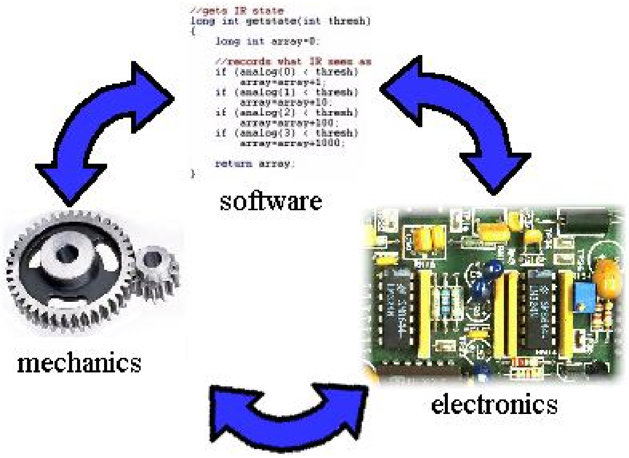
Chapter 1

1.1 Introduction

The word “robot” has been used since almost one century ago. It appeared for the first time in Karel Capek’s play R.U.R. (Rossum’s Universal Robots). This man was a very influential Czech writer, and maybe for this reason it is believed that he was the first person who used that word. However, some investigations have showed that the idea came from his older brother Josef Capek, a painter, who proposed him the word “robot” (derived from the Czech noun “robota”, which means “labor”) to call some kind of “artificial workers” that appeared in his play (1). Robotics are becoming more and more widely used in the automation, medical, manufacturing industries, also in many science fiction films and many others fields. All are based on the microcontroller technology that enables manufacturers to put an entire CPU on one chip. Nowadays, robots are constructed tended to be human-like. Generally, robots have three main parts known as mechanical, electronic and electronic system. If robot is replaced by human, sensor is represented eye, controller is represented brain and actuator is represented leg.



```
//gets IR state
long int getstate(int thresh)
{
    long int array=0;

    //records what IR sees as
    if (analog(0) < thresh)
        array=array+1;
    if (analog(1) < thresh)
        array=array+10;
    if (analog(2) < thresh)
        array=array+100;
    if (analog(3) < thresh)
        array=array+1000;

    return array;
}
```

Figure 1-1: Robot Component

This project is more about to design and development a line following mobile robot for multipurpose application. It included the mechanism, circuit and programming. After the base robot was built, the robot will undergo test run and than from the test we can collects data and identify the weakness and further improvement. The project goal is to design and build robots that work for multipurpose application that can use in offices, hospital, airports and museums but elderly and disabled which in care here. For education propose it can be use to get knowledge's or information's about the robot world. It also can help doing the task for assignment and made technology experiment to improve their understanding about robot technology in the future.

In this chapter brief explanation about flow of this project will be discussed specially. There are the problem statements, objectives, scope of the project and methodology of the project.

1.2 Literature review

The use of robotics in human healthcare is growing interest. In most institutions, services provided to the elderly and disabled are unsatisfactory, mainly due to their dependence on human assistance. Utilizing robots will spare human work force and help decreasing the risk of infection; and subsequently, economizing on personnel and hygiene expenses.

Hagadam, hamed, Hussein ET.AL 26-28 Aug[2]. 2013 IEEE: This paper discussed these issues in three points, in order to present a low cost autonomous robotic wheelchair design to transport the elderly and disabled within hospitals. First, a design of a robot was implemented on a small prototype. The robot was navigated using line-following with obstacle detection technique; which is simple and easily executed. Second, a design of a robotic wheelchair was presented. The wheelchair was developed from manual wheelchairs used in hospitals after proper adjustments, and a design of two gearboxes has been proposed. Third, the advantage of using robotic wheelchairs inside hospitals was discussed by utilizing findings from previous studies. The robot prototype circuit was built using optical and ultrasonic sensors, two DC motors and L293D motor driver; and programmed with Atmega16L Microcontroller. But the main problem of this designed is more expensive because used wheelchairs.

Román Osorio C., José A. Romero,ET.AL(2006)[3]. This paper was show a prototype development of an intelligent line follower mini-robot system, the objective is to recognize, understand and modify the actual performance of the movements of the robot during its pathway by way of getting information in real time from different magnetic sensors implemented in the system and based in a V2X digital compass, microcontroller and audiometric measurements. But there is no obstacle detection.

Another previous work similar to this project is M. S. Islam & M. A. Rahman (2013) [4]. This paper presents a 700gm weight of a 9W LDR Sensor based line follower robot design and fabrication procedure which Always directs along the black mark on the white surface. The problem here LDR is not working perfect, the environment light change their redoing so it give error reading sometime.

1.3 Problem Solutions

At the turn of the millennium, the number of elderly and disabled in need of care is increasing dramatically. Currently, the services provided to the elderly and disabled in most public institutions are unsatisfactory; this is largely because of their dependence on human assistance and costs. One obvious area that needs immediate attention is transportation in public areas like airport terminals, hospitals, museums, office buildings etc. The problem statement in this project is to build a mobile robot that can reach the elderly and disabled in those public area from one point to another. In another Sid the products of a lot of manufactories thy need to curry from one point to other continuously by human this is teddies and slow, so this robot can overdo the human work. This thesis will introduce a low cost mobile robotic platform. It offers hardware and software features to build a robot to transport the elderly and disabled people in airports, hospitals, industries and other similar facilities. And it also use in manufactories to cry the product from point to other.

1.4 Objective

Robotic technology is going through major revolutions. Sparked by a dramatic increase of computation and the substantial decrease in costs of major sensor technologies (e.g. IR sensor) the goal of intelligent service robots that can assist people in their daily living activities is closer than ever. In under graduate (B.sc) I was build line following robot but there is a lot of messing point. Now I wanted to complete some of my recommendations such as (object detection) and transfer it to be more useful. The main objective for this low cost fundamental electronic component based line Sensing robot can carry a load without getting off the line.

1.5 Scope of the Project.

The scope of this project is to build a robot structure that will function properly referring to the objective where the need of the design must be build in good shape to make sure it suitable with the motor and other stuff. Next to create a computer program (brain) that will process the whole data from input to the output. The program must be declared as the mechanical part is done to make sure the robot will function properly.

Lastly, perform an experiment that is including testing and commissioning the product to make sure the product well function as stated in the objective. There are many kinds of methods that can be implemented to develop this project. For this project, the scopes can be described as follows:

i) Design and develop a prototype mobile robot that transfer anything, support a little Weight of grams which applied power window motor to this mobile robot as navigator.

ii) Design and develop the program using the 16F877A microcontroller (PIC).

iii) Design and develop a mobile robot that must be built with the capability to self navigate from a starting point to an end point.

iv) Design and develop a mobile robot to travel along a dark line using sensors and at smoothly right and left turning.

1.6 Approach

Hardware components such as microcontroller, sensors and motors will be used to build the robot. Software programs such as code microC compiler and ISIS will be written to direct and control the robot to move movement from a certain point to another transporting elderly /disabled people.

1.7 Layout of Project

Chapter 2: Mechanic designed of mobile robot this chapter shows about mechanical designed such as dimensions, calculation of power, speed, and torque and kinematic of the robot.

Chapter 3: Software for the mobile Robot will show the use of microC compiler and ISIS simulation and flowchart of the robot.

Chapter 4: Electronic circuit Design this part represent the control circuit (brain) and sensors (vision system).

Chapter 5: assembling and implementation of the robot. Show the complete assembling of chassis with electronic and sensor and software.

Chapter 6: Result, Conclusion and Recommendation This chapter state output of the result that be obtained, discussion and analysis of the result and also talk about conclusion and recommendation.

Chapter2

Mechanic designed of mobile robot

2.1 Introduction

The vehicle has a custom-made metal frame, which is mounted with two independent left and right driving wheels driven by two DC motors, respectively. It does not have any steering mechanism, but can change its navigation direction by changing the velocity of the driving wheels independently. If the velocity of the right wheel is greater than the velocity of the left wheel, the robot will turn left vice versa. Using micro controller with the input from sensors controls the two motors of mobile robot [5]. The 9,s volt battery is used to provide power to the motors and the control part respectively to reduce the switching noise induced by the motors. The control circuit and the driver circuit are provided on board.

The platform of the mobile robot is simply a 10 * 5-inch piece of plywood, cut from a larger piece. The robot had to have a minimum of three wheels in order to work, and how the wheels are powered and mounted affects the control of the robot. Figure 2.1 shows the Location of the wheels in the chassis. All the combinations require two motorized wheels and a caster wheel for balance.

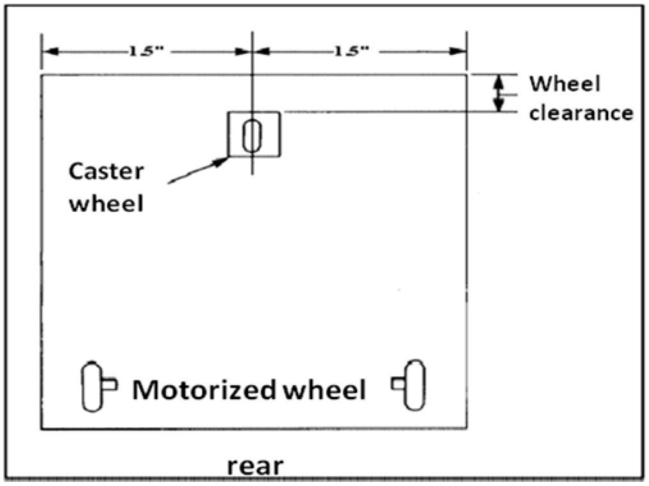


Figure 2-1: Location of wheels

The robot uses 3 IR sensors at the middle to sense the black strip path and one on either side to sense white background. The sensors are tuned to corresponding black and white colors to keep track of predefined path. The rules for the robot to keep track and avoid hitting obstacle is feed to the micro controller in the form of programmer, micro controller decides the next move according to the algorithm. The vehicle is equipped with three sets of IR sensor provided the vehicle to keep track of black strip on white background, the proximity sensor is provided at the front to detect obstacle during navigation.

2.2 Defining the Line

The width of the line to be followed should be 3cm. The line must contrast well with the floor. Both the line and the floor should be solid colors, not patterns. In fact, solid white on solid black would make the robot happiest (see Figure 2-2). The line and floor must remain the same color throughout the course. It doesn’t matter what shade of floor you have because the robot is designed to follow either a darker line on a lighter floor or a lighter line on a darker floor.

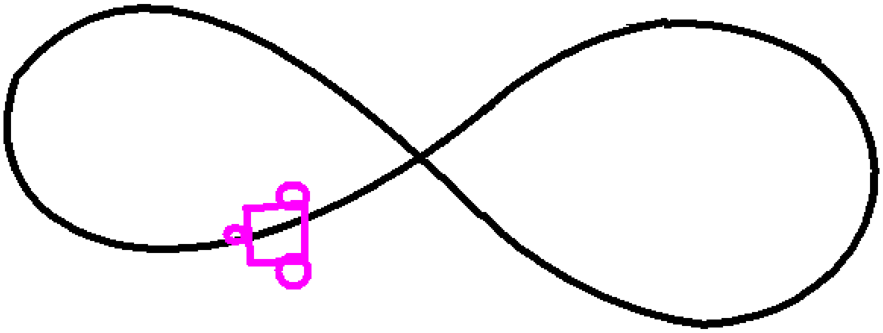


Figure 2-2. Unacceptably

2.2.1Curving and Crossing Lines

Straight-aways are easy for any properly operating line follower. Sharp turns, however, can derail even a competent robot. With reasonable brains and an array of sensors or a vision system, the line-following robot to handle turns 90° or greater and also broken lines (see Figure 2-3). However, for the robot building at this project, limit turn angles to 45° or less. Lines should always be contiguous.

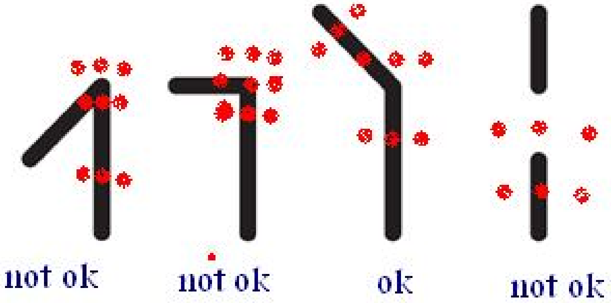


Figure 2-3. Unacceptably sharp turns: (left to right) 135°, 90°. (Far right) broken line

Given a series of smaller turns over a longer distance, the robot can guide in any direction (see Figure 2-4).

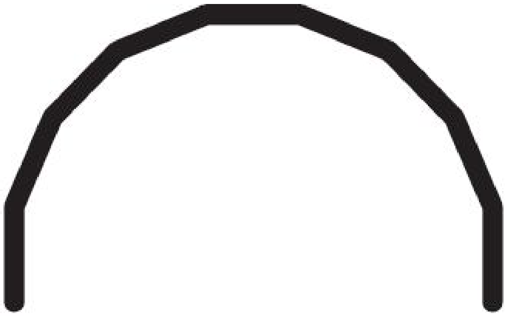


Figure 2-4 Acceptable 180° turn made gradually from 22.5° segments

The better the surface contrast and the slower the robot's pace, the steeper the turns the robot can perform. After all, the worst thing that can happen is that the robot wanders away. Many designers fret over line crossings or line splits in the course (see Figure (2-5). It's possible for some robots to head between the lines as the brightness averages out. But, the robot circuit presented in this project isn't confused by such intersections, and quickly settles into one path choice or the other.

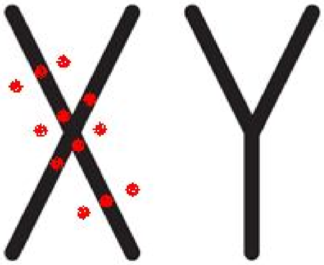


Figure 2-5 A crossing and a split

2.3 The DC Motors

The DC motors witch used here spin at about 8000 RPM. At the high end, that means the motor can rotate its shaft around 8000 times before a clock's second hand completes a single rotation. That's really fast! Initially it may seem that faster is better, but that's not the case. For a line-following robot with slow-reacting sensors, fast speed could drive the robot off the line before the brain has time to react.

2.3.1Gear designed

The type of gear which was used here is compound gear .in a simple train of gear do not affect the speed ratio of the system. But these gears are useful in bridging over the space between the driver and the driven [6]. In this case, each intermediate shaft has two gear rigidly fixed to it so that they may have the same speed. One of these two gears meshes with the driver and the other with driven or follower attached to the next shaft as show in figure (2-6) below

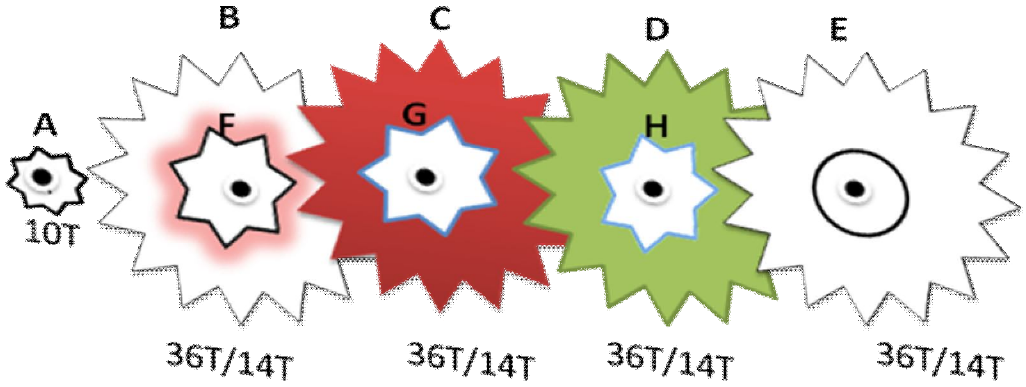


Figure (2-6) show gear design

The motor shaft is connected to the gear A and rotted at 8000 r.p.m. the gear wheels B, C, D, E, F, G and F are fixed to parallel shaft rotated together. The final gear E is fixed on the output shaft. What is speed of gear E? The gear A, F, G, H is driver while the gear B, C, D and E are driven or followers. So these gears are compound gears are compound gear.

Na = speed of driving gear A

Ta = number of teeth in driving gear A

Na, Nb … Ne = speed of respective gears in r.p.m.

Ta, Tb… Te = number of teeth on respective gears. Since gear A is in mish with gear B, there for its speed ratio is

$\frac{Na}{Nb} = \frac{Tb}{Ta}$……………………………………………………………….(2.1)

Similarly F is in mish with gear C, there for its speed ratio is

$\frac{Nf}{Nc} = \frac{Tc}{Tf}$………………………………………………………………. (2.2)

Similarly G is in mish with gear D, there for its speed ratio is

$\frac{Ng}{Nd} = \frac{Td}{Tg}$…………………………………………………………….... (2.3)

Similarly H is in mish with gear E, there for its speed ratio is

$\frac{Nh}{Ne} = \frac{Te}{Th}$…………………………………………………………….... (2.4)

The speed ratio of compound gears trins is obtained by multiplying the equation (1), (2), (3) and (4).

$$\frac{Na}{Ne} = \frac{Tb}{Ta} * \frac{Tc}{Tf} * \frac{Td}{Tg} * \frac{Te}{Th}$$

$$\frac{8000}{Ne} = \frac{36}{10} * \frac{36}{14} * \frac{36}{14} * \frac{36}{14}$$

$$\frac{8000}{Ne} = \frac{1679616}{27440} = 61.2105$$

Then **Ne** = 8000/61.2105 = 131 r.p.m

So the output speed is 131 r.p.m which represent the two back wheels velocity. But this speed must be converted to liner velocity.

2.3.2Converting RPM to a Metric Unit

The metric unit for angular velocity is radian per second, or rad/s. Even though it doesn’t appear often in advertisements, rad/s sometimes shows up in datasheets. For motors, RPM and rad/s are used to measure the same thing, but rad/s is an internationally accepted unit in the metric system. To convert RPM to rad/s, multiply by 0.10472 (which is an approximation of π/30).

RPM × 0.10472 = rad/s, so: 131 RPM × 0.10472 = 13.71832 rad/s

To convert rad/s to RPM, multiply by 9.54929 (which is an approximation of 30/π).

rad/s × 9.54929 = RPM , so:13.71832 rad/s × 9.54929 = 131 RPM

2.3.3 Calculating Linear Speed

Here’s the formula for linear speed:

(RPM of loaded motor / 60 seconds in a minute) × Wheel circumference in meters = linear speed in meters per second

To determining the wheel circumference is easy there are two way the first with a cloth tape measure (see Figure 2-7). Or used a tape measure, wrap a strip of paper around the tire. Mark the spot of overlap and unroll the paper. Then use a ruler to determine the length of the flat piece of paper with the mark. So the wheel circumference of robot is about 11 cm, which is 0.11 m.

The second way by use the law of circular surface $l = 2\pi r$, r is radios of wheel. The diameter of wheel is 35mm, apply it to equation above. L=2*3.14*1.75=10.99cm

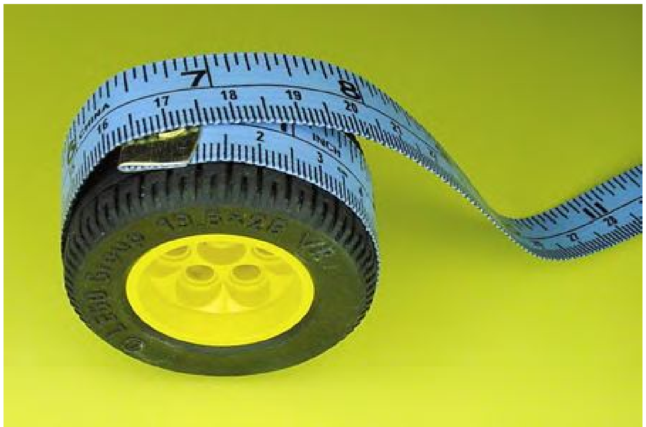


Figure 2-7 Measuring wheel circumference with a cloth tape measure

Plugging the numbers into the linear speed formula results as follows:

Linear speed= (131 RPM / 60) × 0.11 m = 0.24 m/s. For example the robot should be able to complete a straight, 4-meter course in about…

Linear course length in meters / linear speed = number of seconds to complete .as follows

4 m / 0.24m/s = 16.667 s.

