بسم الله الرحمن الرحيم

# SUDAN UNIVERSITY OF SCIENCE & TECHNOLOGY

## *COLLEGE OF ENGNEERING*

### *ELECTRIACL ENGINEERING DEPARTMENT*

## LINE FOLLOWING PRODUCT VAN CONTROL SYSTEM

## استخدام عربة تابعة المسار للتحكم فى نقل الانتاج

*A project Submitted in Partial Fulfillment of The Requirements of B.Sc. Degree in Electrical Engineering*


**إعداد الطلاب:**

1. مزمل محمد بابكر محمد
2. عبدالله آدم محمد عبدالرسول
3. محمد عمر ابراهيم المهدي
4. عرفات بابكر مركز السيد

الاشراف :

أ.جعفر بابكر عثمان


October 2018

# ..... الاية ....

بسم الله الرحمن الرحيم

(( فَتَعَالَى اللَّهُ الْمَلِكُ الْحَقُّ وَلَا تَعْجَلْ بِالْقُرْآنِ مِن قَبْلِ أَن يُقْضَىٰ إِلَيْكَ وَحْيُهُ وَقُل رَّبِّ زِدْنِي عِلْمًا ))

**سورة طه – الآية 114**

# DEDICATION

**Thankful appreciation for support,**

**encouragement and understandings to our**

**beloved mothers, fathers, brothers and sisters,**

# ACKNOWLEDGEMENT

*First and foremost, we would like to express our gratitude to the most Gracious and Most Merciful ALLAH S.W.T for helping me to complete this report. It has been an honor and pleasure to have* ***U. Gaffar Babiker Osman*** *as our supervisor. We are grateful to him for the time given to us to make this requirement and for his valued suggestion. In addition to his huge knowledge and experience. We enjoyed his support and patience during the very tough moment of the research work and writing of the report. Last but certainly not least, we would like to deeply acknowledge our beloved parents for their untiring efforts in providing moral and financial assistance that inspired to finish this work and also to all our friends that's been really helpful in providing us some help along with their kind opinion.*

# ABSTRACT

*The line following van is a device that detects and follows a line. The path may be visible like a black line on a white surface or vice versa or it can be invisible like a magnetic field, it can also sense the presence of an obstacle on the path, and take special measures. A close loop control system is used in the van. The van must sense a line and maneuvers accordingly to stay on course while correcting the wrong moves using feedback mechanism thus forming a simple but yet effective closed loop System. The van is designed to follow very tight curves as the data from the sensors are continuous in nature. This van is simple but effective having straightforward design to perform line following task.*

# المستخلص

العربة تابعة المسار هي عباره عن جهاز يستطيع أن يتحسس وجود خطٍ على الأرض وتقفى أثره، يمكن أن يكون الخط أسود اللون على سطح أبيض أو العكس، ويمكن استخدام مجال مغناطيسي كدليل بدلاً عن الخط المرئي.
يمكن للعربة أيضاً كشف وجود عائق على الطريق واتخاذ الإجراءات مناسبة.
العربة تعمل بنظام تحكّم ذو حلقة مغلقة، حيث على العربة تحسس مكان الخط والمناورة للبقاء على المسار الصحيح، فيما تقوم بتصحيح الحركات الخاطئة عن طريق منظومة تغذية عكسية لتشكل نظام حلقة مغلقة.
صممت العربة لتقوم باتباع الخط حتى في المنحنيات الضيقة طالما تتدفق الإشارات من الحساسات.
تصميم العربة بسيط ومباشر لكنه كافٍ وفعال للقيام بالمهمة.

## CONTENTS

# List of tables

# List of Figures

# CHAPTER ONE

# INTRODUCTION

## 1.1 Overview

Line following van is a device that can detect and follow line which is drawn on the floor while sensing if there is any presence of an obstacle in pre-defined distance. The path can be visible like a black line on a white surface (or vice-versa) or it can be invisible like a magnetic field. Robot should read for identifying his position in some time, after that, the control system (AVR) will create for the line follower some orders to respect components in purposes of making correct movements of the line follower. Sensing a line and maneuvering the line follower to stay on course, while constantly correcting wrong moves using feedback mechanism forms a simple yet effective closed loop system.

Although a Line Follower's purpose in life is to follow a line, other issues have to be discussed for example, if the track that it is following is on a table, it should also have some form of detecting the edge of the table so it won't fall Off. If the robot is moving fast, it has to have fast reflexes so that if the line turns it will be able to compensate its direction so it won't leave the track. Line Follower needs light to move, but sometimes if an external light source is strong enough it might confuse the robot. If the robot is doing its preferred activity outdoors, a whole set of issues have to deal with, such as sunlight, dust, uneven terrain and other issues. [6]

The Line Follower in this project may not be able to handle all the conditions listed above, but as long as it follows the line and detects the presence of obstacles; it will be satisfied.

## 1.2 Research problem

In large stores there is a continuous movement for the transport of goods and put them in places allocated and requires a great time and effort multiplier to work with high efficiency and this requires an increase in the working hands and thus increase the spending of money and this is trying to reduce it.

## 1.3 Research objective

The main aim of the project is to implement a line follower which is a simple robotics application, the follower designed for repetitive tasks which use the same path a lot and that helps to reduce the need for a human operator

## 1.4 Research methodology

Microcontroller is used to make decisions according to signals that delivered or collected through sensors (IR sensors and an ultrasonic sensor), decisions like go forward, rotate or stop are being made as a response to the form of signals on the microcontroller bins.

## 1.5 Research structure

- Chassis and body.
- Sensors and signal processing circuits.
- Microcontroller.
- Motor drivers
- Actuators (Motors and wheels).

# CHAPTER TWO

## COMPONENTS OF THE LINE FOLLOWER

First of all, the circuit built on a breadboard so as known, in electronics some experimentation is required to prototype (trial) specific circuits. A prototype circuit is needed before a PCB is designed for the final circuit. A breadboard is used to prototype the circuit. It has holes which components can be inserted and has electrical connections between the holes as per the figure 2.1 below shows that. Using a breadboard means no soldering and a circuit can be constructed quickly and modified easily before a final solution is decided upon.

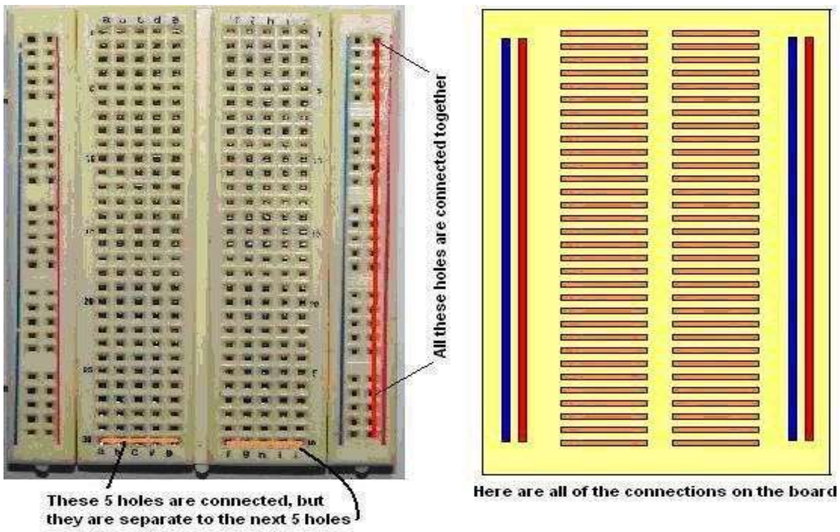


Figure **2.1** breadboard

## 2.1 Microcontroller

A microcontroller is a computer-on-a-chip or a single-chip computer. Micro suggests that the device is small, and controller tells that the device might be used to control objects, processes, or events. Another term to describe a microcontroller is embedded controller, because the microcontroller and its support circuits are often built into, or embedded in, the devices they control.

