....	iii
ABSTRACT.....	iv
ACKNOWLEDGEMENT.....	v
LIST OF FIGURES	viii
LIST OF TABLES.....	ix
CHAPTER 1: INTRODUCTION.....	1
1.1 Background of Study.....	1
1.2 Problem Statement.....	2
1.3 Objective.....	3
1.4 Scope of Study.....	3
CHAPTER 2: LITERATURE REVIEW.....	4
2.1 Overview.....	4
2.2 Mobile Robotic System.....	4
2.3 Navigation System	5
2.3.1 Collision Detection Concept.....	5
2.3.2 Collision Avoidance Concept.....	5
2.3.3 Sensor.....	6
2.3.4 Ultrasonic sensor.....	7
2.4 Locomotion System.....	10
2.4.1 Steering system.....	11
2.4.2 Servo motor.....	14
2.4.3 DC Motor.....	16

	2.4.4	<i>Driving system</i>	17
	2.5	Control System	18
	2.5.1	<i>Microcontroller Overview</i>	18
	2.5.2	<i>Microcontroller features</i>	19
CHAPTER 3:		METHODOLOGY	21
	3.1	Procedure Identification.....	21
	3.2	Tools Required.....	23
	3.3	Basic Building Block Diagram.....	24
CHAPTER 4:		RESULT AND DISCUSSION	25
	4.1	Discussion.....	25
	4.1.1	<i>Navigation System</i>	25
	4.1.2	<i>Line Following Implementation</i>	26
	4.1.3	<i>Analog Distance Sensor</i>	28
	4.1.4	<i>Locomotion System</i>	30
	4.1.4.1	<i>Steering System</i>	31
	4.1.4.2	<i>Driving System</i>	32
	4.1.5	<i>H-Bridge Motor Driver</i>	33
	4.1.6	<i>Control System</i>	35
	4.1.7	<i>Power Regulation</i>	36
CHAPTER 5:		CONCLUSION AND RECOMMENDATION	37
	5.1	Conclusion.....	37
	5.2	Recommendation.....	38
REFERENCES			39
APPENDICES			40

LIST OF FIGURES

Figure 1	LawnBott LB3500.....	4
Figure 2	Ultrasonic receiver circuit.....	8
Figure 3	Ultrasonic transmitter circuit.....	8
Figure 4	Car Type Steering Method.....	11
Figure 5	Dually drive wheels using Differential Drive.....	12
Figure 6	Tricycle Steering Method.....	13
Figure 7	Futaba S-148.....	14
Figure 8	Servo Parts.....	15
Figure 9	Centerline Drive Motor Mount with Two End to End Balancing Caster.....	17
Figure 10	Front-Drive Motor Mount with Single Opposing Balancing Caster.....	18
Figure 11	Flowcharts for Procedure Identification.....	22
Figure 12	Basic Building Block Diagram.....	24
Figure 13	Four pair of infrared sensor	26
Figure 14	Schematic diagram for the four pair infrared sensor.....	27
Figure 15	Four pair Infrared Sensors Transmitters and Receivers	27
Figure 16	SHARP GP2Y0A21YK.....	28
Figure 17	Triangulation method.....	29
Figure 18	Differential drive system for robot.....	31
Figure 19	Centerline Drive type.....	32
Figure 20	H Bridge Motor Driver Schematic Using L298N.....	33
Figure 21	H-Bridge L298N motor control part on the PCB.....	34
Figure 22	Power Regulation Schematic.....	36
Figure 23	Power Regulation part on PCB.....	36

LIST OF TABLES

Table 1	Hardware and software required.....	23
Table 2	Comparison between ultrasonic sensor and infrared sensor.....	25
Table 3	Comparison between leg, track and wheel.....	30
Table 4	PIC16F877A parameters.....	35

CHAPTER 1

INTRODUCTION

1.1 Background of study

Robot is a virtual or mechanical artificial agent. In practice, it is usually an electro-mechanical machine which is guided by computer or electronic programming, and is thus able to do tasks on its own. Another common characteristic is that by its appearance or movements, a robot often conveys a sense that it has intent or agency of its own.[1]

In general, robots are classified based on their capabilities. Some standard classifications of robots include their main of operation, degree of autonomy, and the goal they are designed to fulfill. Robots can be designed and built for any environment imaginable. One popular way of classifying robots is by what environments they're designed to operate in. Some typical examples include stationary robots that are fixed in one place and cannot move. This category includes robotic arms, computerized machine tools, and most other Industrial Robots. Industrial Robots are robots used in mass production e.g. welding robots, CNC plate cutters or CNC drills. The large majorities of these robots are stationary and connected to a computer. Ground robots are basically designed to operate on the surface of the earth or other planet, and are usually sub categorized by their drive train such as wheels, tracks and legs. Underwater robot also known as Autonomous Underwater Vehicles, these are designed to operate underwater, possibly at great depth. Last but not least, Aerial robot which is Unmanned Aerial Vehicles designed to perform various kinds of robotic flying machines, including planes and helicopters.[3]

1.2 Problem Statement

Robot has been playing its essential roles for a long time in wide range of areas such as in heavy machinery, automation and aviation industries. Furthermore, with the invention of autonomous robots either from the smallest to the largest structure has provides their own functionality to human race to achieve modern life that we had in the present. However, for small scale and domestic use, cost is the most important factor that should be consider before designing an autonomous robot. Hence, the success of building an autonomous robot using considerable amount of cost would attract users domestically.

1.3 Objectives of Study

The objective of this project is to build and develop an autonomous robotic system that will operate autonomously and has the ability to sense. There are several of objectives to be achieved in this project, which are:-

- Study and construct suitable electronic circuitry design that can be implemented in the robotic system.
- Study and understand mechanical concept that suite the robotic system and the electronic design.
- Study the usage of software to program PIC so that the robot can perform the required activities.

1.4 Scope of Study

The scope of study for this project is covering several areas such as electronic circuitry design, programming based software and mechanical design. Electronic circuit design is required in order to design an electronic circuit that is suitable for the autonomous system. This part is very important to avoid complex circuits and to achieve maximum usage of a single electronic circuit. Programming based software is needed to algorithms and instruction for the control system. Mechanical design is appropriate to build the physical part of the robot and also to assembly the electronic circuit with the system. These three areas of knowledge are crucial to achieve the objectives of this project. Study of previous projects would become really helpful in order to gain more knowledge and better understanding.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

Autonomous mobile robot can be found in any industries. A high degree of autonomous robotic is preferred to use in the space and underwater exploration. Other more mundane uses benefit from having some level of autonomy and appear to be consumer products like cleaning floors, mowing lawns, and waste water treatment.

2.2 Mobile Robotic System

Mobile robot is designed to move from one place to another. Wheels, tracks and legs allow the robot to traverse a terrain. [5] Basically robot depends on their navigation and locomotion mechanism. These two parameters will give the robot abilities to navigate through their surroundings and because of their unique abilities, autonomous robot also being used as consumer products. For example Kyodo America Industries Co. Ltd produce an autonomous robot called LawnBott that capable to criss cross lawn at random angles.



Figure 1: LawnBott LB3500

2.3 Navigation System

The system is required to provide the 'vision' for the robot. By using sensors, the robot become 'aware' of its surrounding and can detect obstacles that locate in it pathway. If the robot detected obstacles ahead, it should be able to do what it has been programmed for. There are many types of sensors that come with various theories and application. It can be used in the robot to provide detection and proper navigation for the robot.

2.3.1 Collision Detection Concept

This concept is also known as passive detection concept that required physical contact with the obstacles or object. If this concept is applied, robot will change their direction after hitting obstacles. Common techniques that been used nowadays are pressure pad spring, whiskers and also bumper switch.[5]

2.3.2 Collision Avoidance Concept

This concept is also known as active detection concept. It is totally different from passive detection as it did not require any physical contact with obstacles. The robot will detect any obstacles trough the pathway within a pre-designed distance and changes direction to avoid collision. Ultrasonic sensor and infrared sensor are techniques that suite to apply with this concept.[5]

2.3.3 Sensor

A sensor is defined as a device that receives energy form one system and transmits it to another. By other means; the sensor is a device which capable of being actuated by energizing input from one or more transmission media in turn generating a related signal to one or more transmission systems. It provides a usable output response to the specified input measured, which may be physical or mechanical quantity, property, or conditions. The energy transmitted by these systems may be electrical, mechanical or acoustical. The nature of electrical output from the transducers depends on the basic principle involved in the design. The output maybe analog, digital or frequency modulated. The sensor has to be physically compatible with its independent applications. In the process of selecting a suitable sensor, there are few specifications that should be considered:

1. Operating range: Chosen to maintain range requirements and good resolution.
2. Sensitivity: Chosen to allow sufficient output.
3. Frequency response and resonant frequency: Flat over entire desired range.
4. Environment capability: Temperature range, corrosive fluids, pressure, shocks, interaction, size and mounting restriction.
5. Minimum sensitivity: To expected stimulus, other than measured.
6. Accuracy: repeatability and calibration errors as well errors expected due to sensitivity to other stimuli.
7. Electrical parameters: Length and type of cable required, signal to noise ratio when combined with amplifiers, frequency response limitations.

2.3.4 Ultrasonic Sensor

Sound can be used to detect the proximity of objects in much the same as for infrared light. Ultrasonic sound is transmitted from a transducer, reflected by a near object, and then received by another transducer. The advantage of using sound is that it is not sensitive to objects of different color and light reflective properties. However, there are materials reflect sound better than others, and some even absorb sound completely. In comparison, proximity detection with sound is more fool proof. The ultrasonic transmitter work as follows; a stream of 40 kHz pulses are produced by a 555 timer wired up as an astable multivibrator. The receiving transducer is positioned two or more inches away from the transmitter.[5]

For the best results, a piece of foam can be placed between two transducers to eliminate direct interferences. The signal from the receiving transducer needs to be amplified; an operational amplifier (such as an LM741) is more than sufficient for the job. The amplified output of the receiver transducer is directly connected to another 741 operational amplifier wired as comparator. The ultrasonic receiver is sensitive only to sounds in about 40 kHz range. The closer the ultrasonic sensor to object, the stronger the reflected sound will be. The output of the comparator will change between low to high as the sensor is moved closer to or farther away from an object. The potentiometer, which can be found on the operational amplifier on the receiver circuit, can be adjusted to vary the sensitivity of the circuit.

Below are the typical schematic examples of ultrasonic sensor.[7] The circuit behaves slightly different compared to the basic ultrasonic sensor circuit as explained before.

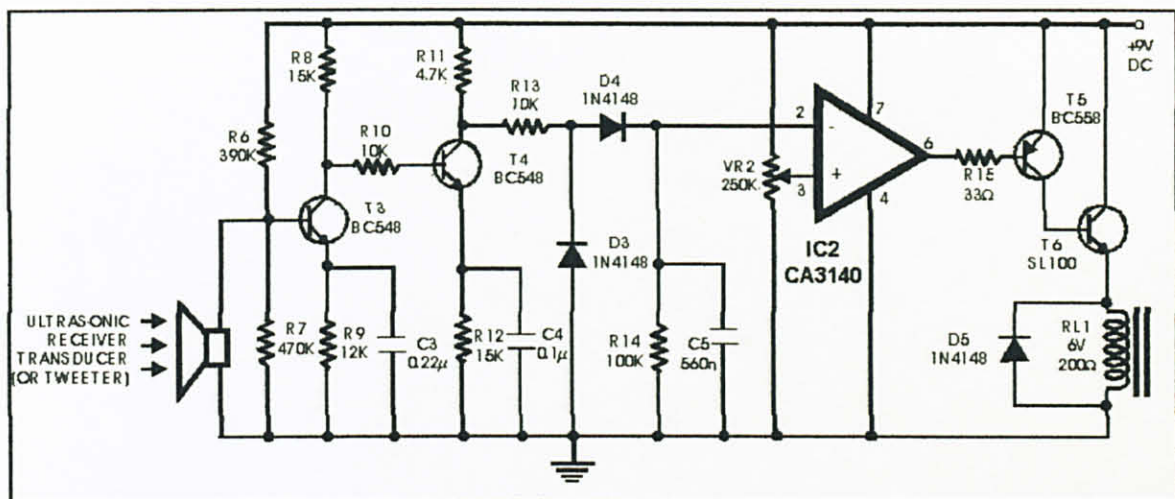


Figure 2: Ultrasonic receiver circuit

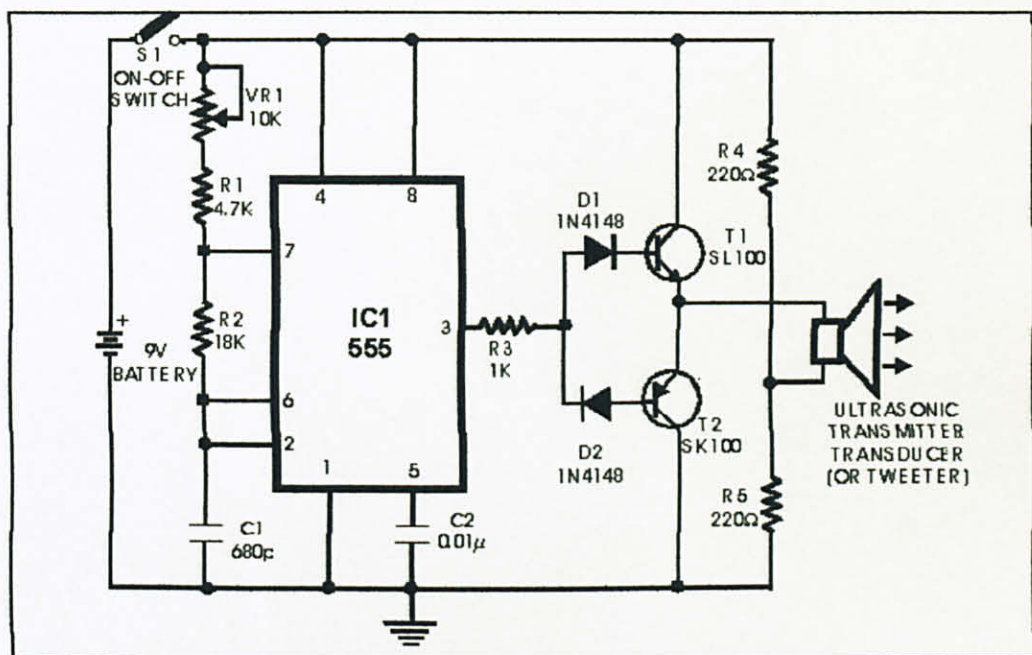


Figure 3: Ultrasonic transmitter circuit

The ultrasonic transmitter uses a 555 timer based astable multivibrator. It oscillates at a frequency of 40 – 50 kHz. An ultrasonic transmitter transducer is used to transmit ultrasonic sound very effectively. The transmitter is powered from a 9V PP3 single cell.

The ultrasonic receiver circuit uses an ultrasonic receiver transducer to sense ultrasonic signals. It also uses a two stage amplifier, a rectifier stage, and an operational amplifier in inverting mode. Output of operational amplifier is connected to a relay through a complimentary relay driver stage. A 9V battery eliminator can be used for receiver circuit, if required. When switch S1 of transmitter is pressed, it generates ultrasonic sound. The sound is received by ultrasonic receiver transducer. It converts it to electrical variations of the same frequency. These signals are amplified by transistors T3 and T4. The amplified signals then rectified and filtered. The filtered DC voltage is given to inverting pin of operational amplifier IC2. The non-inverting pin of IC2 is connected to a variable DC voltage via preset VR2 which determines the threshold value of ultrasonic signal received by receiver for operation of relay RL1. The inverted output of IC2 is used to bias transistor T5. When transistor T5 conducts, it supplies base bias to transistor T6. When transistors T6 conducts, it actuates the relay. The relay can be used to control any electrical or electronic equipment.

2.4 Locomotion System

The way robot move from a point to another point is called locomotion. Robot locomotion takes many forms, but wheels, legs and tracks are the common.[5]

- **Wheels:** The most popular method for providing the robots with mobility. Robot wheels can be just about any size, limited only by the dimensions of the robot and one's imagination. Turtle robots usually have small wheels less than 2 or 3 inches in diameter. Medium sized rover type robot use wheels with diameters up to 7 or 8 inches. The advantages of wheels are they simple to construct, provide great maneuvering ability do not require powerful motors as tracks and legs does. They are preferable especially indoor where space is tight and packed with obstacles.[5]
- **Tracks:** The tracks usually replace common wheels on each side of the robot, act as giant wheels. The track turn, like wheels and the robot, lurches forward and backward. For maximum traction, each track is about as long as the robot itself. Track drive is preferable for many reasons, including the fact that it makes possible to mow through all sorts of obstacles, like rocks, ditches and potholes, the track drive provide excellent traction and greater stability. But the drawback is that powerful motors are required to drive the track.[5]
- **Legs:** Legged robots are challenge to design but they provide an extra level of mobility that sometimes does not offered by wheeled and tracked robots. Due to difficulties and complicated design manner of legged robot, there are seldom legged robots being made compared to wheeled and tracked robots.[5]

2.4.1 Steering system

Steering system is a mechanism that provides direction to the robot. Its path is pre-determined based on pre-determined instructions or detection of obstacles. Basically, there are three common driving mechanisms which are car type, differential drive and tricycle drive. Each has their own advantages and disadvantages.

