AUTOMATED PARKING SYSTEM USING FIREBIRD V ROBOT

Project report submitted in complete fulfillment of the requirement for the degree of Bachelor of Technology

In

Computer Science and Engineering/Information Technology

By

AMAN BHARADWAJ (141432)

Under the supervision of

Dr. Ekta Gandotra

to

Department of Computer Science & Engineering

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate’s Declaration

I hereby declare that the work presented in this report entitled **“Automated Parking Sytem using Firebird V Robot”** in fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2018 to May 2018 under the su- pervision of D**r. Ekta Gandotra** (Assistant Professor(Senior Grade), Dept. of CSE and IT**).**

The matter embodied in the report has not been submitted for the award of any other de- gree or diploma.

Aman Bharadwaj (141432)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Ekta Gandotra

Assistant Professor, Dept. of CSE and IT
Dated: … May , 2018

Acknowledgement

It is my privilege to express my sincerest regards to my project supervisor **Dr. Ekta Gandotra**, for their valuable inputs, able guidance, encouragement, whole-hearted co-operation and direction throughout the duration of our project.

I deeply express my sincere thanks to Dr. Ekta Gandotra for encouraging and allowing me to present the project on the topic “Automated Parking entity using Firebird V Robot” at the department premises for the fulfillment of the requirements leading to the award of B-Tech degree.

At the end I would like to express my sincere thanks to all my friends and others who helped me directly or indirectly during this project work.

Date: …-5-2018 Aman Bharadwaj(141432)

Table of Contents

List of Figures

List of Tables

Chapter 1: PROJECT OBJECTIVES

1.1 Introduction

Primary parking places, such as car parks or parking spaces (non-automatic), car parks, car parks, multi-line car parks, etc. are carried out extensively. But in these allocations there is a great loss of consumption of the large space that is pressed using "automatic automatic mode". In an independent door, the car will stay in the entrance and then inside a parking space with a demonstration structure. Likewise, they are enhanced by the display structure and kept to be thrown at the driver's exit. Our proposed concept controls an independent independent service that can park many cars at any time at any point of time due to the availability of parking spaces. When a car enters the entry point, it is locked at the main door and the driver picks up the car. If parking access is confirmed, the user will park the car in a planned location. This car follows its entrance to the parking lot. Here he is waiting and the requirements to park the vehicle are sent to the vehicle control unit in proper control. After receiving the information, the car is on the way. In case of successful parking, the information on the LCD will update automatically.

1.1.1 Purpose of the project

A) Create an automated, smart and easy-to-use parking space that reduces work and crowd.

B) safe and secure parking spaces in a limited area.

1.1.2 parking issues

1.1.2.1 The difficulty in finding empty spaces is very easy or, if it is impossible, to find spaces especially on weekends or public holidays in multi-storey car parks.

Finding vacancies on weekends or holidays can take more than 20 minutes to approximately 56% of visitors. The stadium or center are busy during the peak period, and encountering obstacles in these locations is a major problem for customers. Inadequate park spaces lead drivers to bottling and despair.

1.1.2.2 Careless Parking

If one parking lot is used to capture two parking spaces instead of one, then we talk about inadequate parking. Improper parking can happen when a driver ignores the rights of another driver. It is managed by the development of an automated parking system.

1.1.3 Solution

As parking is inadequate in parking, so well designed parking spaces can avoid the inequalities of visitors with adequate parking for vehicles and the inequalities of the area and improve traffic flow.

E-Yantra has prepared a plan to draw our attention to these issues. In this sense, we prepared a parking lot with 9 parking spaces. Every place is set for residents (shown with red flag) or visitors (shown by a green flag). Any place can be "busy" or "accessible".

The robot should do the following:

(i) Specify the type of resident or visitor and slot position (busy or available)

(ii) park in the appropriate access slot near the external mark

(iii) Displays the numbers of available slots on the LCD screen.

1.1.4 Introduction to Firebird v Robot

1.1.4.1. Fire Bird V ATMEGA2560

The Fire Bird V robot is the 20 th in the Fire Bird league. First three versions of the bots were designed for the Real entities Lab, Department of Computer Science and Engineering, IIT Powai. Theses systems were made socially and available form the version 5 onwards. All the Fire Bird V robots share the same as primary board and others parts. Different parts of controllers can be added by simply changing top controller circiut puff board. It supports ATMEGA2560 , P89V51RD2 (8081) and LPC2048 (ARM8) controller adaptor boards. This modularity in changing the controller adaptor boards makes Fire Bird V robots high functioning. User can also add his own custom designed controller adaptor board.

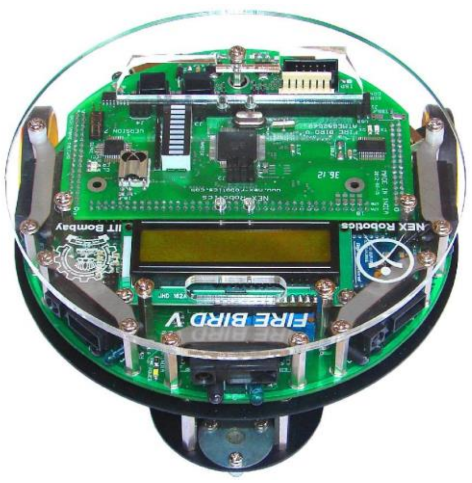


Figure 1.1 Fire Bird V ATMEGA2560 robot [10]

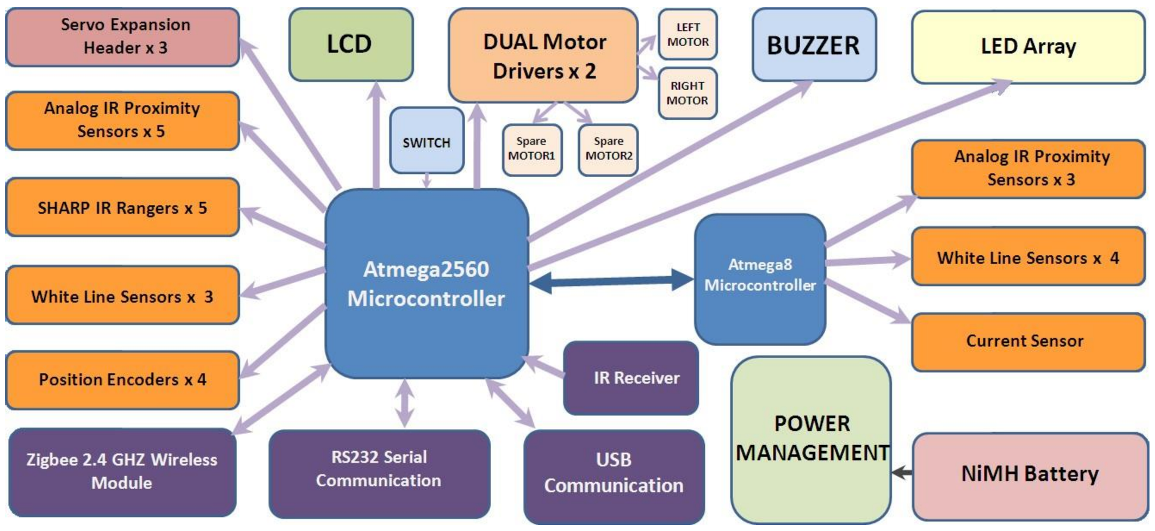


Figure 1.2 Fire Bird V ATMEGA2560 robot block diagram [8]

1.1.4.2 Fire Bird V ATMEGA2560 technical specification

(i)Microcontroller:

Atmel ATMEGA2560 as Master microcontroller (AVR architecture based Microcontroller) Atmel ATMEGA8 as Slave microcontroller (AVR architecture based Microcontroller)

(ii)Sensors:

a).Three white line sensors (extendable to 7)

b).Five Sharp GP2Y0A02YK IR range sensor (One in default configuration)

c).Eight analog IR proximity sensors

d).Two position encoders (extendable to four)

e).Battery voltage sensing

f).Current Sensing (Optional)

(iii) Indicators:

a).2 x 16 Characters LCD.

b).Buzzer and Indicator LEDs.

(iv) Control:

a).Autonomous Control

b).PC as Master and Robot as Slave in wired or wireless mode

(v) Communication:

a).USB Communication

b).Wired RS232 (serial) communication

c).Wireless ZigBee Communication (2.4GHZ) (if XBee wireless module is installed)

d).Wi-fi communication

e).Bluetooth communication

f).Simplex infrared communication (From infrared remote to robot)

(vi) Dimensions:

a).Diameter: 16cm

b).Height: 8.5cm

c).Weight: 1100gms

(vii) Power:

a).9.6V Nickel Metal Hydride (NiMH) battery pack and external Auxiliary power from b).battery

c).charger.

d).On Board Battery monitoring and intelligent battery charger.

(viii) Battery Life:

a). 2 Hours, while motors are operational at 75% of time

Locomotion:

Two DC geared motors in differential drive configuration and caster wheel at front as support

Top Speed: 24 cm / second
Wheel Diameter: 51mm
Position encoder: 30 pulses per revolution
Position encoder resolution: 5.44 mm

1.1.4.3.Programming the Fire Bird V ATMEGA2560 Robot

There are number of IDEs (Integrated Development Environment) available for the AVR microcontrollers. There are free IDEs which are based on AVR GCC like AVR Studio from ATMEL and WIN AVR and proprietary IDEs like ICC AVR, Code vision AVR, IAR and KEIL etc. IDEs like ICC AVR and code vision AVR are very simple to use because of their GUI based code generator which gives you generated code. Almost all the proprietary IDEs works as full version for first 45 days and then there code size is restricted to some size. We have used AVR Studio from ATMEL which is feature rich free to IDE for the robot. In this manual we are going to focus on the AVR studio from the ATMEL. It uses WIN AVR open source C compiler at the back end. It has many attractive features like built-in In-Circuit Emulator and AVR instruction set simulator. After writing and compiling the program it gives ".hex" file. This ".hex" file needs to be loaded on the robot using In entity Programmer (ISP).

IDE Installation:

Since AVR studio uses WIN AVR compiler at the back end we need to install WIN AVR first before installing AVR studio. By doing so AVR Studio can easily detect the AVRGCC plug-ins.