Microcontrollers are characterized by how many bits of data they process at once, with a higher number of bits generally indicating a faster or more powerful chip. Eight-bit chips are popular for simpler designs, but 4-bit, 16-bit, and 32-bit architectures are also available.

### 2.1.1 Architectures of Microcontrollers

Every microcontroller must have memory space to store a program (code) and data. While code provides instructions to the CPU, the data provides the information to be processed. The CPU uses buses (wire traces) to access the code. [1]

ROM (Read Only Memory) and RAM (Random Access Memory) memory space. There are two types of microcontroller architectures Harvard architecture and von Neumann architecture.

### 2.1.2 Atmel AVR

The AVR is a modified Harvard architecture 8-bit RISC (Reduced Instruction Set Computer) single chip microcontroller which was developed by Atmel in 1996. The AVR was one of the first microcontroller families to use on-chip flash memory for program storage, as opposed to One-Time Programmable ROM, EPROM(Erasable Programmable Read Only Memory) , or EEPROM( Electrically Erasable Programmable Read Only Memory) used by other microcontrollers at the time. The Atmel AVR and the Microchip PIC are very similar (Harvard architecture) lines of microcontrollers.

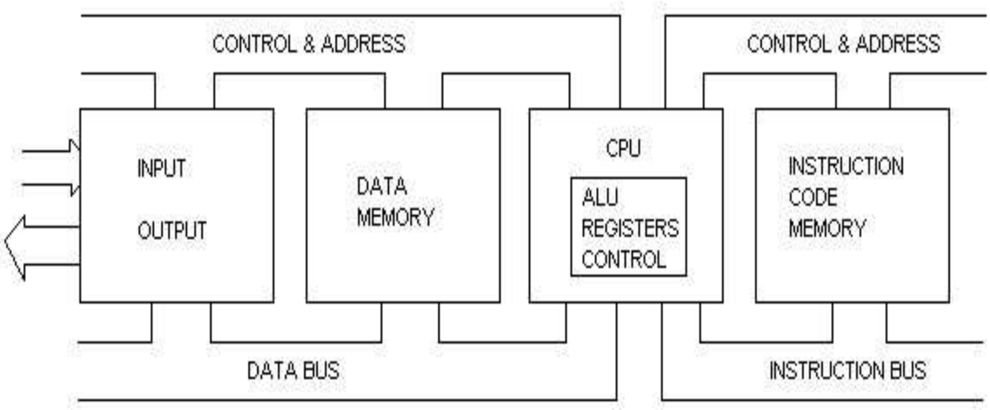


Figure **2.2** Harvard architecture

## 2.1.3 Mega AVR

These are powerful microcontrollers with more than 120 instructions and lots of different peripheral capabilities, which can be used in different designs. Table 2.1 shows the types of mega AVR. Some of their characteristics are as follows:

- Program memory: 4K to 256K bytes.
- Package: 28 to 100 pins.
- Extensive peripheral set. Extended instruction set: They have rich instruction set.

Table 2.1 types of mega AVR

| Part Num. | Code ROM | Data RAM | Data EEPROM | I/O pin | ADC | Timers |
|---|---|---|---|---|---|---|
| AT mega 8 | 8K | 1K | 0.5K | 23 | 8 | 3 |
| AT mega 16 | 16K | 1K | 0.5K | 32 | 8 | 3 |
| AT mega 32 | 32K | 2K | 1K | 32 | 8 | 3 |
| AT mega 64 | 64K | 4K | 2K | 54 | 8 | 4 |
| AT mega 1280 | 128K | 8K | 4K | 86 | 1 | 6 |

## 2.1.4 ATmega16

The Atmel AT mega 16 is equipped with 32 general purpose 8-bit registers that are tightly coupled to the processor's arithmetic logic unit within the CPU. Also, the processor is designed following the Harvard Architecture format. That is, it is equipped with separate, dedicated memories and buses for program and data information. The register-based Harvard Architectures coupled with the RISC-based instruction set allows for fast and efficient program execution and allows the processor to complete an assembly language instruction every clock cycle. Atmel indicates the AT mega 16 can execute 16 million instructions per second when operating at a clock speed of 16 MHz. [2]

## 2.1.5 Atmega16 architecture overview

The AT mega 16 has external connections for power supplies (Vcc, GND, Avcc, and AREF), an external time base (XTAL1 and XTAL2) input pins to drive its clock, processor reset (active low RESET), and four 8-bit ports (PA0PA7, PB0-PB7, PC0-PC7, PD0-PD7), which are used to interact with the external world, these ports may be used as general purpose digital input/output (I/O) ports or they may be used for the alternate function. The ports are interconnected with the ATmega16's CPU and internal subsystems via an internal bus. The AT mega 16 also contains a timer subsystem, an analog to digital converter (ADC), an interrupt subsystem, memory components, and a communication subsystem.

### 2.1.5.1 Analog to digital converter

The AT mega16 is equipped with an eight-channel ADC. The ADC converts an analog signal from the outside world into a binary representation suitable for use by the microcontroller. The AT mega16 ADC has 10-bit resolution. This means that an analog voltage between 0 and 5V will be encoded into one of 1024 binary representations between $(000)_{16}$ and $(3FF)_{16}$. This provides the AT mega16 with a voltage resolution of approximately 4.88 Mv. [1]

## 2.1.5.2 Pin Description

Figure 2.3 below shows the pins of Atmega16 microcontroller:

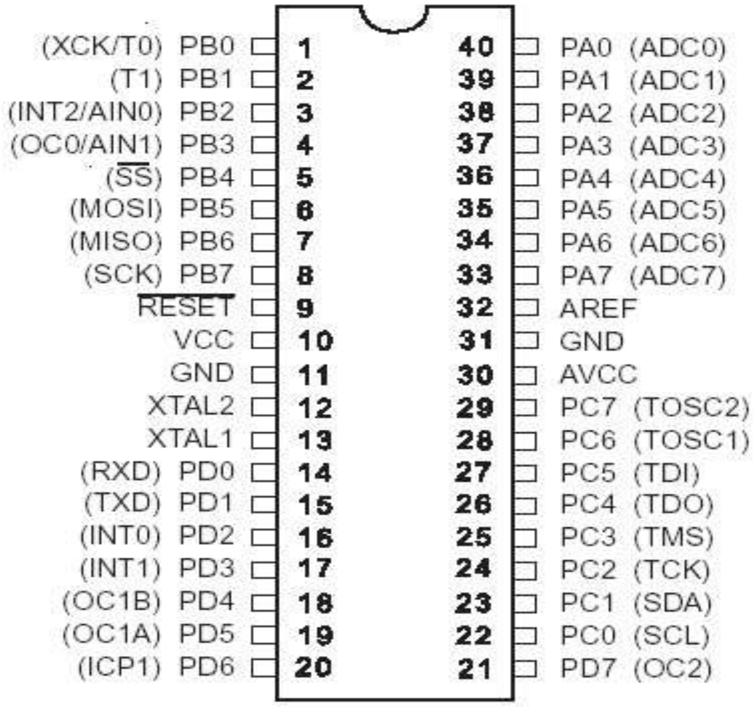


Figure **2.3** pin description of Atmega16

**VCC**

Digital supply voltage.