- **Car-Type:** Pivoting the wheels in the front is one of the methods to steer a robot. Robot with car-type steering is not as maneuverable as differentially steered robots but they are better suited for outdoor uses especially over rough terrain. However, better traction and steering accuracy can be obtained if the wheel on the inside of the turn pivots more than the wheel on the outside.[5]

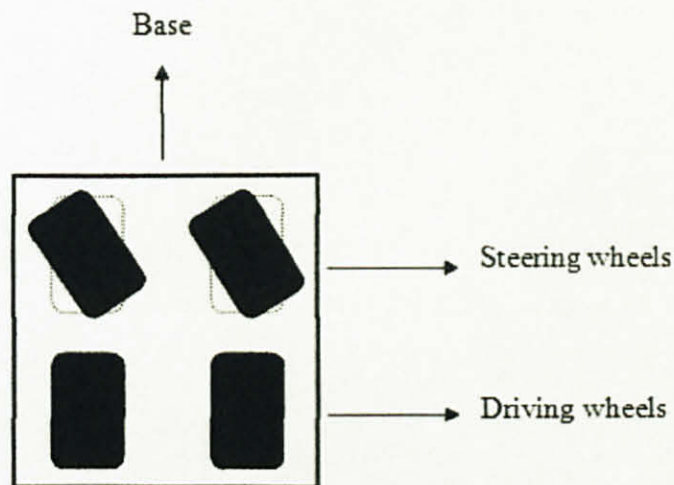


Figure 4: Car Type Steering Method

- **Differential:** For wheeled robots, differential steering is the most common method to getting the machine to go in different direction. The technique is that one side of the wheels stops or reverses direction while the other side keeps on going on. The result is that the robot will turn in the direction of the stopped or reversed wheel. Because of friction effects differential steering is most practical with two-wheel drive systems. Additional set of wheels however, can increase friction during steering.[5]

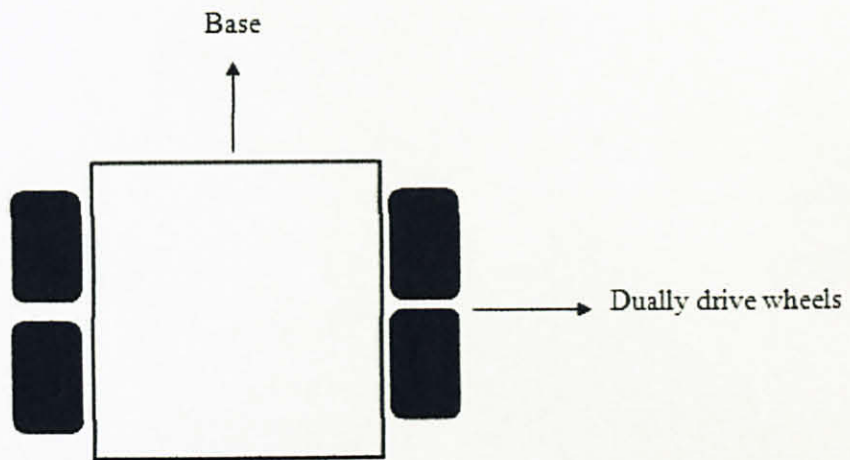


Figure 5: Dually drive wheels using Differential Drive

- **Tricycle:** One of the biggest drawback of the differentially steered robot is that the robot will steer off course if one of the two motor decrease in speed. A better approach is to use a single drive motor powering two rear wheels and a single steering wheel at the front. The robot can be steered in a circle just slightly larger than the width of the machine. Distance from the rear wheels to the front should be measured carefully as short case will cause instability when making turns causing the robot to tip over opposite direction. For tricycle method the front wheel should be steered with high degree of accuracy in order to travel in straight line. Most often the steering wheel is controlled by a servo motor or stepper motor. These two motors can provide a high degree of positional accuracy that is required.[5]

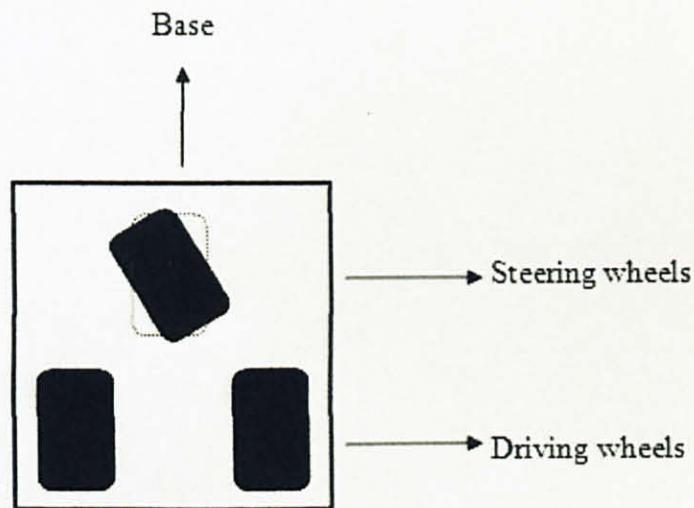


Figure 6: Tricycle Steering Method

2.4.2 Servo Motor

Servo motors are composed of an electric motor mechanically linked to a potentiometer. Pulse-width modulation (PWM) signals sent to the servo are translated into position commands by electronics inside the servo. When the servo is commanded to rotate, the motor is powered until the potentiometer reaches the value corresponding to the commanded position.

The servo is controlled by three wires: ground (usually black/orange), power (red) and control (brown/other color). This wiring sequence is not true for all servos. Servo is wired as brown (negative), red (positive) and orange (signal). The servo will move based on the pulses sent over the control wire, which set the angle of the actuator arm. The servo expects a pulse every 20 ms in order to gain correct information about the angle. The width of the servo pulse dictates the range of the servo's angular motion.

A Servo also has an output shaft. This shaft can be positioned to specific angular positions by sending the servo a coded signal. As long as the coded signal exists on the input line, the servo will maintain the angular position of the shaft. As the coded signal changes, the angular position of the shaft changes. In practice, servos are used in radio controlled airplanes to position control surfaces like the elevators and rudders. They are also used in radio controlled cars, puppets, and also robots.

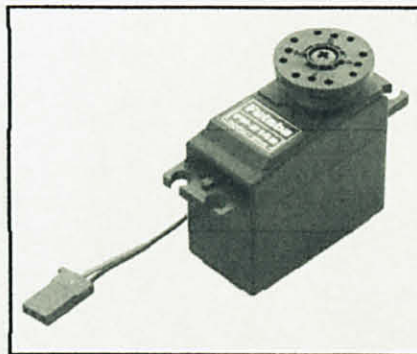


Figure 7: Servo Motor Futaba S-148

Servos are extremely useful in robotics. The motors are small, built in control circuitry, and are extremely powerful for their size. A standard servo such as the Futaba S-148 has 42 oz/inches of torque, which is pretty strong for its size. It also draws power proportional to the mechanical load. A lightly loaded servo, therefore, doesn't consume much energy.

The servo motor has some control circuits and a potentiometer that is connected to the output shaft. The pot allows the control circuitry to monitor the current angle of the servo motor. If the shaft is at the correct angle, then the motor shuts off. If the circuit finds that the angle is not correct, it will turn the motor the correct direction until the angle is correct. The output shaft of the servo is capable of travelling somewhere around 180 degrees. Usually, it's somewhere in the 210 degree range, but it varies by manufacturer. A normal servo is used to control an angular motion of between 0 and 180 degrees. A normal servo is mechanically not capable of turning any farther due to a mechanical stop built on to the main output gear.

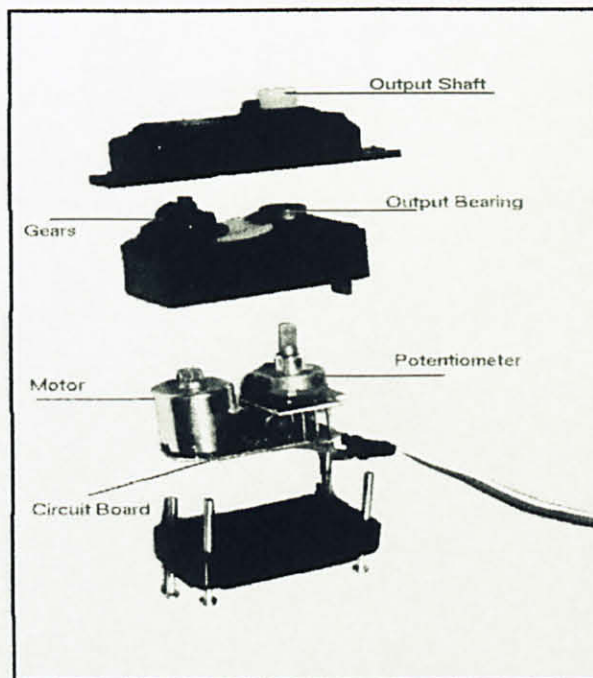


Figure 8: Servo Parts

2.4.3 DC Motor

Powered by continuous direct current, DC motor produces angular force by spinning according to its speed rating. Gears can be used in order to reduce the motor speed if it is too much for the robot to handle. There are two main controls for a DC motor; the direction control and the speed control. The direction control provides forward and reverse control. Direction control techniques such as relay and power mosfet are commonly used techniques.

The speed control is obviously to control the speed of the motor. Basically, without this speed control, the robot might not be able to slow down when required. The best way to use this technique is to chop the power being passed into different sized of chunks. This can be done by setting up a recurring signal; the time for power to be passed to the motor should be a set fraction of this period which known as 'Pulse Width Modulation' or PWM.[5]

There are few things that need to be considered in choosing the right type of DC motor for the robot:

- DC motors can be often effectively operated at voltages above and below their specified rating. If the motor is rated for 12V and it run at 6V, the odds are the motor will still turn but at reduced speed and torque. Conversely, if the motor is running at 18 to 24V the motor will turn faster and will have increased torque. Significantly, overdriving a motor may cause it ti wear out much faster than normal. However, it is usually fairly safe to run at 10 V motor at 12 V or 6V motor at 5V.[5]
- DC motors draw the most current when they are stalled. Stalling occurs if the motor is supplied current, but the shaft does not rotate. If there is no stall detection circuitry built in the motor driver, the electronic parts used with the motor must be able to deliver current at stall condition or else the electronic circuit could damage.[5]

- DC motors vary in term of efficiency. Many of least expensive motors are meant to be used in applications where huge force is needed rather than conservation of electricity.[5]

The rational speed of a DC motor is usually too fast to be directly applied in a robot. Few types of gear reduction is necessary to slow down the speed of the motor shaft. Gearing down the output speed has the positive side effect of increasing torque.[5]

2.4.4 Driving system

Mobilizing the robots is the function of this system. Basically, there are two types of drive motor mount that are centerline and front drive. For centerline type, the weight is evenly distributed across the robot body or platform. It also has no back or front part. For the front-drive type, the weight is concentrated on the motor side of the platform. The advantage is that it will simplify the construction of the robot.

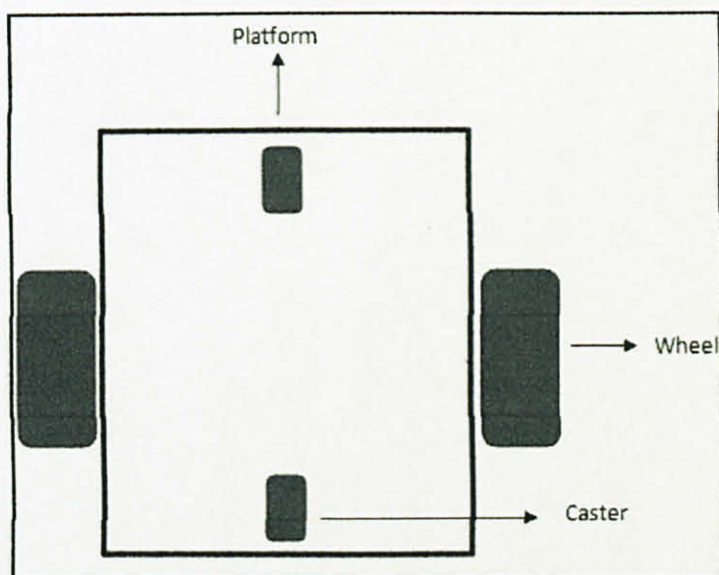


Figure 9: Centerline Drive Motor Mount with Two End to End Balancing Caster

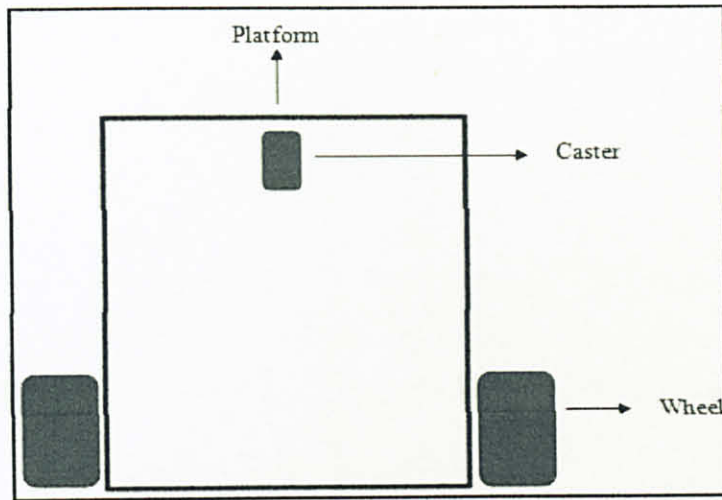


Figure 10: Front-Drive Motor Mount with Single Opposing Balancing Caster

2.5 Control System

A control system is a device or set of devices to manage, command, direct or regulate the behavior of other devices or systems. There are two common classes of control systems, with many combination and variation such as logic and sequential control, feedback and also linear control. A control system also should contain data algorithms, programs and logic analysis that will enable devices to perform activities. For autonomous robot, a controller should be able to work as the robot's brain thus it can control the robot movements.

2.5.1 Microcontroller Overview

A microcontroller (also microcomputer, MCU or μC) is a small computer on a single integrated circuit consisting internally of a relatively simple CPU, clock, timers, I/O ports, and memory. Program memory in the form of NOR flash or OTP ROM is also often included on chip, as well as a typically small amount of RAM. Microcontrollers are designed for small or dedicated applications.[4]

Some microcontrollers may use four-bit words and operate at clock rate frequencies as low as 4 kHz, as this is adequate for many typical applications, enabling low power consumption (milliwatts or microwatts). They will generally have the ability to retain functionality while waiting for an event such as a button press or other interrupt; power consumption while sleeping (CPU clock and most peripherals off) may be just nanowatts, making many of them well suited for long lasting battery applications.[4]

2.5.2 Microcontroller features

Microcontrollers must provide real time (predictable, though not necessarily fast) response to events in the embedded system they are controlling. When certain events occur, an interrupt system can signal the processor to suspend processing the current instruction sequence and to begin an interrupt service routine (ISR, or "interrupt handler").[4]

Microcontroller programs must fit in the available on-chip program memory, since it would be costly to provide a system with external, expandable, memory. Compilers and assembler are used to turn high-level language and assembler language codes into a compact machine code for storage in the microcontroller's memory. Depending on the device, the program memory may be permanent, read-only memory that can only be programmed at the factory, or program memory may be field-alterable flash or erasable read-only memory.[4]

Microcontrollers usually contain from several to dozens of general purpose input/output pins (GPIO). GPIO pins are software configurable to either an input or an output state. When GPIO pins are configured to an input state, they are often used to read sensors or external signals. Configured to the output state, GPIO pins can drive external devices such as LED's or motors.[4]

A dedicated Pulse Width Modulation (PWM) block makes it possible for the CPU to control power converters, resistive loads, motors, etc., without using lots of CPU resources in tight timer loops.[4]

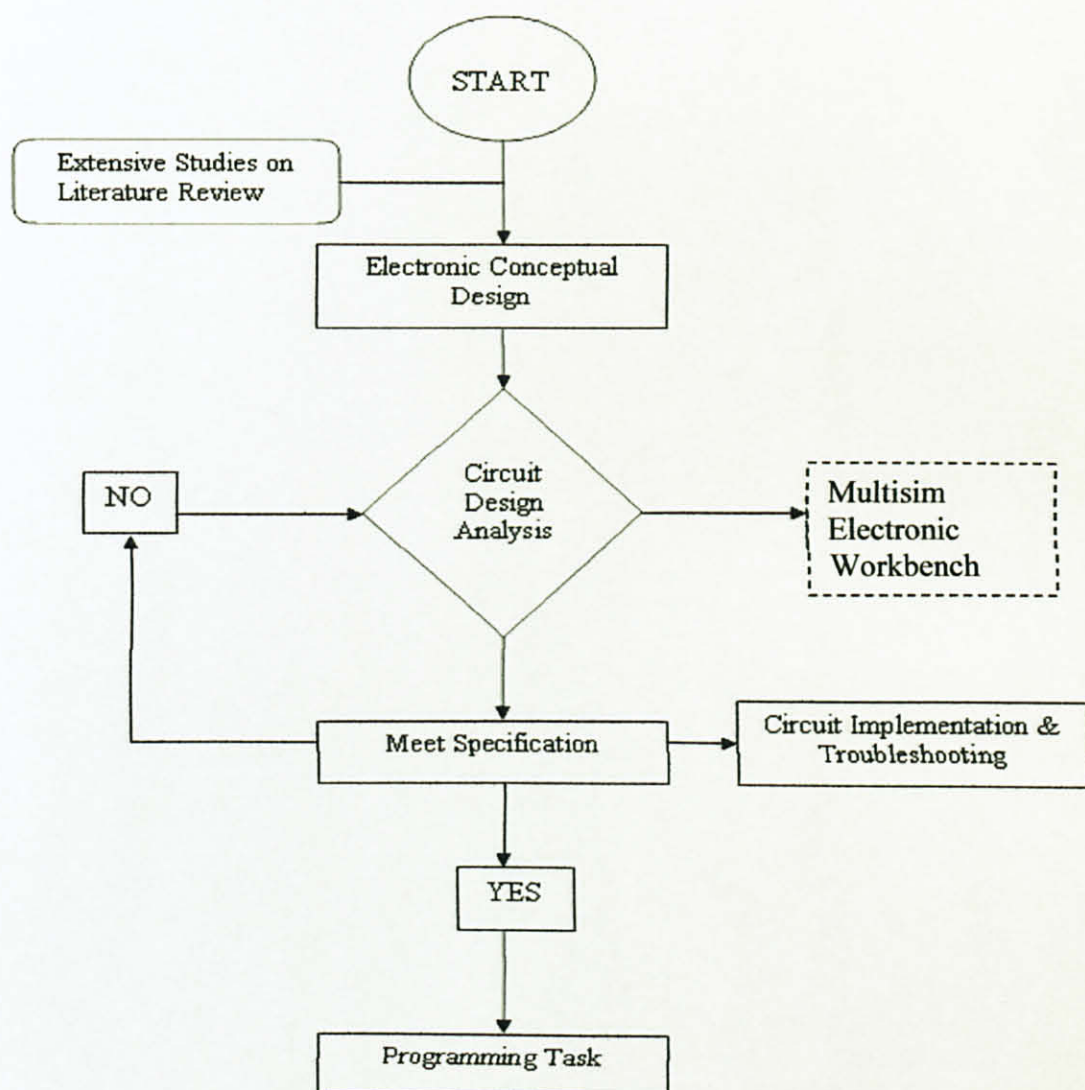
Universal Asynchronous Receiver/Transmitter (UART) block makes it possible to receive and transmit data over a serial line with very little load on the CPU. Dedicated on-chip hardware also often includes capabilities to communicate with other devices (chips) in digital formats such as I2C and Serial Peripheral Interface (SPI).[4]

CHAPTER 3

METHODOLOGY

3.1 Procedure Identification

Figure 2 below is a process flow chart describing the project procedure step-by-step. The procedure need to be followed in order to make sure that the project can be completed in the period of time given.