The calculation indicates 16.667 seconds. In reality, it took the robot about 17 seconds. The difference between the calculated value and the measured value is the difference between no-load RPM and loaded RPM. You can work the measured value backwards to determine the approximate loaded RPM.

(Linear course length in meter / number of seconds to complete) / Wheel circumference in meters) × 60 seconds in a minute = RPM of loaded motor so (4 m / 17 s) / 0.11 m) × 60 = 128 RPM.

That means the loaded RPM is about 128 RPM. Of course, the robot wiggles a bit rather than going in a straight line. So, the true loaded RPM is a little higher, but 128 RPM is accurate enough for predictive calculations. Now that we know the loaded speed, let's plug it back into the original formula to make sure it will now predict a 17-second course time.

(128 RPM / 60) × 0.11 m = 0.234 m/s

4 m / 0.234 m/s = 17 s

What happens if larger (22 cm) wheels are installed?

(128 RPM / 60) × 0.22 m = 0.469 m/s

4 m / 0.469 m/s = 8.5 s.

3.3.4 Torque vs. Speed Relationship

The figure 2.8 and 2.9 show the relation between torque and speed and power

$$\tau_{motor} = \tau_s - \omega\tau_s/\omega_n$$

$$\omega_{motor} = (\tau_s - \tau)\omega_n/\tau_s$$

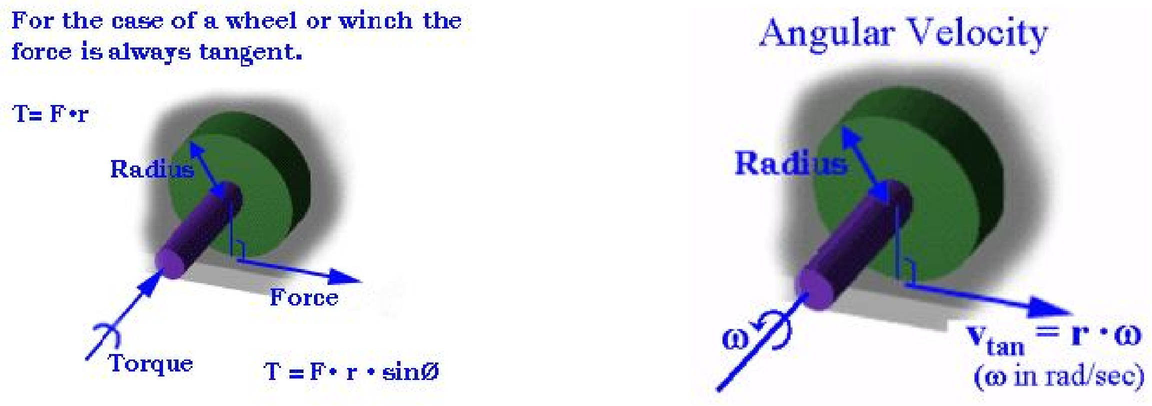

For the case of a wheel or winch the force is always tangent.
T= F•r
Radius
Force
Torque
T = F• r • sinØ
Angular Velocity
Radius
ω
$v_{tan} = r \cdot \omega$
(ω in rad/sec)

Figure 2-8 the wheel rotation

Power = Torque * Rotational Velocity (in radians)

Max Power Occurs at ½ Stall Torque and ½ Rotational Speed

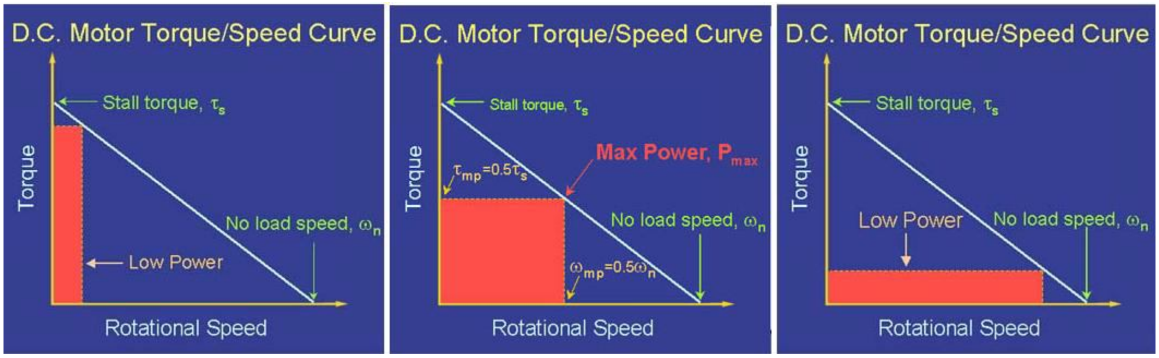


Figure 2-9 relation between torque, speed and power

2.3.5 Calculate the Required Wheel Torque

From the technical specification of the DC motor wich was chsen can be capable of accelerating a 5lb, two wheel drive robot with wheel diameters of 1.4" at a rate of 1ft/sec/sec. So to determine the torque (force) that must be generated at each wheel in order to meet the functional requirement.

Convert all units to SI units	Imperial	SI
Mass =	5lbs	2.2727 kg
Acceleration =	1.5ft/square seco	0.4573 meter/square second
Torque =	ft/lbs	Newton/meters

F = MA

$$Force_{Total=mass_{kg}*Acceleration_{meter/second^{2}}}$$

$$Force_{Total=}2.2727\ .0.4573\ _{m/S^{2}}$$

$$Force_{Total}= 1.04\ _{newtens}$$

The Total force required to meet the Functional Requirement is 1.04 Newton's. However, the vehicle has 2 motors and wheels. Therefore each motor/wheel combination needs only supply half the required force or 2.9 Newton's.

$$Force_{wheel}= 0.52\ _{newtens}$$

2.3.6 Calculate the torque requirement for each wheel

Torque is a twisting or turning force that acts on an axis or pivot. The 2 wheels of this vehicle turn on an axis or shaft. The distance from the center of the turning axis to the outside of the wheel is the radius of the wheel. The torque or turning force of the axle acts through the radial distance as illustrated in the graphic below.

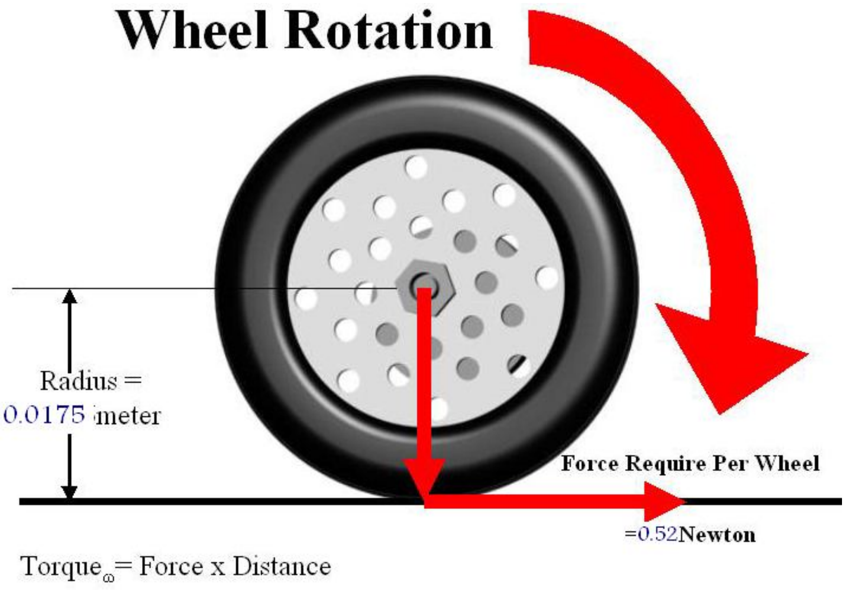


Figure 1

Figure 2-10 wheel rotation analysis

Distance = Wheel Radius = Wheel Diameter/2 = 1.4”/2 inches =0.0175meter

Force = Force required per wheel = 0.52 Newton’s

$$Force_{wheel=0.52_{kg}*0.0175_{meter}}$$

$$Force_{wheel=0.0091newten.meter}$$

2.3.7 Choosing dc motor

According to calculation above the motor was chosen as in the table (2-1) below shows the technical specification of the dc motor which was used; assume no load because the robot is a bit small.

Table (2-1) show the technical specification of dc mot

	Voltage		No Load		At Maximum Efficiency				Stall	
Model	Operating	Nominal	Speed	Current	Speed	Current	Torque	Output	Torque	Current
	Range	V	rpm	A	rpm	A	g·cm	W	g·cm	A
RE-260RA-2760-38	1.5-3.0	1.5	7825	0.31	6004	0.68	8.6	0.45	36.79	2.23

2.4 Architecture of Mobile Robot System

A kinematics model of a mobile robot used in this thesis is shown in Figure 2-11.

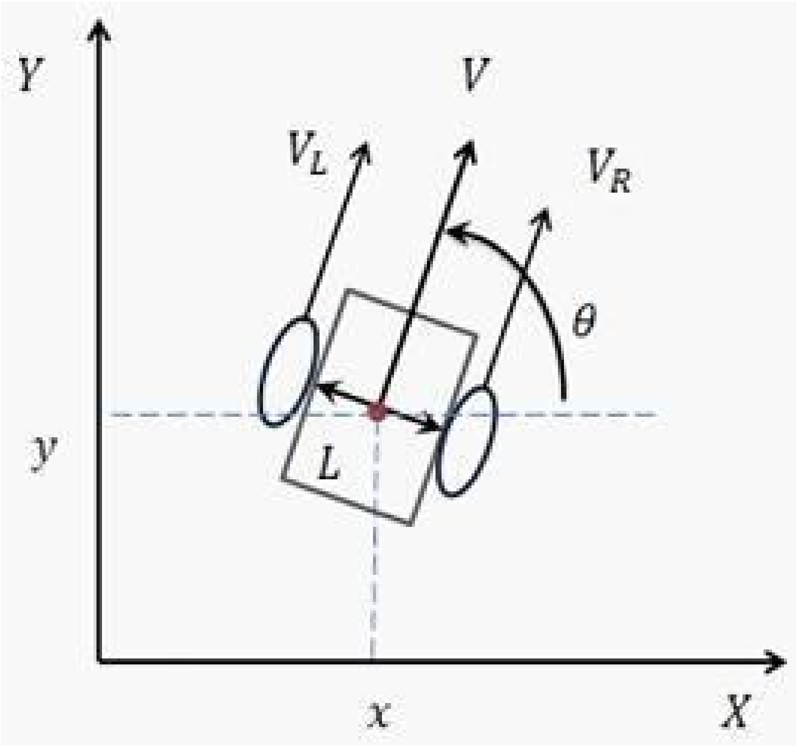


Figure 2-11 Kinematics Model of the Robot

Accelerations of left and right wheels are w_r and, w_l respectively. We assume that the contact between the wheels and the ground is pure rolling and non-slipping [7]. The relationship between left and right wheels of velocity and acceleration is as follows:

$$V_L = r\omega_L \text{ and } V_R = r\omega_R \ . \qquad (2.5)$$

Where r is a radius of the wheel, the linear velocity of the mobile robot is V. Linear velocities for the left and right wheels are V_l andV_r, respectively. The relationship betweenwl, wl, $V_l$and V_r is as follows:

$$V = \frac{V_R + V_L}{2} = r\frac{\omega_R + \omega_L}{2} \qquad \omega = \frac{V_R - V_L}{L} = r\frac{\omega_R - \omega_L}{L} \tag{2.6}$$

Now we can summarize a dynamic model for the mobile robot as follows:

$$x' = V\cos\theta\,,\ y' = V\sin\theta\,,\ \theta' = \omega\,,$$

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} V \\ \omega \end{bmatrix}$$

Chapter 3

Software for the mobile Robot

3.1 Introduction

The software developed for the robot used both microC for PIC and Proteus7.7. They are free development software's for programming the PIC Microcontrollers. In this project microC was used for write the code and Proteus 7.7 for simulating the electronic circuit. The figure (3-1) below show the block diagram software Microcontroller Systems.

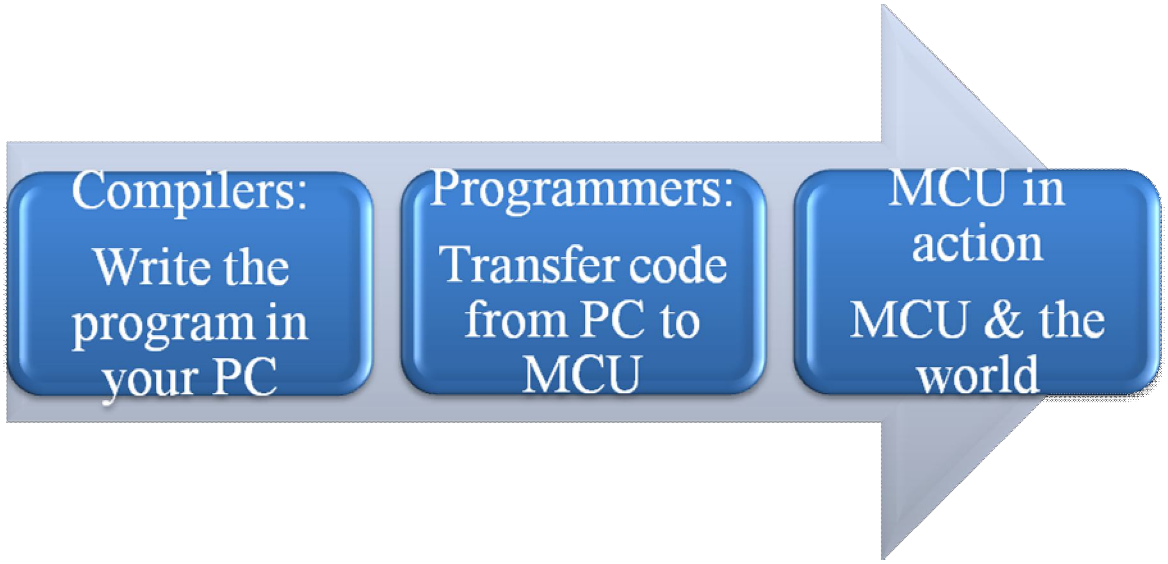


Figure 3-1 Microcontroller software Systems

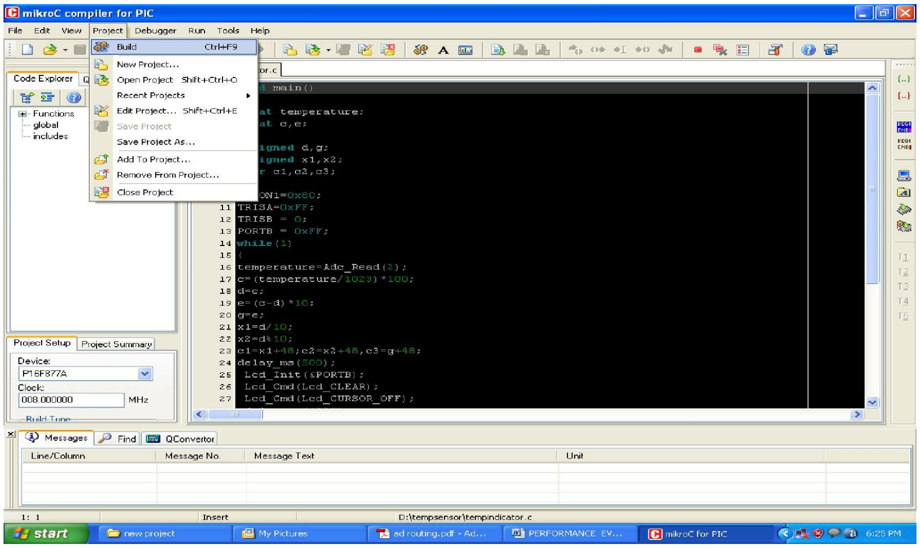
3.2 Introduction to MikroC compiler

Compilers for microcontroller CPUs are available in many programming language. Most used programming languages are C and assembly. Microcontroller manufacturer often provide compilers and simulators that support many of the chips.

MikroC is a powerful, feature rich development tool for PICmicros. It is designed to provide easiest possible solution for developing applications for embedded systems, without compromising performance or control Fig. 3.6 show the the window of micro c . PIC and C fit together well: PIC is the most popular 8-bit chip in the world, used in a wide variety of applications, and C, prized for its efficiency, is the natural choice for developing embedded systems. MikroC provides a successful match featuring highly advanced IDE, ANSI compliant compiler, broad set of hardware libraries, comprehensive documentation, and plenty of ready to run examples.

3.2.1 How to build a project in microC

Step 1: The software MikroC pro for PIC 2009 is opened and the c code for the program is then written on the writing window. After writing the code we go to the topmost toolbar Project > Build or press Ctrl+ F9 to build the code as shown in the figure (3-2). The following pictures show the above process:



Device: P16F877A

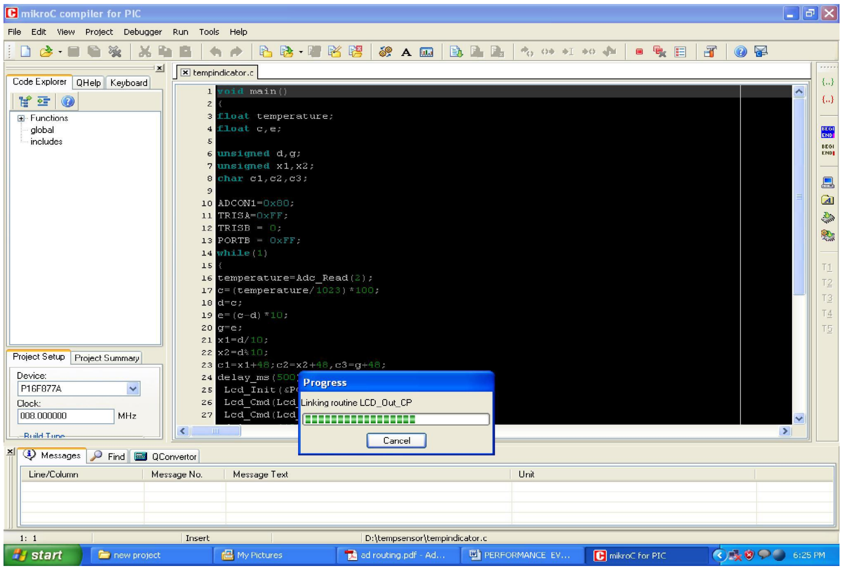
Clock: 008.000000 MHz

D:\tempsensor\tempindicator.c

```
d main()

at temperature;
at c,e;

igned d,g;
igned x1,x2;
r c1,c2,c3;

ON1=0x80;
TRISA=0xFF;
TRISB = 0;
PORTB = 0xFF;
while(1)
{
temperature=Adc_Read(2);
c=(temperature/1023)*100;
d=c;
e=(c-d)*10;
g=e;
x1=d/10;
x2=d%10;
c1=x1+48;c2=x2+48,c3=g+48;
delay_ms(500);
 Lcd_Init(&PORTB);
 Lcd_Cmd(Lcd_CLEAR);
 Lcd_Cmd(Lcd_CURSOR_OFF);
```

Fig. 3-2 How to build a project

Step 2: After building the code the compiler starts to compile the program as in figure (3-3). After checking the initial errors and rectifying them the entire code was 'Built' by the software to generate the corresponding HEX code of the c program the functions used in the above program are linked together into a subroutine which generates the HEX code from the raw c code.


