INTELLIGENT CAR PARKING MANAGEMENT SYSTEM

Project Report submitted in partial fulfilment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

By

Pushpender Singh (141064)

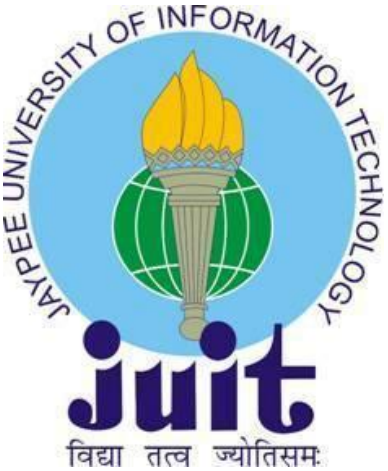
Keshav Saini (141106)

Mayank Ameta (141087)

UNDER THE GUIDANCE OF

Dr. Rajiv Kumar

Associate Professor
(ECE Department)



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT
SOLAN – 173 234, HIMACHAL PRADESH INDIA
May-2018

TABLE OF CONTENTS

DECLARATION

We hereby declare that the work reported in the B-Tech Project Report entitled **"Intelligent Car Parking Management System"** submitted at **Jaypee University of Information Technology, Waknaghat, India,** is an authentic record of our work carried out under the supervision of **Dr. Rajiv Kumar**. We have not submitted this work elsewhere for any other degree or diploma.

Pushpender Singh Mayank Ameta Keshav Saini

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT, SOLAN, H.P

Date:

SUPERVISOR'S CERTIFICATE

This is to certify that the work reported in the B-Tech. Report entitled **"Intelligent Car Parking Management System"**, submitted by **Pushpender Singh, Mayank Ameta and Keshav Saini** at **Jaypee University of Information Technology, Waknaghat, India** is a bonafide record of their original work carried out under my supervision. This work has not been submitted elsewhere for any other degree or diploma.

Dr. Rajiv Kumar

Associate Professor

Electronics and Communication Engineering

JUIT Waknaghat

Date:

ACKNOWLEDGEMENT

We owe a great many thanks to a great many people who have helped and supported us during this project. Our deepest thanks to **Dr. Rajiv Kumar (Associate Professor)**, our project guide for his exemplary guidance, monitoring and constant encouragement throughout the course of this project work. He has taken great pains to go through the project and make necessary corrections as and whenever needed. We are also grateful to **Mr. Pramod Kumar (ECE Project lab**) for their practical help and guidance. We would also like to thank all the faculty members of ECE department without whom the progress of this project would have been a distant reality. We also extend our heartfelt thanks to our family and well-wishers.

Name of the students:

Pushpender Singh (141064)

Keshav Saini (141106)

Mayank Ameta (141087)

Date:

LIST OF ACRONYMS & ABBREVIATIONS

1. APS – “Automated (car) Parking System”
2. LCD – “Liquid Crystal Display”
3. LED – “Light Emitting Diode”
4. AVR – “Advanced Virtual RISC”
5. ISIS –“Intelligent Schematic Input System”
6. IDE – “Integrated Development Environment”
7. VSM – “Virtual System Modelling”
8. RISC – “Reduced Instruction Set Computer”
9. IR – “Infrared”
10. PCB – “Printed Circuit Board”
11. GSM – “Global System for Mobile Communication”
12. OTP – “One Time Password”

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This project report presents the idea and implementation of “Intelligent Car Parking Management System”. Automation, is the use of the various systems for operating components or objects with minimal human hindrance. The biggest advantage of automation is that it saves labour and time, however, it is also used to save energy and materials used and to improve quality, accuracy, precision and implementation time.

In our this project we are basically designing a parking system where one has to enter the details of his car (like car registration number) at the entry gate and then leave his car there and the car will automatically get parked at the available parking slot. If the parking area is full, a warning message of “Parking is full!” will be displayed. In this way that person will be able to save his time and energy.

We have divided our project into three parts i.e. planning part, software implementation, research paper publication and hardware implementation. First of all we planned layout of our project. Then we implemented our project on software. For software implementation we used Atmel AVR Studio (for writing code in embedded C) and ISIS professional proteus (for circuit simulation). Finally we have also tested our project on hardware and we got positive results on it.

CHAPTER 1

INTRODUCTION

Nowadays in many multiplex structures there is a severe trouble for automobile parking systems. There are many lanes for auto parking, so to park an auto one has to see for the all lanes. Conventionally, car parking structures does not have any sensible monitoring system. Parking a lot are monitored via human beings. All vehicles entering into the parking and wasting time for looking for parking slot. Sometimes it creates blockage. Condition emerge as worse when there are lot of parking lines and each & every lane have more than one parking slots. Moreover lots of men labour involved for this manner for which there is lot of investment. So the want is to boost a system where man or woman leaves his/her car at entrance and the auto ought to routinely get parked. Benefits of computerized gadget for automobile parking monitoring will decrease the human efforts.