3.2 Tools required

In order to ensure that this project can be run smoothly suitable hardware, software and tools has been identified based on the procedure identification criteria.

Table 1: Hardware and software required

CRITERIA	HARDWARE	SOFTWARE/TOOLS
Electronics circuits Simulation and Analysis	Personal computer	<ul style="list-style-type: none">• Multisim Electronic Workbench
PIC Programming and Analysis	Computer connected to PIC communication hardware	<ul style="list-style-type: none">• MPLAB IDE• HI TECC C PRO• PICKit 2
Mechanical design	Perspex Bolts & nuts Stand screw	<ul style="list-style-type: none">• Google SketchUp

3.3 Building Block Diagram

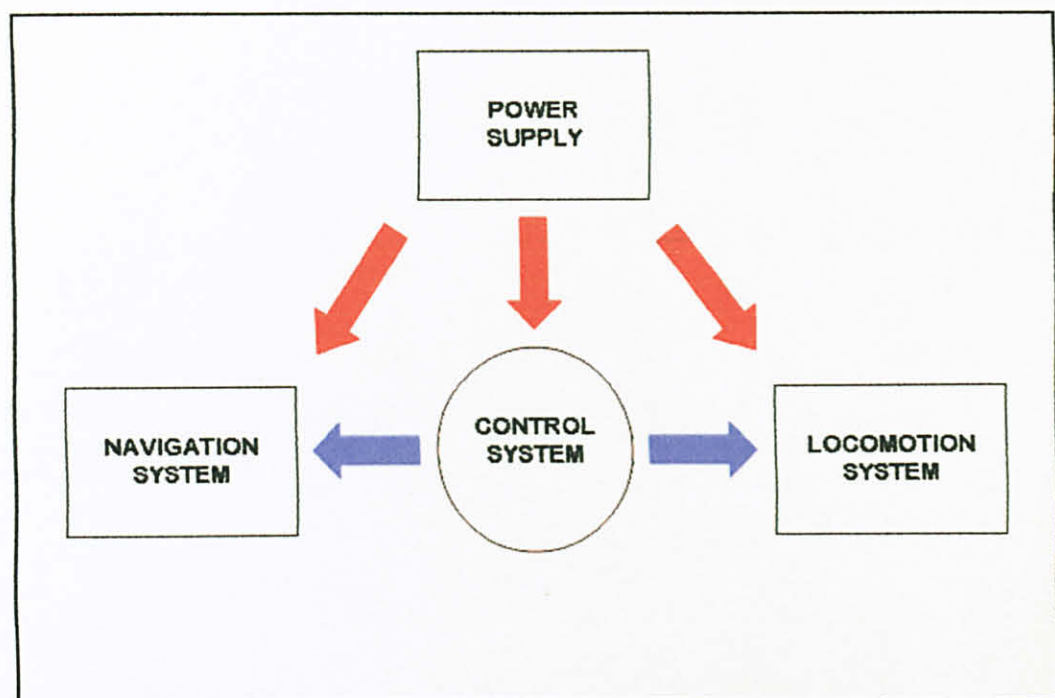


Figure 12: Basic Building Block Diagram

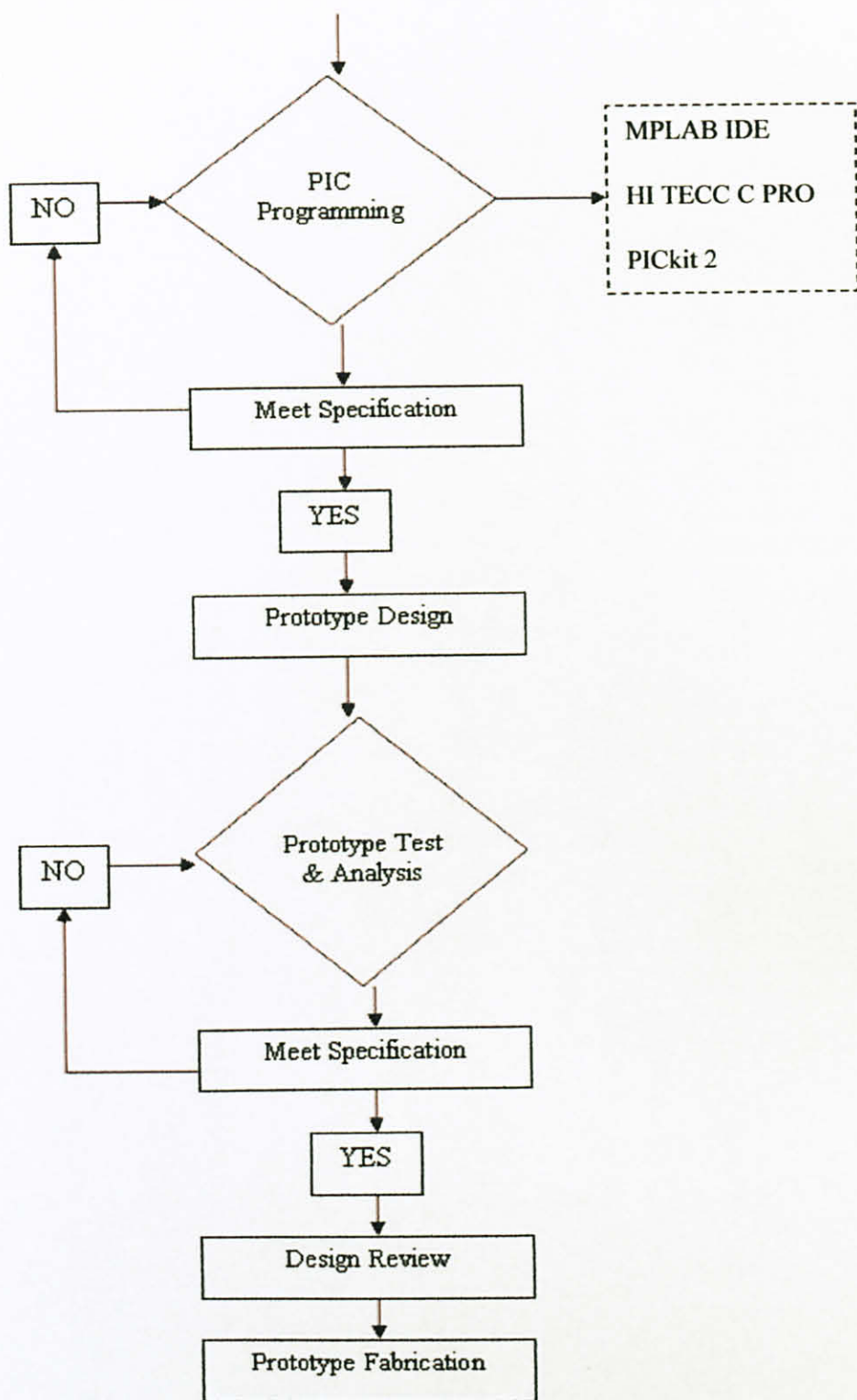


Figure 11: Flow chart for Procedure Identification

CHAPTER 4

RESULT AND DISCUSSION

4.1 DISCUSSION

4.1.1 Navigation System

This system will provide detection that will prevent autonomous robot from collide with obstacles. Thus, obstacles avoidance concept is apply. In consequences, risks and effects on the robot can be minimize. Comparison between ultrasonic sensor and infrared sensor had been made on certain criteria.

Table 2: Comparison between ultrasonic sensor and infrared sensor

Criteria	Ultrasonic sensor	Infrared sensor
Detection range	7cm – 300cm (3m)	10cm – 75cm
Power Supply (Current)	30mA	30mA
Environment	<ul style="list-style-type: none">- Sound absorption by obstacles- Minor dependence on humidity and temperature- Hardness of a obstacle	<ul style="list-style-type: none">- Ambient lighting and brightness- Color of the obstacles

As from the table above, ultrasonic sensor is better compared to infrared. However, ultrasonic sensor is much more expensive and need an external circuit in order to work. On the other hand, many kind of infrared sensor is cheaper and easy to handle as it has its own built-in circuit.

4.1.2 Line Following Implementation

For line following implementation, four pair of infrared sensor is used. It is installed on the bottom of the robot. The four paired sensor is identified as left sensor, middle left sensor, middle right sensor and right sensor. The placement of these sensors is shown in the figure below from top view of the robot.

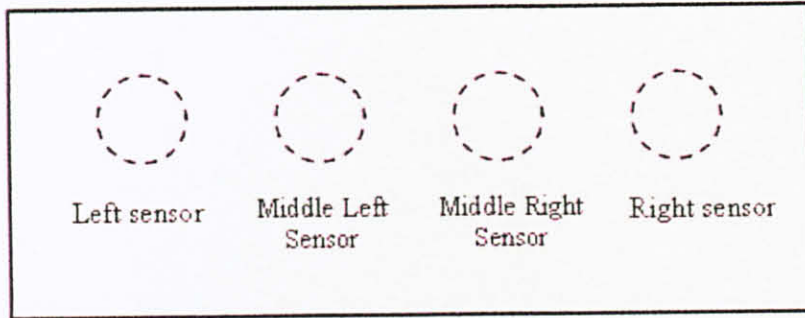


Figure 13: Four pair of infrared sensor

Before the robot can be used for line following mode, it should be calibrated or in other words “teaching” the robot on how to follow black line. Below are the steps that need to be followed:

1. Adjust the robot so that the center infrared sensor is on top of white floor; make sure the wheels and the caster of the robot touches the floor properly.
2. Adjust the preset of the center sensor until center LED is ON
3. Then, adjust the robot to move the center sensor towards the black line where reflection of infrared is poor.
4. Make sure that the LED indicator is OFF. If the LED is still on, that means it is over tune. Tune the preset back until the indicator is OFF.
5. Repeat step 1-4 for a few time and make sure that the LED indicator is OFF and ON correctly.
6. Repeat step 1-5 for left and right sensor.

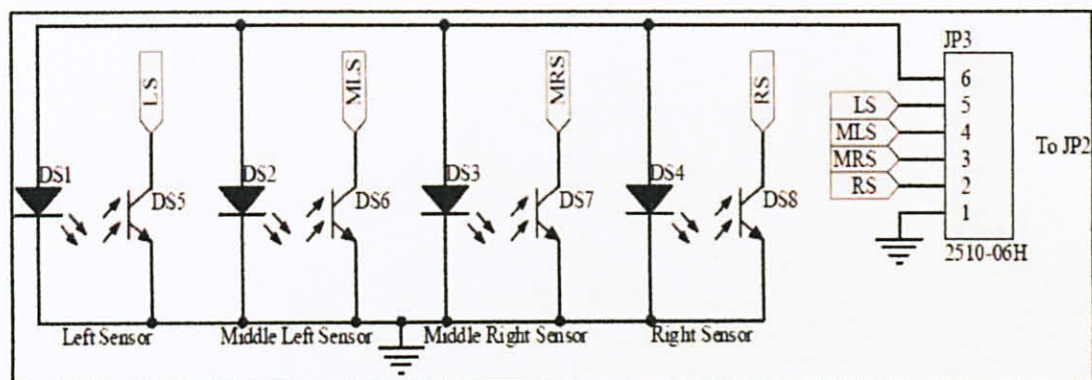


Figure 14: Schematic diagram for the four pair infrared sensor.

JP3 is 2510 is refer as connector for the 6-ways cable that connected to the other end of the cable. LS = left sensor, MLS = middle right sensor, MRS = middle left sensor and RS = right sensor. Figure 13 below show how Infrared sensors are constructed on PCB.

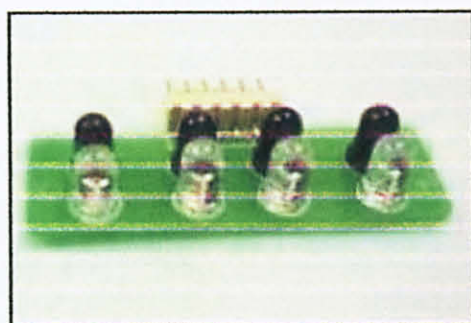


Figure 15: Four pair Infrared Sensors Transmitters and Receivers.

4.1.3 Analog Distance Sensor

In order to provide a 'vision' for the prototype, an analog distance sensor is going to be used. The suitable choice of the sensor is SHARP GP2Y0A21YKOF. This device is composed of an integrated combination of position sensitive device, infrared emitting diode and signal processing circuit. It takes continuous reading and reports the distance as an analog voltage with a distance range of 10cm to 80cm. This sensor has 3-pin that is power, ground and output. Voltage use is 5V and since this sensor does not requires external control circuit; output pin is directly connected to PIC.

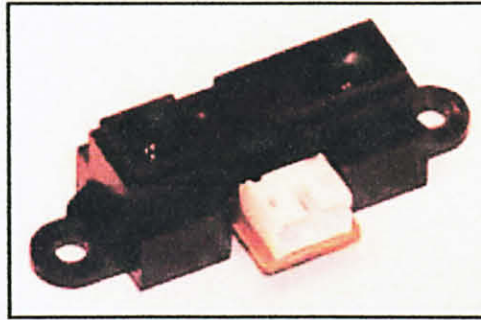


Figure 16: SHARP GP2Y0A21YK

The operation of this sensor is based on the process of triangulation. At first, the sensor will emit a light (wavelength range of 850nm \pm 70nm) and if the light hit any object it will be reflected back and if it not there will be reflection at all. When the light is reflected back to the sensor, it will return at an angle that is dependent to the distance of the object. The process of triangulation works by detecting the angle of this reflected beam and when angle is known the distance between the robot and the sensor can be determine.

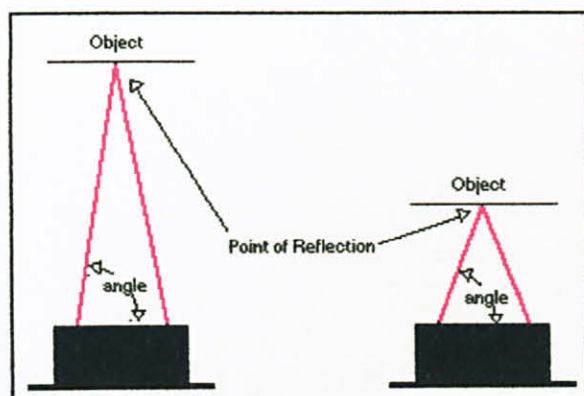


Figure 17: Triangulation method

The lens of the receiver is made from a special lens that transmits the reflected light onto an enclosed linear Charge Coupled Device (CCD) array based on the return angle. The CCD array will determine the angle and the sensor will produce a corresponding analog value that can be read by the microcontroller. The built-in circuit in the sensor applies a modulated frequency to the emitted beam. In consequence, this sensor is immune to interference of ambient light, and also offers amazing indifference to the color of the object being detected.

4.1.4 Locomotion system

Table 3: Comparison between leg, track and wheel

CRITERIA	WHEEL	TRACK	LEG
Size	Does not depend on the robot size.	Similar to the length of the robot	Does not relies on the robot size
Advantages	<ul style="list-style-type: none">- Simple to construct- Provide great maneuvering ability.- Does not require powerful motor	<ul style="list-style-type: none">- Able to mow through sorts of obstacles- Excellent traction and greater stability	Extra level of mobility compared to track and wheel.
Disadvantages	<ul style="list-style-type: none">Unable to drive trough difficult terrain- Easily tip over during turns without proper design	<ul style="list-style-type: none">- Requires powerful motors- Large friction area between track and terrain surface	Difficult to design.

Comparing the size, advantages and disadvantages between those types of locomotion wheel is recommended to be used as part of the driving mechanism.