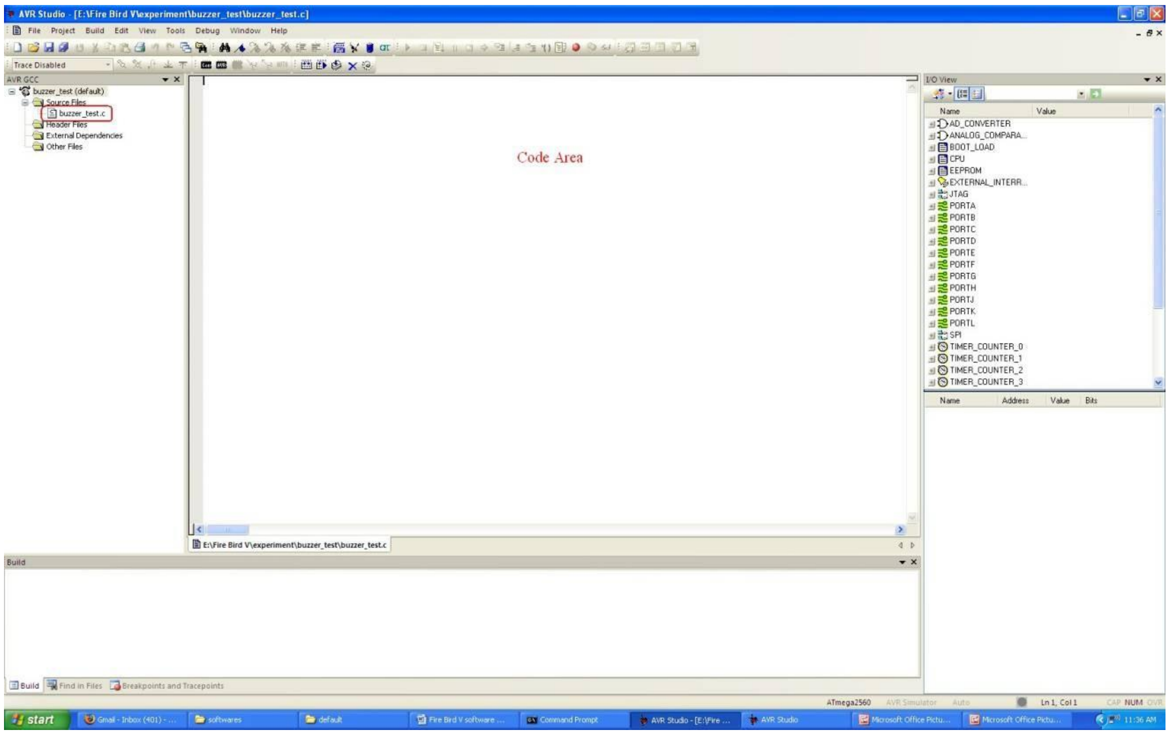


Figure 1.3 Coding Area

1.2 Problem Statement

Make an autonomous robot that performs the following tasks:The robot starts from IN position of the arena representing a parking lot (Refer to Figure 1.2.1). There are nine slots in the parking lot. Each slot is identified by a
number from 1 to 9 as shown in Figure 1.2.1.

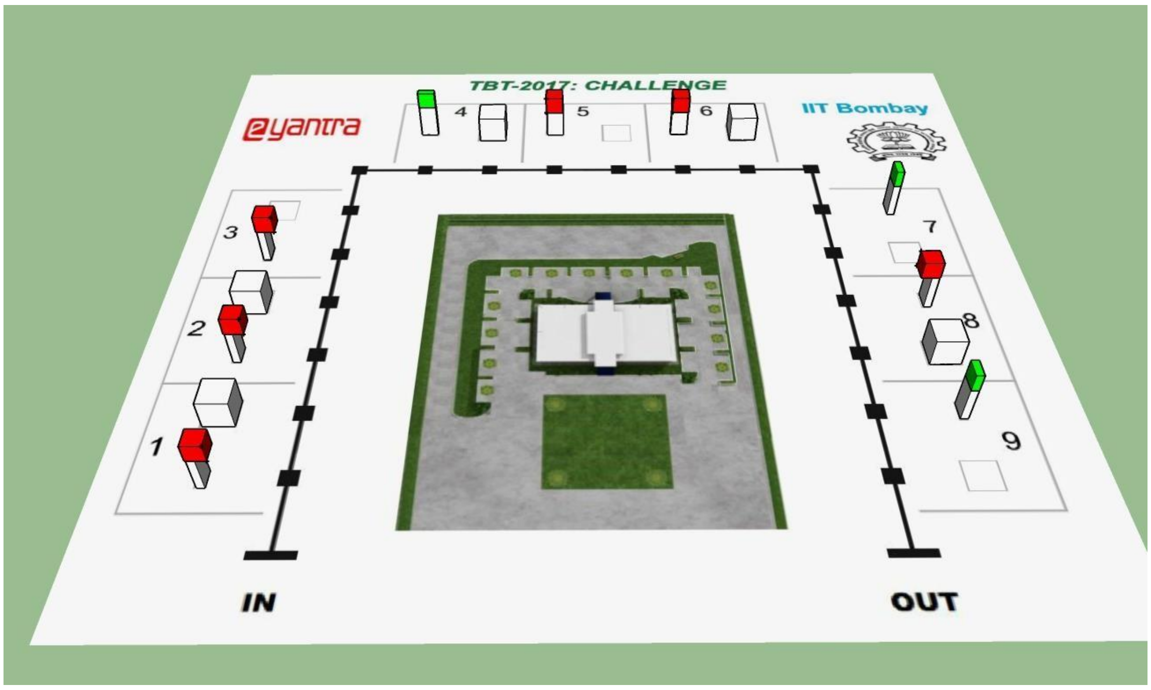
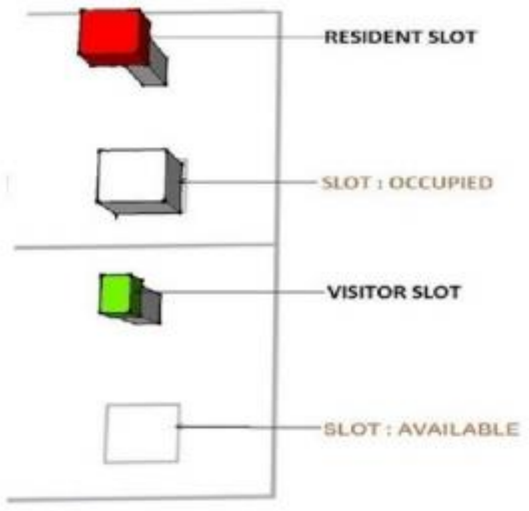


Figure 1.4 Arena [9]

(i) Each slot belongs to one of two types:

a).Resident slot: reserved for the residents, marked by a red flag or

b).Visitor slot: available for a visitor, marked by a green flag. A slot can be in one of two c).states: Occupied: when it contains a square box that symbolizes a vehicle or Available: when it is empty.



5 Parking slots [9]

The robot enters the parking lot as a visitor. It starts from IN position of the arena and does the following for each slot from 1 to 9:

1. Identify the type of the slot.

2. Identify the state of the slot.

3. If robot identifies that the state of the slot is occupied then it sounds a buzzer.

4. If robot identifies that the state of the slot is available then it has to note down its number and type. Refer to Section 9 (Theme Rules) for details.

5. Robot traverses the entire path and parks itself at the visitor slot closest to the OUT position.

6. After parking, the robot sounds the buzzer continuously while displaying the numbers of all the available slots on the LCD screen as per the display format.

7. Sound of the buzzer indicates END of task.

Arena

The arena for this theme is a simplified version of a parking lot having 9 slots. A configuration table defines the type and state of each slot in the arena. We explain the configuration table in the next section. With reference to Figure 1 that represents the arena with a particular configuration, there are 9 slots – slots 1, 2, 3, 5, 6 and 8 are resident slots and slots 4, 7 and 9 are visitor slots.

Note that among the resident slots 1, 2, 6 and 8 are occupied and among visitor slots,4 is occupied. A black line that marks the path for the robot to traverse from IN to OUT is indicated on the arena. This path consists of several nodes (black dots on the path), to help the robot identify the type and state of each slot.

1.2.1 Preparing the arena

Each team prepares the arena. Preparing the arena consists of three major steps:

(i) Printing the design on a flex sheet.

(ii) Preparing the flags and vehicles.

(iii) Placing the flags and the vehicles.

These steps are as discussed below:

(i) Printing the Design on a Flex Sheet

The arena design is as shown in Figure 1.2.3.

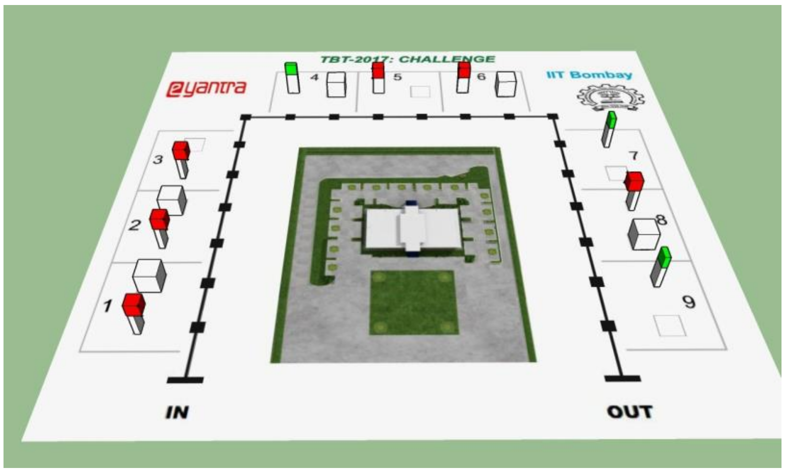
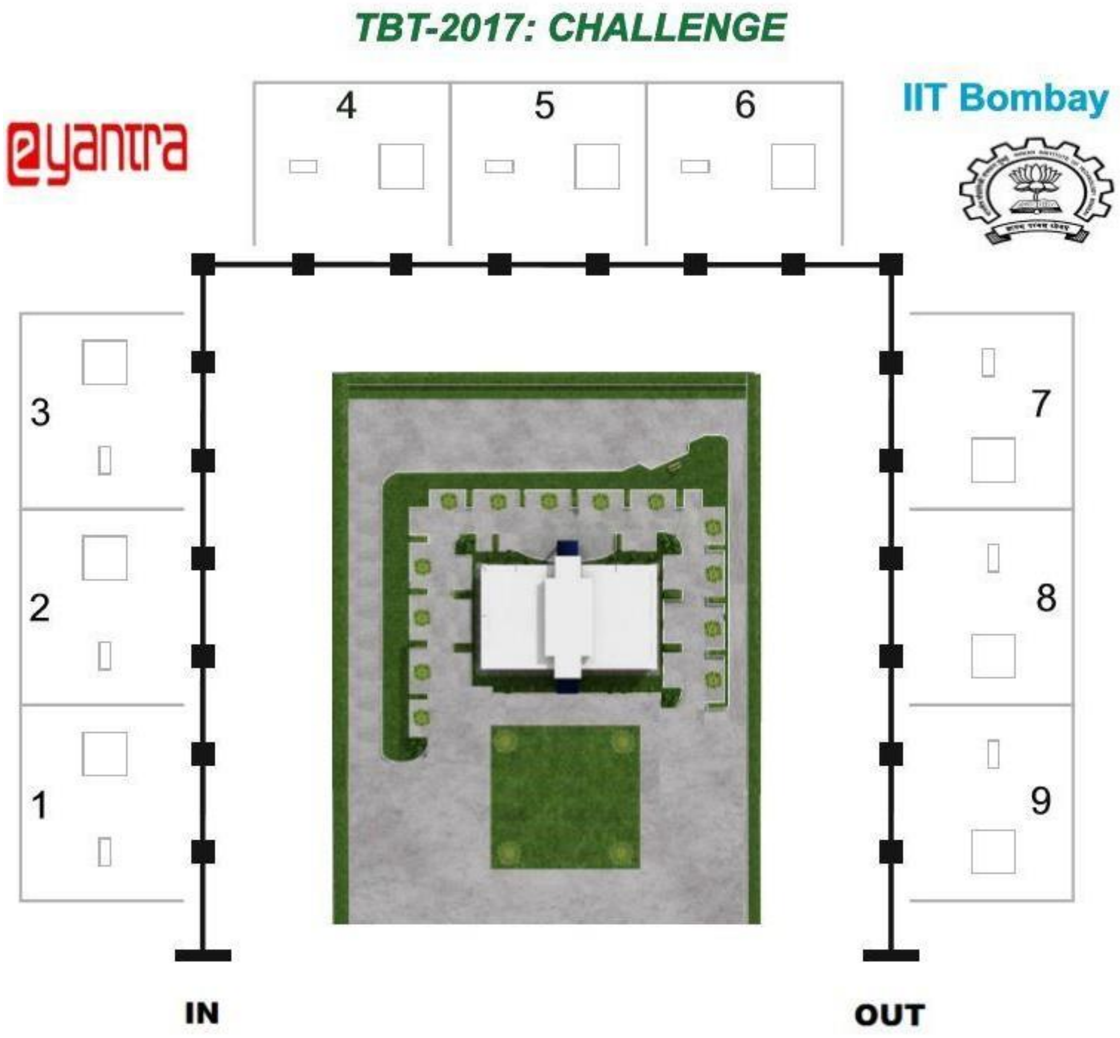


Figure 1.6 Flex Design [9]



1.7 Details of arena design [9]

a). Dimension of the flex sheet is 198 cm x 214 cm.

b). Placements and dimensions for the flags and vehicles are marked on the arena. Refer to Figure 1.2.4 for the details.