These days in numerous multiplex systems there is an extreme issue car parking. There are numerous paths for car parking, so to stop an auto one needs to search for the all paths. Traditionally, auto stopping frameworks does not have any smart checking framework. Parking garages are observed by individuals. All vehicles go into the stopping and sit idle for hunting down stopping opening. Now and again it makes blockage. Condition turn out to be more awful when there are different stopping paths and every path have numerous stopping openings. Additionally there is a ton of men work required for this procedure for which there is parcel of speculation. So the need is to build up a framework where individual leaves his auto in stopping section and the auto ought to naturally get stopped. Utilization of robotized framework for auto stopping observing will diminish the human endeavours.

The concept for the automated parking system is driven by two factors: a need for parking spaces and a scarcity of available land.

1.1 LITERATURE WORK

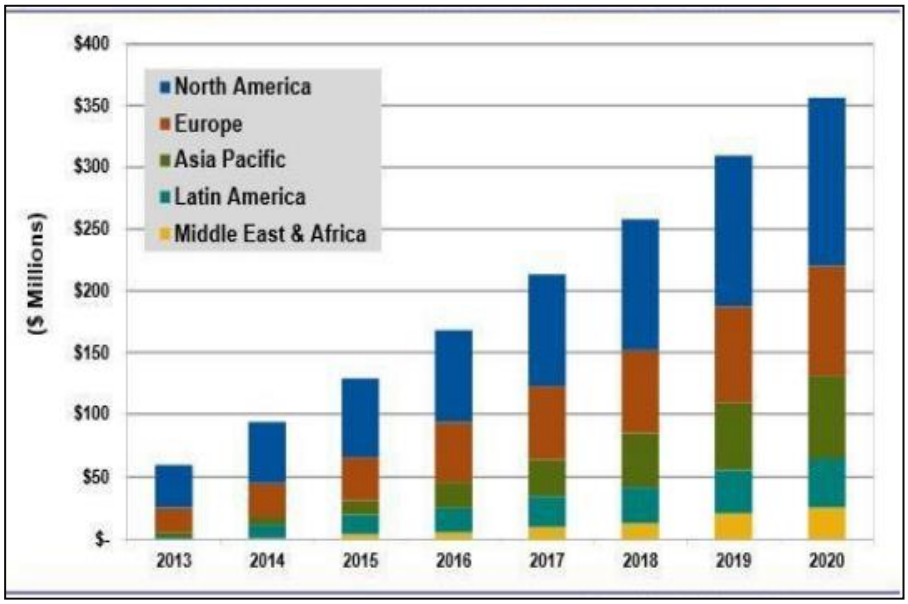
Lots of study and research has been executed in the field of automatic vehicle parking gadget to provide first-class substitute for easy, protected & rapid car parking. One such concept was to scan the registration information of the car from number plate the usage of image processing technique [1] and then the use of this auto registration quantity discovering out vehicle entry time, car exit time and additionally time period for which car was once parked

which will help in accumulating parking fees/fare. “The smart car parking system was once designed using some sensors which worked in three phases” [2]. 1st phase used for the scanning segment in which the parking lot was examined through the Ultrasonic Sensors established on the robot-car and a path is produced if the space is sufficient. 2nd section used to be positioning segment which consisted of the robotic reverses to the part of the parking area warding off any collision. Finally in 3rd segment the robotic moves to the parking role in the parking house in a unified pattern.

One similar plan was given that uses 3 supersonic detectors mounted on front left corner of the automobile for getting the data of encompassing space of automobile parking space [3].These four sensors facilitate to decide the turning angle of the automobile.
Another concept proposed used to be a layout of an automated car parking machine managed by means of an android software [4] that controls the variety of automobiles to be parked/un-parked within a parking region with the help of an Android Application.
Some smart parking structures were additionally embedded with technologies like GSM modules [5-6] for SMS services to grant more secure and person friendly parking system.

Few other researches had been additionally performed the use of nonholonomic vehicles [7] and soft-computing techniques [8].

Graph shows the need of smart parking system in future



Thus, our purpose was to suggest an automated parking system in context of parking which are based on auto drivers which will now not only save time, cash and strength but additionally overcome the barriers of such similar pre-existing systems.

1.2 BENEFITS OF AUTOMATED PARKING SYSTEMS (APS):

- Havoc and waiting during parking is reduced.
- Boon for old-age as nicely as beginner drivers who generally create problems while parking automobile.
- Driving for search of a parking space is reduced, thereby reducing engine emissions.
- Ceiling height is minimized since there is no pedestrian traffic (drivers and passengers) in the parking area.
- No walkways, stairways or elevators are needed.
- The parked cars and their contents are more secure since there is no public access to parked cars.
- Only minimal ventilation and lighting systems are required.
- APS theoretically removes the need for parking assistance or labour.