mikroC compiler for PIC
File Edit View Project Debugger Run Tools Help
Code Explorer QHelp Keyboard
Functions
global
includes
Project Setup Project Summary
Build Type
Messages Find QConvertor
Line/Column Message No. Message Text Unit
start new project My Pictures ad routing.pdf - Ad... PERFORMANCE EV... mikroC for PIC 6:25 PM
Progress
Cancel

tempindicator.c

```
void main()
{
float temperature;
float c,e;

unsigned d,g;
unsigned x1,x2;
char c1,c2,c3;

ADCON1=0x80;
TRISA=0xFF;
TRISB = 0;
PORTB = 0xFF;
while(1)
{
temperature=Adc_Read(2);
c=(temperature/1023)*100;
d=c;
e=(c-d)*10;
g=e;
x1=d/10;
x2=d%10;
c1=x1+48;c2=x2+48,c3=g+48;
delay_ms(500
 Lcd_Init(&P
 Lcd_Cmd(Lcd
 Lcd_Cmd(Lcd
```

Linking routine LCD_Out_CP

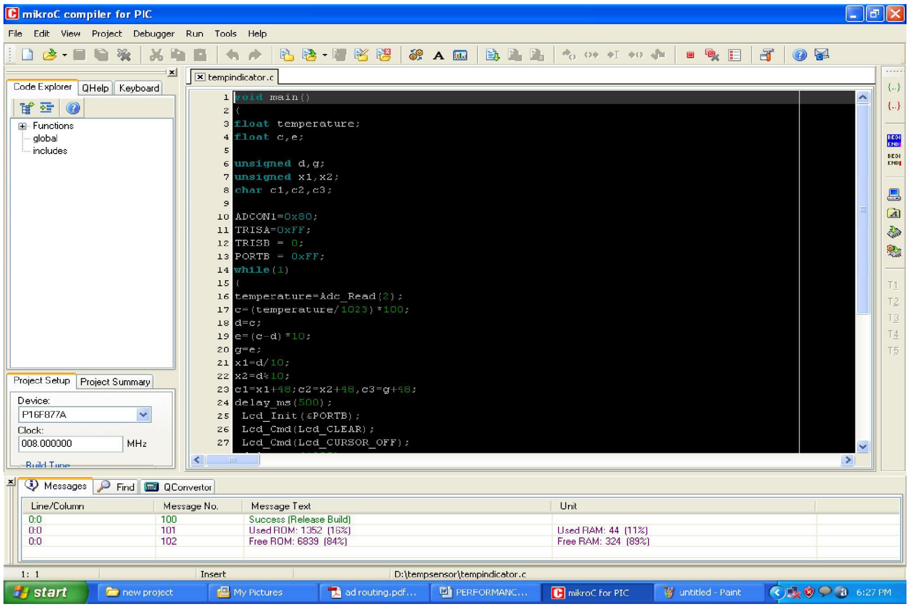
Device: P16F877A

Clock: 008.000000 MHz

1: 1 Insert D:\tempsensor\tempindicator.c

Fig. 3-3 Compilation of the project

Step 3: After creation of subroutine the compiler generates the HEX code and tells the user how much space it occupies and how much RAM it has to be used as in figure (3-4). After the HEX code has been created the code gets ready to be downloaded. After completion it looks like the following figure.



tempindicator.c

```
void main()
{
float temperature;
float c,e;

unsigned d,g;
unsigned x1,x2;
char c1,c2,c3;

ADCON1=0x80;
TRISA=0xFF;
TRISB = 0;
PORTB = 0xFF;
while(1)
{
temperature=Adc_Read(2);
c=(temperature/1023)*100;
d=c;
e=(c-d)*10;
g=e;
x1=d/10;
x2=d%10;
c1=x1+48;c2=x2+48,c3=g+48;
delay_ms(500);
 Lcd_Init(&PORTB);
 Lcd_Cmd(Lcd_CLEAR);
 Lcd_Cmd(Lcd_CURSOR_OFF);
```

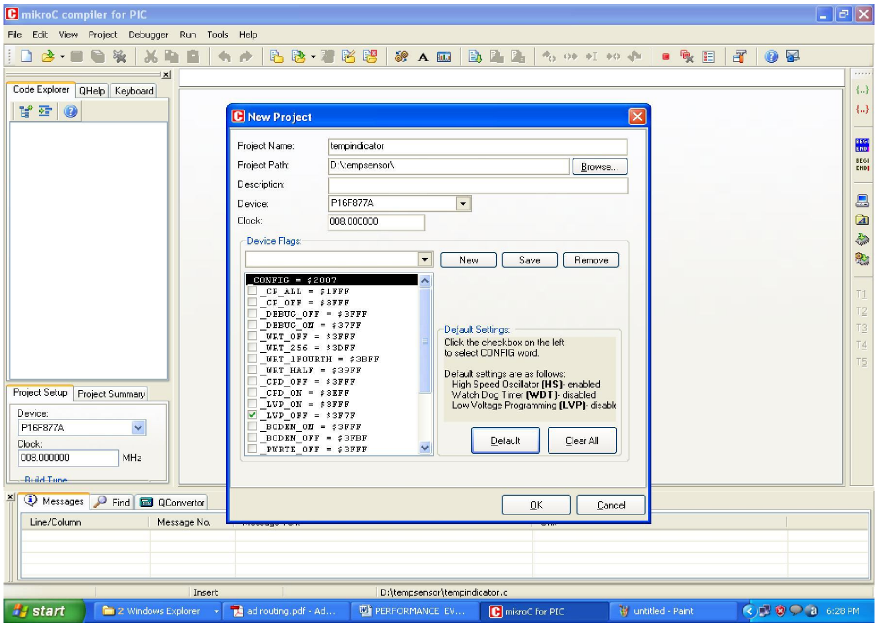
Device: P16F877A
Clock: 008.000000 MHz

Line/Column	Message No.	Message Text	Unit
0:0	100	Success (Release Build)	
0:0	101	Used ROM: 1352 (16%)	Used RAM: 44 (11%)
0:0	102	Free ROM: 6839 (84%)	Free RAM: 324 (89%)

1: 1 Insert D:\tempsensor\tempindicator.c

Fig. 3-4 Messages of Compilation

Step 4: The fourth step is to save the created code into the respective destination as in figure (3-5). The leftmost toolbar exhibits a **Create New Project** option. This project is named as “temp indicator” and saved in the D: / drive. The device name has been given as PIC16F877A and the clock speed is input as 8 MHz.



New Project

Project Name: tempindicator
Project Path: D:\tempsensor\
Description:
Device: P16F877A
Clock: 008.000000

Device Flags:

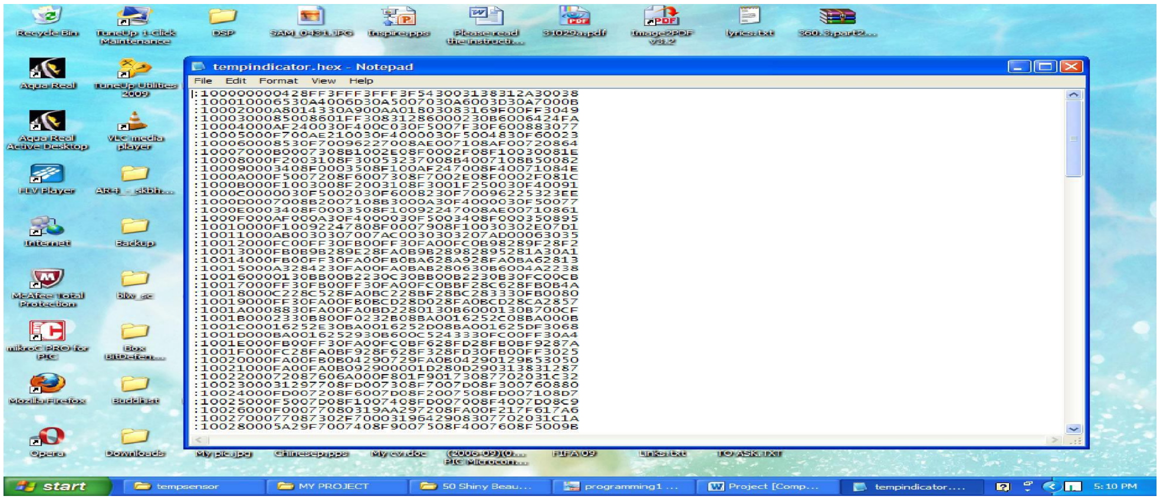
```
_CONFIG = $2007
□ _CP_ALL = $1FFF
□ _CP_OFF = $3FFF
□ _DEBUG_OFF = $3FFF
□ _DEBUG_ON = $37FF
□ _WRT_OFF = $3FFF
□ _WRT_256 = $3DFF
□ _WRT_1FOURTH = $3BFF
□ _WRT_HALF = $39FF
□ _CPD_OFF = $3FFF
□ _CPD_ON = $3EFF
□ _LVP_ON = $3FFF
☑ _LVP_OFF = $3F7F
□ _BODEN_ON = $3FFF
□ _BODEN_OFF = $3FBF
□ _PWRTE_OFF = $3FFF
```

Default Settings:
Click the checkbox on the left to select CONFIG word.

Default settings are as follows:
High Speed Oscillator **(HS)**- enabled
Watch Dog Timer **(WDT)**- disabled
Low Voltage Programming **(LVP)**- disabl

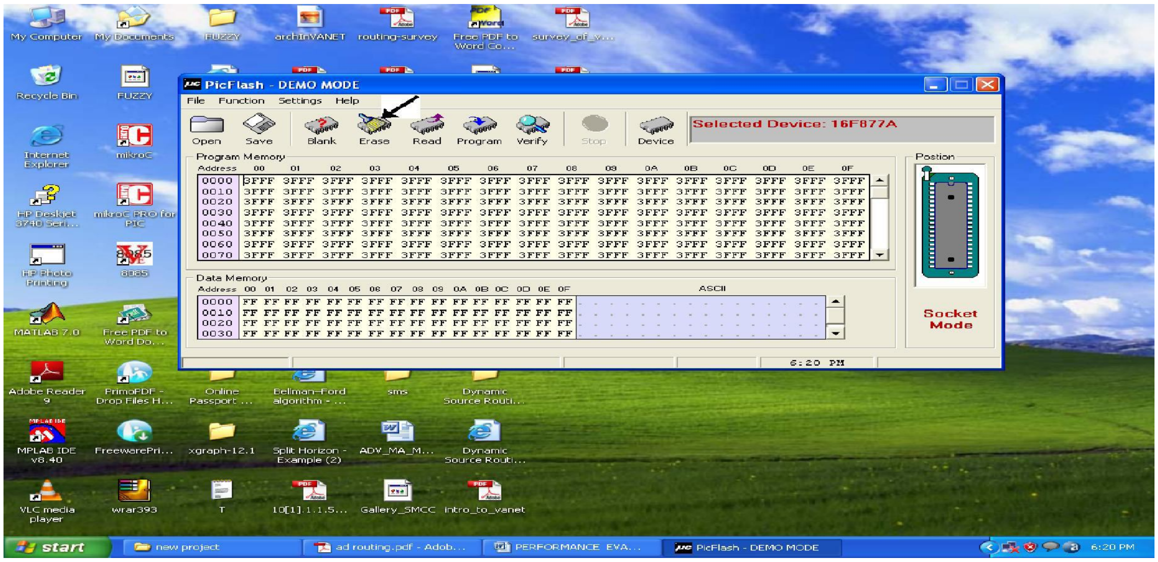
Fig. 3-5 Project setting

After final generation of the HEX code it looks like the following as in figure (3-6).



```
:100000000428FF3FFF3FFF3F543003138312A30038
:1000100006530A4006D30A5007030A6003D30A7000B
:10002000A8014330A900AA01803083169F00FF3049
:1000300085008601FF308312860002 30B6006424FA
:10004000AF240030F400C030F5007F30F600883077
:10005000F700AE210030F4000030F5004830F60023
:1000600008530F7009622700 8AE007108AF00720864
:10007000B0007308B1002E08F0002F08F10030081E
:10008000F2003108F3005323700 8B4007108B50082
:100090003408F0003508F100AF247008F40071084E
:1000A000F5007208F6007308F7002E08F0002F081C
:1000B000F1003008F2003108F3001F250030F40091
:1000C0000030F5002030F6008230F70096225323EE
:1000D0007008B2007108B3000A30F4000030F50077
:1000E0003408F0003508F10092247008AE00710861
:1000F000AF000A30F4000030F5003408F000350895
:10010000F10092247808F0007908F10030302E07D1
:10011000AB0030307007AC0030303207AD00063035
:10012000FC00FF30FB00FF30FA00FC0B98289F28F2
:10013000FB0B9B289E28FA0B9B2898289528 1A30A1
:10014000FB00FF30FA00FB0BA628A928FA0BA62813
:10015000A3284230FA00FA0BAB280630B6004A2238
:10016000 0130BB00B2230C30BB00B2230B30FC00CB
:10017000FF30FB00FF30FA00FC0BBF28C628FB0B4A
:10018000C228C528FA0BC228BF28BC283330FB0080
:10019000FF30FA00FB0BCD28D028FA0BCD28CA2857
:1001A0008830FA00FA0BD22801 30B6000130B700CF
:1001B0002330B800F0232B08BA0016252C08BA000B
:1001C00016252E30BA0016252D08BA001625DF3068
:1001D000BA0016252930B600C5243330FC00FF30A4
:1001E000FB00FF30FA00FC0BF628FD28FB0BF9287A
:1001F000FC28FA0BF928F628F328FD30FB00FF3025
:10020000FA00FB0B04290729FA0B04290129B53050
:10021000FA00FA0B092900001D280D290313831287
:100220007208 7606A000F801F9017308770203 1C32
:100230003129 7708FD007308F7007D08F30076 0880
:10024000FD007208F6007D08F2007508FD007108D7
:10025000F5007D08F1007408FD007008F4007D08C9
:10026000F00077080319AA297208FA00F217F617A6
:1002700007708 7302F70003196429083077 02031C1A
:100280005A29F7007408F9007508F4007608F5009B
```

Step 5: The next software that is used is µC which comes along the Universal Debugger and is used for downloading the code into the debugger. As in figure (3-7)Before loading the previously created HEX code into the microcontroller, it is first erased with the help of this software to remove any previously loaded programs. For this we click on the **ERASE** option available on the middle of the toolbar as shown in the figure.

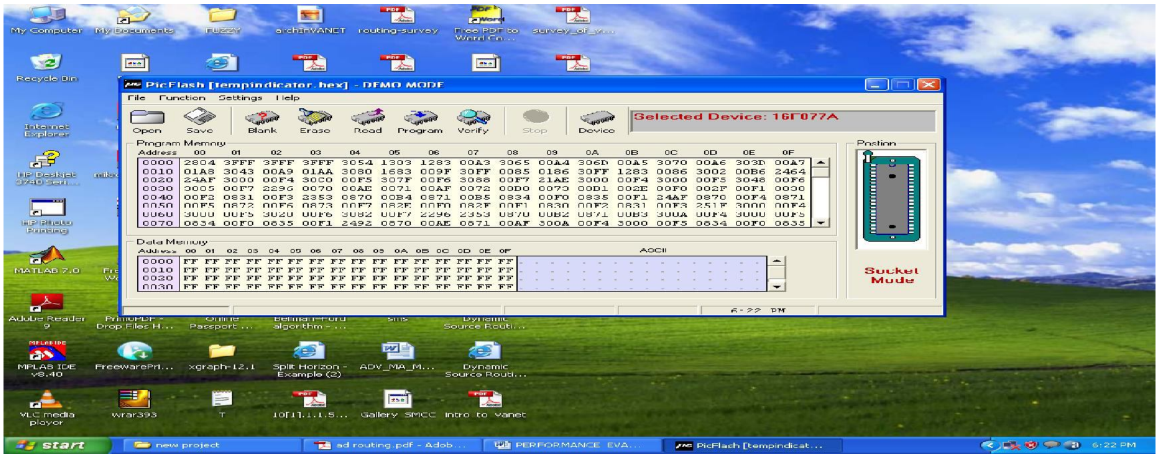


Selected Device: 16F877A

```
Program Memory
Address  00   01   02   03   04   05   06   07   08   09   0A   0B   0C   0D   0E   0F
0000    3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0010    3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0020    3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0030    3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0040    3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0050    3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0060    3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0070    3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF

Data Memory
Address 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F   ASCII
0000    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   . . . . . . . . . . . . . . . .
0010    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   . . . . . . . . . . . . . . . .
0020    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   . . . . . . . . . . . . . . . .
0030    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   . . . . . . . . . . . . . . . .
```

Fig. 3-7 HEX code downloads software

Step 6: The final step is to write the hex code program into the microcontroller as in figure (3-8). For this the debugger is interfaced with the computer via RS 232 cable or USB. After successful interfacing the code is downloaded into the chip via the µC software. After successful downloading it looks like the following.



PicFlash [tempindicator.hex] - DEMO MODE

Selected Device: 16F077A

```
Program Memory
Address  00   01   02   03   04   05   06   07   08   09   0A   0B   0C   0D   0E   0F
0000    2804 3FFF 3FFF 3FFF 3054 1303 1283 00A3 3065 00A4 306D 00A5 3070 00A6 303D 00A7
0010    01A8 3043 00A9 01AA 3080 1683 009F 30FF 0085 0186 30FF 1283 0086 3002 00B6 2464
0020    24AF 3000 00F4 30C0 00F5 307F 00F6 3088 00F7 21AE 3000 00F4 3000 00F5 3048 00F6
0030    3005 00F7 2296 0070 00AE 0071 00AF 0072 00D0 0073 00D1 002E 00F0 002F 00F1 0030
0040    00F2 0831 00F3 2353 0870 00B4 0871 00B5 0834 00F0 0835 00F1 24AF 0870 00F4 0871
0050    00F5 0872 00F6 0873 00F7 082E 00F0 082F 00F1 0830 00F2 0831 00F3 251F 3000 00F4
0060    3000 00F5 3020 00F6 3082 00F7 2296 2353 0870 00B2 0871 00B3 300A 00F4 3000 00F5
0070    0834 00F0 0835 00F1 2492 0870 00AE 0871 00AF 300A 00F4 3000 00F5 0834 00F0 0835

Data Memory
Address 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F   ASCII
0000    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   . . . . . . . . . . . . . . . .
0010    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   . . . . . . . . . . . . . . . .
0020    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   . . . . . . . . . . . . . . . .
0030    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   . . . . . . . . . . . . . . . .
```

Fig. 3-8 HEX code placed in µC software

3.2.2 PWM and A/C

Line following Robotic movement is actually the control of motors. An actuator is essentially a DC motor equipped with an electronic circuit to control the motor rotation and direction. The controller uses PWM to control the seed of motors control. The A/C need for sensors .To achieve such a control simulation of a motor by PWM signal I used software Proteus Isis 7.

A program was also written to change the duty cycle of the pulse width modulation (PWM). By changing the duty cycle of a pulse sequence, one can change the average power supplied by that sequence. Therefore, the speed of the motor will change based on the duty cycle (i.e., average power) of the pulse sequence[6]. The DC motors have high speed therefore the PWM was used to reduce the speed of the motor determine the application.

Pulse-width modulation control works by switching the power supplied to the motor on and off very rapidly. The DC voltage is converted to a square-wave signal, alternating between fully on (nearly Vm) and zero, giving the motor a series of power "kicks" as showing in Figure 3.16. If the switching frequency is high enough, the motor runs at a steady speed due to its fly-wheel momentum. By adjusting the duty cycle of the signal (modulating the width of the pulse, hence the 'PWM'), the time fraction it is "on", and the average power can be varied, and hence the motor speed.