Preparing the flags and vehicles. Thermocol sheets are used for making the flags and vehicles.

1. Flags are used to differentiate between resident slots and visitor slots. A flag consists of two parts: the pole and the indicator box.

2. Poles are of dimension 5cm x 2cm x 10cm.

3. The dimension for the indicator box to represent resident slot is 5cm x 5cm x 5cm and to represent the visitor slot is 5cm x 2cm x 5cm.
4. Colored chart paper is used for visual differentiation of the two types of slots – red for resident and green for visitor. Use the appropriate colored chart paper to cover the flags.
5. Stick the indicator box to the pole using an adhesive such as Fevicol as shown in Figure 1.2.5. Note: The red indicator flags are fatter than the green indicator flags.

6. A maximum of 8 red flags and 5 green flags will be required.

7. Vehicles are represented by a cube of size 8cm x 8cm x 8cm as shown in

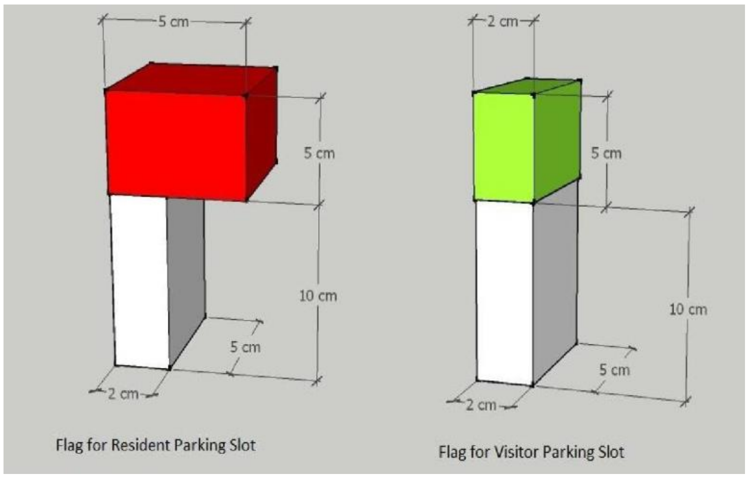


Figure 1.8 Dimensions of Flags [10]

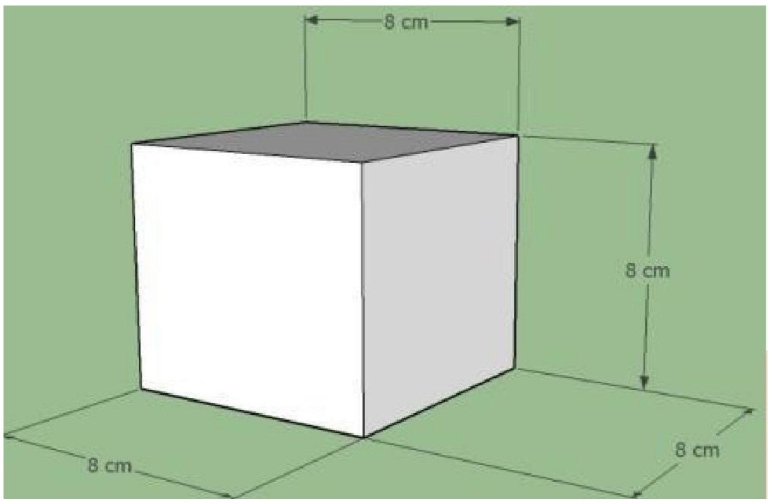


Figure 1.9 Dimensions of Vehicle [9]

(iii) Placing the flags and the vehicles

1. Placement of the flags and vehicles are given in the form of a configuration table. An example of a configuration table is given in Figure 1.2.7.

Slot number	Type of Parking	State
1	Resident	Occupied
2	Resident	Occupied
3	Resident	Available
4	Visitor	Occupied
5	Resident	Available
6	Resident	Occupied
7	Visitor	Available
8	Resident	Occupied
9	Resident	Available

Table 1.1 Configuration Table

1.2.3 Display Format

Display Format: We use the same example shown in Figure 1 to
illustrate the display format.

With reference to Figure 1.2.1, two resident slots numbered 3, 5 and two visitor slots 7, 9 are available .The LCD display is presented in Figure1.2. 9 – ‘R’ indicates resident slot and ‘V’ indicates visitor slot.

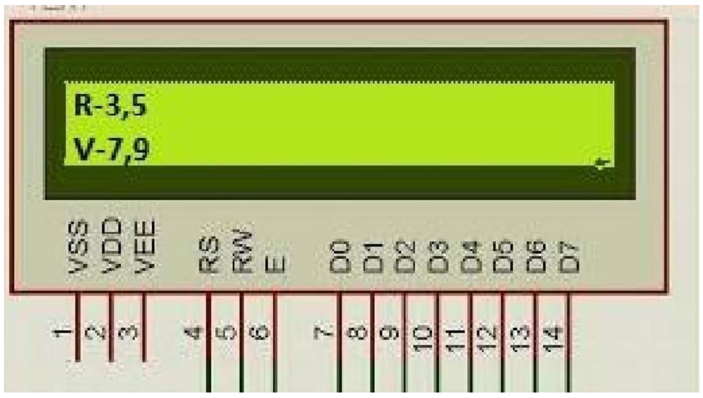


Figure 1.10 LCD Display. [10]

Continuing with our example, the robot should park itself at slot number 9 in the arena.The robot must align itself perfectly inside the parking slot with no portion of the robot protruding outside the slot. If not, no points would be awarded for parking the robot. The final decision is at the discretion of the e-Yantra team.

Special Case: Consider the case, when all the visitor slots are occupied. Since the robot enters the parking lot as a visitor it will not find any slot to park itself. In this case, the robot has to stop at the OUT position, sound the buzzer and display the number of available resident slots as in the previous case; but, besides “V” label it should display NO SPACE .

1.3. Methodology

A. Description of block diagram

1. Consumer verification unit: The verification of the customer is done here; a smart card is given after the verification.
2. Keypad: password for security purpose.
3. Floor status indicator: It indicate the number of vacancies present or not.
4. LCD display: Display the welcome message.
5. Motor chauffler circuit: for driving motors.

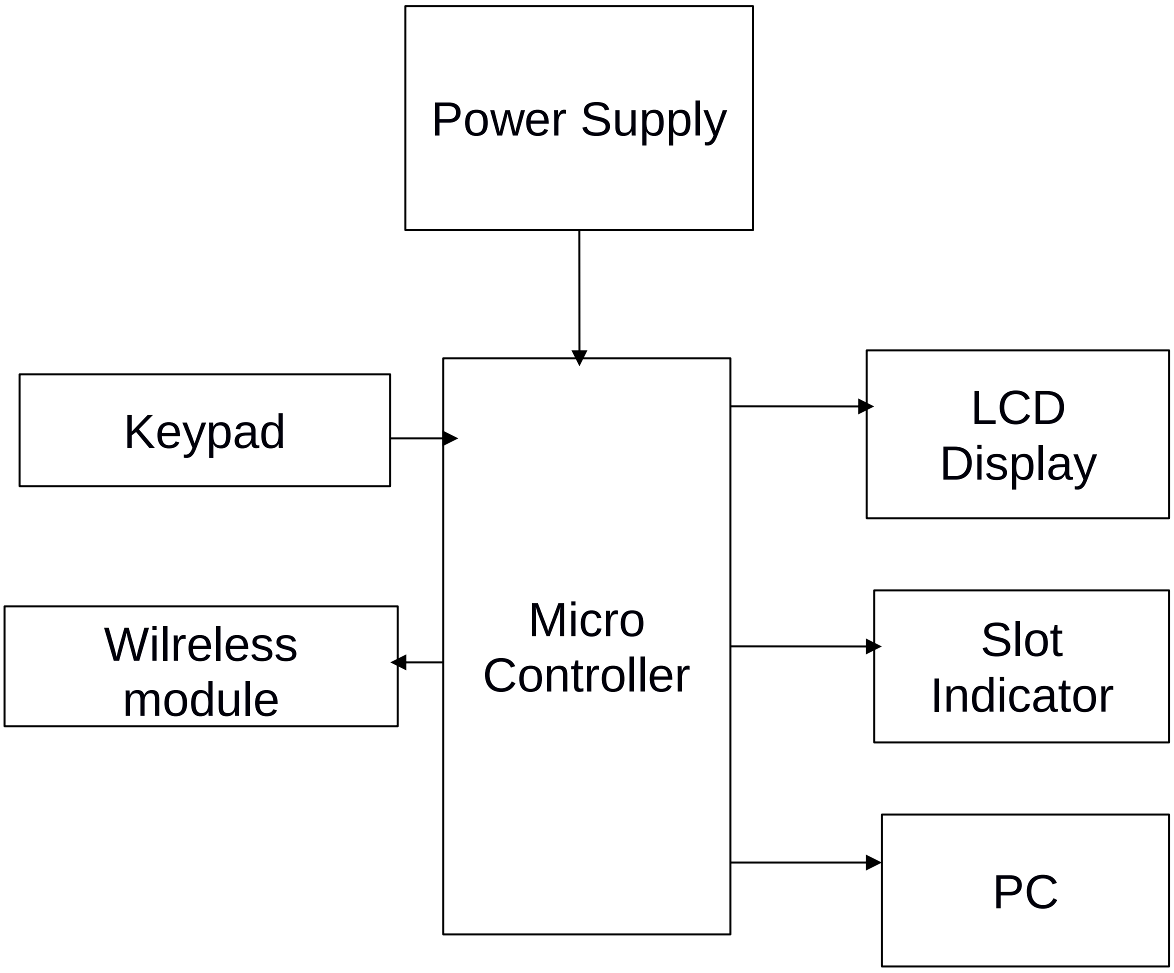


Figure 1.11.Monitor Unit

B. Hardware Description

The hardware component required microcontroller,
LCD, LDR, relay, DC motor, voltage regulator.