**GND**

Ground.

**Port A (PA7.PA0)**

Port A serves as the analog inputs to the digital (ADC). Port A also serves as an 8-bit bi-directional I/O port, if the (ADC) is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability.

When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

**Port B (PB7..PB0)**

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port B also serves the functions of various special features of the Atmega16 as listed on table below.

**Port C (PC7..PC0)**

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs. Port C also serves the functions of the JTAG interface and other special features of the Atmega16 as listed on table below.

**Port D (PD7.PD0)**

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port D also serves the functions of various special features of the Atmega16 as listed on table below.

**RESET**

Reset Input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset.

**XTAL1**

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

**XTAL2**

Output from the inverting Oscillator amplifier.

**AVCC**

AVCC is the supply voltage pin for Port A and the ADC. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter.

**AREF**

AREF is the analog reference pin for the ADC.

## 2.2 Motor Driver (L293D)

One of the first realizations in robotics is that making something move isn't an easy task. Simply can't take a “brain” circuit and connect it to a motor and expect anything to happen. The motor will simply not work at the puny output signal from the brains, and stay stationary. What the brain needs is an enforcer (Muscle). Something to convince the motor to do things the way the brains want it to be done. There are many ways to strengthen (buffer) a signal so it's strong enough to drive a large load like a motor. Transistors H-bridges circuit, buffer chips, and dedicated motor driving chips are all suitable choices.

A motor driver is a device or group of devices that serves to govern in some predetermined manner the performance of an electric motor. A motor driver include automatic means for starting and stopping the motor, selecting forward or reverse rotation, selecting and regulating the speed, regulating or limiting the torque, and protecting against overloads and faults.

### 2.2.1 (L293D) Description

This device is a monolithic integrated high voltage, high current four channel driver. Drive inductive loads (such as relays solenoids, DC and stepping motors and switching power transistors). To simplify use as two bridges each pair of channels is equipped with an enable input. A separate supply input is provided for the logic, allowing operation at a lower voltage and internal clamp diodes are included. This device is suitable for use in switching applications at frequencies up to 5 kHz. The L293D is assembled in a 16 lead plastic package which has 4 center pins connected together and used for heat sinking.

L293D is dual H-Bridge motor drive, so with one IC can interface two DC motors which can be controlled in both clockwise and counter clockwise direction. L293D has output current of 600Ma and peak output current of 1.2A per channel. Moreover, for protection of circuit from back EMF output diodes are included within the IC. The output supply has a wide range from 4.5V to 36V, which has made L293D a best choice for DC motor driver.

### 2.2.2 Working theory of H-Bridge

The name “H-Bridge” is derived from the actual shape of the switching circuit which controls the motion of the motor. It is also known as (Full Bridge). Basically there are four switching elements in the H-Bridge as shown in figure 2.4 below.

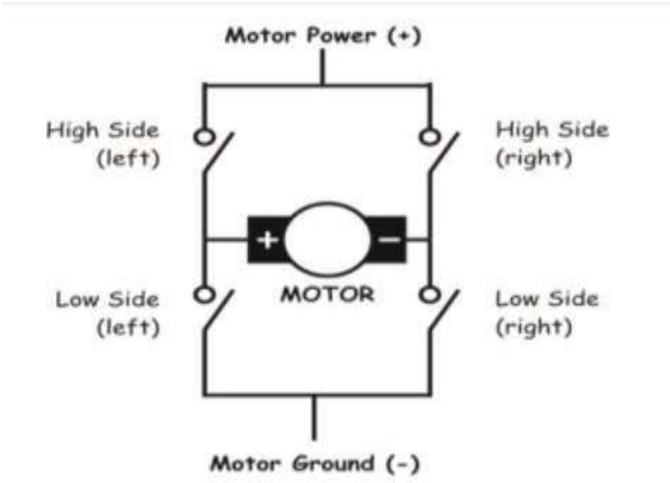


Figure **2.4** H-Bridge circuit

As seen in the figure 2.4 above there are four switching elements named as “High side left”, “High side right”, “Low side right”, “Low side left”. When these switches are turned on in pairs motor changes its direction accordingly. Like, if we switch on High side left and Low side right then motor rotate in forward direction, as current flows from (+) power supply through the motor coil goes to ground via switch low side right. Similarly, when you switch on low side left and high side right, the current flows in opposite direction and motor rotates in backward direction. This is the basic working of H-Bridge. So we have seen that using simple switching elements we can make our own H Bridge, or other option we have is using an IC based H bridge driver.

### 2.2.3 Pin configurations

Figure 2.5 and table 2.2 below shows the pin configuration of L239D.

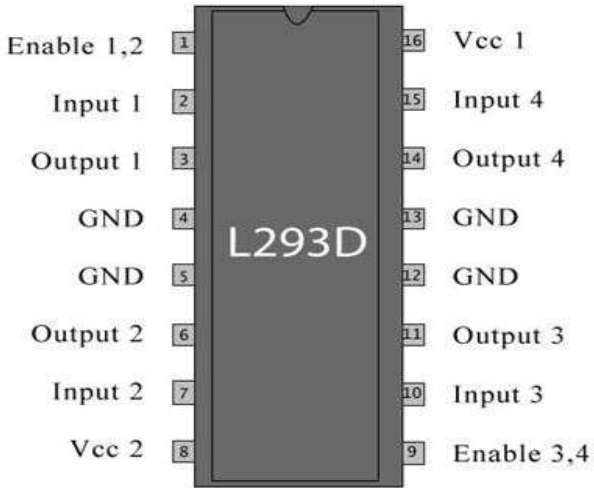


Figure 2.5 pin configurations of (L293D)

Table 2.2 pin configuration of L293D

| **Pin Number** | **Function** | **Name** |
|---|---|---|
| **1** | Enable pin for Motor 1; active high | Enable 1,2 |
| **2** | Input 1 for Motor 1 | Input 1 |
| **3** | Output 1 for Motor 1 | Output 1 |
| **4** | Ground (0V) | Ground |
| **5** | Ground (0V) | Ground |
| **6** | Output 2 for Motor 1 | Output 2 |
| **7** | Input 2 for Motor 1 | Input 2 |
| **8** | Supply voltage for Motors; 9-12V (up t 36V) | Vcc 2 |
| **9** | Enable pin for Motor 2; active high | Enable 3,4 |
| **10** | Input 1 for Motor 1 | Input 3 |
| **11** | Output 1 for Motor 1 | Output 3 |
| **12** | Ground (0V) | Ground |
| **13** | Ground (0V) | Ground |
| **14** | Output 2 for Motor 1 | Output 4 |
| **15** | Input 2 for Motor 1 | Input 4 |
| **16** | Supply voltage; 5V (up to 36V) | VCC 1 |

## 2.3 Sensors

A sensor is often defined as a device that receives and responds to a signal or stimulus. This definition is broad. In fact, it is so broad that it covers almost everything from a human eye to a trigger in a pistol.  Sensors that are used in artificial systems must speak the same language as the devices with which they are interfaced. This language is electrical in its nature. Thus, it should be possible to connect a sensor to an electronic system through electrical wires. Hence, we use a somewhat narrower definition of sensors, which may be phrased as:  A sensor is a device that receives a stimulus and responds with an electrical signal. The stimulus is the quantity, property, or condition that is sensed and converted into electrical signal.