4.1.4.1 Steering system

The steering system to be applied for the prototype is differential drive system. The forward and reverse direction of the DC motor can be manipulated in order to make the prototype to turn left or right. The DC motor can also be switch off. For example, if the prototype need to turn left, the left DC motor can be switch off or spin on reverse and at the same time the right DC motor move in forward direction. The opposite movement will create an arc-turn and the movement will stop when the angle of turn has been met. Figure below show the differential drive system:-

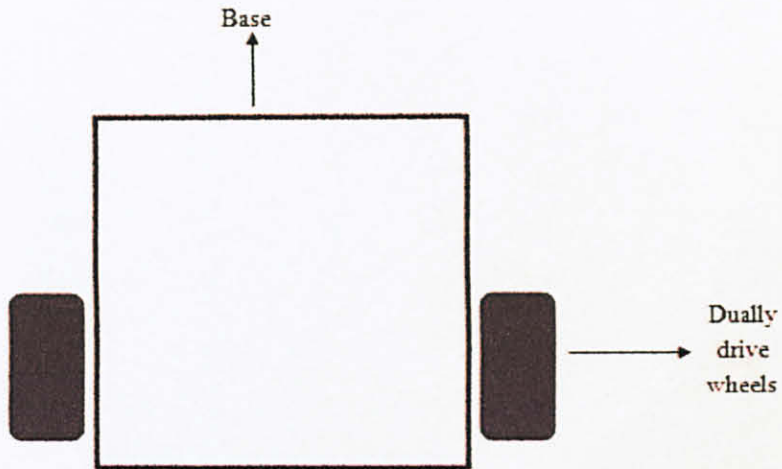


Figure 18: Differential drive system for robot

4.1.4.2 Driving system

The driving system will have two DC motors for each drive wheels. The motor will be mounted according to centerline type as discussed. Two caster wheels will be attached at part of the robot to support the movement. By using this type, the weight is evenly distributed across the robot body or platform. The robot also has no back or front part.

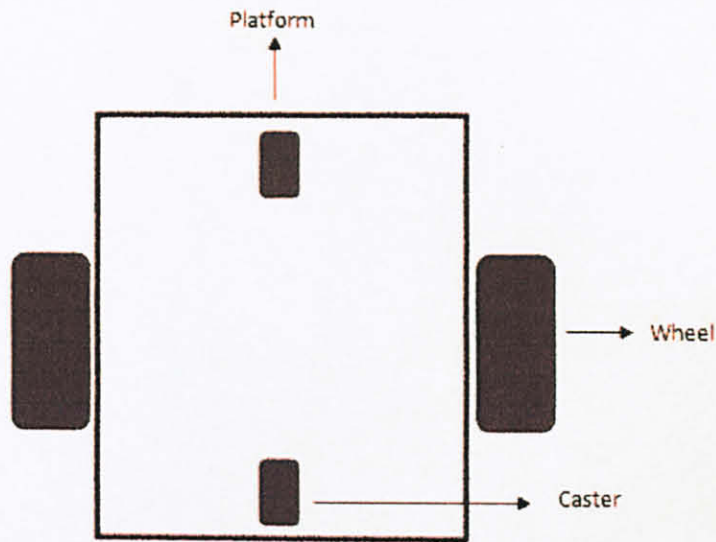


Figure 19: Centerline Drive type

4.1.5 H-Bridge Motor Driver

In this project two modified servo motors are used. The servos can rotate 360 degree which means it can rotate continuously but it is still suitable for small mobile robot. This servo also has only 2 lead wires and do not have the signal wire and it cannot be controlled by pulses. Power supplied for these servos is 6V using four batteries each 1.5V connected in series. Since it has only two lead wires, this servo will act as normal DC motor. Thus, to control a DC motor using microcontroller we need a motor driver. In this case, a motor driver using L298N has been design.

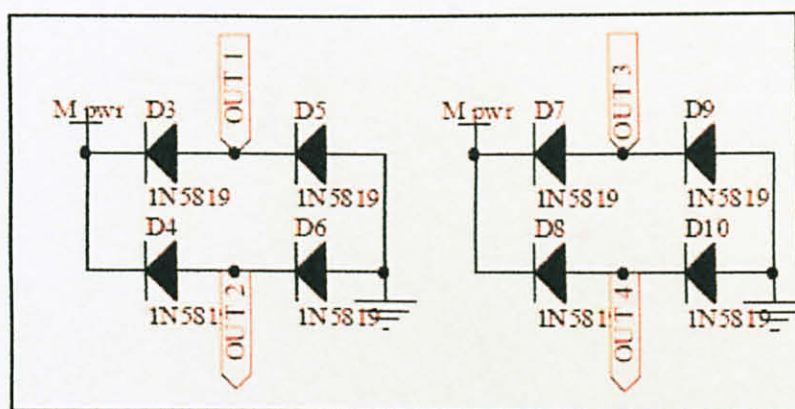
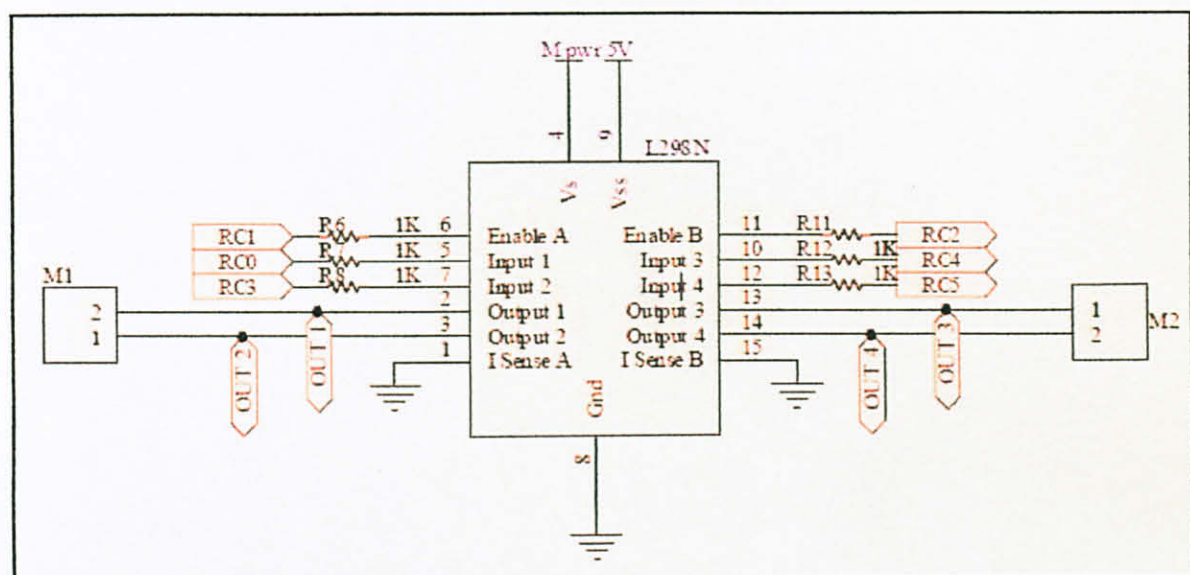


Figure 20: Motor Driver Schematic Using L298N

Based on Figure 16, there are 6 pins out from microcontroller connected to L298N chip, 3 pins for the left servo and the other 3 for the right servo. The other 4 pins which are Output 1 to Output 4 are for power supply for both servos. From the 3 pins that are connected to microcontroller, 2 pins are used to provide direction for servo (RC0, RC3, RC4 and RC5). One more pin is for input of Pulse Width Modulation from microcontroller to control the speed of the motor which are RC1 and RC2. The directions of the servos are depend on the connection of the terminal and can also be determined through the programming part.

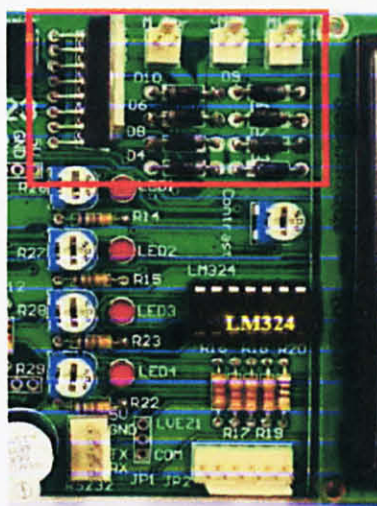


Figure 21: H-Bridge L298N motor control part on the PCB.

4.1.6 Control system

Programmable Interface Controller (PIC) is obviously the best microcontroller that is suitable for this project because of idea availability, large user base, and extensive collection of application notes, availability of low cost or free development tools, and serial programming and re-programming capability. Type of PIC that has been chosen is PIC16F877A. The table below also illustrates few parameters that make PIC16F877A ideal for this project.

Table 4: PIC16F877A parameters

Parameters	Value
Program Memory Type	Flash
Program Memory	4 KB
CPU Speed (MIPS)	5
RAM Bytes	368
Data EEPROM (bytes)	256
Digital Communication Peripherals	1-A/E/USART, 1-MSSP(SPI/I2C)
Capture/Compare/PWM Peripherals	2 CCP
Timers	2 x 8-bit, 1 x 16-bit
Analog To Digital Converter (ADC)	8 ch, 10-bit
Comparators	2
Temperature Range ($^{\circ}\text{C}$)	-40 to 125
Operating Voltage Range (V)	2 to 5.5
Pin Count	40

4.1.7 Power Regulation

Power regulation of a circuit provides the regulation of a set of voltage can supply a minimum required amount of required power and allow for additional features on the application.

In this project, a 9V battery is used to supply the electronic components on Printed Circuit Board (PCB). Microcontroller and sensor are sensitive to the input voltage. If there are any changes in value of voltage supply for both components bad things will happen. Thus, we need to regulate the 9V voltage to 5V using LM7805.

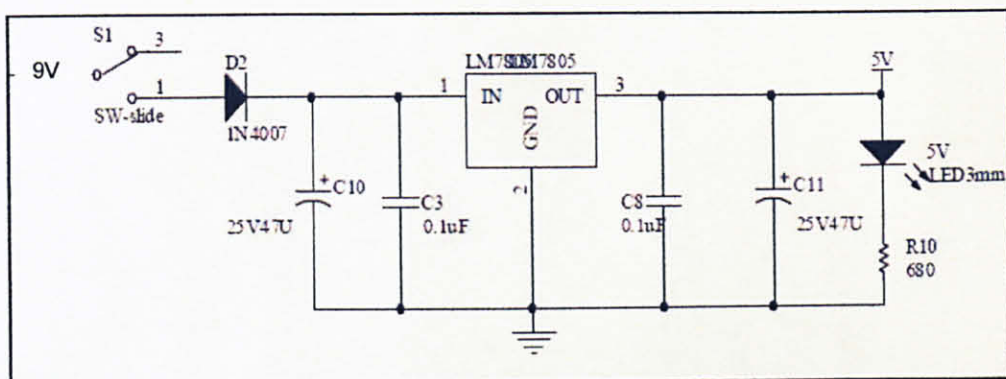


Figure 22: Power Regulation Schematic

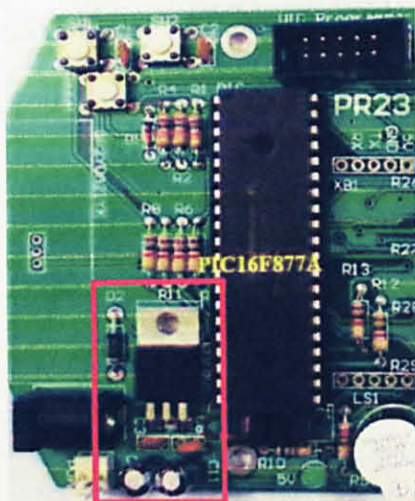


Figure 23: Power Regulation part on PCB

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

Autonomous robots can perform desired tasks without human supervision or guidance. Different kinds of robot will have their own degree of autonomy and have the ability of autonomous in many ways. A robot with high degree of autonomy is desirable to perform difficult task, while lower degree of autonomy robot are more suitable for educational and entertainment purpose.

This project has been developed and able to meet the objectives. The robot has all of the design requirements that has been discuss before. The methodology implemented is important in providing guideline. After several months of research and development, the robot possess the ability of an autonomous robot that can move on its own, detect and avoid obstacles.

5.2 Recommendation

In order to make the prototype to work properly, a few recommendations has been made and listed below:

- The placement of batteries case should be fixed in at the robot platform so that the design could be neater.
- Provide heat sink for component LM7805 and L298N.
- Since the robot has not fully working yet, more tests and troubleshooting should be carry out.
- Testing should be carry out in the target environment such as indoor with not too many obstacles.

If this recommendation is apply, the prototype will achieve the objectives and become more reliable in the future.

REFERENCES

- [1] Robot. 14th August 2009 <<http://en.wikipedia.org/wiki/robot>>
- [2] Robot locomotion. 26th October 2009
<http://en.wikipedia.org/wiki/Robot_locomotion#Bipedal_Walking>
- [3] Robotics/Types of Robots. 26th October 2009
<http://en.wikibooks.org/wiki/Robotics/Types_of_Robots>
- [4] Microcontroller. 25th April 2010
<<http://en.wikipedia.org/wiki/Microcontroller>>
- [5] Gordon McComb, Myke Predko. 2006, *Robot Builder's Bonanza, 3rd Edition*, McGraw-Hill.
- [6] John Lovine, 2004, *PIC Robotics, A Beginner's Guide to Robotics Projects Using the PIC Micro*, McGraw-Hill.
- [7] Schematics Depot – Ultrasonic switch circuit. 15th June 2010
<http://www.reconnsworld.com/ir_ultrasonic_ultraswitch.html>
- [8] Definition of transducer. 4th May 2010
<http://www.ehow.com/about_5097552_definition-transducers.html>
- [9] What a servo: A quick tutorial. 4th May 2010
<<http://www.seattlerobotics.org/guide/servos.html>>
- [10] John M. Holland, 2004, *Designing Autonomous Mobile Robots: Inside the mind of an Intelligent Machine*, Elsevier.
- [11] Control system. 4th May 2010
<http://en.wikipedia.org/wiki/Control_system>

APPENDICES

APPENDIX A – Gantt Chart for FYP II

[illegible]

APPENDIX B

Programming Code

```
#include <pic.h>
```

```
__CONFIG ( 0x3F32 );
```

```
/*define*/
```

```
#define sw1          RE0
```

```
#define sw2          RE1
```

```
#define motor_ra     RC0
```

```
#define motor_rb     RC3
```

```
#define motor_la     RC4
```

```
#define motor_lb     RC5
```

```
#define s_left       RB0
```

```
#define s_mleft      RB1
```

```
#define s_mright     RB2
```

```
#define s_right      RB3
```

```
#define buzzer       RE2
```

```
#define rs           RB7
```

```
#define e            RB6
```

```
#define lcd_data     PORTD
```

```
#define b_light      RB5
```

```
#define SPEEDL       CCPR1L
```

```
#define SPEEDR       CCPR2L
```

```
#define CHANNEL1     0b10001001           // AN1 ( Distance sensor )
```

```
#define RX_PIN       RC7
```

```
#define TX_PIN       RA2
```

```
#define BOT_ADD      100
```


/*global variable*/

unsigned char data[4] = {0};

const unsigned char line [] = {"1.LINE FOLLOW"};

const unsigned char OF[] = {"2.Object Following"};

const unsigned char AS[] = {"3.Analog Sensor"};

const unsigned char *mode [3] = {&line[0],&OF[0],&AS[0]};

unsigned int result;

unsigned int To=0,T=0,TH=0;

unsigned char REC;

unsigned char i=0,raw;

unsigned int us_value (unsigned char mode);

/*function prototype*/

void init(void);

void delay(unsigned long data);

void send_config(unsigned char data);

void send_char(unsigned char data);

void e_pulse(void);

void lcd_goto(unsigned char data);

void lcd_clr(void);

void send_string(const char *s);

void dis_num(unsigned long data);

void line_follow(void);

void object_follow(void);

```
void analog_sen(void);
```

```
void forward(void);
```

```
void stop (void);
```

```
void backward (void);
```

```
void reverse (void);
```

```
void left(void);
```

```
void right(void);
```

```
void read_adc(char config);
```

```
/*interrupt prototype*/
```

```
if (TMROIF)
```

```
{
```

```
    TMROIF = 0;
```

```
    To +=0x100;
```

```
}
```

```
if(RBIF)
```

```
{
```

```
    RBIF = 0;
```

```
    if (RB4)
```

```
    {
```

```
        TMRO = 0;
```

```
        To = 0;
```

```
    }
```

```
    else TH = TMRO + To;
```

```
}
```

```
if(RCIF)
```

```

{
    RCIF = 0;
    if (RCREG == 'R') data[i=0]= RCREG;
    else if (RCREG == 100) data[i=0]= RCREG;
    if ((data[0] == 'R'))data [i++] = RCREG;
    if (i>4) i = 4;
}
}

```

/*main function*/

void main(void)

```

{

    unsigned char m=0,i =0;
    delay(20000);
    init();
    buzzer = 1;
    lcd_clr();
    send_string("Select mode");
    lcd_goto(20);
    send_string(mode[m]);
    buzzer = 0;

    while(1)
    {
        if( !sw1)
        {
            while(!sw1);

```

```

        m++;

        if ( m > 2) m = 0;

        lcd_goto(20);

        send_string(mode[m]);

        send_string("  ");

    }

```

```

    if (!sw2)
    {
        while(!sw2);

        switch(m)
        {
            case 0 : line_follow();
                     break;

            case 1 : object_follow();
                     break;

            case 2 : analog_sen();
                     break;

            default :      ;

        }
    }

```

```

}

```

```

}

```

/*initialize the microcontroller*/

```

void init()