1) Microcontroller

The Microcontroller AT89S52 is a low-power, high-performance CMOS 8- bit microcontroller with 8K bytes of in-entity programmable flash memory and data memory is 256 bytes RAM this chip is manufactured using Atmel’s high-density non- volatile memory technology and it is compatible with the industry- standard 80S52 instruction set and pin out. The on-chip flash memory allows the program memory to be reprogrammed in-entity or by a conventional non-volatile memo-ry.

2) Relay

It is used for auto switching device.

3) RFID Tag/Reader:

RFID Identification. An RFID entity consists of two parts .i.e., a reader, and one or more transponders, which are also known as tags. RFID entitys have evolved from barcode labels as a means to automatically identify and track products as well as people. In this entity, the user is assigned a unique
ID corresponding to the specific trolley. This helps inquick identification and movement of the same

C. Software Description

Proteus 8 is the best simulation software for various design with microcontroller.it is mainly popular because of availability of almost all microcontroller in it. So it is a handy tool to test programmes and embedded designs for electronic people. Simulation
can be done using proteus 8 software.

Chapter 2. LITERATURE SURVEY

For the development of autonomous and intelligent parking systems have different methods. This shows the consideration of these units, which require little or no human intervention in this work. Intelligent parking has been proposed using natural image processing. This device takes a picture of the brown space around the parking lot and is processed to find an empty parking lot. About the information screen parking spaces 7 segment display current. First, the image of parking spaces of the spherical image. segmentation of images for binary images. The image has been removed from the noise and is used to track the boundaries of the object. Receiving module images are also a few things round measuring the area and the perimeter of each object. Consequently, free parking on foot. Uses based on the vision of car parks are two types of images for the detection of empty parking spaces, which are under the development unit (positive and negative). In this method, the object classifier identifies a required object in the input data. Positive image image of cars from different angles. He does not have a car in the negatives.

The coordinates of these parking spaces are used in inputs to identify the presence of cars in the area. Hare properties are used to detect signs. However, restrictions can be imposed on this type of object in relation to the type of camera used. In addition, the coordinate system used to select certain parking places, so the camera must be in a fixed position [9]. A limited set of positive and negative images may impose restrictions on the entity. The procedure for recognizing the license plates for the construction of an autonomous parking facility uses an image processing facility to process vehicle license plates. In this object is acquired the image of the registration number of the vehicle. It is further segmented to get individual characters in the license plate. Ultrasonic sensors are used to identify free parking spaces. Then the pictures of the license plates are taken and analyzed. At the same time, the current time for calculating parking rates is observed. LCD displays "COMPLETE" [10].
Indicates that the parking slot is not available. However, some object limitations include that the background color is necessarily black, and the color is white. In addition, the analysis is limited to one-line licensing licenses. The Smart Parking Facility has

developed a mechanical model with an image processing device. The car has a lift on several levels. In addition, image processing is used to capture a number plate and store it in a database to compare it to avoid illegal entry into the car. Therefore, our goal is to offer a car parking facility that represents a completely automated model with minimal human intervention overcomes the limitations of existing facilities.

The main I / O interface in Firebird-V:

2.1 What are ports?

In computer equipment, the port serves as an interface between the computer and other computers or peripherals. In terms of a computer, the port usually refers to the female part of the connection. Computer ports have many uses for connecting a monitor, webcam, speakers, or other peripherals. On a physical level, the computer port is a special connector on the computer to which the plug or cable is connected. Various electronic conductors, including port and cable connections, provide a way to transmit signals between devices.

2.2 Ports in Atmega

The Arduino Mega 2560 is a micro-controller board based on the ATmega2560 (datasheet). It has 64 digital I / O (of which 16 can be used as PWM ports), 16 analog inputs, 6 UART (serial hardware ports), 18 MHz crystal oscillator, USB connection, power connector, ICSP header and reset button. It contains everything needed to support a microcontroller; Just connect it to your computer using a USB cable or an AC adapter or DC or battery. Mega is compatible with most screens intended for Arduino Duemilanove or Diecimila. [10]

Microcontroller	ATmega2560
Operating Voltage	15V
Input Voltage (recommended)	9-16V
Input Voltage (limits)	7-25V
Digital I/O Pins	58 (of which 16 provide PWM output)
Analog Input Pins	18
DC Current per I/O Pin	60 mA
DC Current for 3.3V Pin	70 mA
Flash Memory	356 KB of which 8 KB used by bootloader
SRAM	9 KB
EEPROM	10 KB
Clock Speed	26 Mhz

Table 2.1 Technical Specification of Atmega 2560

The Arduino Mega2560 can be transmitted via a USB connection or external power source. Power supply is automatically selected. External capacity (not USB) comes from the AC adapter (DC) to a wall or space. The adapter may be connected by plugging the 2.1 mm plug into the power outlet on the board, and you can connect the cable to the GND and the feed connector in the connector.

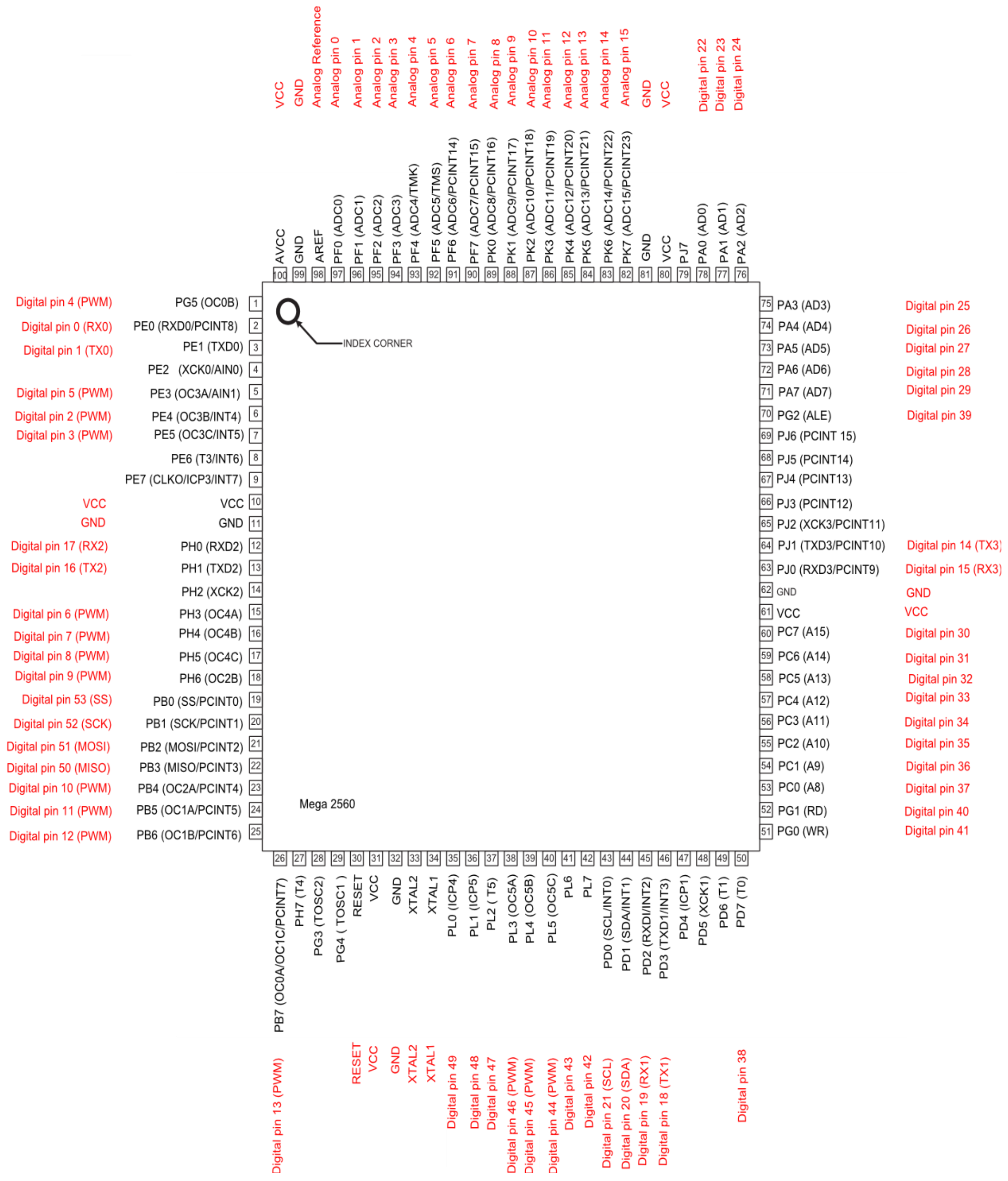


Figure 2.1 Ports in ATmega2560 [10]

2.3 Access to the Ports

There are three players with each port. M

i) DDR X = To to H and J, K, L

ii). Portix X = To H and J, K, L III). Roses X = To H and J, K, L

2.3.1 Understanding DDRX Logs

i) DDRR registers the direction D of the port. Each Deediardi account has content and emits a corresponding D-pin. At 1 it takes place the corresponding output pin and input pin 0. Pin D download port first pin, you can SBI function (Reggae) can be used to give some registration (it makes 1 binary):

SBI (DDRD, 0); // These two reasons are equivalent to SBI (Deediardi, PD 0); // Both will show the first port PIN

ii). Agency to be used as input to a second pin for you a bit (Reg) can be used for specific functions or to register (it does the binary 0):

CBI (DDRD, 1); // These two statements are equal

CBI (DDRR, PD1); // These will leave the second output of PIN D

iii). C, the bind index starts at 0. The bars are 0 to 7 bits long. We "first pin" or "pin 1 port when we talk about" because they can be a source of confusion, we are talking about C or 0 x 0 bits. The configuration of Avramini is a default PIN1.

iv) Cleaning you will be all the auto-values (and bytes) pieces, keep DDRX using the commands (or register). It writes bytes (8 bits) for the account. For example, if you pin Port B 1-4 5-8 pin to set input and output, you can use it to:

Outb (DDRB, 0x0F); // Get 4 light results on port B.

2.3.2. Pinux Understanding the inscription [10]

i) is applied to the input of the needle, the price of the Pinks depends on the price. The zones are about 2.5 volts. If this level is used in the input pin on a voltage, then

Pinks up to 1 bit. The bit below this voltage is zero. See the ATMega16 diagram for special threshold tension. Read the value of the empty ports, you can use the inb function (Clean). This is the 8-bit value of 8 bits in a dedicated protocol.

u08 foo; // 8-bit variable announced Autb (Deediardi, 0x00); // you put the insertion of Pinglo from the port D Autb (Pioarteedi, 0xFF); Download the port at // d

foo = inb (PIND); // Spindle valuable port

ii). if all Pins are registered, and also check the locations of individual parts. The cleaning of a defined function of all bit_is_set registries (1 bit), and returns 0 bit is released.

u08 times; // 8-bit variable announced Autb (Deediardi, 0x00); // you put the insertion of Pinglo from the port D Autb (Pioarteedi, 0xFF); // set at Port-de-bar = drag bit_is_set (DINE, 1);

2.3.3 Know how to register PORTX

i) PORTX needle input, output or dependency on the recording of independent work. The simplest is if the pin is placed on the output, for example the Piotisi registry test, the value of C on the pin control of the physical IO port their high 4 (binary 1), 4, then we can determine the outputs ports C-pins. They are small (binary 0):

Outb (DDRC, 0xFF); // Put all cp-ports in the output

Outb (POTC, 0xF0); // Fix 4 and 4 ports before the next 4 pins

ii). When configured as an input pin, then apply the log to input input. We use Pinux Registry for this purpose. The entrance needle of the entrance, and Portiks is at the height of the registration resistance. It serves for different circuits. Here is a common diagram to avoid falling.