The purpose of a sensor is to respond to some kind of an input physical property (stimulus) and to convert it into an electrical signal which is compatible with electronic circuits. The sensors considered as a translator of a generally nonelectrical value into an electrical value. The sensor‟s output signal may be in the form of voltage, current, or charge. These may be further described in terms of amplitude, frequency, phase, or digital code. This set of characteristics is called the output signal format. Therefore, a sensor has input properties (of any kind) and electrical output properties.

## 2.3.1 Ultrasonic sensor (HC-sr04)

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module [6] as shown in figure (2.6) below


Figure **2.6** – ultrasonic operation

## 2.3.1.1 Ultrasonic HC-sr04 pins

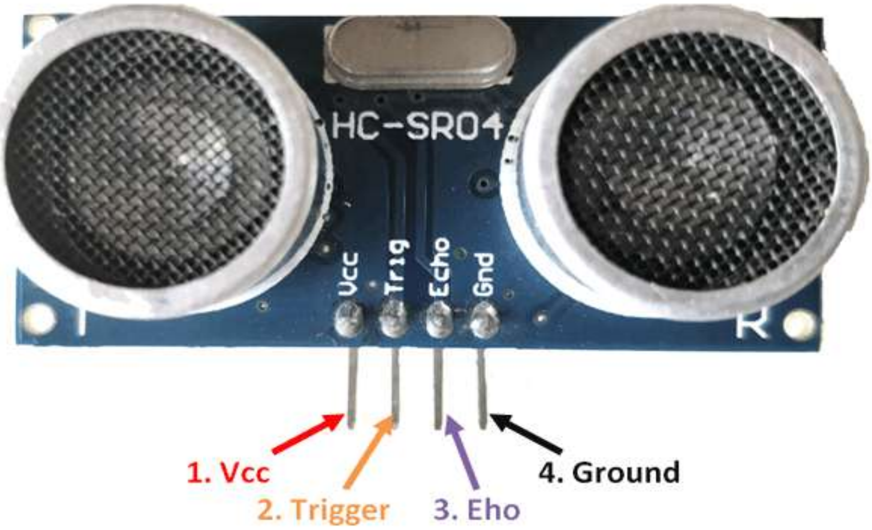


Figure **2.7** – ultrasonic pins

Table 2.3 pin configuration of HC-sr04

| Pin Number | Function | Name |
|---|---|---|
| 1 | The Vcc pin powers the sensor, typic with +5V | Vcc1 |
| 2 | Trigger pin is an Input pin. This pin ha be kept high for 10us to initialize measurement by sending US wave | Trigger |
| 3 | Echo pin is an Output pin. This pin g high for a period of time which will equal to the time taken for the US wa return back to the sensor. | Echo |
| 4 | Ground (0V) | Ground |

## 2.3.2 IR LEDs

A light-emitting diode (LED) is a semiconductor diode that emits incoherent narrow-spectrum light when electrically biased in the forward direction of the p-n junction. The basic function of the emitter is to convert electricity into light. It works on the principle of recombination of the electron-hole pair. As in the conduction band of a diode, electrons are the majority carrier and in the valence band, holes are majority carrier. So when an electron from a conduction band recombines with a hole of valance band, some amount of energy is released and this energy is in the form of light. The amount of energy released is depends upon the forbidden energy gap. The color of the emitted light depends on the composition and condition of the semiconducting material used, and can be infrared (IR LED), visible, or near ultraviolet. LEDs are often used as small indicator lights on electronic devices and increasingly in higher power applications such as flashlights and area lighting. The IR Led has two legs, the leg which is longer is positive and other leg is negative as seen in figure 2.8 below.

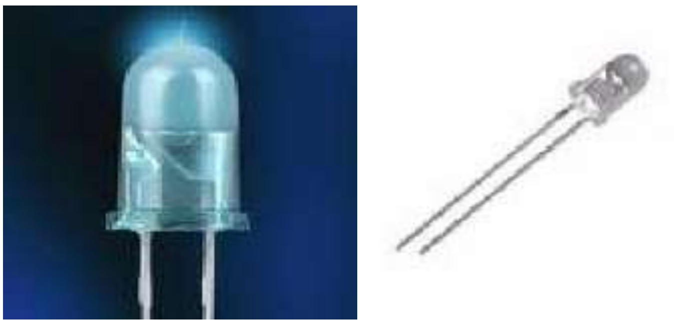

Figure **2.8** IR LED (Tx)

### 2.3.2.1 Photodiodes

The photodiode is a p-n junction diode which is connected in reverse bias direction. The basic function of the detector is to convert light into electricity. As its name implies that it works effectively only when the certain number of photon or certain amount of light falls on it. When there is no fall of light on the photodiode it has an infinite resistance and act as an open switch but as the light starts falling on the photodiode, the resistance becomes low and when the full intensity of light fall on the photodiode then its resistance becomes zero and it starts act like a closed switch. The IR receiver shows in figure 2.9 below:

Figure **2.9** photodiode (Rx)

## 2.3.2.2 Circuit diagram of IR sensors

The circuit diagram is shown below in Figure 2.10 and only one set of emitter/detector sensor is depicted. The remaining two are constructed in the same manner.

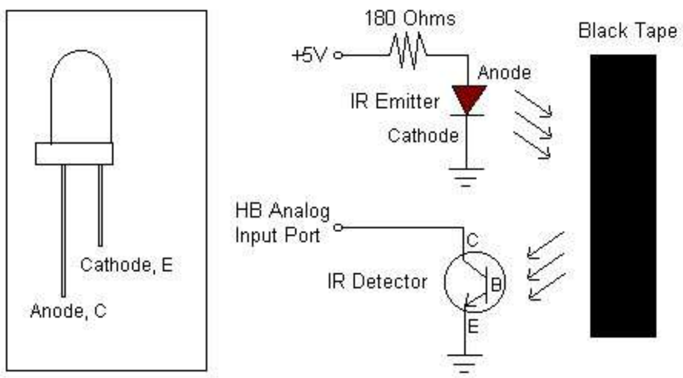


Figure **2.10** circuit diagram of IR sensor

A line sensor in its simplest form is a sensor capable of detecting a contrast between adjacent surfaces, such as difference in color. The theory of operation is simple when light shines on a white surface; most of the incoming light is reflected away from the surface. In contrast, most of the incoming light is absorbed if the surface is black. Therefore, by shining light on a surface and having a sensor to detect the amount of light that is reflected, a contrast between black and white surfaces can be detected. Figure 2.11 shows an illustration of the basics just covered.

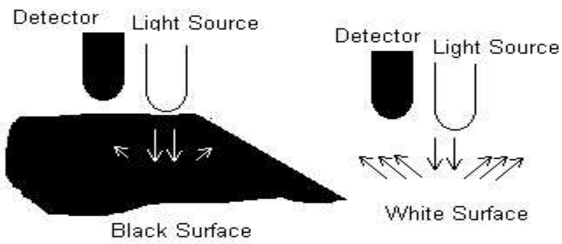


Figure **2.11** the basic IR function

## 2.4 Buzzer

Buzzer is an integrated structure of electronic transducers, DC power supply, widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers and other electronic products for sound devices. Active buzzer 5V Rated power can be directly connected to a continuous sound, this section dedicated sensor expansion module and the board in combination, can complete a simple circuit design, to "plug and play."