```

```

{

```

```

    // ADC configuration

```

```

    ADCON1 = 0b10000100;

```

```

    // setup for capture pwm

```



```
RBIE = 1;

// motor PWM configuration
PR2 = 255;

T2CON =      0b000000100;
CCP1CON =    0b000001100;
CCP2CON =    0b000001100;


// Tris configuration (input or output)
TRISA = 0b000000011;
TRISB = 0b000111111;
TRISC = 0b100000000;
TRISD = 0b000000000;
TRISE = 0b000000011;


// TMR 0 configuration
TOCS = 0;
PSA = 0;
PS2 = 1;
PS1 = 1;
PS0 = 1;
TMROIE = 1;
TMRO = 0;


//setup UART
SPBRG = 0x81;
BRGH = 1;
TXEN = 1;
TX9 = 0;
CREN = 1;
```

```
SPEN = 1;
```

```
RX9 = 0;
```

```
RCIE = 1;
```

```
// enable all unmasked interrupt
```

```
GIE = 1;
```

```
PEIE = 1;
```

```
// LCD configuration
```

```
send_config(0b00000001);
```

```
send_config(0b00000010);
```

```
send_config(0b00000110);
```

```
send_config(0b00001100);
```

```
send_config(0b00111000);
```

```
TX_PIN = 1;
```

```
b_light = 0;
```

```
buzzer = 0;
```

```
stop();
```

```
b_light = 0;
```

```
buzzer = 0;
```

```
stop();
```

```
}
```

```
/*mode 1: line follow*/
```

```
void line_follow()
```

```
{
```

```
    unsigned char memory;
```

```

lcd_clr();

send_string("Position");

while(1)
{
    if ((s_left==1)&&(s_mleft==0)&&(s_mright==0)&&(s_right==0))

    {
        forward();
        SPEEDL = 0;
        SPEEDR = 255;
        memory = PORTB&0b00001111;
        lcd_goto(20);
        send_string ("right ");
    }

    else if ((s_left==1)&&(s_mleft==1)&&(s_mright==0)&&(s_right==0))

    {
        forward();
        SPEEDL = 180;
        SPEEDR = 255;
        memory = PORTB&0b00001111;
        lcd_goto(20);
        send_string ("m_right2");
    }

    else if ((s_left==0)&&(s_mleft==1)&&(s_mright==0)&&(s_right==0))

    {
        forward();
        SPEEDL = 200;
        SPEEDR = 255;
        memory = PORTB&0b00001111;
        lcd_goto(20);
        send_string ("m_right1 ");
    }

    else if ((s_left==1)&&(s_mleft==1)&&(s_mright==1)&&(s_right==0))

    {
        forward();
        SPEEDL = 200;
        SPEEDR = 255;
        memory = PORTB&0b00001111;
        lcd_goto(20);
        send_string ("m_right1 ");
    }

    else if ((s_left==0)&&(s_mleft==1)&&(s_mright==1)&&(s_right==0))

```

```

{
    forward();
        SPEEDL = 255;
            SPEEDR = 255;
                memory = PORTB&0b00001111;
    lcd_goto(20);
    send_string ("middle ");
}

else if ((s_left==0)&&(s_mleft==0)&&(s_mright==1)&&(s_right==0))

{
    forward();
        SPEEDL = 255;
            SPEEDR = 200;
                memory = PORTB&0b00001111;
    lcd_goto(20);
    send_string ("m_left1 ");
}

else if ((s_left==0)&&(s_mleft==1)&&(s_mright==1)&&(s_right==1))

{
    forward();
        SPEEDL = 255;
            SPEEDR = 200;
                memory = PORTB&0b00001111;
    lcd_goto(20);
    send_string ("m_left1 ");
}

else if ((s_left==0)&&(s_mleft==0)&&(s_mright==1)&&(s_right==1))

{
    forward();
        SPEEDL = 255;
            SPEEDR = 180;
                memory = PORTB&0b00001111;
    lcd_goto(20);
    send_string ("m_left2 ");
}

else if ((s_left==0)&&(s_mleft==0)&&(s_mright==0)&&(s_right==1))

{
    forward();
        SPEEDL = 255;
            SPEEDR = 0;
                memory = PORTB&0b00001111;
    lcd_goto(20);
    send_string ("left ");
}

```



```

else if ((s_left==0)&&(s_mleft==0)&&(s_mright==0)&&(s_right==0))

{
    forward();
    if ((memory == 0b00000001) || (memory ==
0b00000011) || (memory ==
0b0000010) || (memory == 0b0000111))
    {
        SPEEDL = 0;
        SPEEDR = 255;
    }

    else if ((memory == 0b00001000) || (memory ==
0b0000100) || (memory == 0b00001100) || (memory == 0b0001110))
    {
        SPEEDL = 255;
        SPEEDR = 0;
    }
}

else if ((s_left==1)&&(s_mleft==1)&&(s_mright==1)&&(s_right==1))
{
    forward();
}

}

}

```

/*mode 2: object following*/

```

void object_follow(void)
{
    int distance;
    lcd_clr();
    send_string("Distance");

    while(1)
    {
        lcd_goto(20);
        read_adc(CHANNEL1);
    }
}

```

```

        distance = result;
        dis_num(result);
        if (distance > 333)
        {

            stop();
            buzzer = 0;

        }
        else if (distance > 280)

        {

            backward();
            SPEEDL = 255;
            SPEEDR = 255;
            buzzer = 0;

        }
    }
}

```

/*mode 3: analog sensor*/

```

void analog_sen(void)
{
    int distance;
    lcd_clr();
    send_string("Distance");
    while(1)
    {
        lcd_goto(20);
        read_adc(CHANNEL1);
        distance = result;
        dis_num(result);
        if (distance < 200)
        {

```

```
        backward();  
        SPEEDL = 255;  
        SPEEDR = 255;  
        buzzer = 0;  
    }  
    else if (distance < 250)  
    {  
        backward();  
        SPEEDL = 230;  
        SPEEDR = 230;  
        buzzer = 0;  
    }  
    else if( distance < 300)  
    {  
        stop();  
        buzzer = 0;  
    }  
    else  
    {  
        right();  
        SPEEDL = 230;  
        SPEEDR = 230;  
        buzzer = 1;  
    }  
}  
}
```

```
/*read adc*/
```

```
void read_adc(char config)
```

```
{
```

```
    unsigned short i;
```

```
    unsigned long result_temp=0;
```

```
    ADCON0 = config;
```

```
    delay(10000);
```

```
    for(i=200;i>0;i-=1)
```

```
    {
```

```
        ADGO = 1;
```

```
        while(ADGO==1);
```

```
        result=ADRESH;
```

```
        result=result<<8;
```

```
        result=result|ADRESL;
```

```
        result_temp+=result;
```

```
    }
```

```
    result = result_temp/200;
```

```
    ADON = 0;
```

```
}
```



```
/*motor control functions*/
```

```
void forward ()
```

```
{  
  
    motor_ra = 0;  
    motor_rb = 1;  
    motor_la = 0;  
    motor_lb = 1;  
}
```

```
void backward ()
```

```
{  
  
    motor_ra = 1;  
    motor_rb = 0;  
    motor_la = 1;  
    motor_lb = 0;  
}
```

```
void left()
```

```
{  
  
    motor_la = 1;  
    motor_lb = 0;  
    motor_ra = 0;  
    motor_rb = 1;  
}
```

```
void right()
```

```
{
```

```
motor_la = 0;
motor_lb = 1;
motor_ra = 1;
motor_rb = 0;
}
```

```
void stop()
```

```
{
    motor_la = 1;
    motor_lb = 1;
    motor_ra = 1;
    motor_rb = 1;
}
```

```
/*LCD functions*/
```

```
void delay(unsigned long data)
```

```
{
    for( ;data>0;data-=1);
}
```

```
void send_config(unsigned char data)
```

```
{
    rs=0;
    lcd_data=data;
    delay(400);
    e_pulse();
}
```

```
void send_char(unsigned char data)
```

```
{
```

```

    rs=1;

    lcd_data=data;

    delay(400);

    e_pulse();
}

void e_pulse(void)
{
    e=1;

    delay(300);

    e=0;

    delay(300);
}

void lcd_goto(unsigned char data)
{
    if(data<16)
    {
        send_config(0x80+data);
    }
    else
    {
        data=data-20;

        send_config(0xc0+data);
    }
}

void lcd_clr(void)
{
    send_config(0x01);

```

```
    delay(350);  
}  
  
void send_string(const char *s)  
{  
    while (s && *s)send_char (*s++);  
}  
  
void dis_num(unsigned long data)  
{  
    unsigned char hundred_thousand;  
    unsigned char ten_thousand;  
    unsigned char thousand;  
    unsigned char hundred;  
    unsigned char tenth;  
    hundred_thousand = data/100000;  
    data = data % 100000;  
    ten_thousand = data/10000;  
    data = data % 10000;  
    thousand = data / 1000;  
    data = data % 1000;  
    hundred = data / 100;  
    data = data % 100;  
    tenth = data / 10;  
    data = data % 10;  
  
    send_char(hundred_thousand + 0x30);  
    send_char(ten_thousand + 0x30);  
    send_char(thousand + 0x30);  
    send_char(hundred + 0x30);  
    send_char(tenth + 0x30);
```



```
send_char(data + 0x30);
```

```
}
```

APPENDIX C
PIC16F87XA DATASHEET



PIC16F87XA

Data Sheet

**28/40/44-Pin Enhanced Flash
Microcontrollers**

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KEELoQ, MPLAB, PIC, PICmicro, PICSTART, PRO MATE and PowerSmart are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


Amplab, FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Application Maestro, dsPICDEM, dsPICDEM.net, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICkit, PICDEM, PICDEM.net, PowerCal, PowerInfo, PowerMate, PowerTool, rLAB, rPIC, Select Mode, SmartSensor, SmartShunt, SmartTel and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2003, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.



Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELoQ® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

28/40/44-Pin Enhanced Flash Microcontrollers

Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

High-Performance RISC CPU:

- Only 35 single-word instructions to learn
- All single-cycle instructions except for program branches, which are two-cycle
- Operating speed: DC – 20 MHz clock input
DC – 200 ns instruction cycle
- Up to 8K x 14 words of Flash Program Memory,
Up to 368 x 8 bytes of Data Memory (RAM),
Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin
PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler,
can be incremented during Sleep via external
crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period
register, prescaler and postscaler
- Two Capture, Compare, PWM modules
 - Capture is 16-bit, max. resolution is 12.5 ns
 - Compare is 16-bit, max. resolution is 200 ns
 - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™
(Master mode) and I²C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver
Transmitter (USART/SCI) with 9-bit address
detection
- Parallel Slave Port (PSP) – 8 bits wide with
external RD, WR and CS controls (40/44-pin only)
- Brown-out detection circuitry for
Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital
Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
 - Two analog comparators
 - Programmable on-chip voltage reference
(VREF) module
 - Programmable input multiplexing from device
inputs and internal voltage reference
 - Comparator outputs are externally accessible

Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced Flash
program memory typical
- 1,000,000 erase/write cycle Data EEPROM
memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™)
via two pins
- Single-supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC
oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

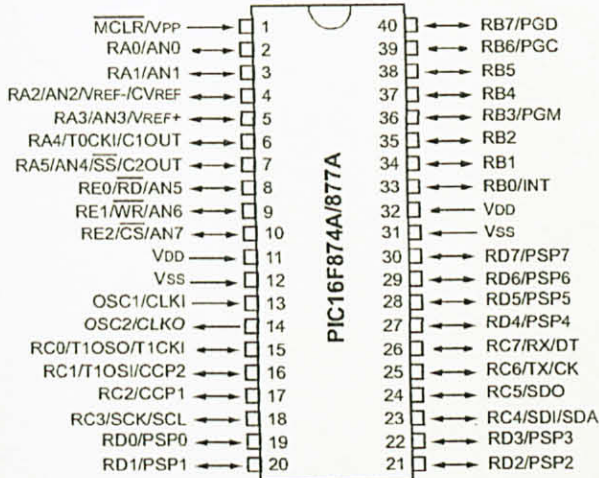
CMOS Technology:

- Low-power, high-speed Flash/EEPROM
technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

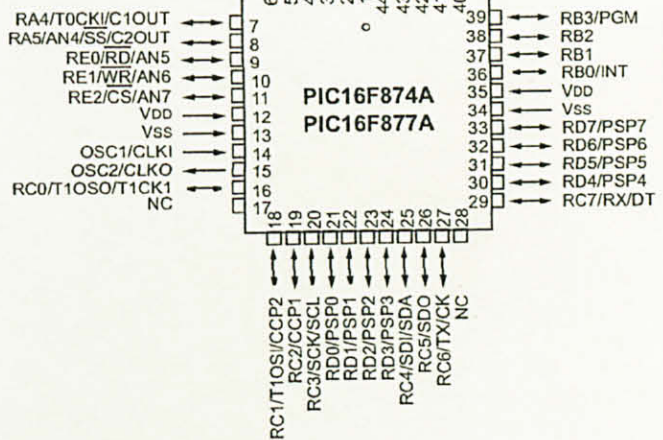
Device	Program Memory		Data SRAM (Bytes)	EEPROM (Bytes)	I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8/16-bit	Comparators
	Bytes	# Single Word Instructions						SPI	Master I²C			
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

Pin Diagrams (Continued)

40-Pin PDIP



44-Pin PLCC



44-Pin TQFP

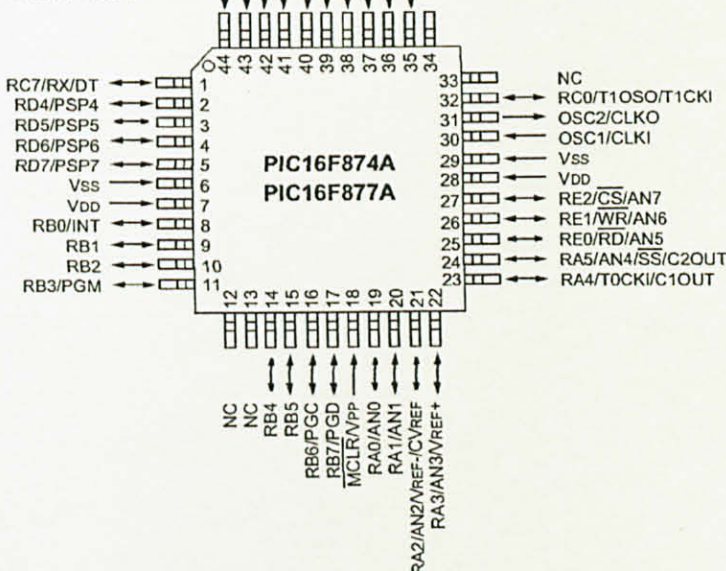
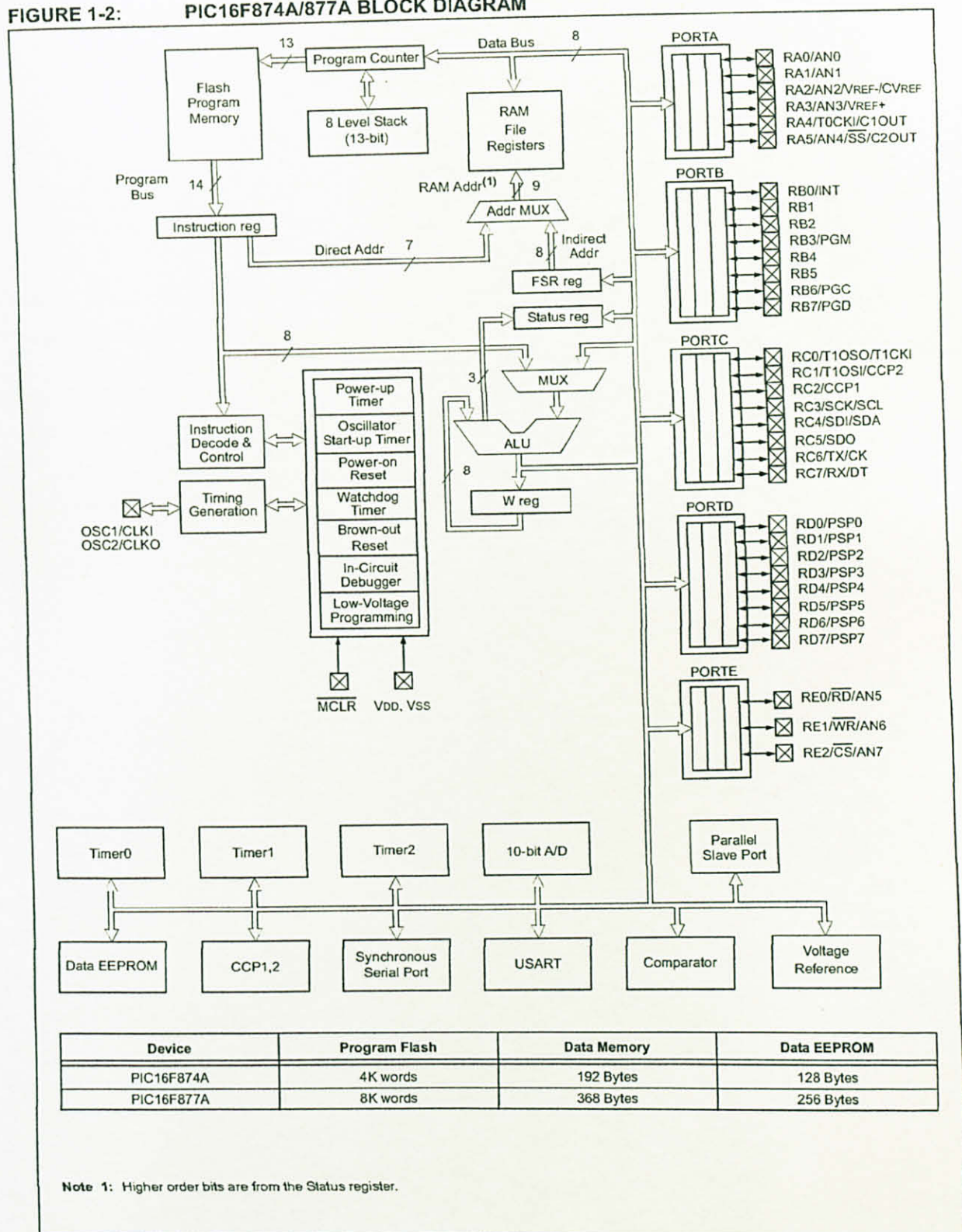


FIGURE 1-2: PIC16F874A/877A BLOCK DIAGRAM



2.0 MEMORY ORGANIZATION

There are three memory blocks in each of the PIC16F87XA devices. The program memory and data memory have separate buses so that concurrent access can occur and is detailed in this section. The EEPROM data memory block is detailed in **Section 3.0 "Data EEPROM and Flash Program Memory"**.