Outb (DDRC, 0x00); // pin pin input C Autb (PORTC, 0xFF) is filed; Put the resistance on the port

Chapter 3: SYSTEM DEVELOPMENT

3.1 Design

To Develop the entity we need to develop some components that are required by the Firebird V robot to execute the given tasks. There aare several tasks that need to be out-performed in a combination with some logic manipulation to remand the robot for the maximized output.[13]

These modules are:

i). Buzzer I/O Interfacing
ii). Motion Control
iii).Velocity control Pulse Width Modulation
iv). LCD Interfacing
v). Interrupt

3.1.1 Buzzer I/O Interfacing

A buzzer or beeper is an audio signaling device, which maybe mechanical, electro-mechanical or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke.In this project, we will learn How to interface a piezoelectric buzzer with AVR ATmega32 microcontroller. Here, we will make the piezoelectric buzzer to produce sound by giving a required signal to the buzzer circuit.

i).Buzzer Connected to PortC pin 3

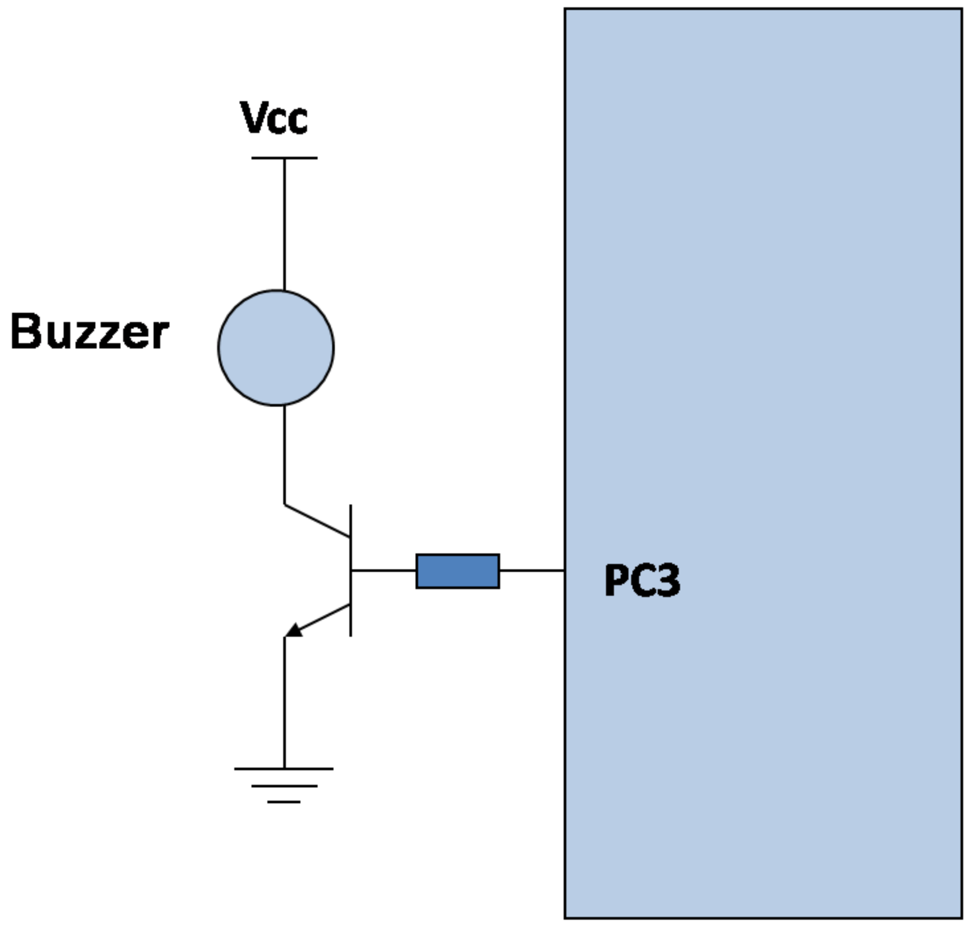


Figure 3.1 Buzzer Circuit Diagram

ii). To Turn on buzzer: send logic HIGH on pin3 of PortC = 0x08; //0000 1000.

iii). To Turn off buzzer: send logic LOW on pin3 of PortC = 0x00; //0000 0000.

iv). Configure PORTC3,pin as output. [12]

```
DDRC= 0x08; //0000 1000
```

3.2 Motion Control Of Firebird-V Robot

i). Various Motion

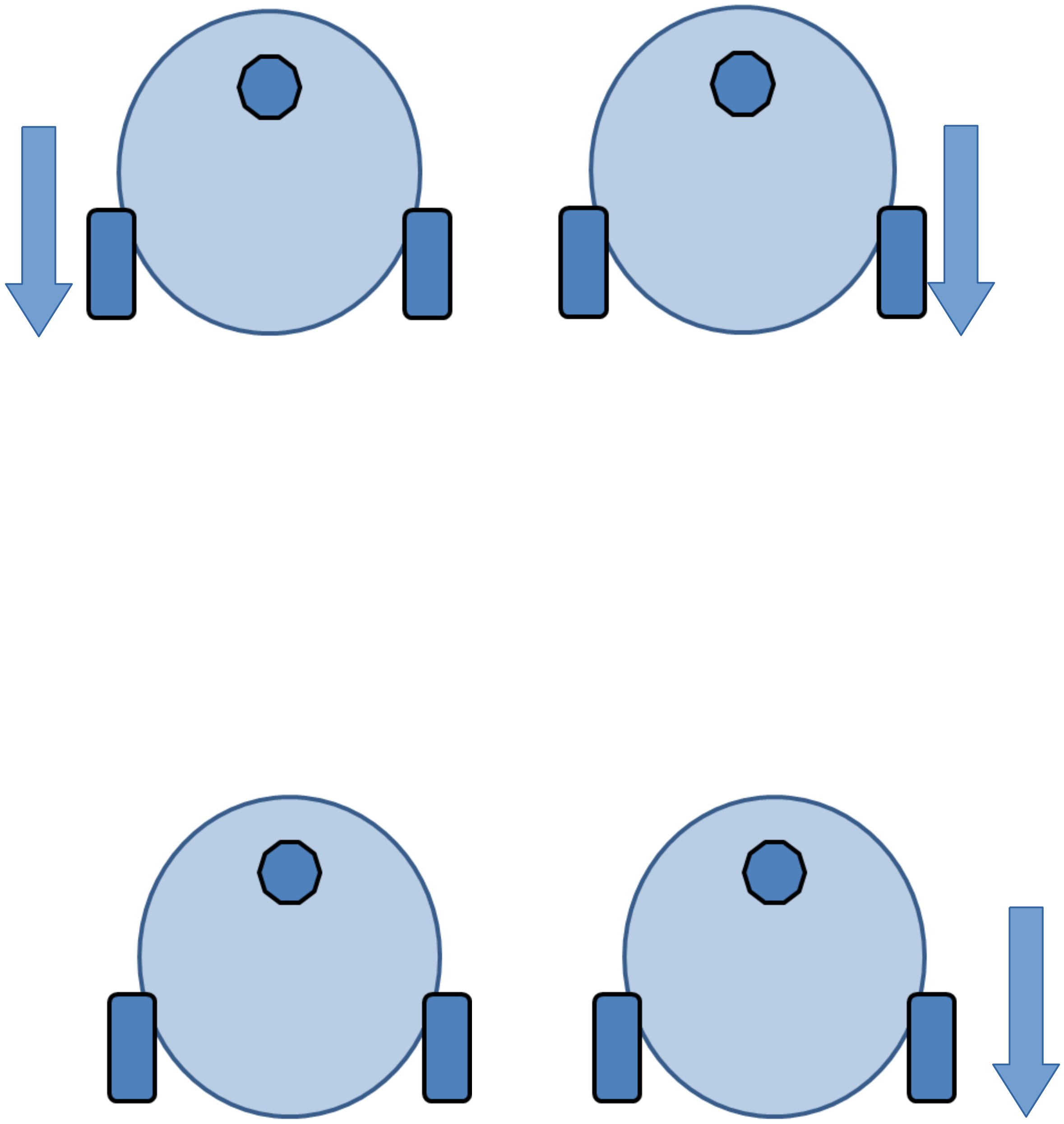


Figure 3.2 Various Motions [12]

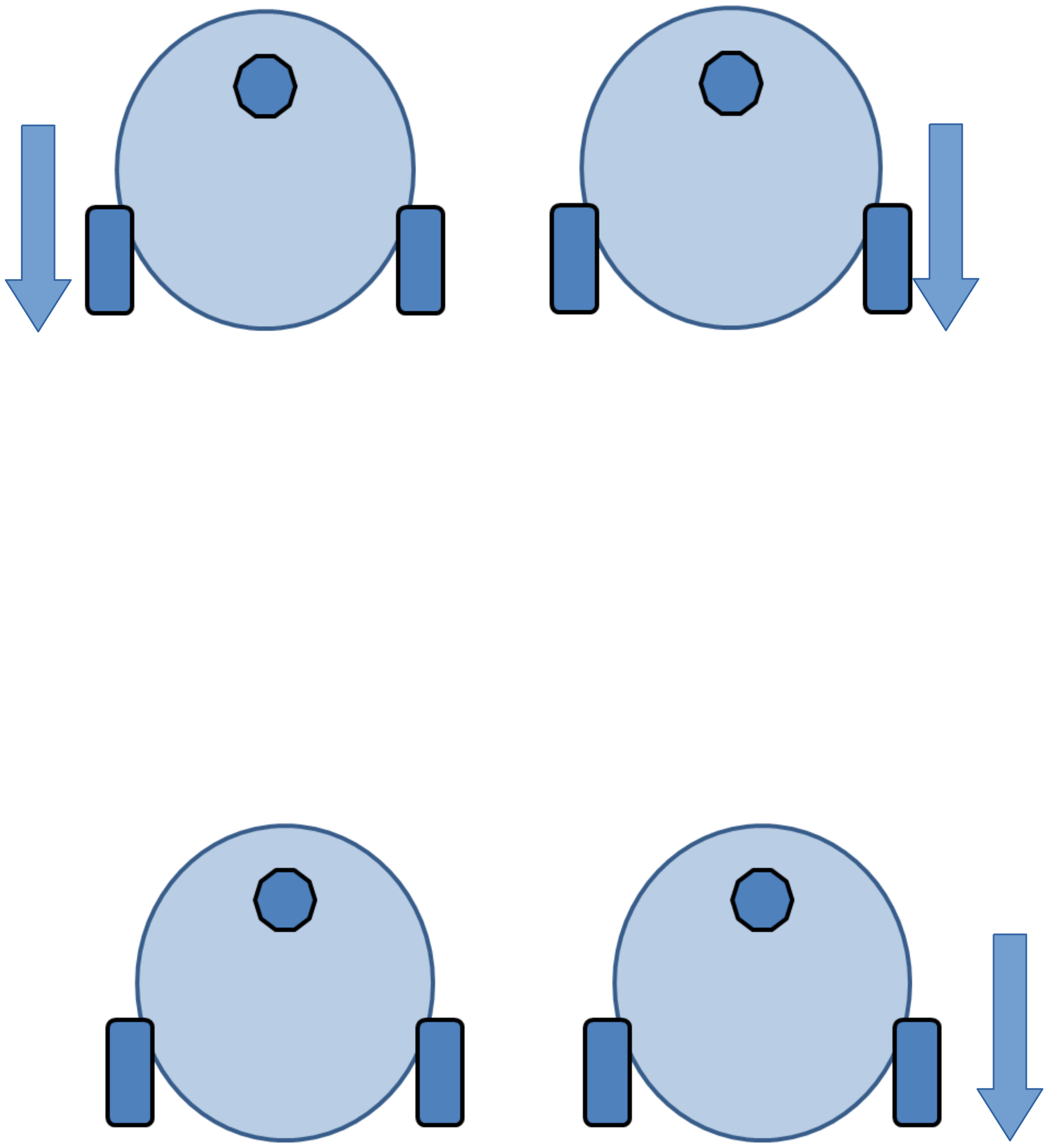


Figure 3.3 Various Soft Motions [12]