## 2.4.1 Buzzer pins

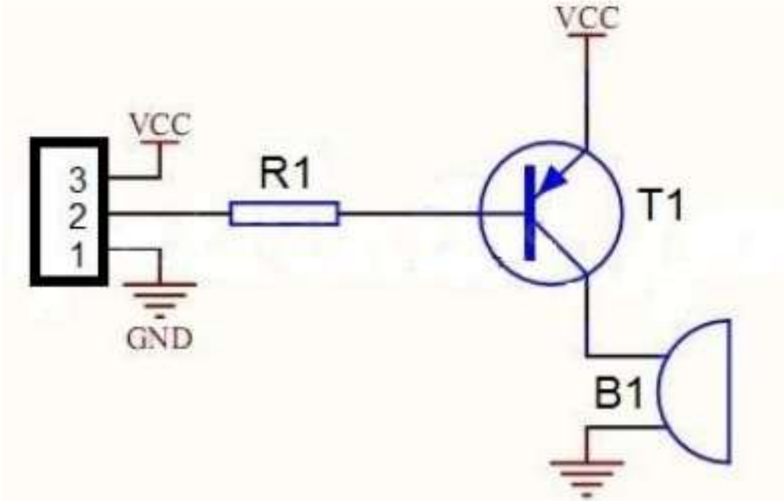


Figure **2.12** – buzzer circuit

Table 2.4 pin configuration of buzzer

| **Pin Number** | **description** | **Name** |
|---|---|---|
| 1 | Ground (0V) | Ground |
| 2 | The input to the buzzer | Input |
| 3 | The Vcc pin powers the buzzer, typic with +5V | vcc |

## 2.5 Interfacing the Components with Microcontroller

The components can be connected with microcontroller directly or through a certain device.

### 2.5.1 IR sensor interface with the microcontroller

The IR sensors array consist of: two LDEs (Tx) and two photo diodes (Rx), the positive legs of the LEDs are connected to the source through ($2K_{\omega}$) resistance to reduce the input voltage, and the negative legs are connected to the ground. By this way the LEDs emitting infrared rays. The photo diodes also having two legs, the positive legs are grounded and the negative legs are connected with the microcontroller to pins (A0, A1) [4].

### 2.5.2 Motor interface with the microcontroller

The microcontroller sends a signal to L239D that acts as a switch. If the signal received by the L239D is high, it will rotate the motor or else it won't do so. Note that microcontroller only sends a signal to a switch which gives the voltage required by the motor to rotate. Here using L293D which can be used to control two motors [7]. Pin connections for L239D shown below:

- En1 & En2 are given logic 1 from microcontroller or give 5V from outside and are used to activate/deactivate one „half" of the H-bridge.
- V is the voltage supply to the motor(s).
- Vcc is the logic 1 or 5V.
- GND

There are two DC motors front motor for front wheels and rear motor for rear wheels. The motors receive a signal from the microcontroller through the motor driver. The dc motors rotation depends on the polarity or the current direction, that"s why the motor driver is used. It has four available inputs and outputs, only two outputs are needed for each motor. The inputs of L239D are connected to the microcontroller ports (D1, D2, D3), D1 and D2 are connected to (PIN1, PIN2) respectively to drive the front motor bidirectional, D3 connected to(PIN4) to drive the rear motor. When the microcontroller sends a signal to the motor driver, this signal actually enables one of the outputs of the motor driver, so L293D allows the motor to rotate in the desired direction. The rear motor only goes forward tracing the line, so it doesn‟t need to run bidirectional and that is the reason of not connecting PIN5 with the microcontroller. The L293D-DCmotor-Interface can be represented in figure 2.13 below bellow:

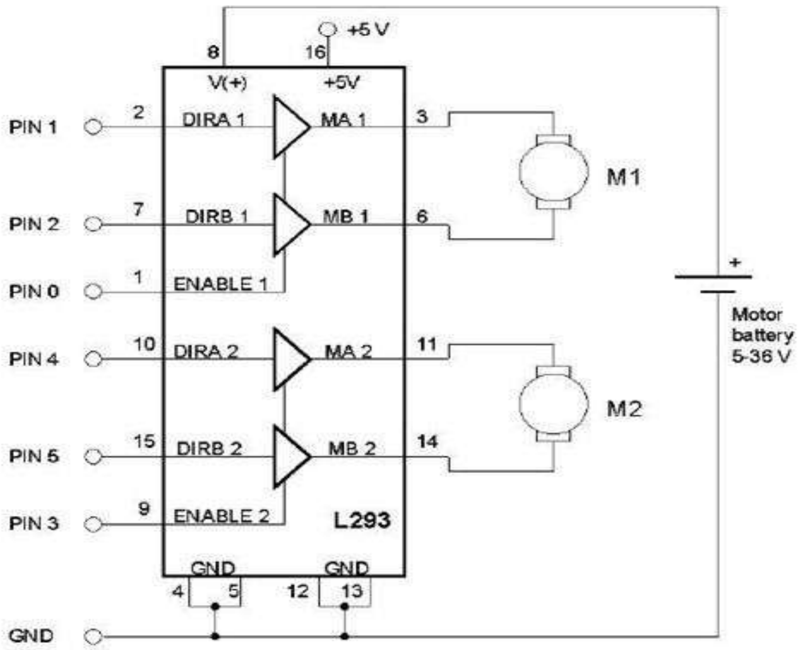


Figure **2.13** L293D and DC-motors interfacing

Table 2.5 Truth table of L293D

| IN1 | IN2 | Description |
|---|---|---|
| 1 | 1 | Motor stops or breaks |
| 0 | 1 | Motor runs anti-clock wise |
| 1 | 0 | Motor runs clockwise |
| 1 | 1 | Motor stops or breaks |

For the above truth table, the enable (EN) has to be set (1).

# Chapter Three

# DESIGN AND PROGRAMMING OF THE LINE FOLLOWER

## 3.1 Circuit Connection

According to the main function of the line follower, which is to detect the path that specified by the user, two pair of IR transmitters and receivers have been connected to the terminals, transmitters connected to the power supply and the receivers connected to microcontroller(portC.0, portC.1) in order to send signal to the microcontroller, with this signal the car can define which case that it goes through, then it gives the suitable output for the motors through the L293D that takes the car back to the black line, when an obstacle is detected by the ultrasonic sensor in less than a predefined length; the microcontroller stops the motors (the operation will be discussed thoroughly later) . The model can be represented in figure 3.1 below. As it is clear, the microcontroller is the main object that the whole components of the circuit are connected to. The devices can be divided into two groups, first; the inputs of the Atmega16 microcontroller which are; the IRs transmitter and receivers, only the receivers are connected directly to the pins of the microcontroller, and the two other IRs transmitters are supplied from the power supply (+ 5v) and an ultrasonic sensor connected to PD3 and PD6 (Input Capture pin). The other group is the output devices; which are; left motor and right motor, connected to motor driver (L293D) and buzzer to give an alarm when an obstacle detected by the ultrasonic sensor.