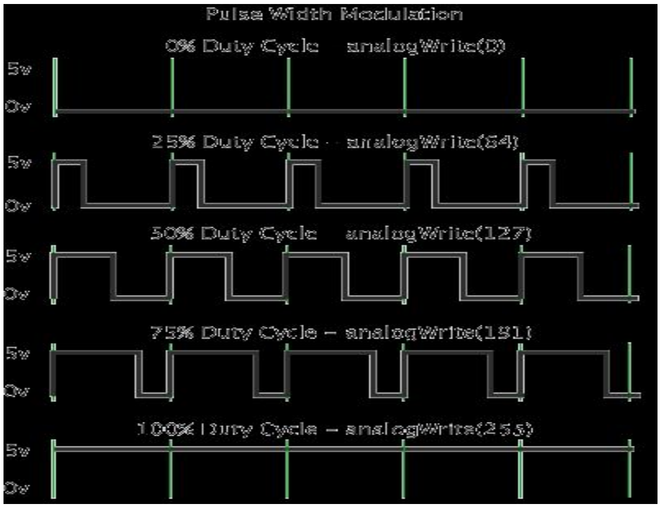


Figure 3-9 PWM signals “on” and “off “

3.2.3 MikroC ADC Libraries

There are so many PIC library function provided by the MikroC pro. But among those huge library functions, we find some of are very important for our project purpose. These are as follows.

ADC library:

ADC (Analog to Digital Converter) module is available with a number of PIC micros. Library function ADC_Read is included to provide you comfortable work with the module in single-ended mode.

Table 3.1: ADC read

Prototype	**unsigned** ADC_Read(**unsigned short** channel);
Returns	10-bit unsigned value read from the specified channel.
Description	Initializes PIC's internal ADC module to work with RC clock. Clock determines the time period necessary for performing AD conversion (min 12TAD). Parameter channel represents the channel from which the analog value is to be acquired. Refer to the appropriate datasheet for channel-to-pin mapping.
Requires	Nothing.

Example	`unsigned tmp;`
	`...`
	`tmp = ADC_Read(2);  // Read analog value from channel 2`

PWM Library:

CCP module is available with a number of PIC MCUs. MikroC PRO for PIC provides library which simplifies using PWM HW Module.

Note: Some MCUs have multiple CCP modules. In order to use the desired CCP library routine, simply change the number **1** in the prototype with the appropriate module number, i.e. PWM1_Start () and PWM2_Start (). PWM1 Library Routines used for left motor.

□ PWM1_Init

□ PWM1_Set_Duty

□ PWM1_Start

□ PWM1_Stop

PWM2 Library Routines used for right motor.

□ PWM2_Init

□ PWM2_Set_Duty

□ PWM2_Start

□ PWM2_Stop

3.3 software flowchart

The microcontroller pic16f877a was programmed to drive the motors and move the robot to follow a line and detect object depending on its input from the sensors. Figure 4-10 shows the microcontroller software flowchart.

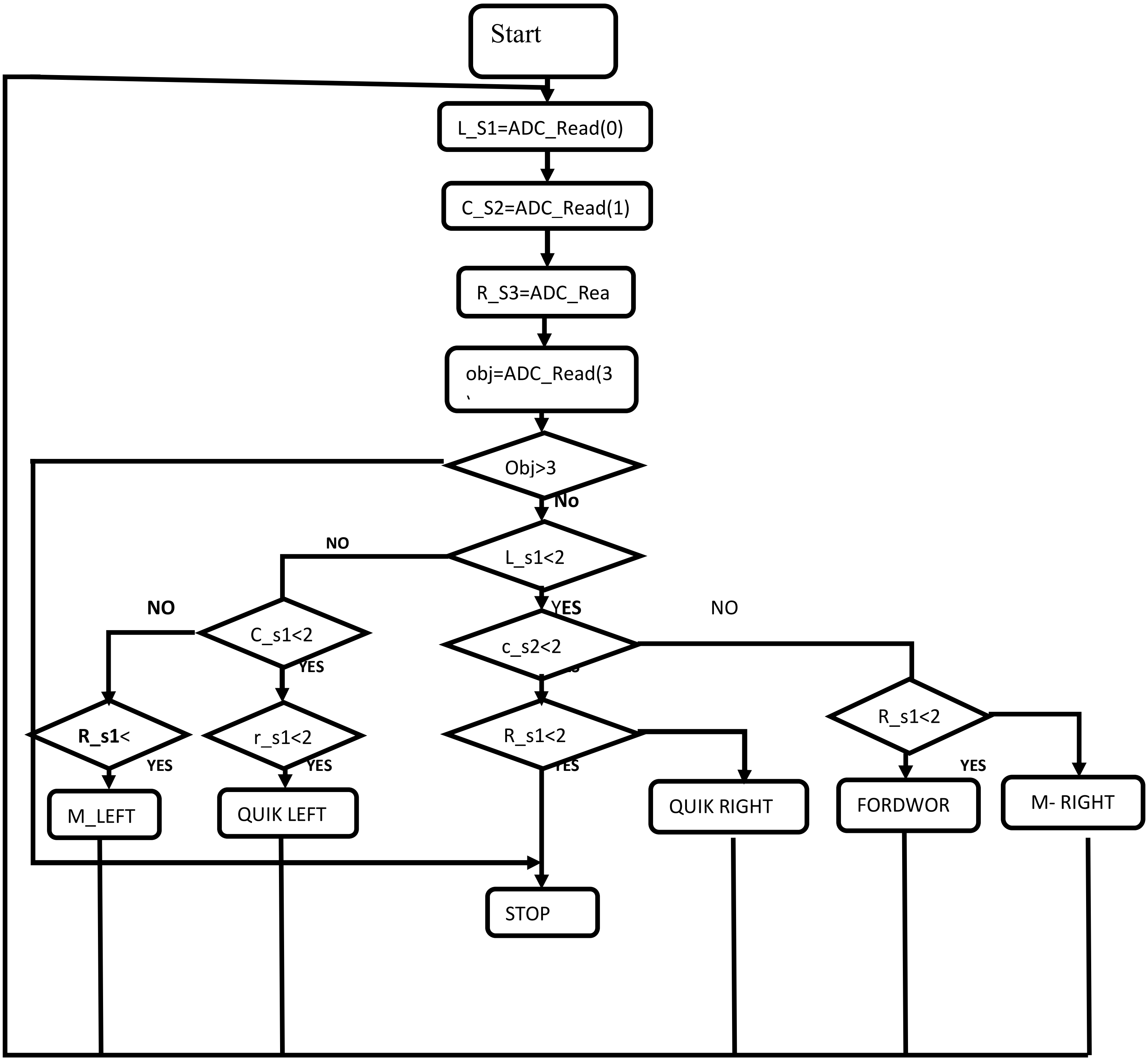


Figure 3-10 Flowchart of the microcontroller to drive the motors

The program initially reads the 4th analogue sensors and converts the data into digital format using the ADC _Read () function of the mikroC . Where the Left _Sensor, Right _Sensor, center sensor and front sensor are connected to channel 0,1,2 and channel 3 of the A/D converter, respectively which supported by PORTA in PIC16F877A.

Left _Sensor =ADC _Read (0).
Center _Sensor =ADC _Read (1).
Right _Sensor =ADC _Read (2).
Obj _Sensor =ADC _Read (3).

The ADC_ read function read the value from sensor and convert it into 10-bits digital .according to the general equation of ADC below.

$$ADC = \frac{Vin+2^\wedge n}{Vref} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (8)$$

Where:

ADC: digital to analogue converting.

$\boldsymbol{Vin}$: Voltage reading from sensor.

n: number of digits (10-bits)

$\boldsymbol{Vref}$**:** Reference voltage

After the value is readied and converted to digital then converted into float values by using the equation (2) which derived from equation(1).

$$Vin = \frac{5*ADC}{1024} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (9)$$

The lines below show the process of writing code.

```
BEGIN
        DO FOR EVER
                Read left sensor and convert to digital
                Read right sensor and convert to digital
                Read center sensor and convert to digital
                Read front sensor and convert to digital
                Calculate the float value for left sensor
                Calculate the float value for right sensor
                Calculate the float value for center sensor
                Calculate the float value for obj-sensor
                Apply the if-then condition
                Calculate the crisp outputs
                Control the motors using the crisp outputs
      END DO
END
```

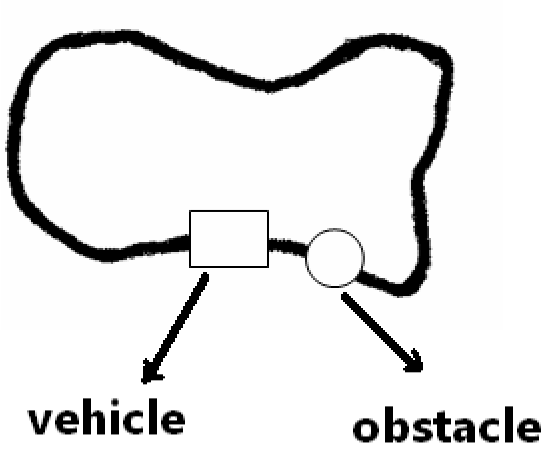


Figure 3-11 controlled line following robot.

3.4 Proteus Isis 7

ISIS is a software program that enables users to design electronic circuit's schemes or printed circuit boards. The main advantage offered by this program to others such as OrCAD, is real-time simulation of all types of electronic circuits, watching clear graphics defaults sets.. For interactive simulation of circuits, it was implemented PROTEUS VSM module. Thus using this module you can draw a complete circuit for a microcontroller based system, and then it can be tested interactively. ISIS also provides to the user options for customizing parts and components involved in making a circuit. ISIS uses the following file types and formats: Design Files (DSN), Backup files (DBK), Section-Files (CES), Module Files (. MOD), Library Files (LIB) and Netlist-Files (SDF). Type design files contain all the information about a circuit. Backup of type design files are created when the saving is carried over an existing file. Section files can be exported and read in another drawing, and those with netlist type are produced through exporting in ProsSpice and ARES. Creating a new design is made by New Order Design. Startup of this command removes all existing design data and displaying a blank standard interfaces A4 size. The created design will be saved under the name UNTITLED.DSN standard. Loading a design can be done in 3 modes. În DOS command line, by ISIS command <design name> or selecting *Open Design* once the program is on, or by double-clicking the file in *Windows Explorer*.

3.4.1 Making the circuit and simulation

If we want to move or repetitive motion, without human intervention, the robot must be one programmable. This means the need for a microcontroller use. Motors connected to the microcontroller will execute movements dictated by PWM signals which are generated by the microcontroller using a code. The code is originally written in mikroC for

PIC program in C + +, and from there is converted to assembler or hexadecimal. For this circuit, the first step is the introduction and positioning of the ISIS components. The first imported component is usually the microcontroller. For this simulation we chose the PIC16F877A microcontroller, it being the one of the most common types of microcontrollers. The most new microcontrollers include from the factory an oscillator to generate a frequency. This frequency is then used to measure the time intervals of clock signals for digital circuits. At every clock cycle the microcontroller processor performs an action. For smotor control by microcontroller it is necessary the adding a driver that realizes the interface between the output pins of microcontroller and actuator. So after adding the driver and the motor the scheme looks like this. TheL293D driver is connected to port B of the microcontroller, and the motor is connected to driver and a 12V power source. 4 LDR connect to portA as input sensor to control the direction movement of the motor, pull-up resistors was added in series with LDR. The need to use resistors to complete circuit of voltage divider to make variable output volt when we change the LDR resistor, as show in figure 3.3.To run the microcontroller simulation, it requires entering the code in hexadecimal format which was written in chapter two. This format is generated by mikroC program in C + + language. Under this program the microcontroller registers are set.

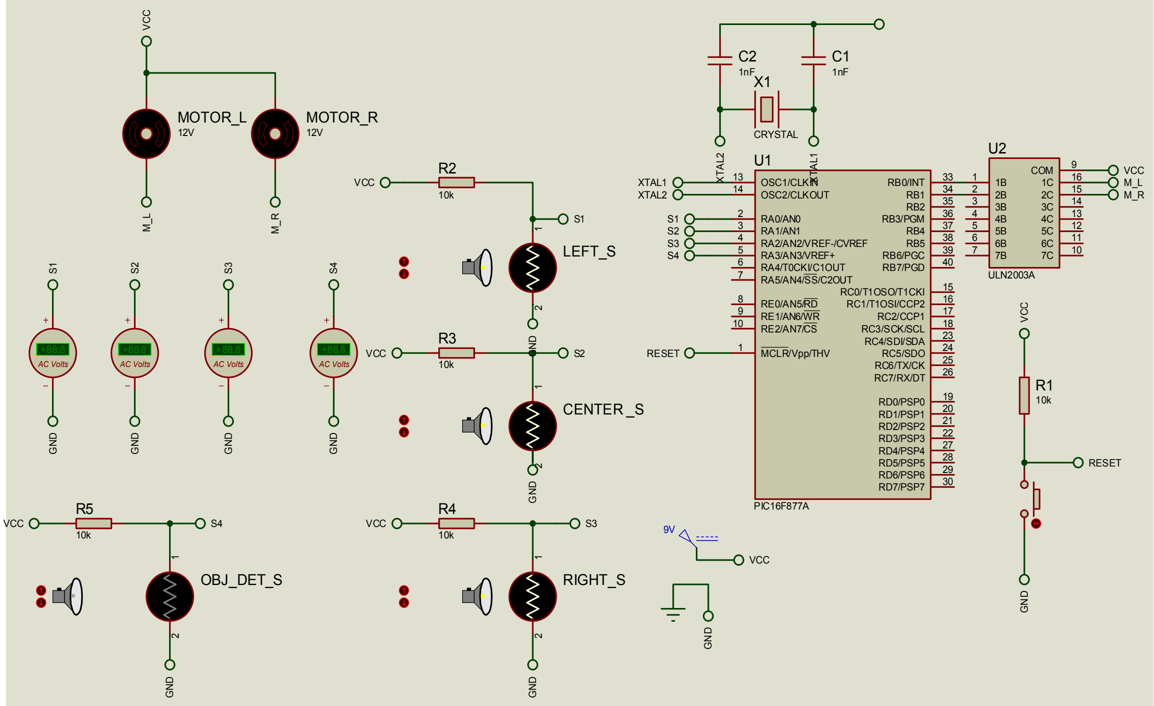


Figure 3-12 simulation circuit

After the simulation circuit was completed and tested successfully, the PCB was designed ISIS itself as in figure (3-12) the PCB has high reliability and guarantee connection.

CHAPTER 4

Electronic circuit Design

4.1 Introduction

The mobile robot is a self operating robot that detects and follows a line that is drawn on the floor and detects the object. The path consists of a black line on a white surface (or vice versa). The control system used must sense a line and maneuver the robot to stay on course, while constantly correcting the wrong moves using feedback mechanism, thus forming a simple yet effective closed loop System. The robot is also designed to follow curves.

4.2 Robot Basic Design and Requirements

The robot was built using a microcontroller PIC16F977A, a motor driver (L293D), optical sensors, IR proximity sensor and two dc motors to controlling wheels as shown in Figure 4-1.

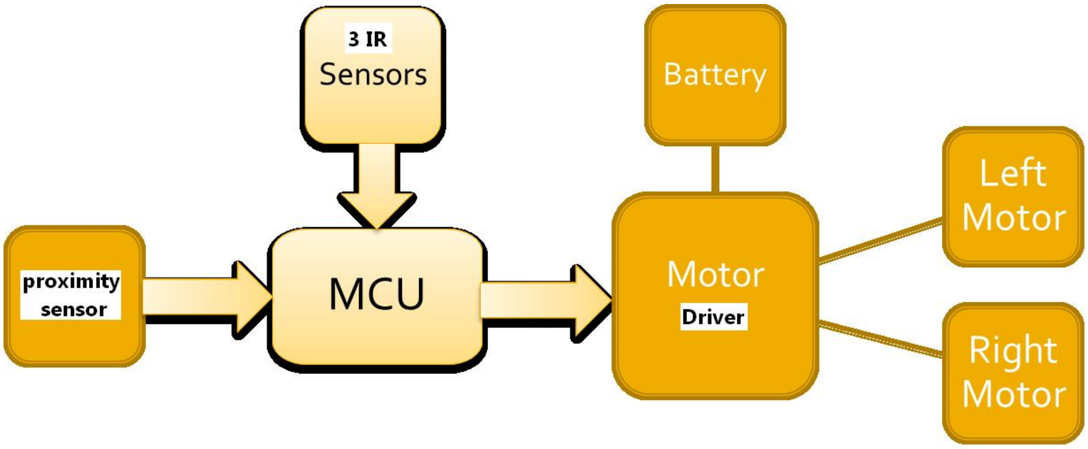


Figure 4-1mobile robot block Design

Line position is captured with the help of the optical sensors mounted at the front end of the robot. (Each sensor consists of an IR LED and an IR Sensor). When the sensors detect a surface it sends a signal to the microcontroller. If the surface detected is black then the output of the microcontroller is high, while for white surface the output is low. The output of the microcontroller controls the steering of the motors. The microcontroller together with the motor driver drives the motors and moves the robot.

4.3 Hardware Components of the Robot

The hardware components of the robot consist of the sensors, microcontroller, motors and the motor driver.

4.3.1 IR sensors

For this project, we will be using three pairs of IR sensors which will be attached to the bottom of the robot. Each sensor has one emitter (IR LED) and one receiver (IR Photo-diode). If white surface is present beneath the IR LED, IR rays are reflected and are sensed by the receiver, while in case of black surface, the light gets absorbed and hence the receiver does not sense the IR rays as shown in Figure 3.3.

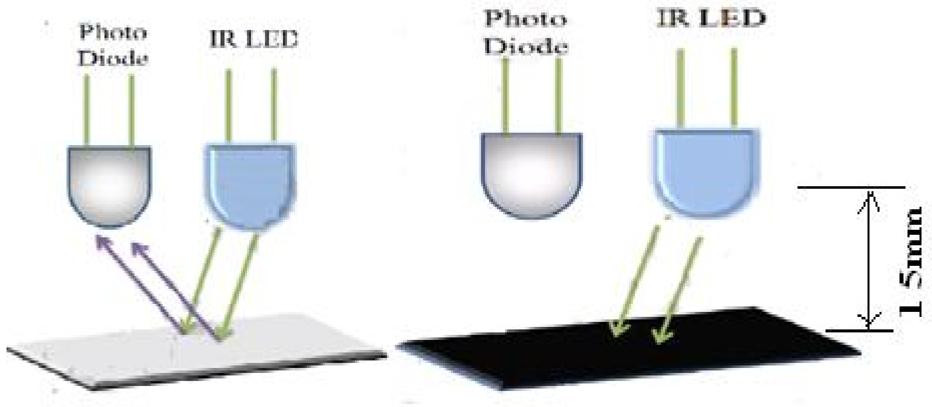


Figure 4-2: Properties of white and black surface.

These 3 sensors will be classified as left sensor, middle sensor and right sensor. A view of the placement of the sensors is as below figure 3.4.

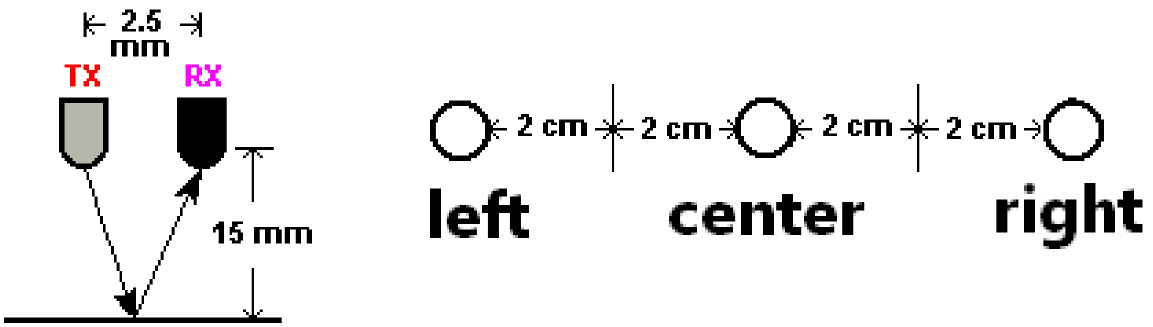
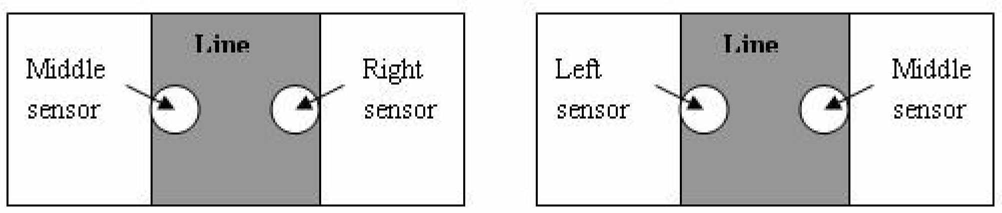


Figure 4-3 position of IR sensor

The distance between each 2 sensor depends on the width. The sensor should be placed in such a way that maximum distance of two sensors is equal to the width of the line as shown in figure 3.5. Exactly The width of line is 3 cm. 4 is the distance between each two sensors plus 0.5 cm of each sensor.