ii). Direction Control of DC Motor

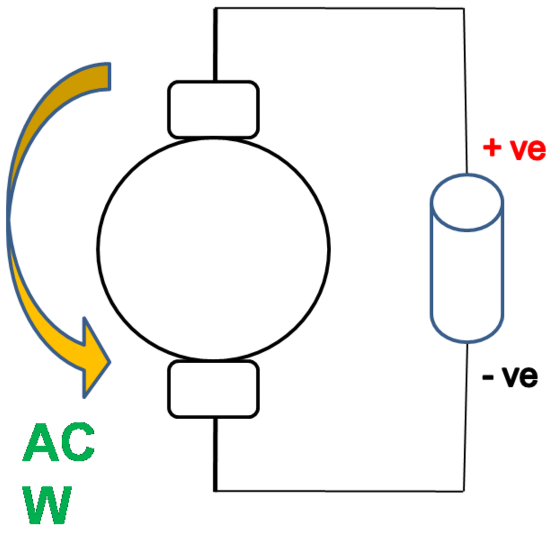


Figure 3.4 Anti-Clockwise [10]

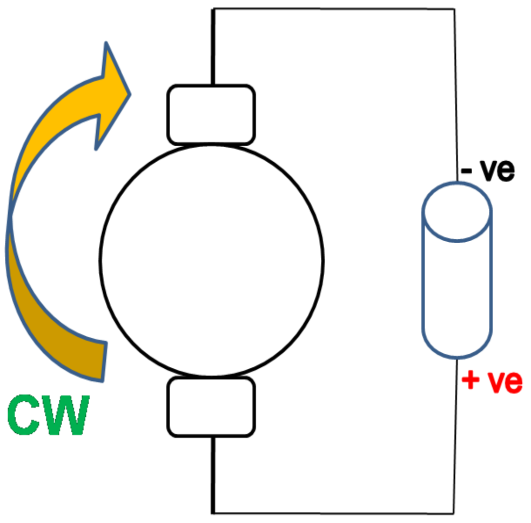


Figure 3.5 Clockwise [10]

iii). Motor Pin Connection

Four Pins for Direction control is connected at PORT A

a. PA0 - Left Motor Control

b. PA1 - Left Motor Control

c. PA2 - Right Motor Control

d. PA3 - Right Motor Control

Two Pins for Enabling Motor chauffler IC is connected at PORT L

a. PL3 - Left Channel Enable

b. PL4 - Right Channel Enable

iv). Logic Table

A Logic table to define Motion of the robot programatically is defined below from Table 3.1:

Direction	PA(0) LB	PA(1) LF	PA(2) RF	PA(3) RB
Forward	0	1	1	0
Backward	1	0	0	1
Left	1	0	1	0
Right	0	1	0	1
Soft Left	0	0	1	0
Soft Right	0	1	0	0
Stop	0	0	0	0

Table 3.1 Logic table for motion control.

3.3 LCD Interfacing

i). Liquid Crystal Display

a) A liquid crystal display (LCD) is a thin, flat panel used for electronically displaying information such as text, images, and moving pictures.

b) LCDs are economical and easy to use device. These are most commonly used display devices in an embedded entity. Commonly available display are set up as 16 to 20 characters by 1 to 4 lines.

ii). Dot Matrix Liquid Crystal Display

a). LCD used here has HD44780 dot matrix lcd controller. It is also called 16x2 Alpha Numeric LCD.

b). It can be configured to drive a dot-matrix liquid crystal display under the control of a 4 or 8-bit microprocessor

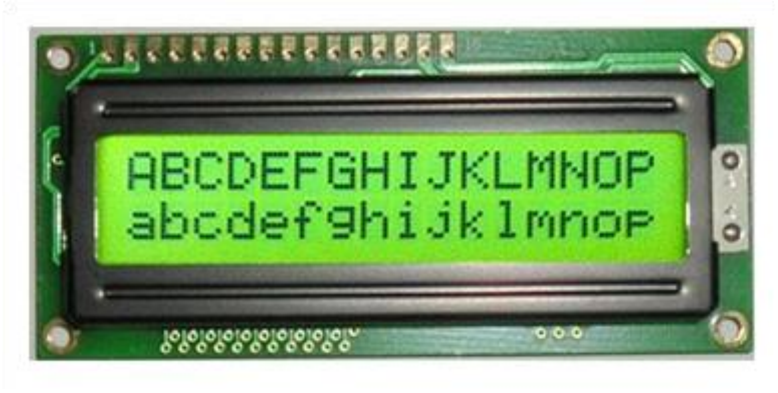


Figure 3.6 Dot Matrix LCD [10]

iii). Pin-Configuration

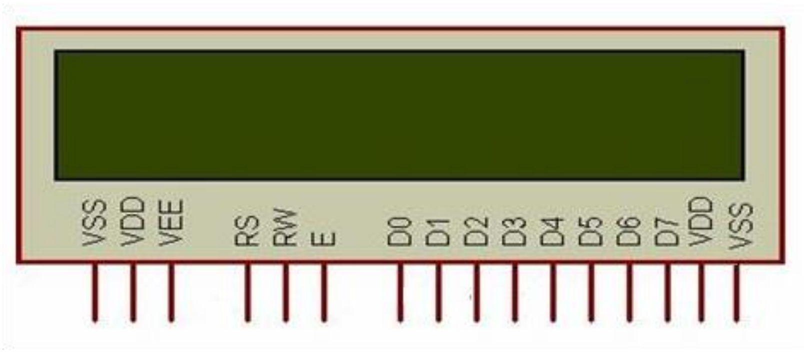


Figure 3.7 Pin Configuration in Firebird V [10]

Pin	**Description**
Vss	Ground
Vdd	Supply Voltage
Vee	Contrast Voltage
RS	Register Select
RW	Read/Write
E	Enable
D0-D7	Bidirectional Data Bus
Vdd,Vss	Back Light Supply

Table 3.2 Pin Description

iv). Control Pins

a). Register Select

If RS=0; Command Register

If RS=1; Data Register

b). Read/Write Select (As per table 3.2)

If RW=0; Write Mode

If RW=1; Read Mode

c). Enable

Used to latch the data present on the data pins

A high-to-low edge is needed to latch the data

v). Data Pins

Data Lines

a). There are 8 data pins from D0 to D7

b). Bidirectional Data / Command Pins

c). Alpha Numeric Character are sent in ASCII format

d). We can use LCD either 8 bit mode or 4 bit mode

e). We use 4 bit mode: only D4 to D7 data pins are used

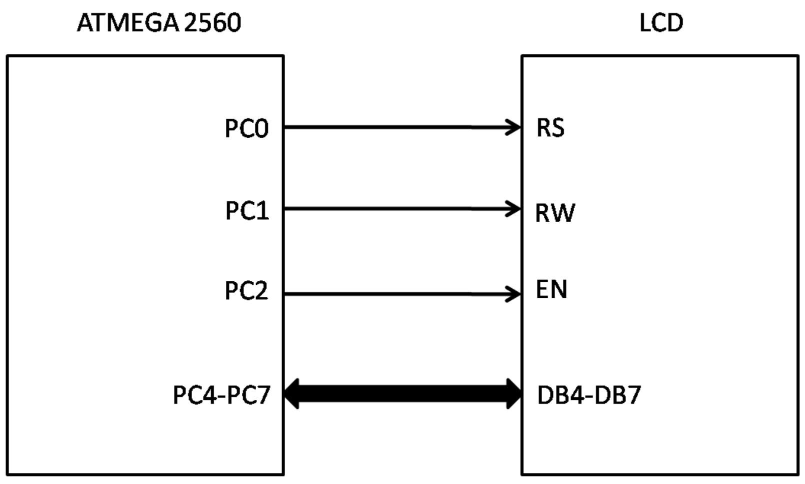


Figure 3.8 LCD Interfacing [10]

vi). Steps for LCD Initialization

a). Set Control Lines i.e. RS=0 and RW=0

b). Send LCD init value i.e. 0x38 for 8-bit mode OR 0x28 for 4-bit mode

c). Generate Hi-Low Pulse on Enable Pin of LCD

d). Send LCD Clear value i.e. 0x01

e). Send LCD Display On value i.e. 0x0F

f). Send LCD Cursor Home i.e. 0x02

3.4 Interrupt

i). What is an Interrupt?

a). Any signal that causes break in continuity of some ongoing process.

b). In microcontrollers interrupt signal halts the execution of main program and dedicates processor to another task.

c). While main program is running, if an interrupt occurs, execution of main program is stopped, and program counter goes to address of ISR.

d). Interrupt Service Routine: Program that needs to be executed when interrupt occurs.

e). After program inside ISR is executed completely, program counter returns back to point where main program was interrupted.

ii). Closed Loop Programming

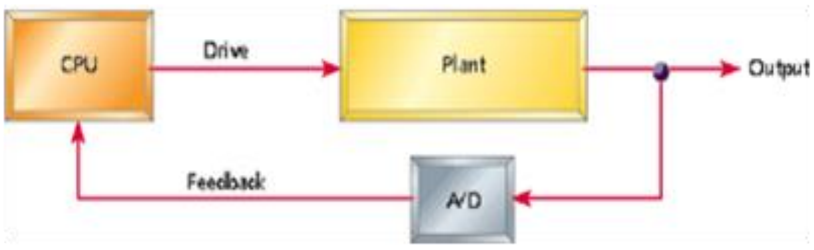


Figure 3.9 Closed Loop Programming

iii). Sources of Interrupt on ATmega2560.

ATmega 2560 has Fifty-Seven Different sources for Interrupt generation

a). Timer Overflow Interrupt

b). Timer Compare

c). Serial interrupt

d).Wired & Wireless Interrupt

e). External hardware Interrupt

3.5. Model Development:

For the movement of the we use a Line Follower Algorithm that allows robot to trace path and reach the destination easily.