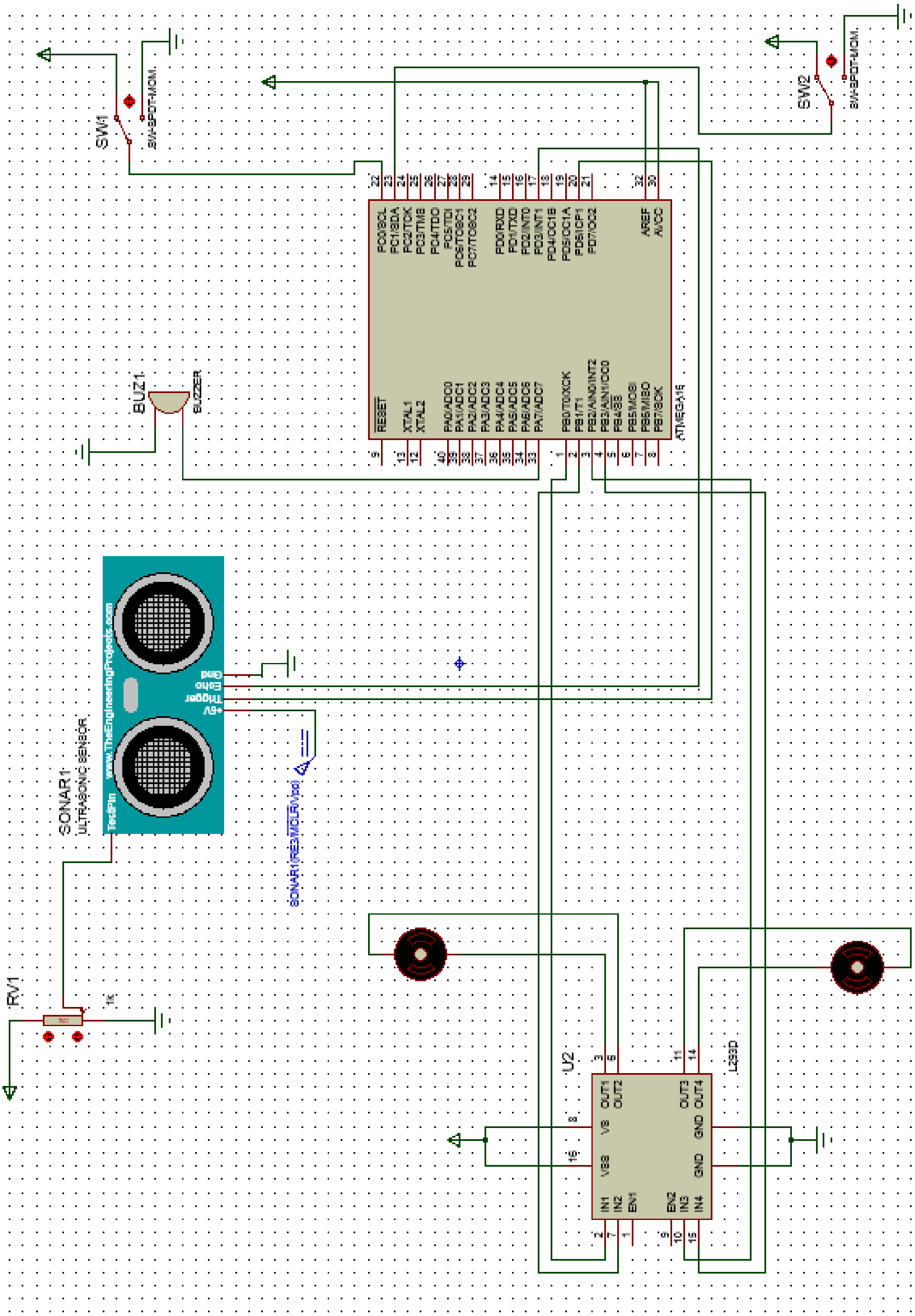


figure **3.1 -** line follower circuit connection

Table 3.1 Ports connections

| Port | I/O | connected to | Runs when | Period |
|---|---|---|---|---|
| | | | | |
| **PortB.0** | O/P | L293(IN 1) | Power is on | - |
| **PortB.2** | O/P | L293(IN 3) | Power is on | - |
| **PortB.1** | O/P | L293(IN 2) | The car got out of path | - |
| **PortB.3** | O/P | L293(IN 4) | The car got out of path | - |
| **PortA.7** | O/P | Buzzer | When an obstacle is detected | 2 sec |
| **PortD.6** | O/P | Ultrasonic (HC-sr04) sensor (trigger) | Sending pulse | 10 u-sec |
| **PortD.3** | I/P | Ultrasonic (HC-SR04) sensor (Echo) | Receive pulse | - |
| **PortC.1** | I/P | Right IR detector | to detect the path | - |
| **PortC.0** | I/P | Left IR detector | to detect the path | - |

## 3.2 The Principle of Operation of the Line Follower

The principle of operation of the follower is to send signals from the right and left IR receivers which are located in the front of the car model to the microcontroller to take an action according to the input. The car should be put where the path (black line) is exactly in between the right and the left IR receivers.

When the microcontroller pins receive a high signal from both receivers in Port C the car go forward, and when the black line is detected by the right IR sensor, the microcontroller will send reversed polarity to motor driver pin that is connected to the right motor so the right motor rotates backward, while sending low signals to motor driver pins that connected to left motor to stop it from rotation. The same scenario happens if the line is detected by the left IR sensor, but this time the left motor will rotate backwards and the right motor stops.

### 3.2.1 The operation of ultrasonic sensor

The microcontroller transmits 10 us trigger pulse to ultrasonic (HC-SR04) trigger pin, the sensor automatically sends eight 40 KHz sound wave and the microcontroller waits for the rising output at Echo pin, when the rising edge occurs at Echo pin; microcontroller timer starts and waits for the falling edge at Echo pin, as soon as the falling edge is captured at Echo pin; microcontroller reads the count of the timer, that count is used to calculate the distance to an object.

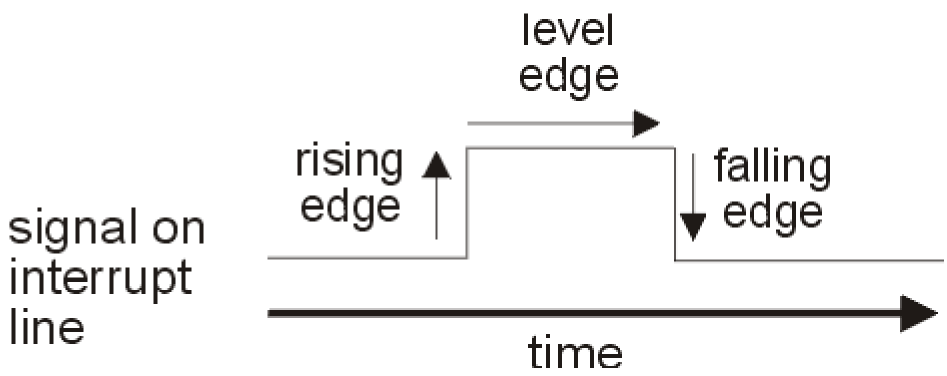


Figure **3.2**

If the distance were less than a predetermined value; microcontroller will stop both motors and enable the buzzer.

We are using timer0 in atmega16 to count the time needed for the sound wave to return, so basically we are counting how much time does the echo pin remain high before falling to 0 (when the sound waves return) so we have to discuss the operation of timer0 at atmega16 and interrupt system.