Figur4-4 the distance between each 2 sensor

4.3.1.1 The Circuit of the sensor

Photo diode has a property that if IR light falls on surface, its electrical resistance changes from 150kΩ in black surface to 10kΩ in white surface. For sensing the change in resistance a voltage divider was used as shown in Figure (3.6).

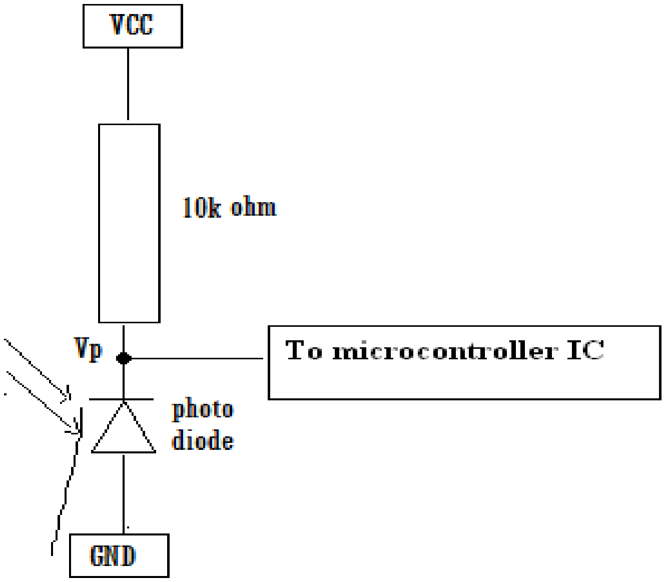


Figure 4-5 Photo detector Circuit.

4.3.1.2 Calculation of the sensor's output

To calculate the output voltage from the sensor to the microcontroller, the following equation was used:-

$$V_P = \frac{Rs}{Rs+R} * \text{Vcc} \qquad \text{......... …………………. (4.1)}$$

Given that

Rs=150kΩ without light (on black surface)

Rs=10kΩ with light (on white surface)

R=10kΩ

Then on a black surface the output voltage is:

$$V_p=\frac{Rs}{Rs+R}*\text{Vcc}=\frac{150}{150+10}*5\text{v}=4.6875\text{V}\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4.2)$$

And on a white surface the output voltage is:

$$V_p=\frac{Rs}{Rs+R}*\text{Vcc}=\frac{10}{10+10}*5\text{v}=2.5\text{V}\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4.3)$$

The output of the sensor gives varying voltages depending on the color sensed; this output is then sent to the microcontroller which determines its logic, high for black surface and low for white surface. The circuit of the sensor contains an IR LED and a Photo detector as shown in Figure 3.7.

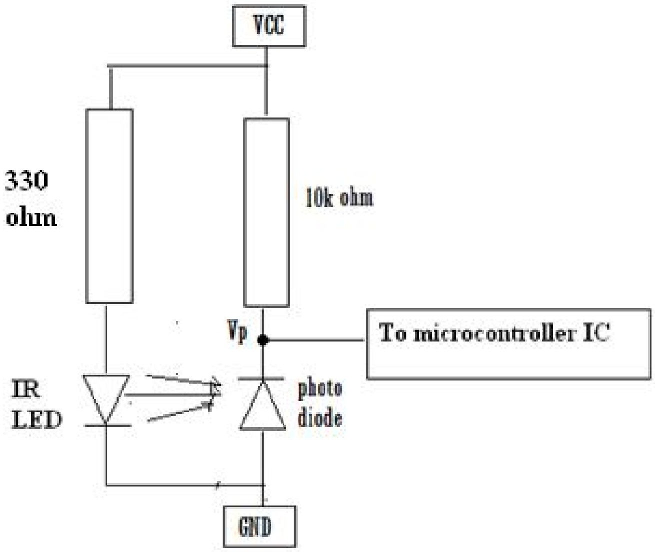


Figure 4-6 Sensor circuit

A veriable resistance can be used to determine the value of resistance to be conected serially with the IR LED, by adjusting value of the variable resistance and read the voltage across the IR detector ,the value of the resistance that provide the highest voltage across IR detector was 330 Ω [9].

4.3.2 IR Proximity sensor detection

There are many ways to realize remote object detection. The simplest is just IR Light Emitting Diode (later LED) and phototransistor. IR LED emits light with wavelength approximately 850 nanometers. Light reaches obstacle and reflects back. There it is picked up with phototransistor [6]. Voltage in measurement point changes (fig. 3.8) and this change is proportional to picked up light intensity. Comparator or ADC might be used to convert it to digital form and decide, if there is an obstacle in front of us. For example if 5 Volts supply is used: 4.5 Volts mean obstacle is far, and 1 Volt means obstacle is close (as phototransistor current flows, voltage dropout on resistor grows according to Ohm's law, and voltage in point of measurement falls).

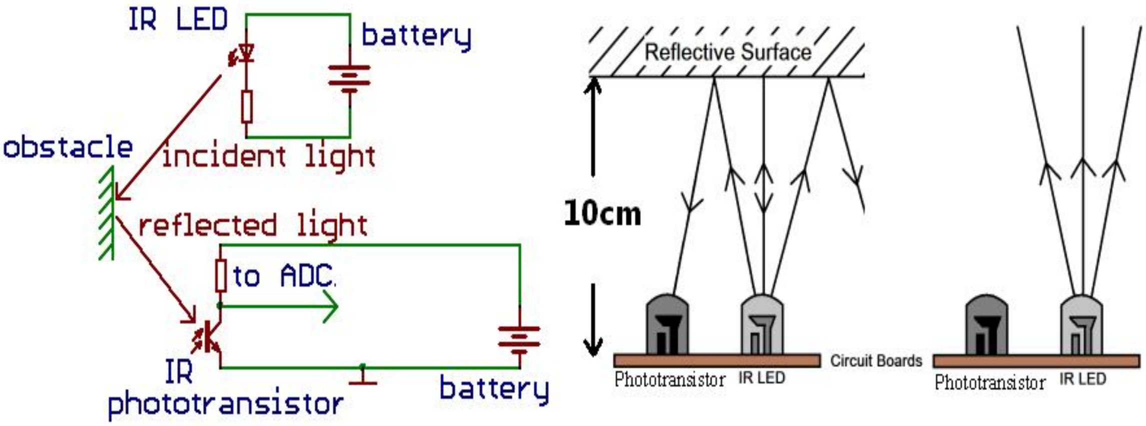


Fig 4-7 schematic of obstacle IR

As TSOP sense 36 KHz modulated light, it turns output low. It is because of the output stage, which is transistor switch. It will keep output low for some time and then again rise high. It not just sense 36 KHz but also determine if it continuous 36 KHz signal, or a burst of square waves. It rejects continuous 36 KHz like an ambient light. TSOP behavior: TSOP reaction on single burst containing 30 pulses. Burst repetition time (T) is 10 ms. Square wave frequency is 36 KHz.

4.3.2.1 Application circuit

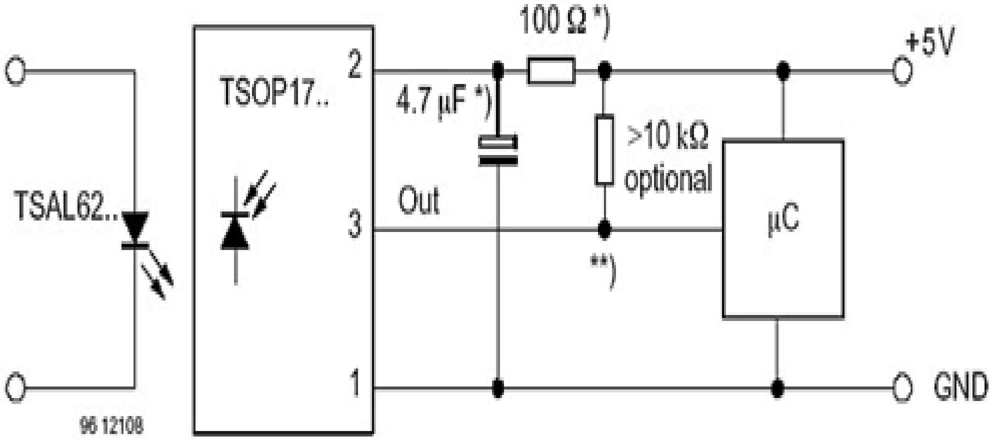


Figure 4-8 Show proximity sensor circuit

TSOP is used typically with high-intensity IR LED. For example, TSAL 6200, TSAL5100 and TSAL5200 are widely used. The best way to turn it on and off is npn transistor, because it can consume high current (up to 400 mA, depending of the current-limiting resistor value) and MCU pin may not to be able to supply it. Current can be easily calculated as (Vsourse-1.95)/Rcurr.lim. If IR LED is soldered close to TSOP false detections may occur through diode side radiation. This can be eliminated using opaque LED coating: black paper, black thermal shrink tube, etc.

For this IR bumper two MCU pins are required. One pin is used to generate 36 KHz square wave bursts 20-30 pulses each. Another pin is used to monitor output of TSOP. Square wave is generated using timers. Remember, that for 36 KHz square wave, you need to toggle LED twice faster: we use timer in microc compiler to generate this frequency as pwm.

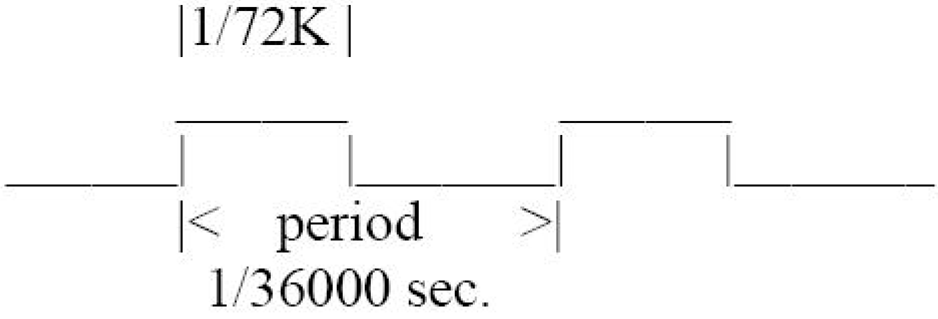


Figure 4-9 show 36 KHz square wave

4.3.3 The Microcontroller

The Microcontroller acts as the Brain of the robot, which generates desired output for corresponding inputs. There are several companies that manufacture microcontrollers, such as ATMEL, Microchip, Intel and Motorola. PIC16F877A microcontroller was shosen in this robot. It is an microchip product.

Why PIC16F877A

The choice of using **PIC16F877A** microcontroller in the robot was due to the following features it posses:-

- 256 bytes of EEPROM data memory
- Self programming
- 2 Comparators
- 8 channels of 10-bit Analog-to-Digital (A/D) converter
- 2 capture/compare/PWM functions

- synchronous serial port can be configured as either 3-wire Serial Peripheral Interface (SPI™) or the 2-wire Inter-Integrated Circuit (I²C™) bus
- Universal Asynchronous Receiver Transmitter

In order to enable the microcontroller to operate properly it is necessary to provide. basic requirement for micro Power Supply, Reset Signal and Clock Signal[9]. Figure3.11 shows the basic requirement for PIC16F877A. For more information about the PIC microcontroller, please refer to the datasheet. The datasheet can be found in Microchip web site at.

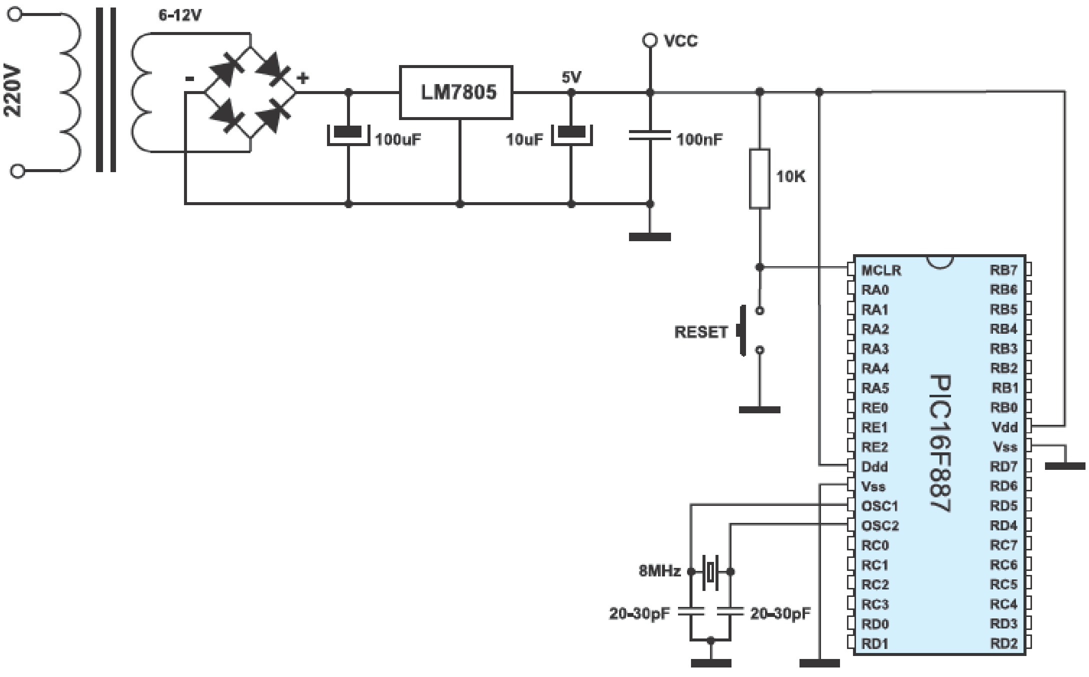


Figure 4-10 basic requirement for PIC16F877A

4.3.4 DC motor driver module

L293D is a dual H-bridge motor Driver, so with one IC we can interface two DC Motors which can be controlled in both clockwise and counter clockwise direction and I have the motor with fix direction of motion. I can make use of all the four I/Os to connect up to four DC motors. L293D has output current of 600mA and peak output current of 1.2A per channel. Moreover for protection of circuit from back E.M.F output diodes are included within the IC. The output supply (VCC2) has a wide range from 4.5V to 36 V, which has made L293d a best choice for DC motor driver. Figure 3.12 show the L293d pin configuration.

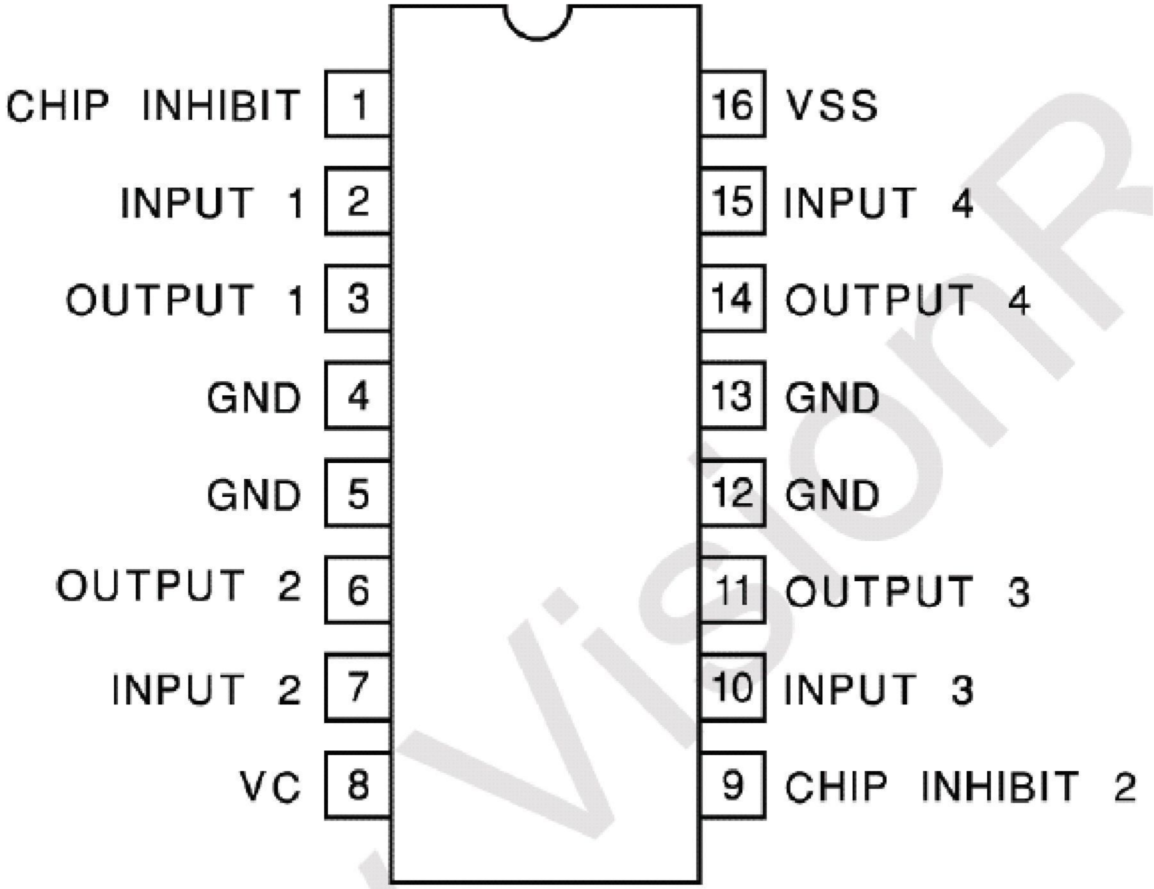


Fig. 4-11 L293D Dual H-bridge Motor Driver pin configuration

A simple schematic for interfacing a DC motor using L293D is shown below in figure 3.13.

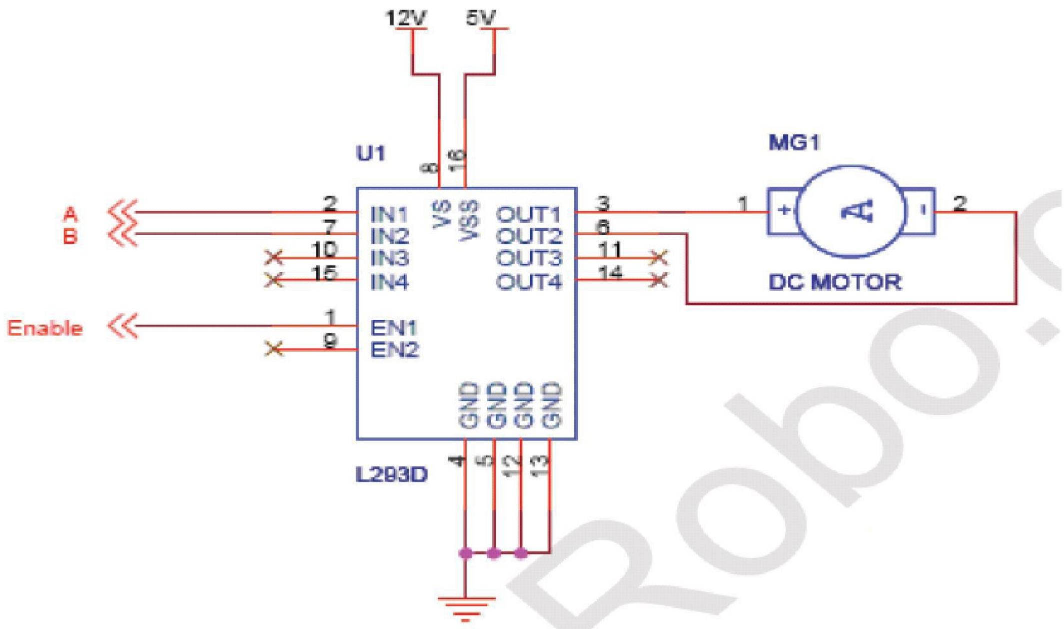


Figure 4-12 a simple schematic for LD293

Three Pins are needed for interfacing a DC motor (A, B, Enable).Output to be enabled completely then I can connect to VCC and only two pins needed from controller to make the motor work. Truth table1.1 is same for both BJT and microcontroller.