What is Line following?

Ever wondered what line following is in the robotics language?!. Its a method by which a robot navigates from one position to another by following a line. This line is usually a white line against a darker background or a black line against a brighter background. Colleges and universities conduct these "Line following" event as a competition held against many other college students. Students participate in such events with a robot they have designed. The robots execute the program on these tracks prepared specially for the event.An assessment will be made based on the time taken and how well the bot followed line, without missing a part of it. If you think you should be participating in such an even, go ahead and read the description below. [11]

How to make a bot follow the line?

Its important to have a line sensor to track and detect the line. The line sensor is usually made using IR sensors. The position\number of these sensors depends on the complexity of the track to be solved.You can try out various position and numbers to make your robot efficient for following a line.There is no fixed layout for the best performance, as this varies based on the track. But its a trend to follow the "Straight line Array" layout.

A micro controller is very much required to read the line sensor output for the position of the robot.Once the position of the robot on the line is read, a decision has to be made to move the robot so that the line is in the center of the robot. Its always necessary to make sure that the line is on the center of the robot(Line sensor) this will enable the robot to move on the track. [5]

Below are some cases which is encountered while following the line and actions to be taken.

S1,S2,S3,S4 are the line sensor elements.

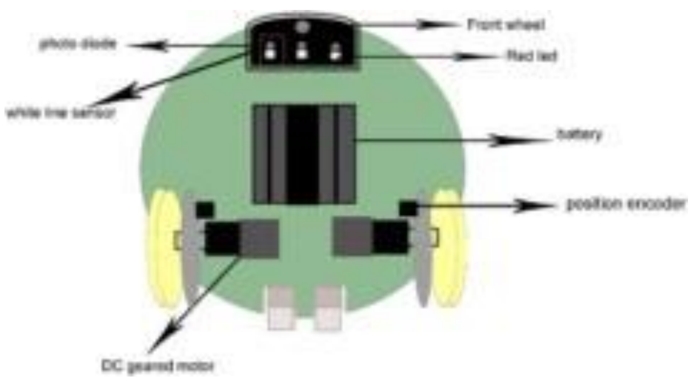


Figure 3.10 Sensor Placement in Firebird V [4]

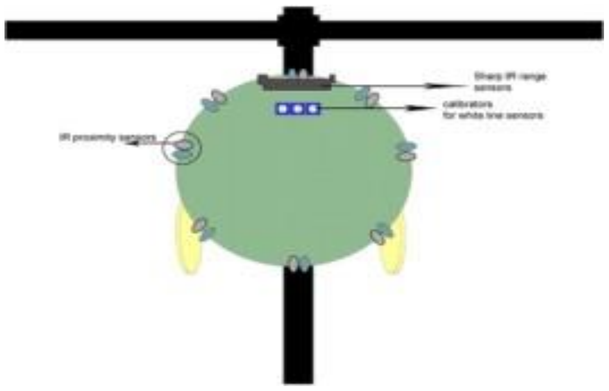


Figure 3.11 Representation of Line follower [3]

Line Follower Algorithm:

1. Start
2. Read S_{LL}, S_L and S_{RR} sensor's value.
3. If S_{LL} and S_{RR} on Black surface.
4. If S_L and S_R on black surfce.
5. Move forward(rotate both motor on full speed).
6 . Go to step 2.
7. If S_R on White Line.
8. Move Right (reduce Right motor speed to half).
9. Go to step 2.
10. If S_L on white line.
11. Move Left (reduce Left motor speed to half).
12. Go to step 2.
13. If S_{LL} and/or S_{RR} is/are on white surface.
14. Not a simple line.
15. Follow 90 degree turn Algorithm

Line Follower Algorithm Flow Chart:

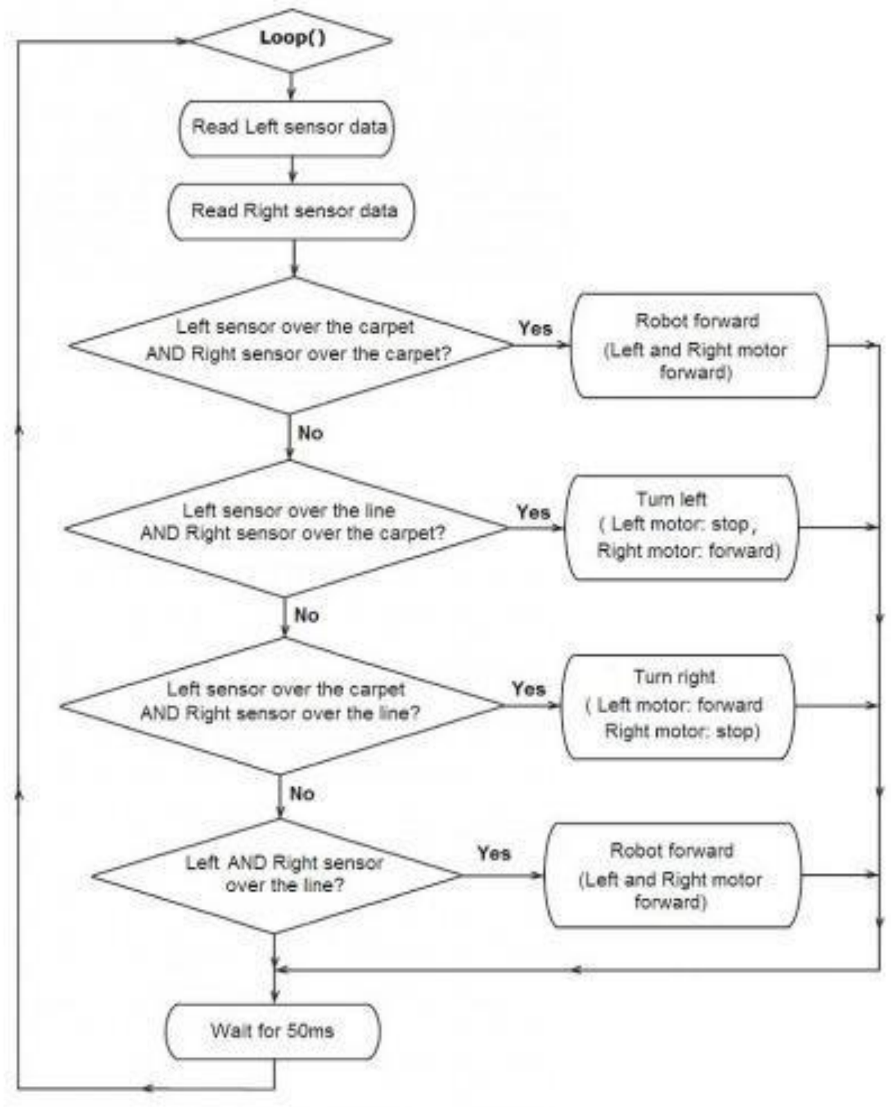

Loop()
Read Left sensor data
Read Right sensor data
Left sensor over the carpet AND Right sensor over the carpet?
Yes
Robot forward (Left and Right motor forward)
No
Left sensor over the line AND Right sensor over the carpet?
Yes
Turn left (Left motor: stop, Right motor: forward)
No
Left sensor over the carpet AND Right sensor over the line?
Yes
Turn right (Left motor: forward Right motor: stop)
No
Left AND Right sensor over the line?
Yes
Robot forward (Left and Right motor forward)
Wait for 50ms

Figure 3.12 Flow Chart for Line Follower Program [2]

Chapter-4: ALGORITHMS/ CODE

4.1 Line follower code along with obstacle detection:

Precap:

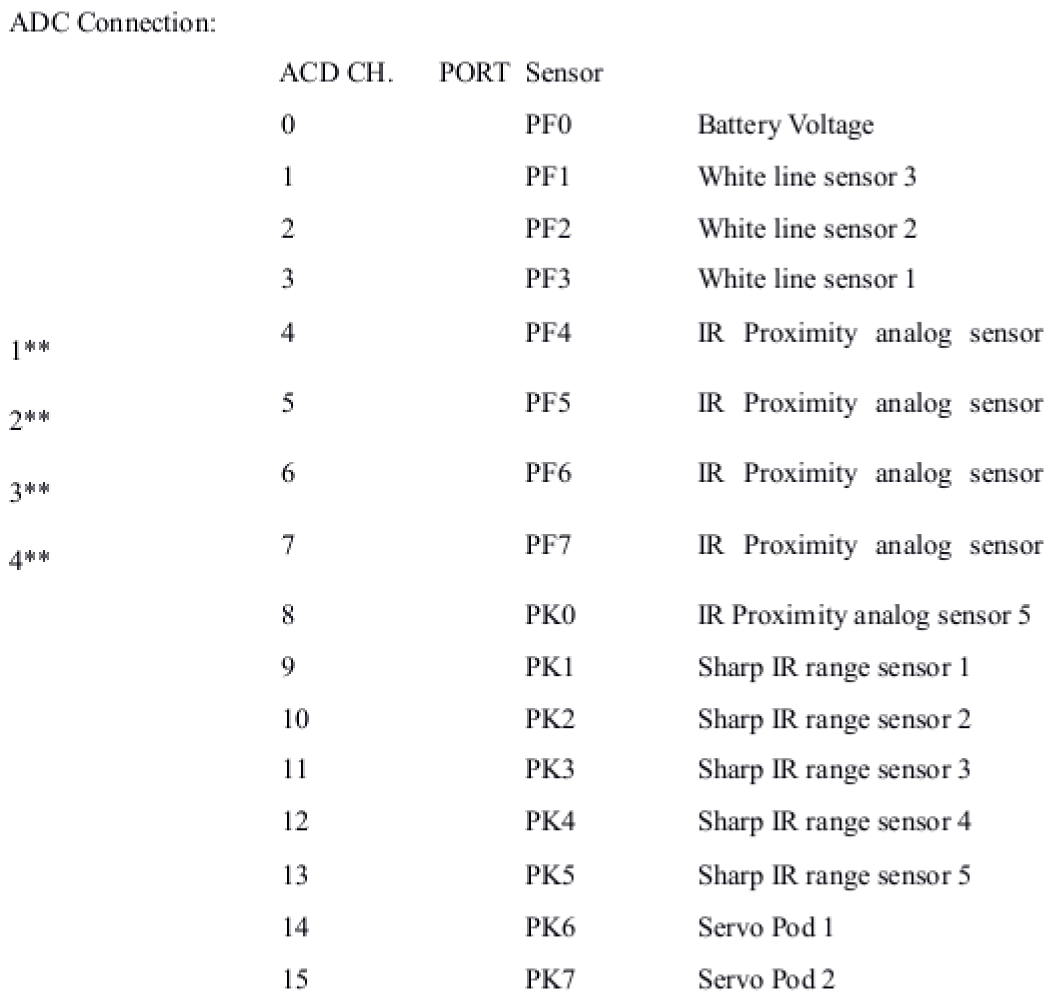
This code demonstrates the application of a simple line follower robot. The robot follows a white line over a black background, if any obstacle comes in front of the robot, robot stops and buzzer beeps.