#### 3.2.1.1 Timer0 operation and interrupt

Timer 0 in an 8-bit timer. It basically means it can count from 0 to 255, the operation of timer 0 is straight forward, the TCNT0 register as shown in figure (3.3) holds the timer count and it is incremented on every timer tick, if the timer is turned on; it ticks from 0 to 255 and overflows, if it does so; a timer overflow flag (TOV) is set. which is going to be used by interrupt.

| **TCNT0** | | | | | | | |
|---|---|---|---|---|---|---|---|
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

Figure **3.3**

The configuration of the timer can be set using TCCR0 register shown below in figure (3.4), with it you can select the frequency of the clock source with CS02, CS01 and CS00 bits as it shown in table (3.2). Also you can select the mode of the timer.

| **TCCR0** | | | | | | | |
|---|---|---|---|---|---|---|---|
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| FOC0 | WGM00 | COM01 | COM00 | WGM01 | CS02 | CS01 | CS00 |

Figure **3.4**

| D2 | D1 | D0 | Clock Source |
|---|---|---|---|
| **CS02** | **CS01** | **CS00** | **Freq** |
| 0 | 0 | 0 | No Clock (Stopped) |
| 0 | 0 | 1 | Clk |
| 0 | 1 | 0 | Clk/8 |
| 0 | 1 | 1 | Clk/64 |
| 1 | 0 | 0 | Clk/256 |
| 1 | 0 | 1 | Clk/1024 |
| 1 | 1 | 0 | Clk/T0-Falling edge |
| 1 | 1 | 1 | Clk/T0-Rising Edge |

Table **3.2**

The timer / counter interrupt flag register (TIFR) in figure (3.5) holds the two basic flags we need the TOV and OVF. Other bits are corresponded to the timer interrupts.

| TIFR | | | | | | | |
|---|---|---|---|---|---|---|---|
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| OCF2 | TOV2 | ICF1 | OCF1A | OCF1B | TOV1 | OCF0 | TOV0 |

Figure **3.5**

## 3.3 Programming of the Microcontroller

The program has been written by C programming language, which is one of the simplest and easiest programming languages that deal with Atmel-AVR microcontrollers. [3]

the instructions of the program show the logical statements that the car should physically execute according to the cases that the car might face in its way going forward.

### 3.3.1 The code of the Line Follower

```
#define F_CPU 8000000UL
```

Sets the Microprocessor to work with 8 MHZ.

```
#include <avr/io.h>
#include<avr/wdt.h>
#include<avr/interrupt.h>
#include<util/delay.h>
```

These are libraries needed by AVR compiler to compile the code.

```
volatile long avg = 0;
volatile unsigned char up = 0;
volatile uint32_t running = 0;
volatile uint32_t timercounter =0;
unsigned int c;
```

declaring variable that is going to be used later in code. And giving them initial values.

```
ISR(TIMER0_OVF_vect)
{
 if (up) {
 timercounter++;
  }
  }
```

ISR stands for Interrupt Service Routine which is going to be invoked when timer0 of Atmega 16 goes overflow and overflow flag is set.

```
DDRD = 0xf0;
PORTD = 0x00;
DDRB = 0xff;
PORTB = 0x00;
DDRA = 0xff;
PORTA = 0x00;
    DDRC=0x00;
    PORTC=0x00;
```

(DDRD = 0xf0) configures the high order bits in register D as an output and the low order bit as an input.

(PORTD = 0x00) right wall zeros to the pins of register D.

(DDRB = 0xff) configures register B as an output.

(the same technique were used for PORTB DDRA PORTA DDRC PORTC).

```
MCUCR |= (0 << ISC11) | (1 << ISC10)
```

Set (ISC11) bit to zero and (ISC10) bit to 1 in MCU control register so it generates interrupt at INT1 if it goes high or low and that can be seen in figure (3.6) below:

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| **SM2** | **SE** | **SM1** | **SM0** | **ISC11** | **ISC10** | **ISC01** | **ISC00** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

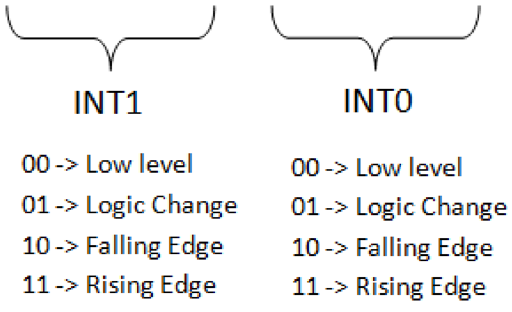


Figure **3.6**

Now interrupt was enabled at INT1 pin by:

```
GICR |= (1 << INT1);
```

GICR stands for - General Interrupt Control Register - which controls interrupt operations. As seen below:

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| INT1 | INT0 | INT2 | – | – | – | IVSEL | IVCE |

Figure **3.7**

```
TIMSK |= (1 << TOIE0);
```

TOIE0 bit in timer mask (TIMSK) is set to high, which enables interrupt to happen when the timer overflow as it shown in figure (3.4).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | OCIE2 | TOIE2 | TICIE1 | OCIE1A | OCIE1B | TOIE1 | OCIE0 | TOIE0 | TIMSK |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

Figure **3.8**

```
sei()
```

Enables all global interrupts.

```
void send_trigger()
{
PORTD = 0x00;
_delay_us(5);
 PORTD = 0xf0;
running = 1;
_delay_us(10);
PORTD = 0x00;
}
```

send signal for trigger the ultrasonic for 10uS.

```
SIGNAL(INT1_vect){
if(running){
if (up == 0) {
up = 1;
timercounter = 0;
TCCR0 |= (0 << CS02)|(0 << CS01)|(1 << CS00);
TCNT0 = 0;
} else {
up = 0;
avg = (timercounter*256+TCNT0)/58;
light_flashing();
running = 0;
```

high voltage rise was assumed to come before low drop, so when voltage rise at INT1 the timer starts and when and when voltage fall

the timer stops and gives count.

`TCCR0 |= (0 << CS02)|(0 << CS01)|(1 << CS00);` this set no pre-scaling for frequency in timer counter control register as it shown in figure (3.5)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | FOC0 | WGM00 | COM01 | COM00 | WGM01 | CS02 | CS01 | CS00 | TCCR0 |
| Read/Write | W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

Figure **3.9**

`TCNT0 = 0;` starts counter.

`avg = (timercounter*256+TCNT0)/58;`

$$\frac{\text{SoundVelocity} * \text{TIMER}}{2}$$

$$\frac{34300 * \text{TIMER}}{2}$$

$$17150 \ * \ \text{TIMER}$$

17150 * TIMER* 10^-6

= TIMER / 58

Timecounter = number of overflow flags

Timer0 has 8 bits so it counts from 0 to 255 means 256 times.

TCNT0 is a register where timer send its momentarily value when stops.

```
void light_flashing()
{
PORTA = 0x00;
if ((avg>0) && (avg <= 15))
{
PORTB = 0b00000000;
PORTA = 0b10000000;
}
else {
c=(PINC&0x03);
if(c==1)
{
PORTB=0b00000100;
_delay_ms(500);
}
else if(c==2)
{
PORTB=0b00000001;
_delay_ms(500);
}
else if(c==3)
{
```

PORTB=0B00001010;

}

Else

{

PORTB=0b00000000

The code above guides the car to follow the line and maneuver around curves.

if ((avg>0) && (avg <= 15)) {

PORTB = 0b00000000; stops the car if there was an obstacle in less than **15 CM**.

PORTA = 0b10000000; turn on the buzzer.

c=(PINC&0x03); to check input pins status.

if(c==1){

PORTB=0b00000100; turn right.

_delay_ms(500); delay the turn for half a second to enable the car to get back to the right direction without jerking.

else if(c==2){

PORTB=0b00000001; turn left.

delay_ms(500); delay the turn for half a second to enable the car to get back to the right direction without jerking

else if(c==3) {

PORTB=0B00001010; move forward.

Else{ both sensors read black, means the car have reached the end of the line in T shape.

PORTB=0b00000000 stops the car.