Table (4-1) Truth Table of DC motor operation

A	B	Direction
0	**0**	**Motor stops or Breaks.**
0	**1**	**Motor runs Anti-Clockwise**
1	**0**	**Motor runs Clockwise**
1	**1**	**Motor stops or Breaks**

4.3.5 Voltage regulator (KA7805)

The KA78XX/KA78XXA/LM78XX 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, make 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 to obtain adjustable voltages and currents.

Features

□ Output Current up to 1A.

□ Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V.

□ Thermal Overload Protection. □ Short Circuit Protection.

The KA7805 was chosen because the microcontroller need 5V DC

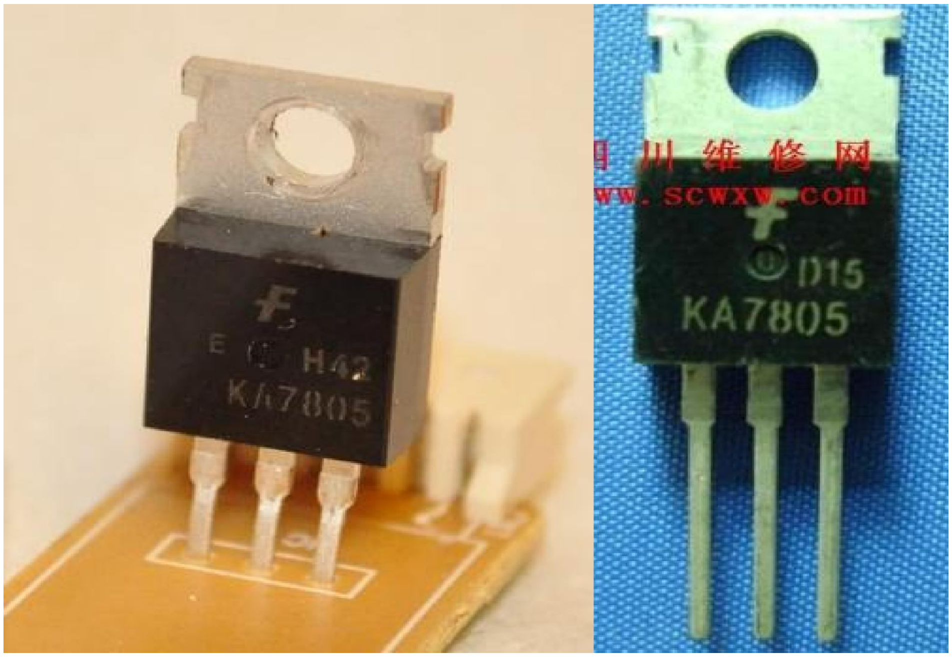


Fig. 4-13 Voltage regulators (KA7805)

4.3.6 Building the control circuit

This is the actual brain of the robot. It is the device that is programmed, to control all the other parts of the robot. Figure 3.15 shows the schematic of the circuit.

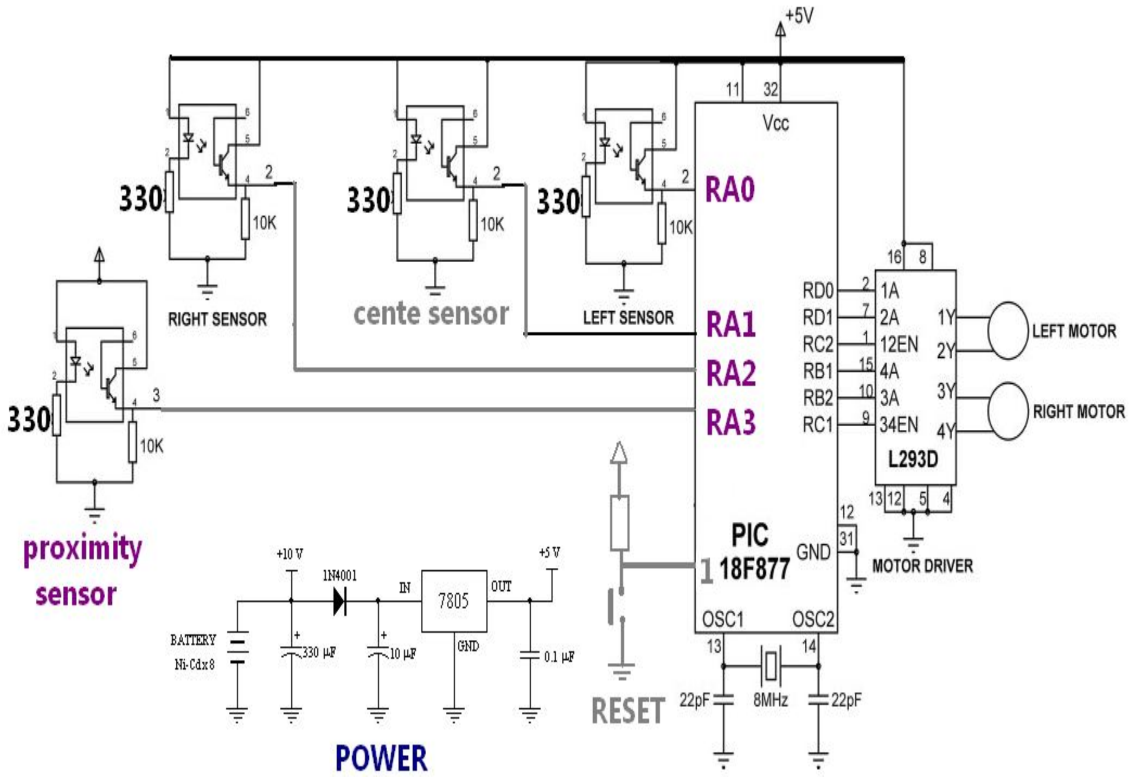


Figure 4-14 the schematic circuit

4.3.6.1 Circuit description

Being the circuit powered from a 9V battery, the regulator LM7805 provides regulated 5V for the microcontroller and for the logic gates of the motor driver. The capacitors witch added between the input/output of the regulator and the ground is used to absorb the noise caused by the presence of motors in the system.

The light sensor is composed of 2 cells, and is based on the IR emission/reception technique described in the chapter 3. D1 and D2 are used as receivers and each connects with IR LED used as emitters. The output of the line sensor is directly fed to the microcontroller ADC. The outputs of the microcontroller are connected to the motors driver and out pout of Driver connect to motors.

CHAPTER 5

Assembling and implementation of the mobile robot

5.1 Introduction

This chapter describes the assembling of mechanic and electronic parts of this robot. In this section, I will concentrate more on the implementation of mechanical and electronic and then equipped them together with software figure 5.1 shows the diagram of the robot assembling.

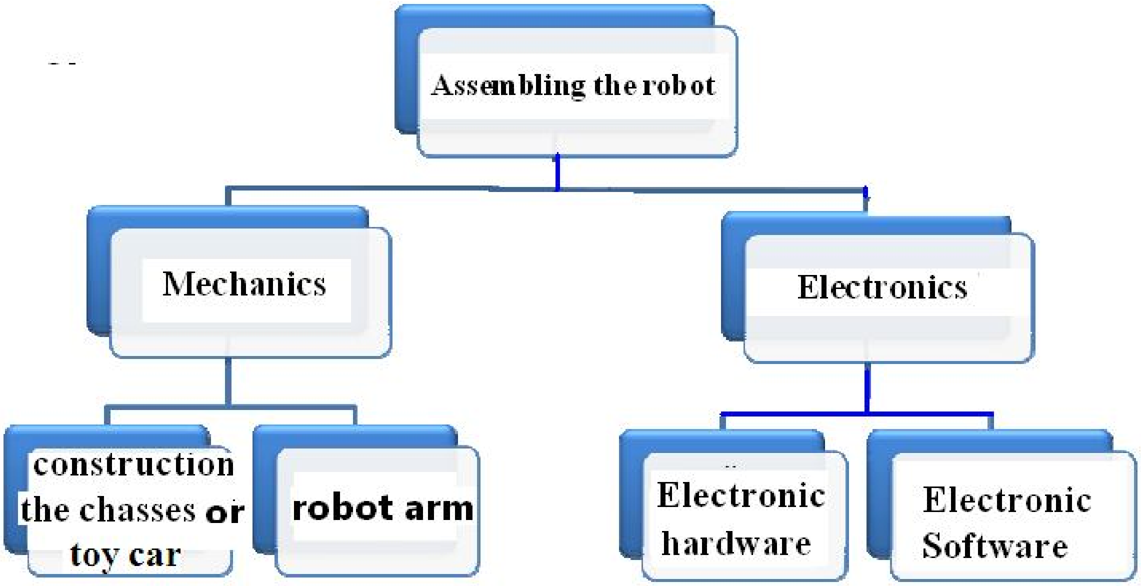


Figure 5-1: Robot assembling parts

5.2 Assembling of the electronic part

The implementation of electronics parts of the robot consist of two components; the software and the hardware components the hardware itself divided into four part, the brain of the robot which the microcontroller represent the main component of it, the sensors to view the track, IR proximity sensor to detect object and finally the hardware programmer .and the software consist from microC compiler and protus7.7 simulator and PIC KIT2 software programmer.

5.2.1 Control circuit (brain)

This control circuit is actual brain of the robot. It is the device that is programmed, to control all the other parts of the robot. After the circuit is complete designed in chapter 4 and simulated in chaper3 then the PCB is constructed by use ISIS software program.

PCB board

A printed circuit board, or PCB, is used to mechanically support and electrically connect Electronic components using conductive pathways, or traces, etched from copper sheets Laminated onto a non-conductive substrate. Conducting layers are typically made of thin copper foil. Insulating materials have a wider scale.

The construction steps of building a PCB includes; drawing the circuit Layout with the aid of special software and then printing the circuit on a transparent paper to form a mask as shown in figure 5-2.

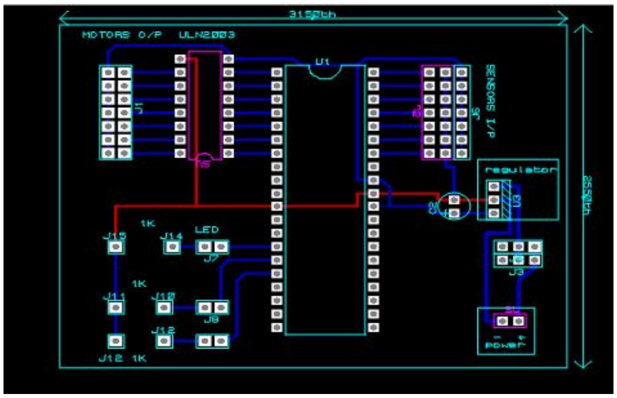


Figure 5-2: The circuit layout

Then expose the copper coated board to Ultra Vitol (UV) rays for a while, where the circuit draw mask the board making the areas of the circuit covered from UV. After that the board is dipped into several chemicals such as (HCL Fecl) to etch the unwanted copper layer leaving only the circuit drawn on board as shown in figure 5-3.

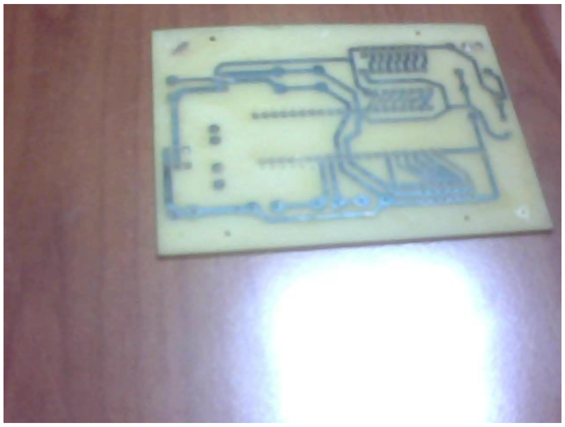


Figure 5-3: The circuit on PCB

Figure 5-4 shows all the electronic components placed and soldered onto the PCB.

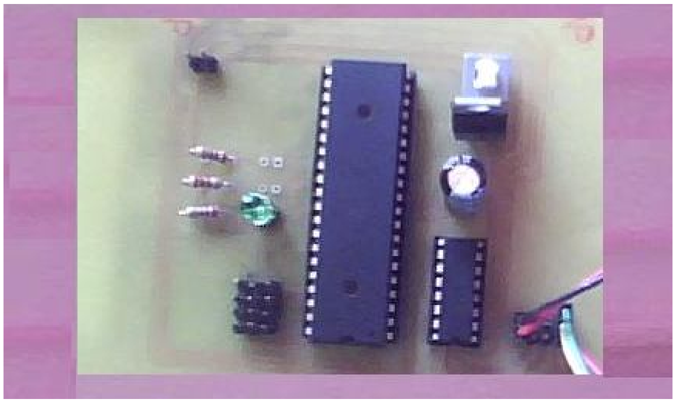


Figure 5-4: component placed

5.2.2 Sensors

For any control system both a sensor for the robot to 'see' the world and actuators to react to what it sees are needed. The IR sensor to see a line which designed in chapter 4 was implemented on a transparent paper and on a PCB board as shown in Figure 5-5 and Figure 5-6 represent three sensor build together in one PCB port.

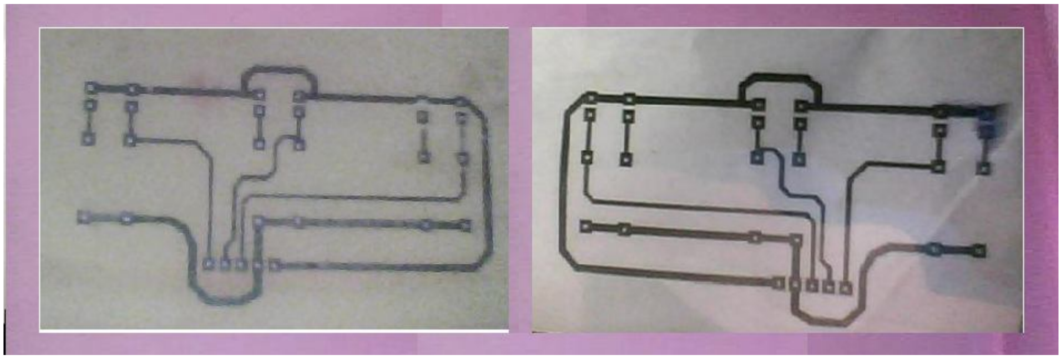


Figure 5-5: The PCB before component placed

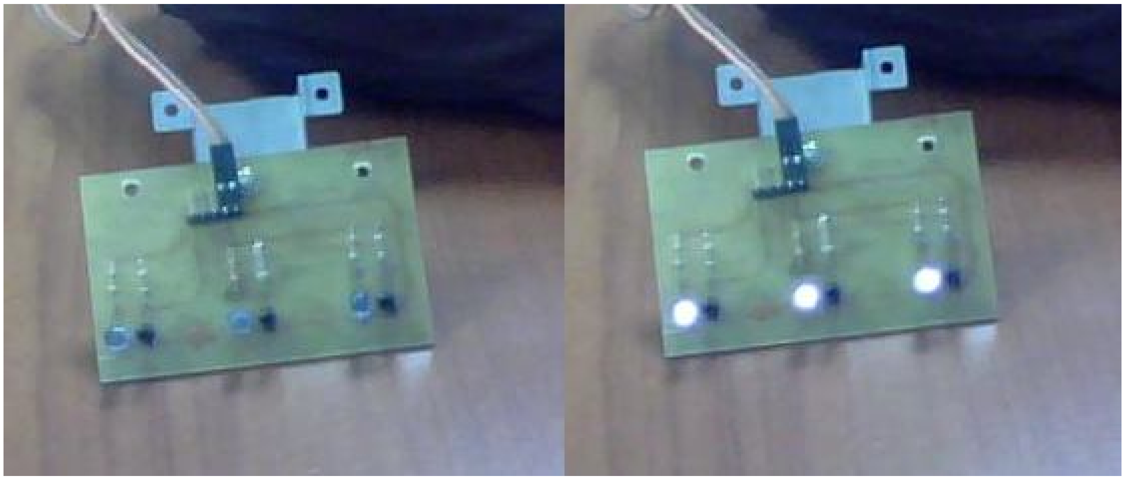


Figure 5-6: IR sensors component placed and soldered

5.2.3 IR Proximity sensor

The IR proximity sensor used for detecting object at distance abut 8cm which consist of photo transistor and IR LED the IR LED must be operated at 38Kh frequency which generated by microcontroller. The figure 5.8 show implementation of it and table 5.7 show the deferent distances vs voltages.

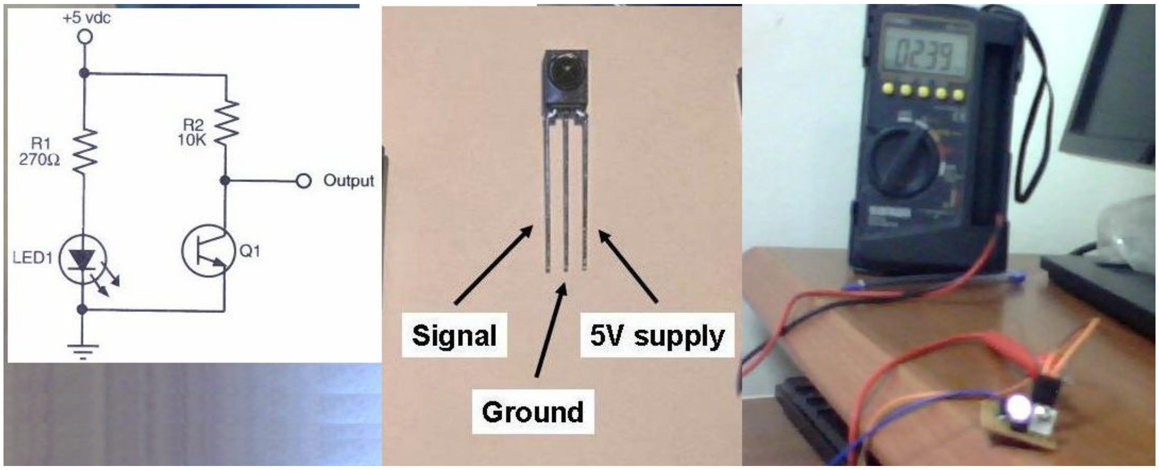


Figure 5.7 show implementation of IR proximity

5.2.4 Power supply

The power supply ensures that the robot will always get enough power and the correct voltages to run all the other parts without interruption.

Figure 5-20 shows the 9-volt battery. The robot used one rechargeable battery 9 volt and a regulator 5 volts was used to supply the 5 volts needed to power the drive wheels, microcontroller, and sensors. The 9 volts battery has the advantage of being readily available and being designed to be safe, because they are sealed. Another advantage is that the battery charger and plug come with the battery.

Figure 5-8: The 9-volt battery

5.2.5 Burner or programmer

Burner or (programmer) is device used to transfer the code which was written by microc compiler from the computer to the microcontroller IC. To transfer the code into a microcontroller, there are different burners available such as; STK-200, STK-500, STK-300, Jtag2 and PIC KIT2, PIC KIT2. But PIC KIT2 programmer was used due to its availability and suitability.