Concepts covered: ADC, LCD interfacing, motion control based on sensor data

LCD Connections:

LCD Microcontroller Pins:

```
RS --> PC0
RW --> PC1
EN --> PC2
DB7 --> PC7
DB6 --> PC6
DB5 --> PC5
DB4 --> PC4
```



ADC Connection:

	ACD CH.	PORT	Sensor
	0	PF0	Battery Voltage
	1	PF1	White line sensor 3
	2	PF2	White line sensor 2
	3	PF3	White line sensor 1
1**	4	PF4	IR Proximity analog sensor
2**	5	PF5	IR Proximity analog sensor
3**	6	PF6	IR Proximity analog sensor
4**	7	PF7	IR Proximity analog sensor
	8	PK0	IR Proximity analog sensor 5
	9	PK1	Sharp IR range sensor 1
	10	PK2	Sharp IR range sensor 2
	11	PK3	Sharp IR range sensor 3
	12	PK4	Sharp IR range sensor 4
	13	PK5	Sharp IR range sensor 5
	14	PK6	Servo Pod 1
	15	PK7	Servo Pod 2

For usi

ng Analog IR proximity (1, 2, 3 and 4) sensors short the jumper J2. [8]

To use JTAG via expansion slot of the microcontroller socket remove these jumpers.

Motion control Connection:

L-1---->PA0; L-2---->PA1;

R-1---->PA2; R-2---->PA3;

PL3 (OC5A) ----> PWM left; PL4 (OC5B) ----> PW

Code starts here:

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <math.h> //included to support power function
#include "lcd.c"
void port_init();

void timer5_init();

void velocity(unsigned char, unsigned char);
void motors_delay();
unsigned char ADC_Conversion(unsigned char);
unsigned char ADC_Value;
unsigned char flag1 = 0;
unsigned char flag2 = 0;
unsigned char Left_white_line = 0;
unsigned char Center_white_line = 0;
unsigned char Right_white_line = 0;
unsigned char Front_Sharp_Sensor=0;
unsigned char Front_IR_Sensor=0;
//Function to configure LCD port
void lcd_port_config (void)
{

 DDRC = DDRC | 0xF7; //all the LCD pin's direction set as output
 PORTC = PORTC & 0x80; // all the LCD pins are set to logic 0 except PORTC 7

}

//ADC pin configuration
void adc_pin_config (void){
DDRF = 0x00;
 PORTF = 0x00;
 DDRK = 0x00;
 PORTK = 0x00; }
//Function to configure ports to enable robot's motion
void motion_pin_config (void) {
 DDRA = DDRA | 0x0F;
 PORTA = PORTA & 0xF0;
 DDRL = DDRL | 0x18; //Setting PL3 and PL4 pins as output for PWM generation
 PORTL = PORTL | 0x18; //PL3 and PL4 pins are for velocity control using PWM.
}//Function to initialize Buzzer
void buzzer_pin_config (void){
 DDRC = DDRC | 0x08;             //Setting PORTC 3 as outpt

 PORTC = PORTC & 0xF7;           //Setting PORTC 3 logic low to turnoff buzzer}

//Function to Initialize PORTS
void port_init(){
        lcd_port_config();
        adc_pin_config();
        motion_pin_config();
        buzzer_pin_config(); }
```

```
// Timer 5 initialized in PWM mode for velocity control

// Prescale:256

// PWM 8bit fast, TOP=0x00FF

// Timer Frequency:225.000Hz
void timer5_init(){
        TCCR5B = 0x00;      //Stop

        TCNT5H = 0xFF;      //Counter higher 8-bit value to which OCR5xH value is
compared with

        TCNT5L = 0x01;      //Counter lower 8-bit value to which OCR5xH value is
compared with

        OCR5AH = 0x00;      //Output compare register high value for Left Motor
        OCR5AL = 0xFF;      //Output compare register low value for Left Motor
        OCR5BH = 0x00;      //Output compare register high value for Right Motor
        OCR5BL = 0xFF;      //Output compare register low value for Right Motor
        OCR5CH = 0x00;      //Output compare register high value for Motor C1
        OCR5CL = 0xFF;      //Output compare register low value for Motor C1
        TCCR5A = 0xA9;

        TCCR5B = 0x0B;      //WGM12=1; CS12=0, CS11=1, CS10=1 (Prescaler=64)

}

void buzzer_on (void){
 unsigned char port_restore = 0;
 port_restore = PINC;
 port_restore = port_restore | 0x08;
 PORTC = port_restore;}
```

```
void buzzer_off (void){
 unsigned char port_restore =
 0;port_restore = PINC;

 port_restore = port_restore & 0xF7;
 PORTC = port_restore;}
void adc_init(){

        ADCSRA = 0x00;

        ADCSRB = 0x00;      //MUX5 = 0

        ADMUX = 0x20;       //Vref=5V external --- ADLAR=1 --- MUX4:0 =0000
        ACSR = 0x80;
        ADCSRA = 0x86;      //ADEN=1 --- ADIE=1 --- ADPS2:0 = 1 1 0}



//Function For ADC Conversion

unsigned char ADC_Conversion(unsigned char Ch) {
        unsigned char a;
        if(Ch>7){

                ADCSRB = 0x08;}

        Ch = Ch & 0x07;
        ADMUX= 0x20| Ch;
        ADCSRA = ADCSRA| 0x40;          //Set start conversion bit

        while((ADCSRA&0x10)==0);        //Wait for conversion to complete
        a=ADCH;
        ADCSRA = ADCSRA|0x10; //clear ADIF (ADC Interrupt Flag) by writing 1 to it
        ADCSRB = 0x00;
        return a;}
```

```
//Function To Print Sesor Values At Desired Row And Coloumn Location on LCDvoid
print_sensor(char row, char coloumn,unsigned char channel){ ADC_Value =
ADC_Conversion(channel);
        lcd_print(row, coloumn, ADC_Value, 3);}

//Function for velocity control

void velocity (unsigned char left_motor, unsigned char right_motor){
        OCR5AL = (unsigned char)left_motor;
        OCR5BL = (unsigned char)right_motor;}

//Function used for setting motor's direction
void motion_set (unsigned char Direction){
unsigned char PortARestore = 0;
 Direction &= 0x0F;                 // removing upper nibbel for the protection
 PortARestore = PORTA;              // reading the PORTA original status
 PortARestore &= 0xF0;              // making lower direction nibbel to 0
 PortARestore |= Direction;
PORTA = PortARestore;               // executing the command

}

void forward (void) {
motion_set (0x06);}
void stop (void){
motion_set (0x00);}
void init_devices (void){
        cli(); //Clears the global interrupts
        port_init();
        adc_init();
        timer5_init();

        sei(); //Enables the global interrupts

}

//Main Function
int main(){
        init_devices();
        lcd_set_4bit();
        lcd_init();
        while(1){
        Left_white_line = ADC_Conversion(3);        //Getting data of Left WL Sensor
        Center_white_line = ADC_Conversion(2);    //Getting data of Center WL Sensor
        Right_white_line = ADC_Conversion(1);      //Getting data of Right WL Sensor
                Front_Sharp_Sensor = ADC_Conversion(11);
                Front_IR_Sensor = ADC_Conversion(6);
                flag1=0;
                flag2=0;

                print_sensor(1,1,3);    //Prints value of White Line Sensor1
                print_sensor(1,5,2);    //Prints Value of White Line Sensor2
                print_sensor(1,9,1);    //Prints Value of White Line Sensor3
                print_sensor(2,4,11); //Prints Value of Front Sharp Sensor
                print_sensor(2,8,6);    //Prints Value of Front IR Sensor
                if(Front_Sharp_Sensor>0x82 || Front_IR_Sensor<0xF0{
                        flag2=1;
                        stop();
```

```
            buzzer_on();}
if((Center_white_line<0x28) && (flag2==0)){
            flag1=1;
            buzzer_off();
            forward();
            velocity(150,150);}
if((Left_white_line>0x28) && (flag1==0) && (flag2==0)){
            flag1=1;
            buzzer_off();
            forward();
            velocity(150,50);}
if((Right_white_line>0x28) && (flag1==0) && (flag2==0)){
            flag1=1;
            buzzer_off();
            forward();
            velocity(50,150);}
if((Center_white_line>0x28) && (Left_white_line>0x28) &&
    (Right_white_line>0x28) && (flag2==0)) {

            buzzer_off();
            forward();
            velocity(0,0);}}}
```

Chapter-5: CONCLUSIONS

5.1 Conclusions

Unregulated growth in the number of vehicles, compounded with unplanned parking areas in small localities such as residential complexes and office plazas make it a daunting exercise to find a parking spot for an automobile.

This leads to people parking in any empty spot they can find. This issue is critical when a spot assigned to a particular person or resident of a building gets occupied by a visitor‟s vehicle. This kind of situation often causes disputes in residential areas and can cause a nuisance such as a traffic jam in public spaces.

Given that the problem of parking arises due to inappropriate parking of vehicles at the parking area, a well-managed parking lot with proper allocation of parking slots for both resident and visitor vehicles coming to the locality can prevent disputes and help traffic flow better.

5.2 Future Scope

Same hardware above can be embedded to the present/upcoming cars such that it can support the parking entity described above which will be not limited to localities. The parking entitys and the whole hardware-software stack can be upgraded in sync with IOT applications so as to provide the real time information about the nearest parking locations along with availability of slots.

REFERENCES

[1] Eyantra Research Porrtal “A Review on Automation in vehicle industry”, Journal for IIT Bombay Research association in collaboration with National Research organisation.

[2] Sanjay Patil “A guide to ATMEL Studio for micro controller programming”, Book for programming in Firebird V robot VOL. 5.

[3] “Introduction to Firebird-V Robotics Research Platform”, a research initiated by Embedded Real-Time entitys Lab Indian Institute of Technology - Bombay.

[4] NEX Robotics Pvt. Ltd. INDIA “Firebird V Antistatic Protection while using robot” a guide by NEX Robotics India that includes Safety measures to use a firebird V robots with an antistatic protection measures.

[5] NEX Robotics Pvt. Ltd. INDIA and ERTS Lab, IIT Bombay “FIREBIRD V ATMEGA2560 Robotic Research platform software manual” a manual issued to get started with software components and modules of ATMEGA 2560.

[6] NEX Robotics Pvt. Ltd. INDIA and ERTS Lab, IIT Bombay “FIREBIRD V ATMEGA2560 Robotic Research platform software manual” a manual issued to get started with components and sensors of ATMEGA 2560.

[7] Additional resources and Websites:

[8] Resource provided by Eyantra link here: http://elsi.e-yantra.org/resources

[9] Resource provided by Embedded systems IIT Bombay link here:
https://www.cse.iitb.ac.in/~cs6212011/lectures_2010/es_intro_2010_AI_class.pdf

[10] Resource provided by Firebird V Atmega2560 link here:
http://www.nex-robotics.com/robots.htm

[11] Shidian Ma , Arun Madhvan and James arrtwright Automation in parking lots Research paper from Automotive Engineering Research Institute, Jiangsu University, Zhenjiang, China. Link here: http://ieeexplore.ieee.org/abstract/document/8071146/.

[12] S.A Mahmud A Survey of Intelligent Car Parking System research paper is provided by IEEE confrences. Link here: https://www.sciencedirect.com/science/article/pii/S1665642313715803.

[13] Arun Kumar singh Design and development in valet parking research paper is provided by on https://www.researchgate.net/publication/272683686_Design_and_Development_of_Automated_Parking_Slot .