Additional information on device memory may be found in the PICmicro® Mid-Range MCU Family Reference Manual (DS33023).

2.1 Program Memory Organization

The PIC16F87XA devices have a 13-bit program counter capable of addressing an 8K word x 14 bit program memory space. The PIC16F876A/877A devices have 8K words x 14 bits of Flash program memory, while PIC16F873A/874A devices have 4K words x 14 bits. Accessing a location above the physically implemented address will cause a wraparound.

The Reset vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 2-1: PIC16F876A/877A PROGRAM MEMORY MAP AND STACK

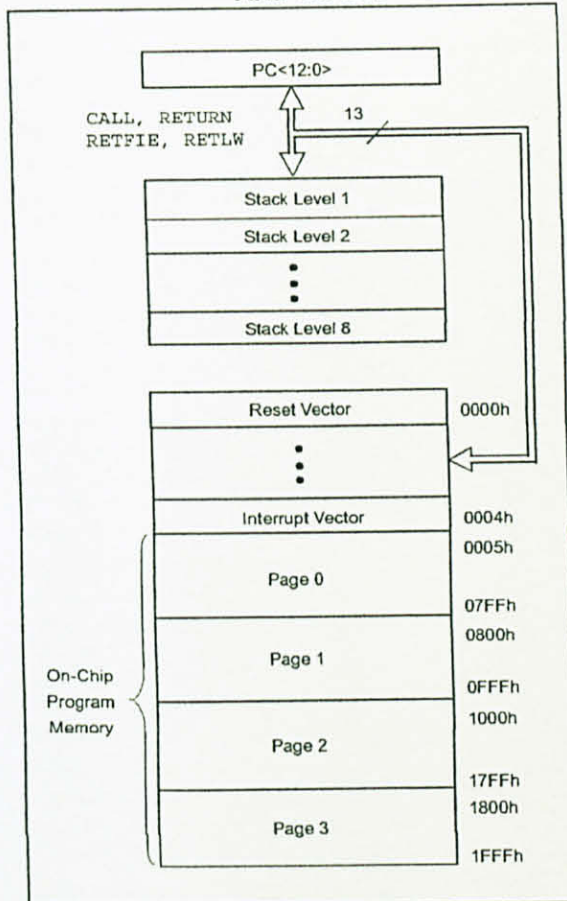
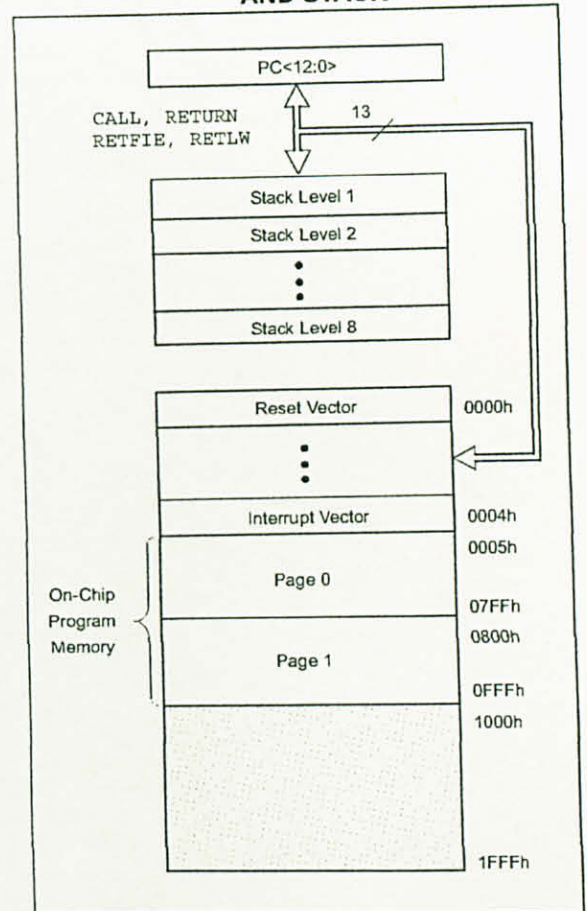


FIGURE 2-2: PIC16F873A/874A PROGRAM MEMORY MAP AND STACK



3.0 DATA EEPROM AND FLASH PROGRAM MEMORY

The data EEPROM and Flash program memory is readable and writable during normal operation (over the full VDD range). This memory is not directly mapped in the register file space. Instead, it is indirectly addressed through the Special Function Registers. There are six SFRs used to read and write this memory:

- EECON1
- EECON2
- EEDATA
- EEDATH
- EEADR
- EEADRH

When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write and EEADR holds the address of the EEPROM location being accessed. These devices have 128 or 256 bytes of data EEPROM (depending on the device), with an address range from 00h to FFh. On devices with 128 bytes, addresses from 80h to FFh are unimplemented and will wraparound to the beginning of data EEPROM memory. When writing to unimplemented locations, the on-chip charge pump will be turned off.

When interfacing the program memory block, the EEDATA and EEDATH registers form a two-byte word that holds the 14-bit data for read/write and the EEADR and EEADRH registers form a two-byte word that holds the 13-bit address of the program memory location being accessed. These devices have 4 or 8K words of program Flash, with an address range from 0000h to 0FFFh for the PIC16F873A/874A and 0000h to 1FFFh for the PIC16F876A/877A. Addresses above the range of the respective device will wraparound to the beginning of program memory.

The EEPROM data memory allows single-byte read and write. The Flash program memory allows single-word reads and four-word block writes. Program memory write operations automatically perform an erase-before-write on blocks of four words. A byte write in data EEPROM memory automatically erases the location and writes the new data (erase-before-write).

The write time is controlled by an on-chip timer. The write/erase voltages are generated by an on-chip charge pump, rated to operate over the voltage range of the device for byte or word operations.

When the device is code-protected, the CPU may continue to read and write the data EEPROM memory. Depending on the settings of the write-protect bits, the device may or may not be able to write certain blocks of the program memory; however, reads of the program memory are allowed. When code-protected, the device programmer can no longer access data or program memory; this does NOT inhibit internal reads or writes.

3.1 EEADR and EEADRH

The EEADRH:EEADR register pair can address up to a maximum of 256 bytes of data EEPROM or up to a maximum of 8K words of program EEPROM. When selecting a data address value, only the LSByte of the address is written to the EEADR register. When selecting a program address value, the MSByte of the address is written to the EEADRH register and the LSByte is written to the EEADR register.

If the device contains less memory than the full address reach of the address register pair, the Most Significant bits of the registers are not implemented. For example, if the device has 128 bytes of data EEPROM, the Most Significant bit of EEADR is not implemented on access to data EEPROM.

3.2 EECON1 and EECON2 Registers

EECON1 is the control register for memory accesses.

Control bit, EEPGD, determines if the access will be a program or data memory access. When clear, as it is when reset, any subsequent operations will operate on the data memory. When set, any subsequent operations will operate on the program memory.

Control bits, RD and WR, initiate read and write or erase, respectively. These bits cannot be cleared, only set, in software. They are cleared in hardware at completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental, premature termination of a write operation.

The WREN bit, when set, will allow a write or erase operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write (or erase) operation is interrupted by a MCLR or a WDT Time-out Reset during normal operation. In these situations, following Reset, the user can check the WRERR bit and rewrite the location. The data and address will be unchanged in the EEDATA and EEADR registers.

Interrupt flag bit, EEIF in the PIR2 register, is set when the write is complete. It must be cleared in software.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the EEPROM write sequence.

Note: The self-programming mechanism for Flash program memory has been changed. On previous PIC16F87X devices, Flash programming was done in single-word erase/write cycles. The newer PIC18F87XA devices use a four-word erase/write cycle. See Section 3.6 "Writing to Flash Program Memory" for more information.

4.0 I/O PORTS

Some pins for these I/O ports are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

Additional information on I/O ports may be found in the PICmicro™ Mid-Range Reference Manual (DS33023).

4.1 PORTA and the TRISA Register

PORTA is a 6-bit wide, bidirectional port. The corresponding data direction register is TRISA. Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin).

Reading the PORTA register reads the status of the pins, whereas writing to it will write to the port latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, the value is modified and then written to the port data latch.

Pin RA4 is multiplexed with the Timer0 module clock input to become the RA4/T0CKI pin. The RA4/T0CKI pin is a Schmitt Trigger input and an open-drain output. All other PORTA pins have TTL input levels and full CMOS output drivers.

Other PORTA pins are multiplexed with analog inputs and the analog VREF input for both the A/D converters and the comparators. The operation of each pin is selected by clearing/setting the appropriate control bits in the ADCON1 and/or CMCON registers.

Note: On a Power-on Reset, these pins are configured as analog inputs and read as '0'. The comparators are in the off (digital) state.

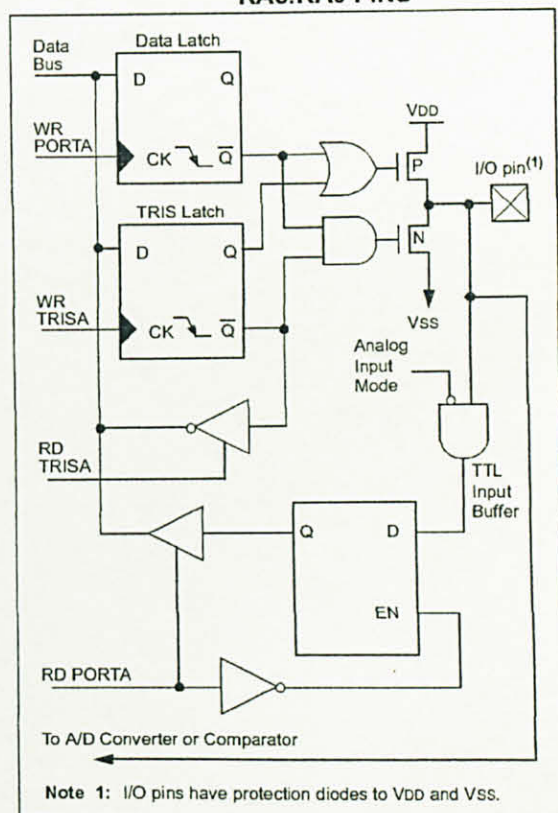
The TRISA register controls the direction of the port pins even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs.

EXAMPLE 4-1: INITIALIZING PORTA

```
BCF    STATUS, RP0 ;
BCF    STATUS, RP1 ; Bank0
CLRF   PORTA       ; Initialize PORTA by
                   ; clearing output
                   ; data latches

BSF    STATUS, RP0 ; Select Bank 1
MOVLW  0x06        ; Configure all pins
MOVWF  ADCON1       ; as digital inputs
MOVLW  0xCF        ; Value used to
                   ; initialize data
                   ; direction
MOVWF  TRISA        ; Set RA<3:0> as inputs
                   ; RA<5:4> as outputs
                   ; TRISA<7:6> are always
                   ; read as '0'.
```

FIGURE 4-1: BLOCK DIAGRAM OF RA3:RA0 PINS



8.0 CAPTURE/COMPARE/PWM MODULES

Each Capture/Compare/PWM (CCP) module contains a 16-bit register which can operate as a:

- 16-bit Capture register
- 16-bit Compare register
- PWM Master/Slave Duty Cycle register

Both the CCP1 and CCP2 modules are identical in operation, with the exception being the operation of the special event trigger. Table 8-1 and Table 8-2 show the resources and interactions of the CCP module(s). In the following sections, the operation of a CCP module is described with respect to CCP1. CCP2 operates the same as CCP1 except where noted.

CCP1 Module:

Capture/Compare/PWM Register 1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. The special event trigger is generated by a compare match and will reset Timer1.

CCP2 Module:

Capture/Compare/PWM Register 2 (CCPR2) is comprised of two 8-bit registers: CCPR2L (low byte) and CCPR2H (high byte). The CCP2CON register controls the operation of CCP2. The special event trigger is generated by a compare match and will reset Timer1 and start an A/D conversion (if the A/D module is enabled).

Additional information on CCP modules is available in the PICmicro® Mid-Range MCU Family Reference Manual (DS33023) and in application note AN594, "Using the CCP Module(s)" (DS00594).

TABLE 8-1: CCP MODE – TIMER RESOURCES REQUIRED

CCP Mode	Timer Resource
Capture	Timer1
Compare	Timer1
PWM	Timer2

TABLE 8-2: INTERACTION OF TWO CCP MODULES

CCPx Mode	CCPy Mode	Interaction
Capture	Capture	Same TMR1 time base
Capture	Compare	The compare should be configured for the special event trigger which clears TMR1
Compare	Compare	The compare(s) should be configured for the special event trigger which clears TMR1
PWM	PWM	The PWMs will have the same frequency and update rate (TMR2 interrupt)
PWM	Capture	None
PWM	Compare	None

10.0 ADDRESSABLE UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (USART)

The Universal Synchronous Asynchronous Receiver Transmitter (USART) module is one of the two serial I/O modules. (USART is also known as a Serial Communications Interface or SCI.) The USART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers, or it can be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The USART can be configured in the following modes:

- Asynchronous (full-duplex)
- Synchronous – Master (half-duplex)
- Synchronous – Slave (half-duplex)

Bit SPEN (RCSTA<7>) and bits TRISC<7:6> have to be set in order to configure pins RC6/TX/CK and RC7/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter.

The USART module also has a multi-processor communication capability using 9-bit address detection.

REGISTER 10-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER (ADDRESS 98h)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

- bit 7 **CSRC:** Clock Source Select bit
Asynchronous mode:
 Don't care.
Synchronous mode:
 1 = Master mode (clock generated internally from BRG)
 0 = Slave mode (clock from external source)
- bit 6 **TX9:** 9-bit Transmit Enable bit
 1 = Selects 9-bit transmission
 0 = Selects 8-bit transmission
- bit 5 **TXEN:** Transmit Enable bit
 1 = Transmit enabled
 0 = Transmit disabled
Note: SREN/CREN overrides TXEN in Sync mode.
- bit 4 **SYNC:** USART Mode Select bit
 1 = Synchronous mode
 0 = Asynchronous mode
- bit 3 **Unimplemented:** Read as '0'
- bit 2 **BRGH:** High Baud Rate Select bit
Asynchronous mode:
 1 = High speed
 0 = Low speed
Synchronous mode:
 Unused in this mode.
- bit 1 **TRMT:** Transmit Shift Register Status bit
 1 = TSR empty
 0 = TSR full
- bit 0 **TX9D:** 9th bit of Transmit Data, can be Parity bit

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared x = Bit is unknown

11.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) Converter module has five inputs for the 28-pin devices and eight for the 40/44-pin devices.

The conversion of an analog input signal results in a corresponding 10-bit digital number. The A/D module has high and low-voltage reference input that is software selectable to some combination of VDD, VSS, RA2 or RA3.

The A/D converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D clock must be derived from the A/D's internal RC oscillator.

The A/D module has four registers. These registers are:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)

The ADCON0 register, shown in Register 11-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 11-2, configures the functions of the port pins. The port pins can be configured as analog inputs (RA3 can also be the voltage reference) or as digital I/O.

Additional information on using the A/D module can be found in the PICmicro® Mid-Range MCU Family Reference Manual (DS33023).