Programmer Requirement:

the PICKit has six holes at its end, five of these holes are connected to five respective pins in the microcontroller. The other two, AVDD and AVSS are used to debug, but we won’t need these today. The PICKit comes with a manual that shows a schematic of how to connect it to the a microcontroller. Pin one of the PICKit is located where the white triangle on the pickit is pointing not where the circuit is actually showing it.

.

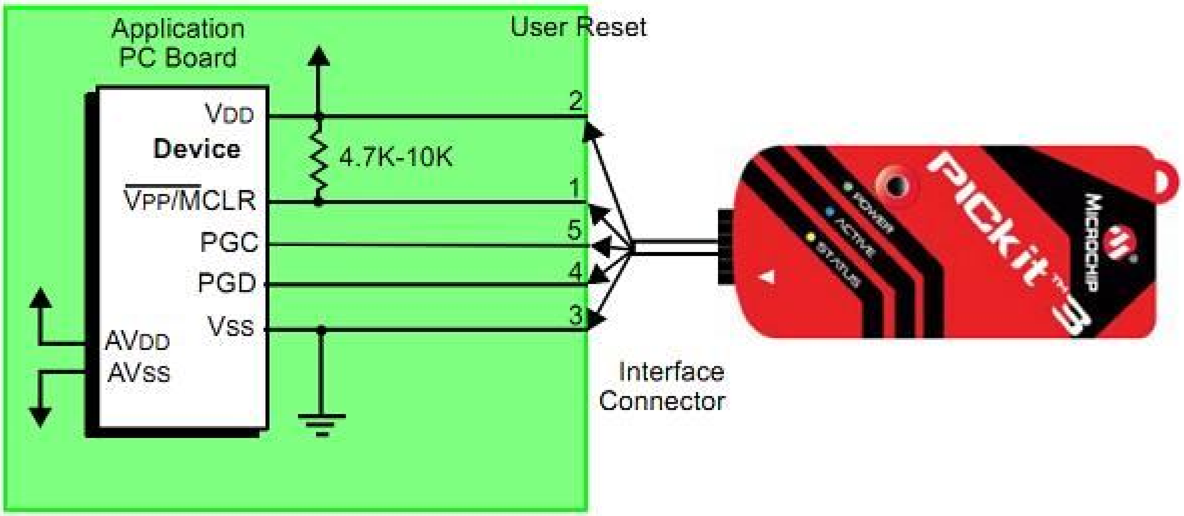


Figure 5-9: Pin connections of PIC16F877A with PIC KIT3.

After that the microcontroller was located on the breadboard, and 5volt power was supplied and all previous connections were applied. Then the code was installed by using the software interface (PIC KIT2 software programmer). Figure 5-10 shows the programmer.

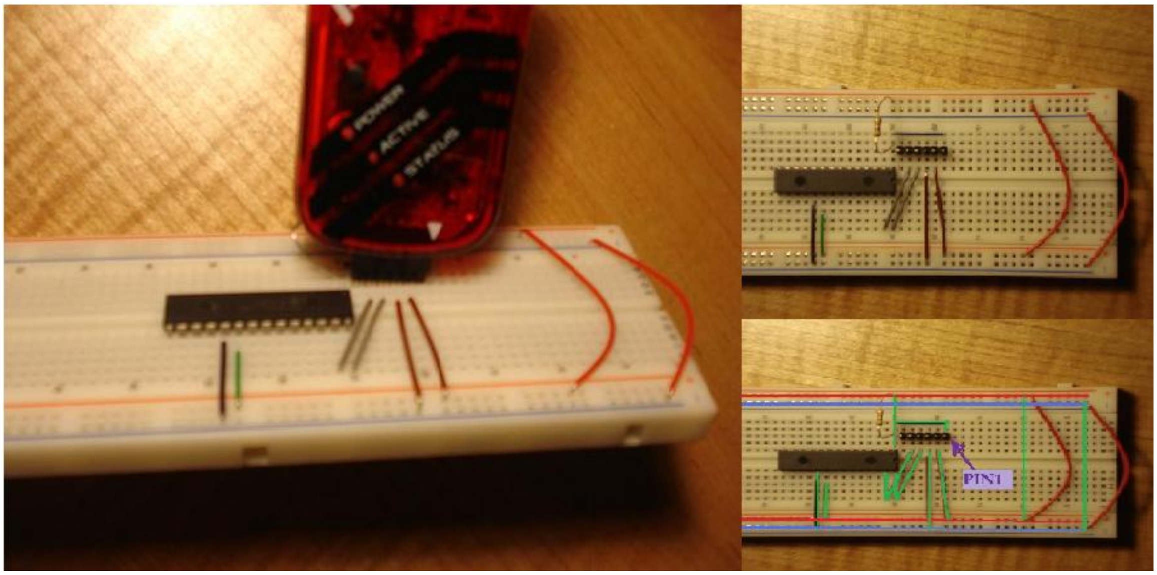


Figure 5-10: PIC KIT2 programmer.

5.3 Mechanics

5.3.1 Chassis construct

The robot body is made from different materials, such as wood, plastic, and metal. The chassis of the robot can be built in two different ways; either by building it from scratch or by simply using a toy car. The second option was chosen to buy a toy car from the market. Two conditions had to be found in it, two dc motors to move a car and one free wheel in the front end of car. Figure 5.11 shows the disassembling of car.

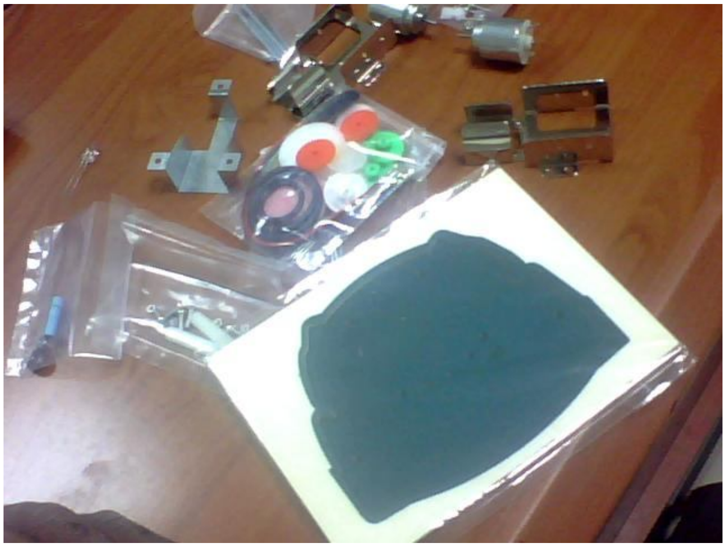


Figure 5-11: Schematic of the chassis of the robot

This option was considered in this project since it is cheaper and the design is optimal. Figure 5.12 chow top and bottom of a assembling of the chasses

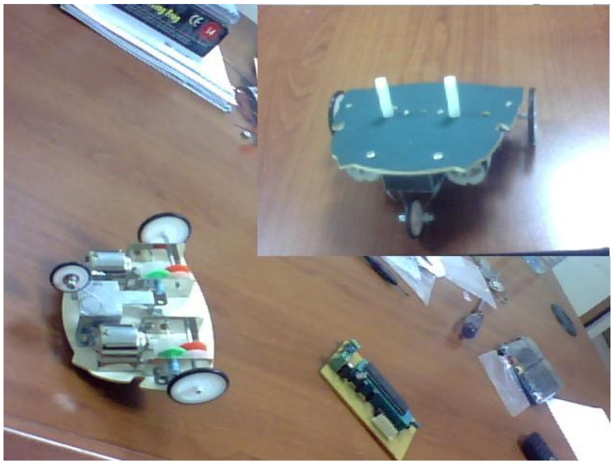


Figure 5.12 show top and bottom of a assembling of the chasses

5.5 assembling of the robot

After chasses, control circuit (brain), line sensor and proximity sensor are completed the microcontroller in programmed and the four output of sensors are connected as input to the microcontroller three for the line following and one for proximity. Then the output from L293D IC connected to the motors as in circuit designed in chapter three. Figure 5.13 shows assembling of all parts together.

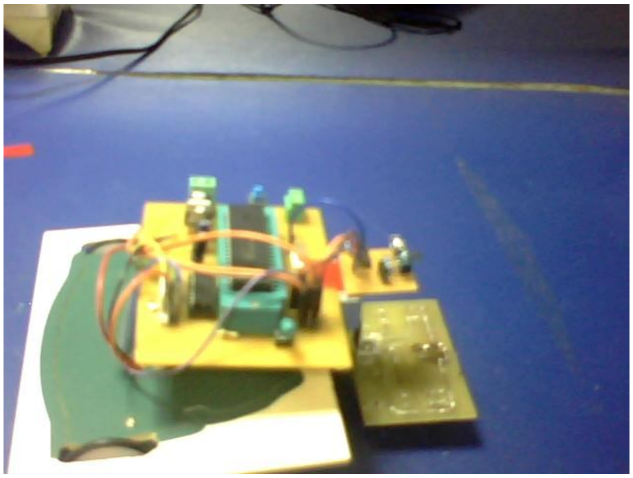


Figure 5.13 shows assembling of all parts together.

The cost of the robot vehicle is around 55$.

Table (5-1) shows the cost of the project components.

Item	Cost each ($)	Cost each SDG	Quantity
5V Linear Regulator	2	5	1
PIC16F877A	12	30	1
Micro- DIP socket	2	5	1
Uln-DIP socket	1.5	3	1
Uln2003A	2	5	1
1000µf Capacitor	0.5	1	1
20 male header	1.6	4	1
330ohm resistor	.2	0.5	2
10Kohm resistor	.2	0.5	3
LED	.5	1	2
9V Battery	2	5	1
(On/Off)switch	1.5	3	1
IR Photo resistors	2.4	6	3
Prototyping board	2.6	7	1
bread board	6	15	1
IR LED	.8	2	3
PCB	20	50	2

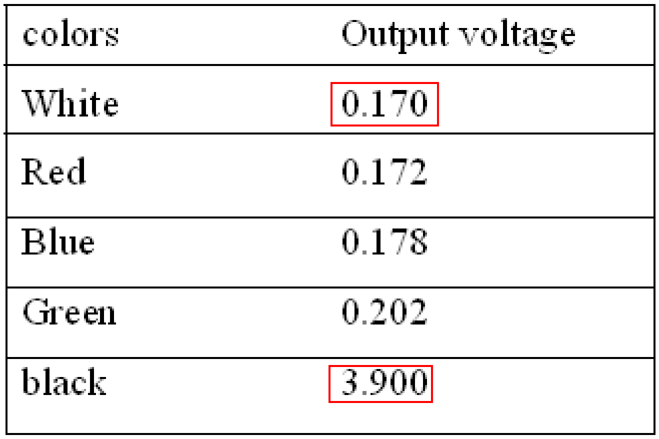
Chapter6

Result, Conclusion and Recommendation

6.1 Result

After the sensors of the track is designed and implemented there are a lot of color used for testing black, green, red, blue and white each color gives different output voltage so the black and white was chosen because high contrast between them. As shown in table 6.1 and figure 6.1.

Table 6.1 show the color vs. voltage



colors	Output voltage
White	0.170
Red	0.172
Blue	0.178
Green	0.202
black	3.900

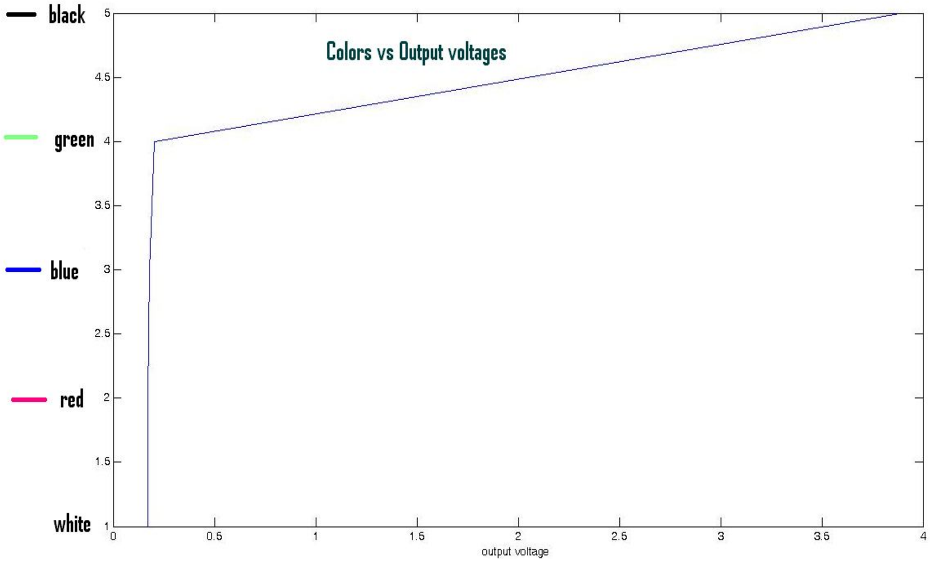
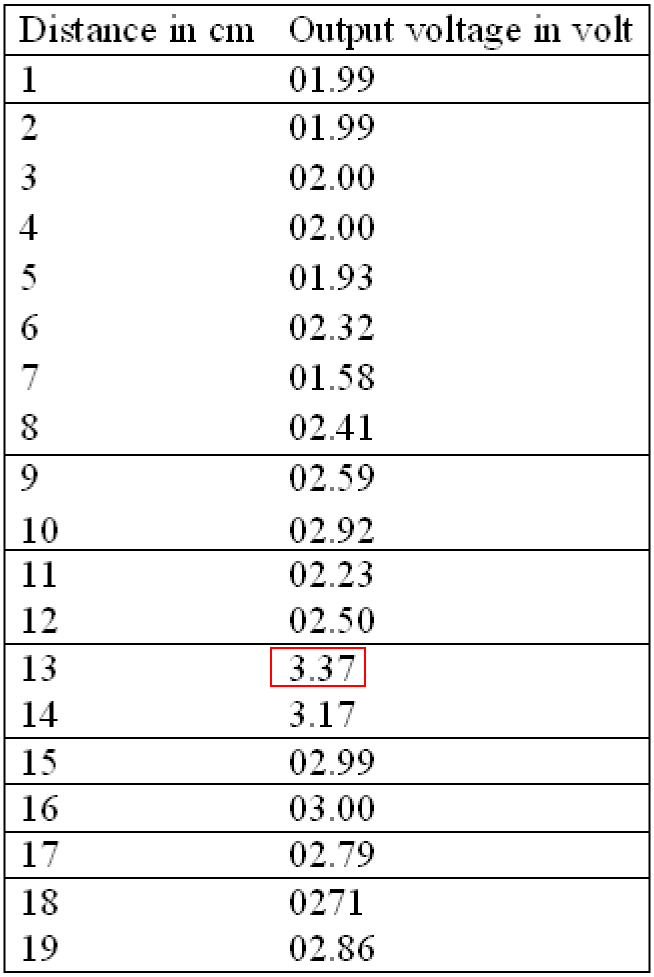


Figure 6.1 shows the color vs. voltage

And also proximity sensor witch used for detect object is tested with different distance, but the effective output voltage is fount at distance 13cm So the object detection occurs at 13cm as show in table 6.2.

Table 6.2 .show distance vs output voltage of proximity sensor



Distance in cm	Output voltage in volt
1	01.99
2	01.99
3	02.00
4	02.00
5	01.93
6	02.32
7	01.58
8	02.41
9	02.59
10	02.92
11	02.23
12	02.50
13	3.37
14	3.17
15	02.99
16	03.00
17	02.79
18	0271
19	02.86

6.2 Robot Testing

For testing the robot a white paper with a black line drawn on it was made, the assembled robot was then held in the air and the white paper was placed underneath the sensors. In doing so, both wheels of the robot rotated as expected and they slowed down when either the paper was moved away or the sensors passed across the black line. But it was realized that when the robot was placed down on the track, it didn’t move. It was found that the current was not enough to drive the motor. For solving this problem, a suitable battery with large current was placed. The robot was then able to move.

6.3 Curve error correction

To force the robot to follow a line there are a lot of error position need to correct [10]. The position of thee sensor on line determined the position of the robot figure 6.3 shows that. The maximum left and right deviation is -45 and 45 respectively.

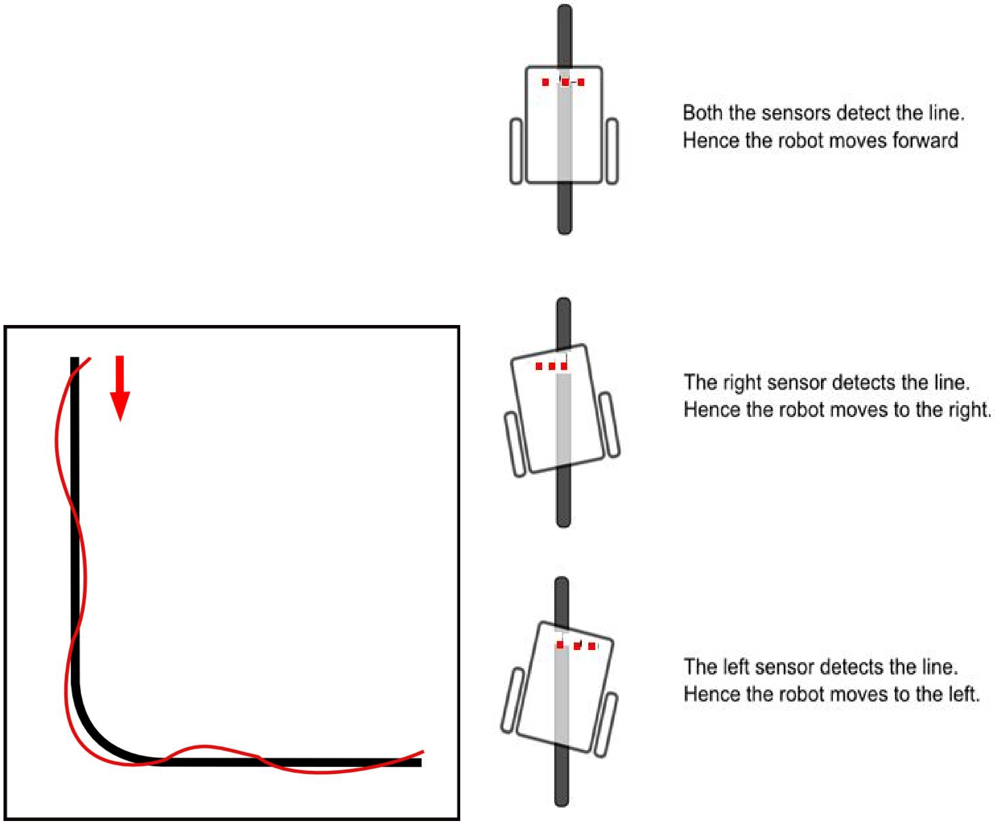

Both the sensors detect the line.
Hence the robot moves forward
The right sensor detects the line.
Hence the robot moves to the right.
The left sensor detects the line.
Hence the robot moves to the left.

Figure 6.2 show robot tracing line

To calculate the error the equation (1) and (2) was used

Error = target_pos – current_pos //calculate erro…….........………… (1)

Target – It is the position you want the line follower to always be (or try to be), that is, the center of the robot.

Current Position – It is the current position of the robot with respect to the line.

Error - It is the difference between the current position and the target. It can be negative, positive or zero. As we may know, the equation of a straight line is:

y = mx + b…………………………...……..……………………………….. (1)

Where y is the distance up (or down) the Y-axis, x the distance on the X-axis, m is the slope of the line and b is the Y intercept, the point where the line crosses the Y-axis when x is zero. The slope of the line is defined as the change in the y value divided by the change in the x value using any pair of points on the line.