# Chapter 4

# Line following van applications

The line follower van is used in many applications. It might be used in a warehouse where the follower follows 'tracks' to and from the shelves they stock and retrieve from. The line follower can also be used as a guide to guide the visitors from the entrance to the main office. There are cases where smarter versions of line followers are used to deliver mail within an office building and deliver medications in a hospital.

## 4.1 Line following van in medical field

In medical field there is no doubt that speed and accuracy are essential elements, and luckily a line follower van can be implemented in various ways to provide these elements, and also lower treatment payments for patients by using it

### 4.1.1 Smart patient's beds

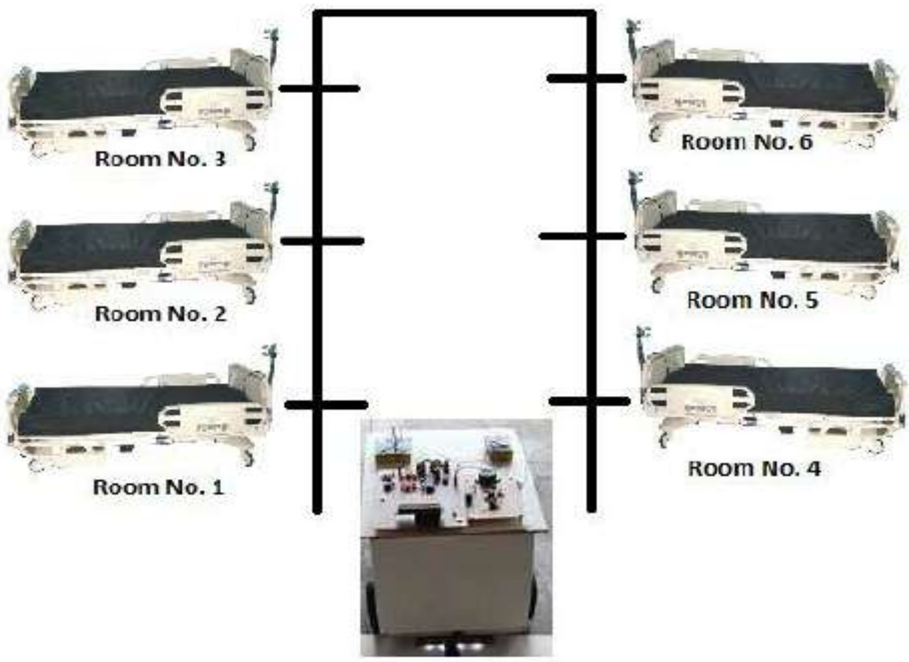


Figure **4.1**

Line follower can be implemented in patient's beds and moves them through hospital rooms to a desired place. A job like that can be stressful to doctors and nurses but using this technology can save time, muscles and money.

### 4.1.2 Smart nurse cart

In hospitals, nurses are assisting the doctors to provide sufficient health treatment to the patients. Each nurse has her own responsibility for the delivery duties in the hospital. In general, nurses bring several medical documents, reports and instruments to patients using cart.

The conventional cart is required some external force to push and pull the cart to the patient bed and bring it back many times in a day. This can be a tiresome for nurse because she needs to service several patients in the hospital. In order to assist the nurses on resolving this problem, a line following cart can be developed to assist nurses while bringing with all the documents, instruments or reports. And can be programed to bring medical equipment's all alone to operation room when operation time is due without any monitoring. Thereby, the work load for nurse can be reduced and she can work in other productive activities.

## 4.2 In industry

Line following van based industrial manufacturing can play a vital role in the field of industry. Using this van in the government organization and Manufacturer Company, the cost for the manpower can be reduced. This line following van can be used as carrying the load to deliver the goods from one place to another smoothly without any damage.

Recently most of famous companies like AMAZON have started using it.

Figure **4.2**

## 4.3 General applications

Line follower van can be used as a guide wither it was in an airport or as a tour host in museums and can also be used as a waiter in restaurants, also as a home floor cleaner and so many applications that anyone can be creative with it.

Figure **4.3** – waiter line follower

# Chapter 5

## 5.1 Conclusion

- A follower scheme has been done which mean that this van can run forward, turn right and turn left.
- (l293D) represents an interface unit with a microcontroller to control the movements of the two motors.
- C programing language is an easy language compared with the other high level language.

## 5.2 Recommendations

- A GSM chip can be integrated in the van to enable it to make calls or send an SMS when it's needed.
- The code can be improved to enable the van to avoid obstacles rather than just a stopping.
- More than two sensors can be implemented for more detecting accuracy.
- Embedded magnets path can be used instead of drown line on the surface, when its supplied with magnetic detectors, the follower follows the magnets path.
- A timer can be used to stop the van is specific places on the line for specific periods or permanently.

# References

[1]- Dhananjay V.Gadre, “*Programming and customizing the AVR microcontroller*” McGraw-Hill 2001.

[2]- Steven F.Barrett, Daniel J.Pack, “*Atmel AVR microcontroller primer Programming and Interfacing*”, Morgan & Claypool 2008.

[3]-Muhammad Ali Mazidi, Sarmad Naimi, Sepeher Naimi, “*The AVR microcontroller and embedded system using assembly and C*”, Prentice hall.

[4]-Jan Axelson, “*The microcontroller idea book*”, Lakeview Research 1997.

[5]- Priyank Patill, , K.J.Somaiya,Line following robot, Mumbai India.

[6]- Jacob Fraden, “*Hand book of modern sensors*” (third edition), San Diego California

[7]-Krishna Nand Gupta and others, “*Motor driver L293D*”.

# APPENDIX

## The full control program:

```
#define F_CPU 8000000UL
#include <avr/io.h>
#include<avr/wdt.h>
#include<avr/interrupt.h>
#include<util/delay.h>

volatile long avg = 0;
volatile unsigned char up = 0;
volatile uint32_t running = 0;
volatile uint32_t timercounter =0;
unsigned int c;
ISR(TIMER0_OVF_vect)
{
if (up) {
timercounter++;
}
}

void light_flashing()
{
PORTA = 0x00;
if ((avg>0) && (avg <= 15))
{
PORTB = 0b00000000;
PORTA = 0b10000000;
}
```

```
else {
c=(PINC&0x03);
if(c==1)
{
PORTB=0b00000100;
_delay_ms(500);
}
else if(c==2)
{
PORTB=0b00000001;
delay_ms(500);
}
else if(c==3)
{
PORTB=0B00001010;
}
Else
{
PORTB=0b00000000;
}
}
}
SIGNAL(INT1_vect){
if(running){
if (up == 0) {
up = 1;
timercounter = 0;
```

```
TCCR0 |= (0 << CS02)|(0 << CS01)|(1 << CS00);
TCNT0 = 0;
} else {
up = 0;
avg = (timercounter*256+TCNT0)/58;
light_flashing();
running = 0;
}
}
}

void send_trigger()
{
   PORTD = 0x00;
   _delay_us(5);
   PORTD = 0xf0;
   running = 1;
   _delay_us(10);
   PORTD = 0x00;
}

int main()
{
     DDRD = 0xf0;
     PORTD = 0x00;
     DDRB = 0xff;
     PORTB = 0x00;
     DDRA = 0xff;
```

```
    PORTA = 0x00;
      DDRC=0x00;
      PORTC=0x00;
    MCUCR |= (0 << ISC11) | (1 << ISC10);
    GICR |= (1 << INT1);
    TIMSK |= (1 << TOIE0);
  sei();
  while(1)
  {
if(running == 0) {
_delay_ms(60);
send_trigger();
}
}
}
```

**The line following van final design:**