REGISTER 11-1: ADCON0 REGISTER (ADDRESS 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0:** A/D Conversion Clock Select bits (ADCON0 bits in **bold**)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	FRC (clock derived from the internal A/D RC oscillator)

bit 5-3 **CHS2:CHS0:** Analog Channel Select bits

000 = Channel 0 (AN0)
 001 = Channel 1 (AN1)
 010 = Channel 2 (AN2)
 011 = Channel 3 (AN3)
 100 = Channel 4 (AN4)
 101 = Channel 5 (AN5)
 110 = Channel 6 (AN6)
 111 = Channel 7 (AN7)

Note: The PIC16F873A/876A devices only implement A/D channels 0 through 4; the unimplemented selections are reserved. Do not select any unimplemented channels with these devices.

bit 2 **GO/DONE:** A/D Conversion Status bit

When ADON = 1:

1 = A/D conversion in progress (setting this bit starts the A/D conversion which is automatically cleared by hardware when the A/D conversion is complete)
 0 = A/D conversion not in progress

bit 1 **Unimplemented:** Read as '0'

bit 0 **ADON:** A/D On bit

1 = A/D converter module is powered up
 0 = A/D converter module is shut-off and consumes no operating current

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

APPENDIX D

78XX VOLTAGE REGULATOR DATASHEET

KA78XX/KA78XXA

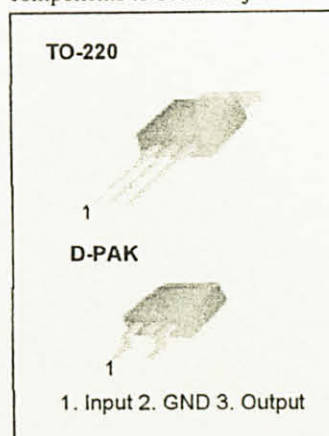
3-Terminal 1A Positive Voltage Regulator

Features

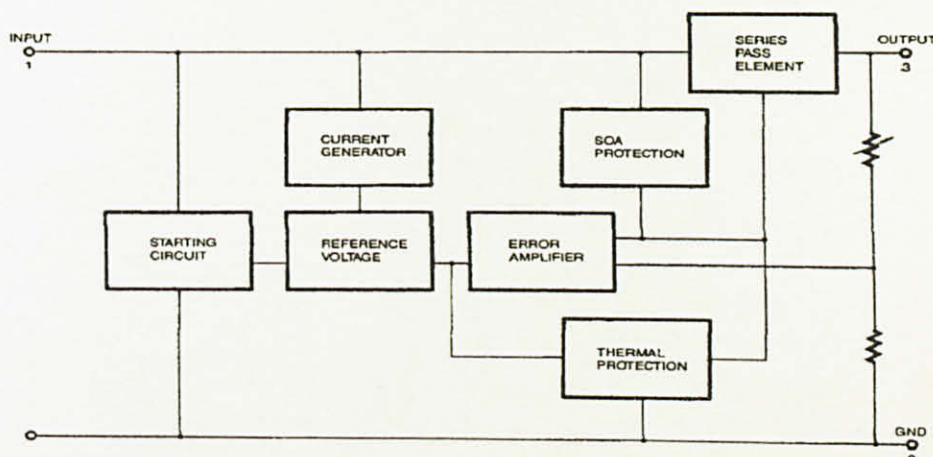
- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

Description

The KA78XX/KA78XXA series of three-terminal positive regulator are available in the TO-220/D-PAK package and with several fixed output voltages, making them useful in a wide range of applications. Each type employs internal current limiting, thermal shut down and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.



Internal Block Diagram



Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Input Voltage (for $V_O = 5V$ to $18V$)	V_I	35	V
(for $V_O = 24V$)	V_I	40	V
Thermal Resistance Junction-Cases (TO-220)	$R_{\theta JC}$	5	$^{\circ}C/W$
Thermal Resistance Junction-Air (TO-220)	$R_{\theta JA}$	65	$^{\circ}C/W$
Operating Temperature Range (KA78XX/A/R)	T_{OPR}	$0 \sim +125$	$^{\circ}C$
Storage Temperature Range	T_{STG}	$-65 \sim +150$	$^{\circ}C$

Electrical Characteristics (KA7805/KA7805R)

(Refer to test circuit, $0^{\circ}C < T_J < 125^{\circ}C$, $I_O = 500mA$, $V_I = 10V$, $C_I = 0.33\mu F$, $C_O = 0.1\mu F$, unless otherwise specified)

Parameter	Symbol	Conditions	KA7805			Unit
			Min.	Typ.	Max.	
Output Voltage	V_O	$T_J = +25^{\circ}C$	4.8	5.0	5.2	V
		$5.0mA \leq I_O \leq 1.0A$, $P_O \leq 15W$ $V_I = 7V$ to $20V$	4.75	5.0	5.25	
Line Regulation (Note1)	Regline	$T_J = +25^{\circ}C$	$V_O = 7V$ to $25V$		100	mV
			$V_I = 8V$ to $12V$		50	
Load Regulation (Note1)	Regload	$T_J = +25^{\circ}C$	$I_O = 5.0mA$ to $1.5A$		100	mV
			$I_O = 250mA$ to $750mA$		50	
Quiescent Current	I_Q	$T_J = +25^{\circ}C$	-	5.0	8.0	mA
Quiescent Current Change	ΔI_Q	$I_O = 5mA$ to $1.0A$	-	0.03	0.5	mA
		$V_I = 7V$ to $25V$	-	0.3	1.3	
Output Voltage Drift	$\Delta V_O / \Delta T$	$I_O = 5mA$	-	-0.8	-	mV/ $^{\circ}C$
Output Noise Voltage	V_N	$f = 10Hz$ to $100KHz$, $T_A = +25^{\circ}C$	-	42	-	$\mu V/V_O$
Ripple Rejection	RR	$f = 120Hz$ $V_O = 8V$ to $18V$	62	73	-	dB
Dropout Voltage	V_{Drop}	$I_O = 1A$, $T_J = +25^{\circ}C$	-	2	-	V
Output Resistance	r_O	$f = 1KHz$	-	15	-	m Ω
Short Circuit Current	I_{SC}	$V_I = 35V$, $T_A = +25^{\circ}C$	-	230	-	mA
Peak Current	I_{PK}	$T_J = +25^{\circ}C$	-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Typical Performance Characteristics

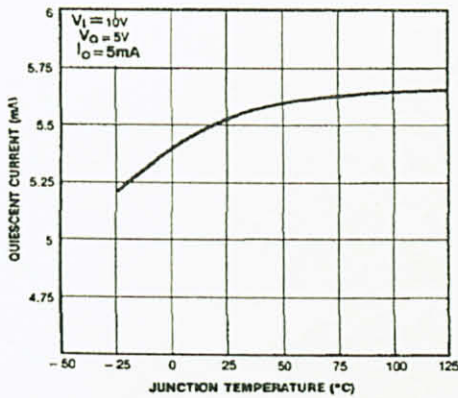


Figure 1. Quiescent Current

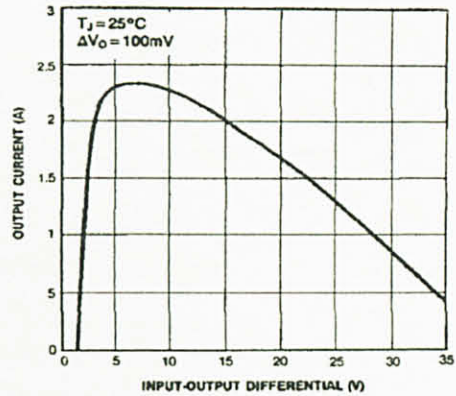


Figure 2. Peak Output Current

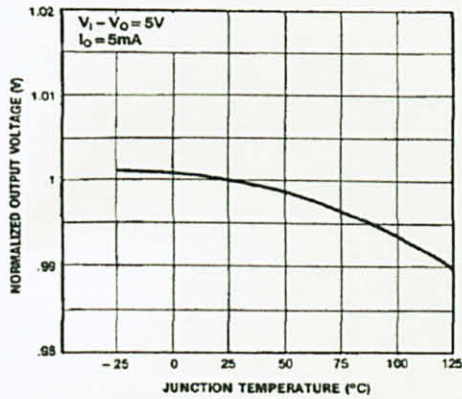


Figure 3. Output Voltage

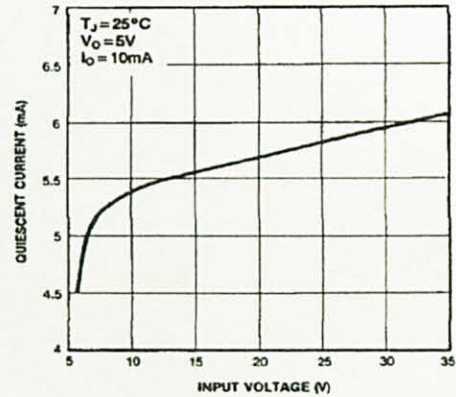


Figure 4. Quiescent Current

Typical Applications

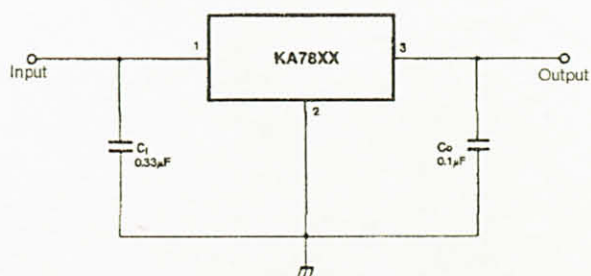


Figure 5. DC Parameters

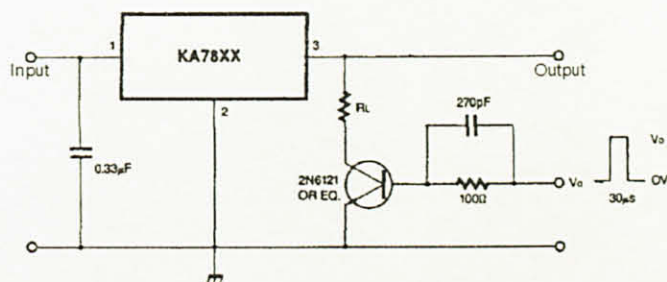


Figure 6. Load Regulation

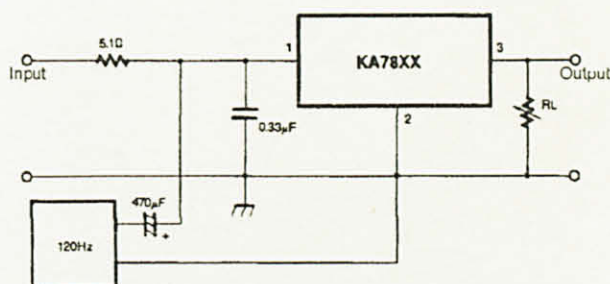


Figure 7. Ripple Rejection

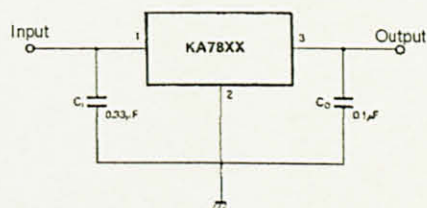
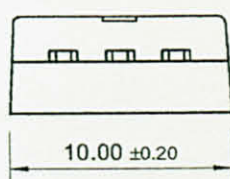
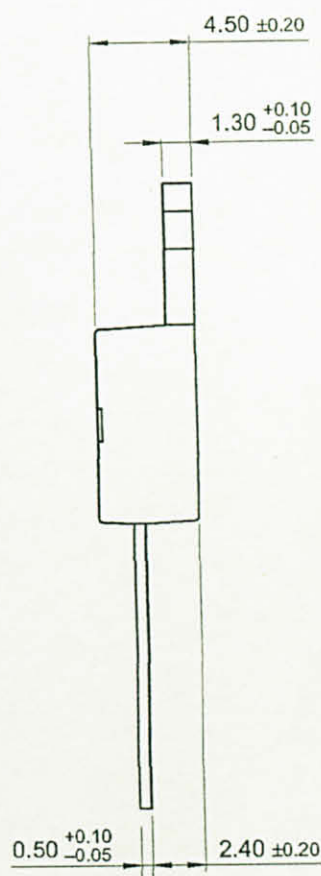
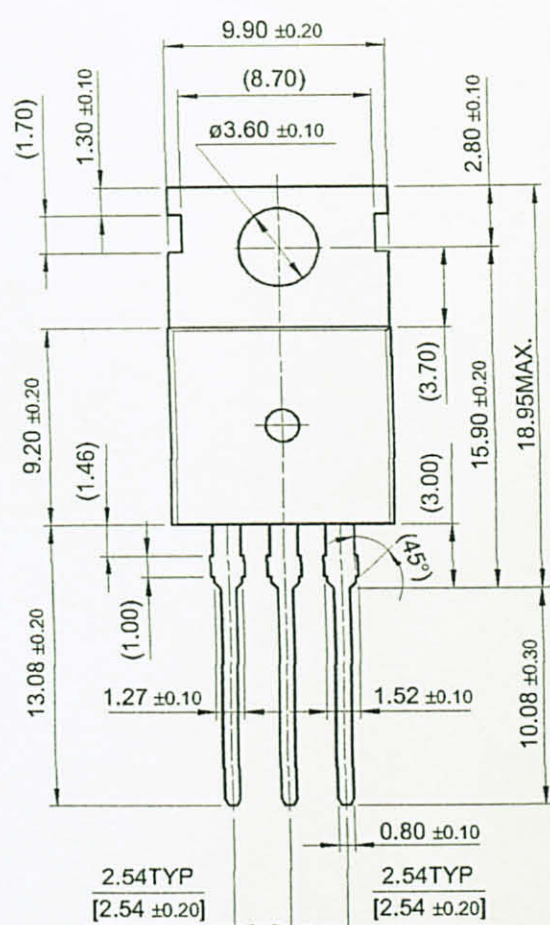