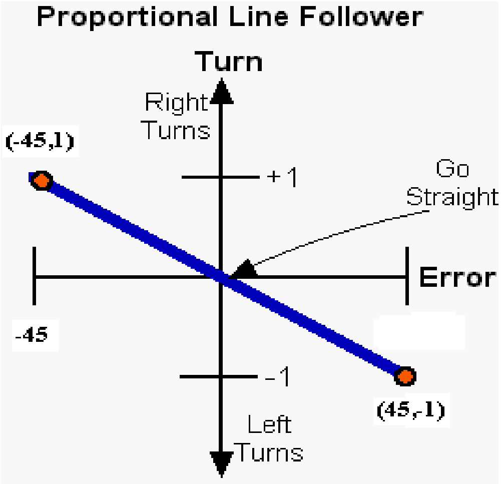


Figure 6.3 proportional line followers

In the above figure 6.3 I have shifted the axis by converting it to an error scale. Since the line now crosses the Y-axis at zero that mens b is is zero and the equation for the line is a bit simpler;

y = mx:

Turn = m***error**

We haven't yet defined what the turn axis means so for now we will just say the turns range from -1 (hard turn to the left) to +1 (hard turn to the right) and a zero turn means we are going straight. The slope of the line in the graph above can be calculated using the two points marked in red (any two points on the line will work);

Slope = m = (change in y)/ (change in x) = (1- (-1)) / (-45 - 45) = 2/-90 = -0.0222

6.3 Conclusion

As the number of elderly and disabled in need of care is increasing dramatically, services to assist them is decreasing in quality. Currently, the services provided to the elderly and disabled in most public institutions are unsatisfactory; this is largely because of their dependence on human assistance and costs. One obvious area that needs immediate attention is transportation in public areas like airport terminals, hospitals, museums, office buildings etc.

Since that the robotic technology is going through major revolutions, the goal of producing intelligent service robots that can assist people in their daily living activities is closer than ever. In this thesis a robot in the form of a vehicle was built. It uses the approach of sensing a black line drawn on a white background to move from one point to another. It is very reliable and can follow a curve.

The robot was built from scratch using the electronic components such as the microcontroller, sensors and motors. The Software such as microc and proteus 7.7 were used in programming and simulating the movement of the robot. Proteus 7.7 was also used to draw the schematic of both the circuit and the sensors which were printed on a printed circuit board.

6.4 Recommendation

While the robot vehicle was built and it moves from one point to another some possible directions for future implementation exists, which will allow variety of different and more complex algorithms to be implemented on the robot and these are:-

1. Replace the line by GPS because the GPS system is more intelligence and reliable than line tracing.
2. Adding the fuzzy logic control, neural network and genetic algorithm.

References

[1] Karel capel. [Online] [Cited: 03 10, 2010.] http://capek.misto.cz/english /biography-e.html.

[2] Hagadam, hamed, Hussein ET.AL 26-28 Aug [2]. 2013 IEEE

[3] Román Osorio C., José A. Romero, Mario Peña C., Ismael López-Juárez, ET.AL (2006) "Inteligent Line Follower Mini-Robot System"

[4] M. S. Islam & M. A. Rahman (2013) [4]

[5] Mirats Tur, J.M., Pfeiffer, C.F "Mobile robot design in education" *IEEE Robotics & Automation Mag.*, v. 13, no 1, March 2006, pp. 69.

[6] *Fundamentals of Machine Elements*, 3rd ed. S.R. Schmid, B.J. Hamrock, and B. Jacobson

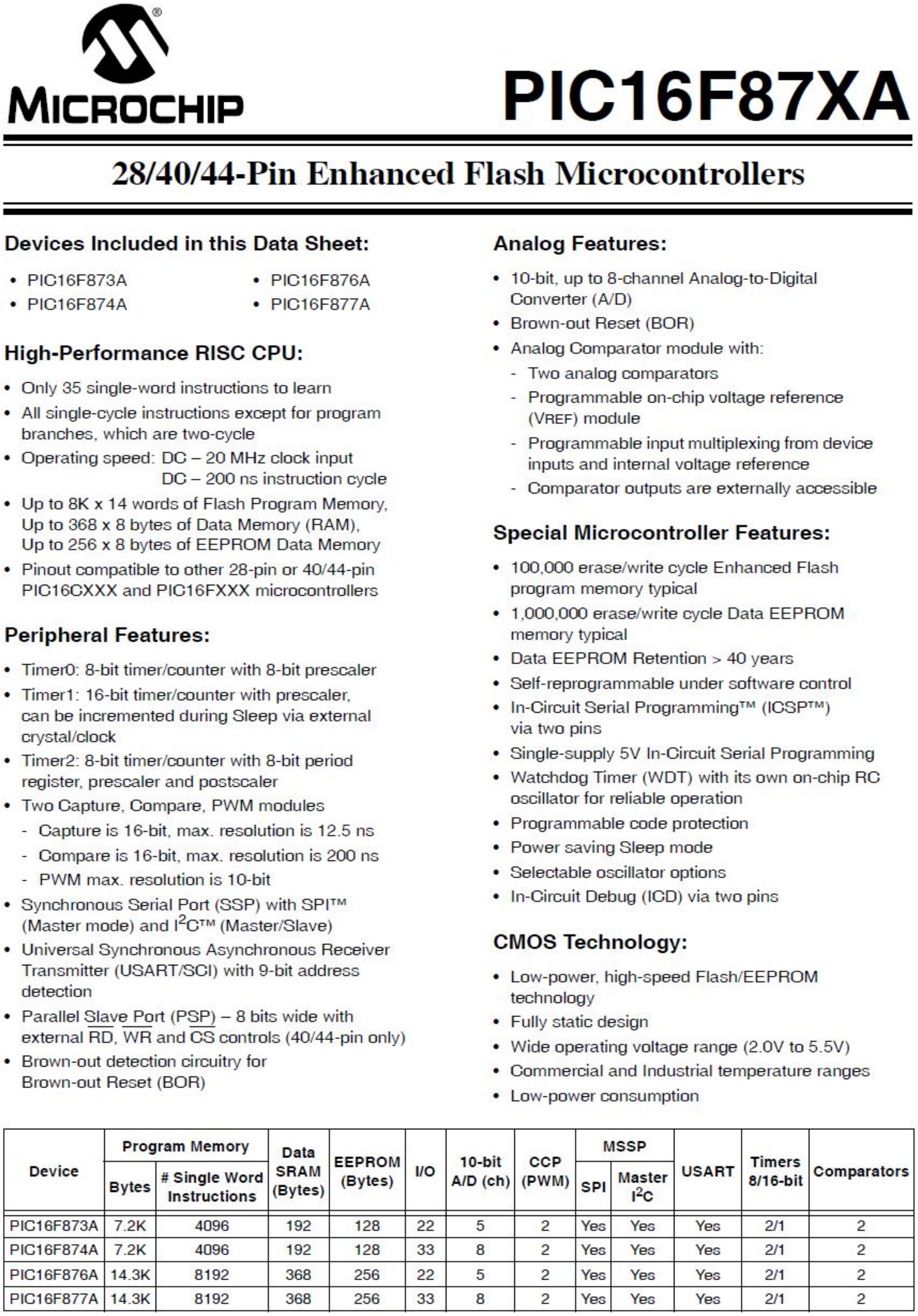
[7] C. G. Rusu and I. T. Birou, "Obstacle Avoidance Fuzzy System for Mobile Robot with IR Sensors", 10th International Conference, (2010) May 27-29.

[8] http://www.societyofrobots.com/robot_tutorial.shtml

[9]file:///E:/imortant/counter/Chapter%204%20%20Examples%20%20Book %20%20PIC%20Microcontrollers%20 %20Programming%20in%20C.htm

[10] http://ieeerobotics.wikidot.com/line-following-competition

Appendix A: PIC16F877A data sheet



MICROCHIP

PIC16F87XA

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 I^{2}C™ (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:

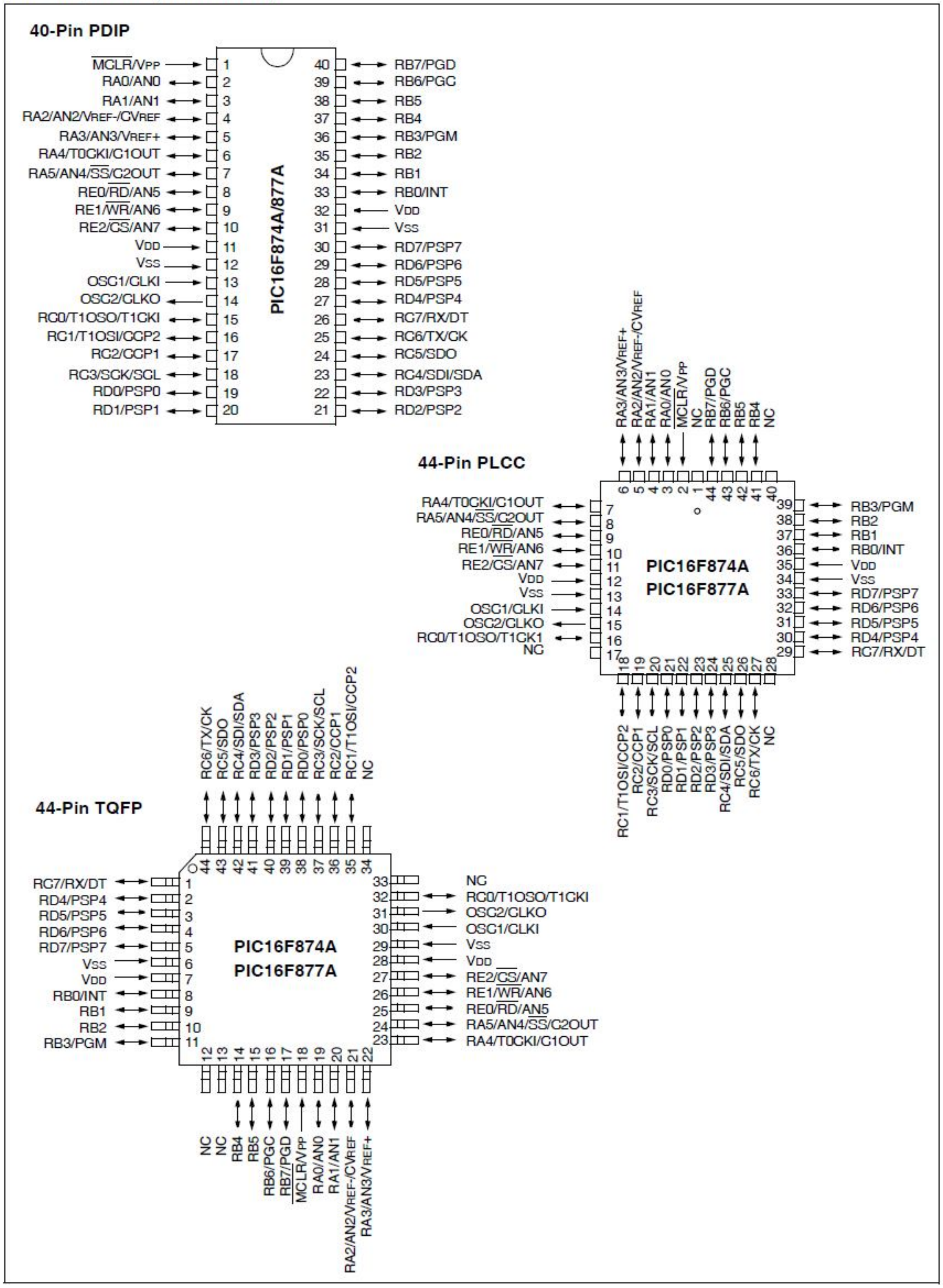
- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single-supply 5V In-Circuit Serial Programming
- 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:

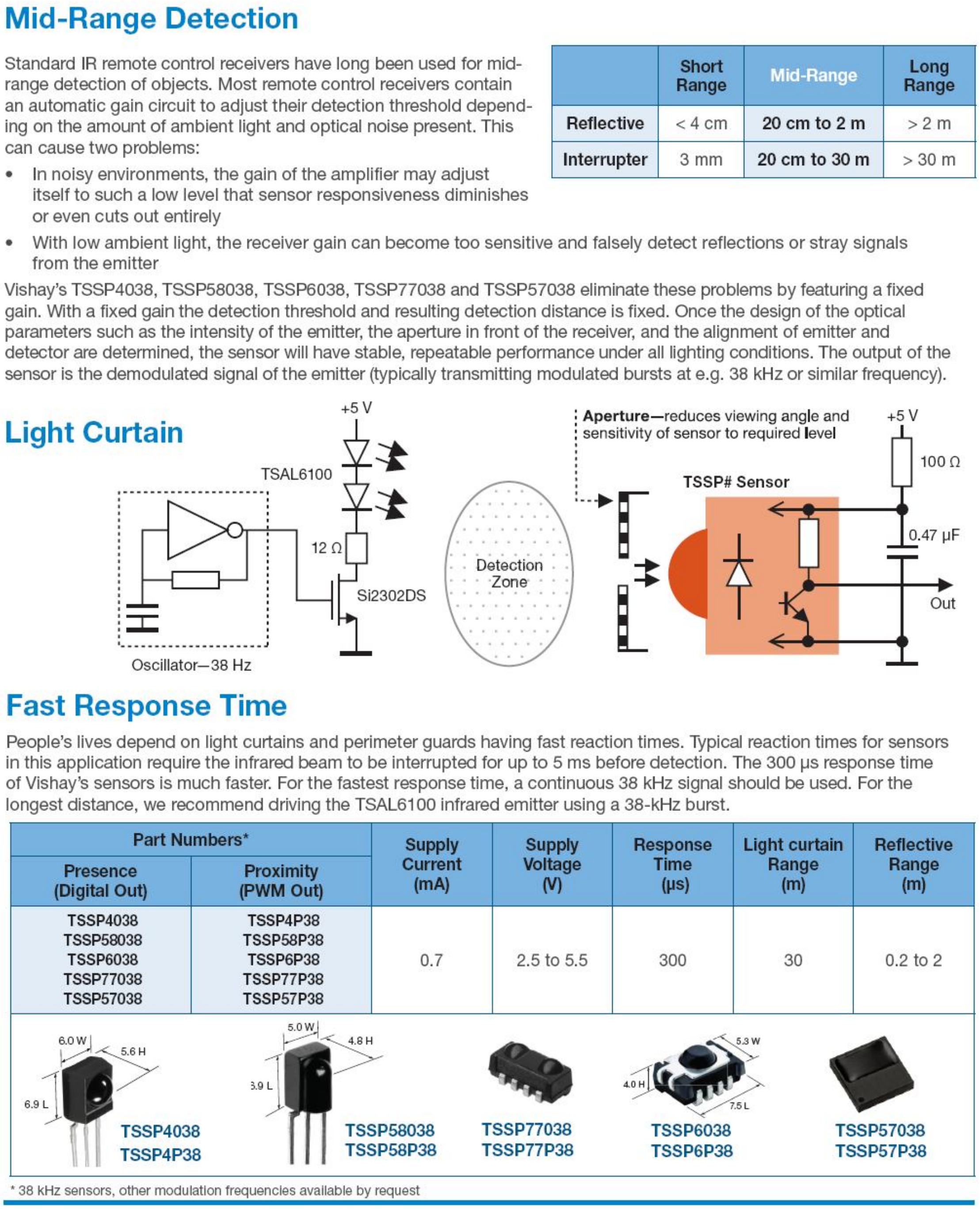
- Low-power, high-speed Flash/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

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 I^{2}C			
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)



Appendix B: Infrared Sensors for Object Detection and Ranging



Mid-Range Detection

	Short Range	Mid-Range	Long Range
Reflective	< 4 cm	20 cm to 2 m	> 2 m
Interrupter	3 mm	20 cm to 30 m	> 30 m

Standard IR remote control receivers have long been used for mid-range detection of objects. Most remote control receivers contain an automatic gain circuit to adjust their detection threshold depending on the amount of ambient light and optical noise present. This can cause two problems:

- In noisy environments, the gain of the amplifier may adjust itself to such a low level that sensor responsiveness diminishes or even cuts out entirely
- With low ambient light, the receiver gain can become too sensitive and falsely detect reflections or stray signals from the emitter

Vishay's TSSP4038, TSSP58038, TSSP6038, TSSP77038 and TSSP57038 eliminate these problems by featuring a fixed gain. With a fixed gain the detection threshold and resulting detection distance is fixed. Once the design of the optical parameters such as the intensity of the emitter, the aperture in front of the receiver, and the alignment of emitter and detector are determined, the sensor will have stable, repeatable performance under all lighting conditions. The output of the sensor is the demodulated signal of the emitter (typically transmitting modulated bursts at e.g. 38 kHz or similar frequency).

Light Curtain

Fast Response Time

People's lives depend on light curtains and perimeter guards having fast reaction times. Typical reaction times for sensors in this application require the infrared beam to be interrupted for up to 5 ms before detection. The 300 µs response time of Vishay's sensors is much faster. For the fastest response time, a continuous 38 kHz signal should be used. For the longest distance, we recommend driving the TSAL6100 infrared emitter using a 38-kHz burst.

Part Numbers*		Supply Current (mA)	Supply Voltage (V)	Response Time (µs)	Light curtain Range (m)	Reflective Range (m)
Presence (Digital Out)	Proximity (PWM Out)					
TSSP4038 TSSP58038 TSSP6038 TSSP77038 TSSP57038	TSSP4P38 TSSP58P38 TSSP6P38 TSSP77P38 TSSP57P38	0.7	2.5 to 5.5	300	30	0.2 to 2

* 38 kHz sensors, other modulation frequencies available by request

Appendix C: CODE

```
FW ()
 { portb.f0=1; portb.f1=1;
 delay_ms(500);
  portb.f0=1; portb.f1=1;
 delay_ms(500);
 }
  RI ()
 { portb.f0=1; portb.f1=0;
 delay_ms(500);
 portb.f0=1; portb.f1=0;
 delay_ms(500);
 }
  LF ()
 { portb.f0=0;  portb.f1=1;
 delay_ms(500);
 portb.f0=0; portb.f1=1;
 delay_ms(500);
 }
  M_L ()
 { portb.f0=0; portb.f1=1;
 delay_ms(500);
 portb.f1=0;
 delay_ms(500);
 }
```

```
  M_R()
 { portb.f1=0; portb.f0=1;
 delay_ms(500);
 portb.f0=0;
 delay_ms(500);
 }
 STOP ()
 {portb.f0=0; portb.f1=0;
 }
 TURN ()
 { STOP ();
 delay_ms(1000);
 RI ();
 delay_ms(1000); }
 AVOID_L()
 {M_L ();}
 AVOID_R ()
 { M_R(); }

int x,xx,xxx,xxxx;
Float T, TT, TTT, TTTT;

Void main(){
ADCON1 = 0x80; // Configure analog inputs and Vref
```

```
trisb=0;

PORTB=0x00;

trisa=0x0f;


while(1){

x=ADC_read(0);

T=5*x/1023;

xx=ADC_read(1);

TT=5*xx/1023;

xxx=ADC_read(2);

TTT=5*xxx/1023;

xxxx=ADC_read(3);

TTTT=5*xxxx/1023;

if(T<=2&&TT>2&&TTT<=2&&TTTT<3)

{

FW();}

if(T<=2&&TT<=2&&TTT>2&&TTTT<3)

{

RI();}

if(T>2&&TT<=2&&TTT<=2&&TTTT<3)

{

LF();}

if(T>2&&TT>2&&TTT<=2&&TTTT<3)

{

M_R();}
if(T<=2&&TT>2&&TTT>2&&TTTT<3)
{
M_L();}
if(T<=2&&TT>2&&TTT<=2&&TTTT>=3)
{
TURN();}
if(T<=2&&TT<=2&&TTT>2&&TTTT>=3)
{
TURN();}
if(T>2&&TT<=2&&TTT<=2&&TTTT>=3)
{
TURN () ;}
if(T>2&&TT>2&&TTT<=2&&TTTT>=3)
{
TURN () ;}
if(T<=2&&TT>2&&TTT>2&&TTTT>=3)
{
TURN () ;}
If (T<=2&&TT<=2&&TTT<=2&&TTTT<=3)
{
AVOID_L () ;}
Else
{
STOP ();
}}}
```