Figure 8. Fixed Output Regulator

Mechanical Dimensions

Package

TO-220



APPENDIX E

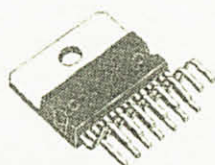
L298N DUAL FULL-BRIDGE DRIVER

DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the con-



Multiwatt15

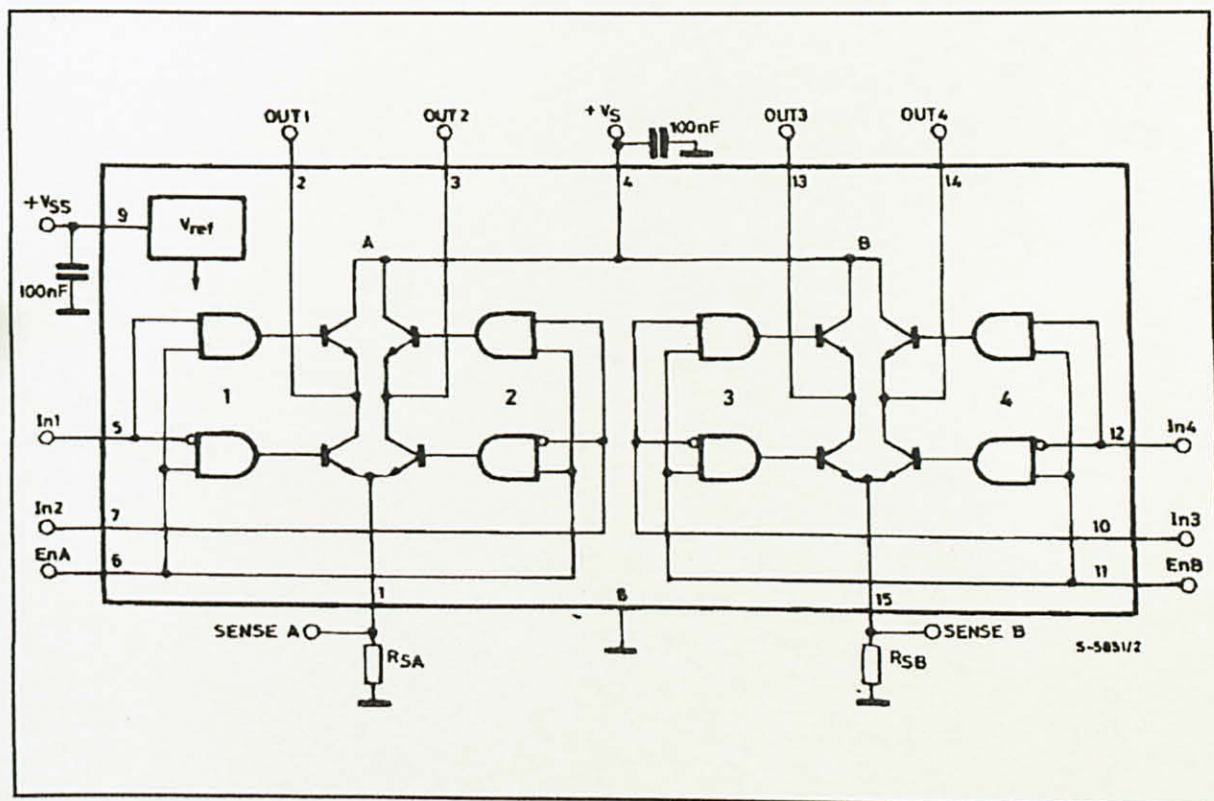


PowerSO20

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

nection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

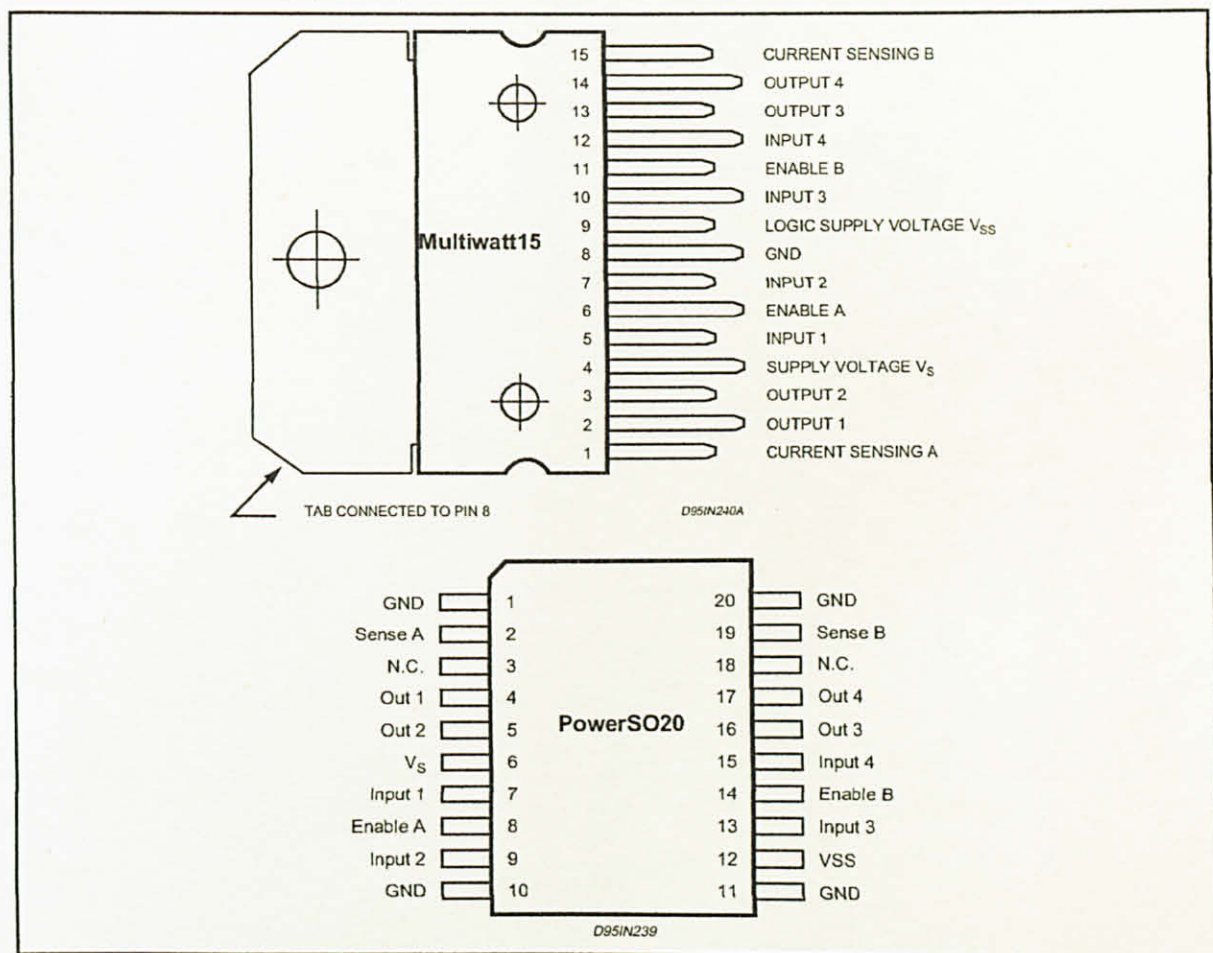
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_I, V_{en}	Input and Enable Voltage	-0.3 to 7	V
I_O	Peak Output Current (each Channel)		
	- Non Repetitive ($t = 100\mu s$)	3	A
	- Repetitive (80% on -20% off; $t_{on} = 10ms$)	2.5	A
	- DC Operation	2	A
V_{sens}	Sensing Voltage	-1 to 2.3	V
P_{tot}	Total Power Dissipation ($T_{case} = 75^\circ C$)	25	W
T_{op}	Junction Operating Temperature	-25 to 130	$^\circ C$
T_{stg}, T_j	Storage and Junction Temperature	-40 to 150	$^\circ C$

PIN CONNECTIONS (top view)



THERMAL DATA

Symbol	Parameter	PowerSO20	Multiwatt15	Unit
$R_{th j-case}$	Thermal Resistance Junction-case	Max.	3	$^\circ C/W$
$R_{th j-amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	$^\circ C/W$

(*) Mounted on aluminum substrate

PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V _S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	V _{SS}	Supply Voltage for the Logic Blocks. A 100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V _S	Supply Voltage (pin 4)	Operative Condition	V _{IH} +2.5		46	V
V _{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I _S	Quiescent Supply Current (pin 4)	V _{en} = H; I _L = 0		13	22	mA
		V _i = L		50	70	mA
		V _i = H				
		V _{en} = L			4	mA
I _{SS}	Quiescent Current from V _{SS} (pin 9)	V _{en} = H; I _L = 0		24	36	mA
		V _i = L		7	12	mA
		V _i = H				
		V _{en} = L			6	mA
V _{IL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V _{IH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		V _{SS}	V
I _{IL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V _i = L			–10	μA
I _{IH}	High Voltage Input Current (pins 5, 7, 10, 12)	V _i = H ≤ V _{SS} –0.6V		30	100	μA
V _{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V _{en} = H	Enable High Voltage (pins 6, 11)		2.3		V _{SS}	V
I _{en} = L	Low Voltage Enable Current (pins 6, 11)	V _{en} = L			–10	μA
I _{en} = H	High Voltage Enable Current (pins 6, 11)	V _{en} = H ≤ V _{SS} –0.6V		30	100	μA
V _{CEsat} (H)	Source Saturation Voltage	I _L = 1A	0.95	1.35	1.7	V
		I _L = 2A		2	2.7	V
V _{CEsat} (L)	Sink Saturation Voltage	I _L = 1A (5)	0.85	1.2	1.6	V
		I _L = 2A (5)		1.7	2.3	V
V _{CEsat}	Total Drop	I _L = 1A (5)	1.80		3.2	V
		I _L = 2A (5)			4.9	V
V _{sens}	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
$T_1 (V_i)$	Source Current Turn-off Delay	$0.5 V_i$ to $0.9 I_L$ (2); (4)		1.5		μs
$T_2 (V_i)$	Source Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (2); (4)		0.2		μs
$T_3 (V_i)$	Source Current Turn-on Delay	$0.5 V_i$ to $0.1 I_L$ (2); (4)		2		μs
$T_4 (V_i)$	Source Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (2); (4)		0.7		μs
$T_5 (V_i)$	Sink Current Turn-off Delay	$0.5 V_i$ to $0.9 I_L$ (3); (4)		0.7		μs
$T_6 (V_i)$	Sink Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (3); (4)		0.25		μs
$T_7 (V_i)$	Sink Current Turn-on Delay	$0.5 V_i$ to $0.9 I_L$ (3); (4)		1.6		μs
$T_8 (V_i)$	Sink Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (3); (4)		0.2		μs
$f_c (V_i)$	Commutation Frequency	$I_L = 2A$		25	40	KHz
$T_1 (V_{en})$	Source Current Turn-off Delay	$0.5 V_{en}$ to $0.9 I_L$ (2); (4)		3		μs
$T_2 (V_{en})$	Source Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (2); (4)		1		μs
$T_3 (V_{en})$	Source Current Turn-on Delay	$0.5 V_{en}$ to $0.1 I_L$ (2); (4)		0.3		μs
$T_4 (V_{en})$	Source Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (2); (4)		0.4		μs
$T_5 (V_{en})$	Sink Current Turn-off Delay	$0.5 V_{en}$ to $0.9 I_L$ (3); (4)		2.2		μs
$T_6 (V_{en})$	Sink Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (3); (4)		0.35		μs
$T_7 (V_{en})$	Sink Current Turn-on Delay	$0.5 V_{en}$ to $0.9 I_L$ (3); (4)		0.25		μs
$T_8 (V_{en})$	Sink Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (3); (4)		0.1		μs

- 1) Sensing voltage can be $-1 V$ for $t \leq 50 \mu s$; in steady state $V_{sens} \min \geq -0.5 V$.
 2) See fig. 2.
 3) See fig. 4.
 4) The load must be a pure resistor.

Figure 1 : Typical Saturation Voltage vs. Output Current.

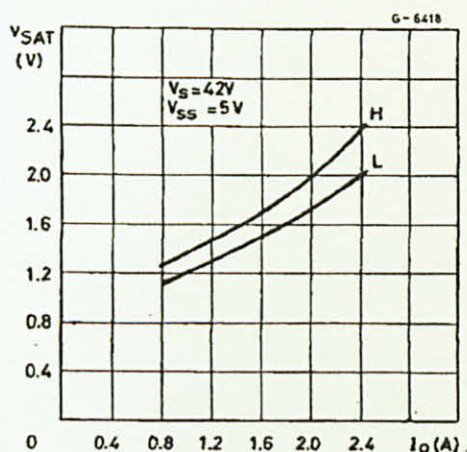
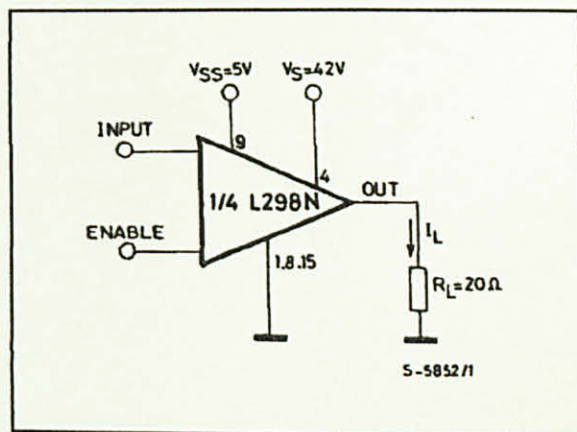


Figure 2 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
 For ENABLE Switching, set IN = H

Figure 3 : Source Current Delay Times vs. Input or Enable Switching.

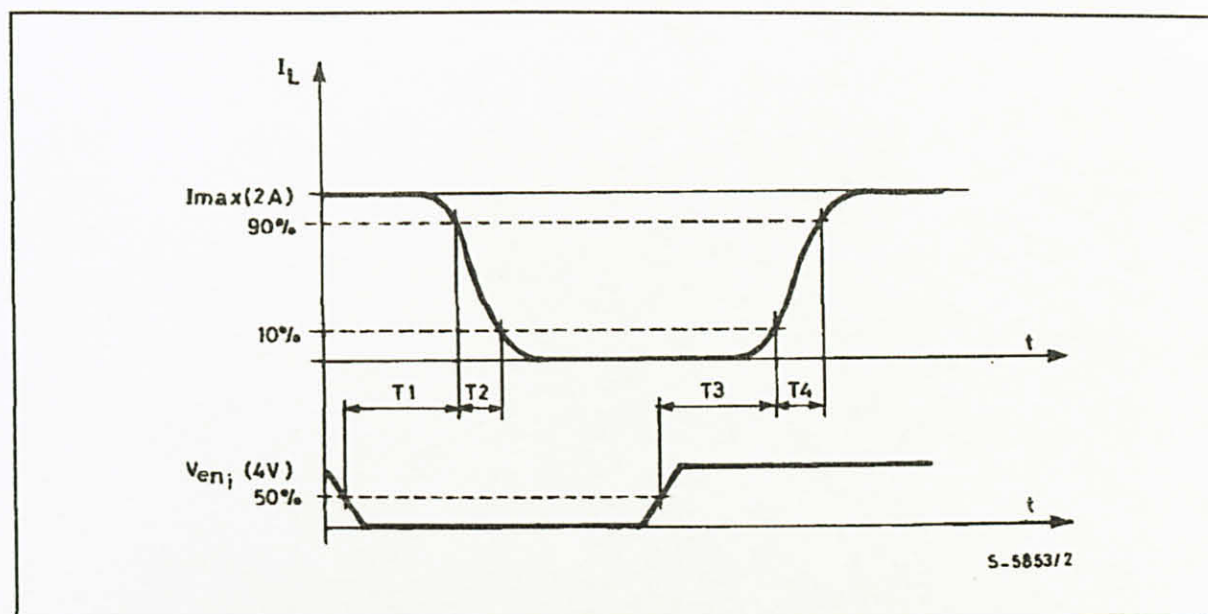
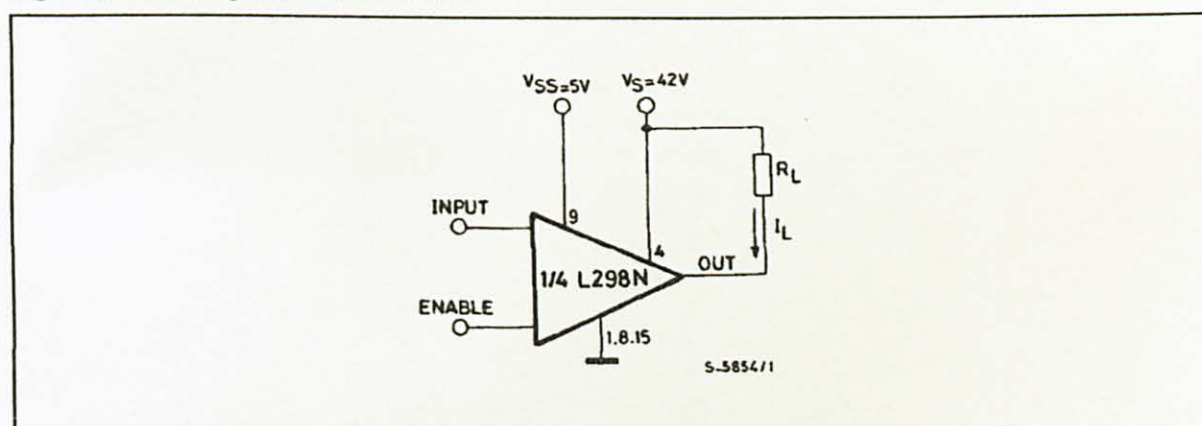
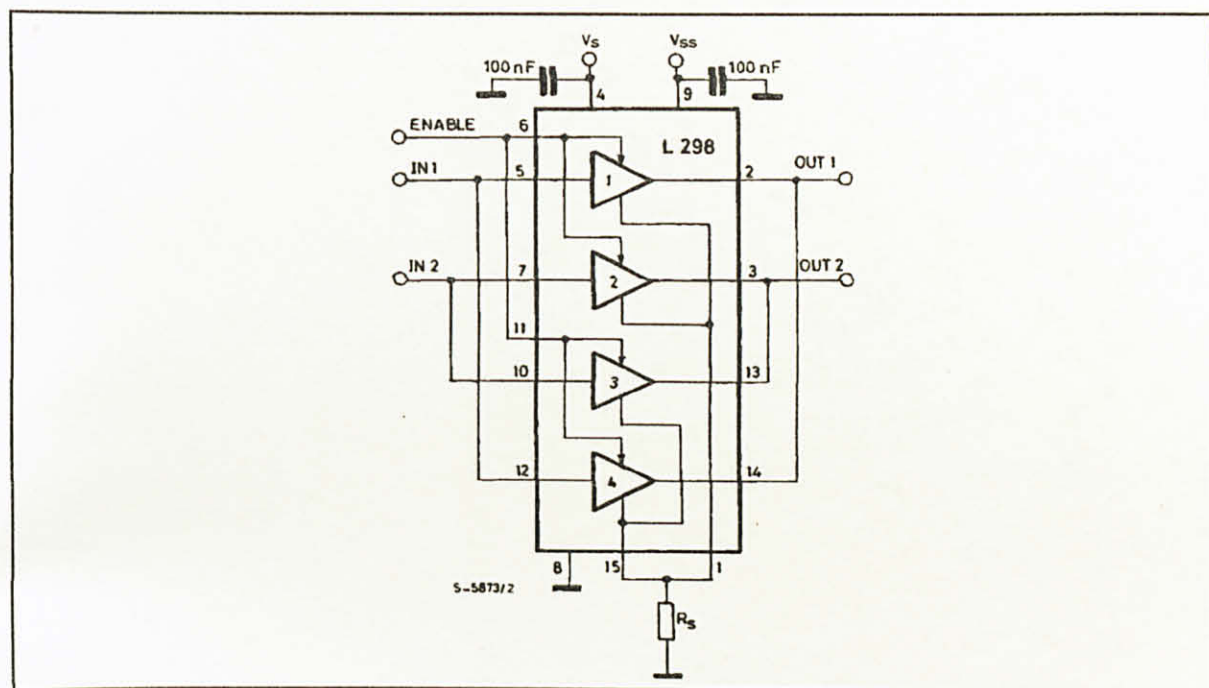


Figure 4 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
 For ENABLE Switching, set IN = L

Figure 7 : For higher currents, outputs can be paralleled. Take care to parallel channel 1 with channel 4 and channel 2 with channel 3.



APPLICATION INFORMATION (Refer to the block diagram)

1.1. POWER OUTPUT STAGE

The L298 integrates two power output stages (A; B). The power output stage is a bridge configuration and its outputs can drive an inductive load in common or differenzial mode, depending on the state of the inputs. The current that flows through the load comes out from the bridge at the sense output : an external resistor (R_{SA} ; R_{SB} .) allows to detect the intensity of this current.

1.2. INPUT STAGE

Each bridge is driven by means of four gates the input of which are In1 ; In2 ; EnA and In3 ; In4 ; EnB. The In inputs set the bridge state when The En input is high ; a low state of the En input inhibits the bridge. All the inputs are TTL compatible.

2. SUGGESTIONS

A non inductive capacitor, usually of 100 nF, must be foreseen between both Vs and Vss, to ground, as near as possible to GND pin. When the large capacitor of the power supply is too far from the IC, a second smaller one must be foreseen near the L298.

The sense resistor, not of a wire wound type, must be grounded near the negative pole of Vs that must be near the GND pin of the I.C.

Each input must be connected to the source of the driving signals by means of a very short path.

Turn-On and Turn-Off : Before to Turn-ON the Supply Voltage and before to Turn it OFF, the Enable input must be driven to the Low state.

3. APPLICATIONS

Fig 6 shows a bidirectional DC motor control Schematic Diagram for which only one bridge is needed. The external bridge of diodes D1 to D4 is made by four fast recovery elements ($t_{rr} \leq 200$ nsec) that must be chosen of a VF as low as possible at the worst case of the load current.

The sense output voltage can be used to control the current amplitude by chopping the inputs, or to provide overcurrent protection by switching low the enable input.

The brake function (Fast motor stop) requires that the Absolute Maximum Rating of 2 Amps must never be overcome.

When the repetitive peak current needed from the load is higher than 2 Amps, a paralleled configuration can be chosen (See Fig.7).

An external bridge of diodes are required when inductive loads are driven and when the inputs of the IC are chopped ; Schottky diodes would be preferred.