Comparison of manual and automatic parking:

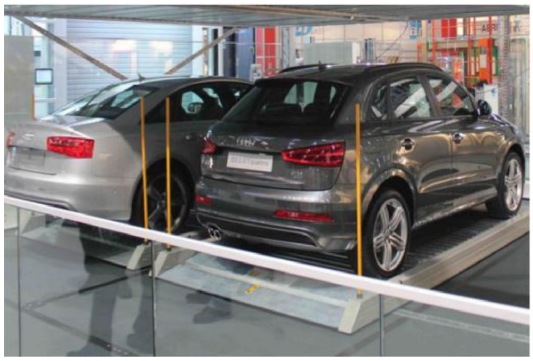


Figure 1.1: Automatic Parking [16]

Figure 1.2: Manual Parking [17]

1.3 DISADVANTAGES OF AUTOMATED PARKING SYSTEMS (APS):

- It may be costly due to involvement of complex devices for automation.
- If the sensing devices go out-of-order or non-functional then there may be great difficulty in getting cars out of parking, more over it may cause accidents inside parking also.

CHAPTER 2

PROJECT REVIEW AND BACKGROUND MATERIAL

2.1 PROJECT REVIEW:

This project has 2 parts. First phase is to make a Parking Lot {Smart/Automated} and Second phase is to make a Vehicle {Smart/Automated}. As soon as auto will arrive at the entrance of the parking place the driver will come out of the car and then he will enter his car's registration quantity at that entry gate. As he enters the vehicle registration wide variety a slot variety corresponding to that automobile wide variety will be routinely allocated to him. This dispensed slot number will appear on LCD screen of parking area. If parking vicinity is full than a warning message displaying "Parking is full!" will be displayed on the LCD screen. Now the driver will enter the allotted slot range into his vehicle the using the keypad embedded on his car. The car will robotically enter the parking place and will get parked to its allocated slot. Also as soon as the automobile enters the parking area, a timer will automatically start, which will be helpful in making decisions regarding parking time, parking fees and future prediction imprints.

2.2 SOFTWARE USED:

2.2.1 AVR STUDIO 4

Atmel AVR Studio is an IDE for developing and debugging embedded Atmel AVR applications. The AVR Studio IDE provides you a seamless and easy-to-use surroundings to jot down, build, and right your C/C++ and computer programme code. We will burn this code on hardware circuit microcontroller exploitation coder to form real time applications. We have a tendency to area unit exploitation AVR Studio four for our C codes. It supports over three hundred Atmel AVR and Atmel sensible ARM-based devices .It has in-system programming and debugging provides interface to any or all Atmel in-circuit programmers and debuggers.

2.2.2 CV AVR

Code vision AVR (CV AVR) compiler is a C cross-compiler, Integrated Development Environment (IDE) and Automatic Program Generator designed for the Atmel microcontroller series made AVR. Code vision AVR can be run on Operating System Windows 95, 98, Me, NT4, 2000, XP and 7. C cross-compiler is able to

translate almost all orders of ANSI C language, to the extent permitted by the architecture of the AVR, with the addition of some special features to take advantage of the AVR architecture and the needs of the embedded system. It can be used for debugging purposes at the level of C.

2.2.3 ISIS PROFESSIONAL PROTEUS

Proteus 7.0 could be a Virtual System Modelling (VSM) that mixes circuit simulation, animated parts and chip models to co-simulate the whole microcontroller primarily based styles. This is often the proper tool for engineers to check their microcontroller styles before constructing a physical epitome in real time. This program permits users to move with the planning victimization on-screen indicators and/or semiconductor diode and digital display displays and, if connected to the laptop, switches and buttons. We tend to square measure victimization version 7.9 for our simulations.

The VSM, Virtual System Modelling, provides a graphical SPICE circuit simulation and animation directly in the ISIS environment. The SPICE simulator is based on the Berkeley SPICE3F5 model.

It may microprocessor-based systems are simulated. With the VSM-Engine can interact during the simulation directly to the circuit. Changes of buttons, switches or potentiometers are queried in real time as well as LED indicators, LCD display and Hot / Cold -Wires displayed.

The microcontrollers are in the periphery and in the code fully supported (interrupt, ADC, I2C, USB, comparators, etc.). It includes a debugging environment for the program code of the microcontroller. To simulate the .HEX and .COF file of the compiled software are necessary. The clock is simulated in real time.

2.3 HARDWARE COMPONENTS USED:

2.3.1 MICROCONTROLLER (ATMEL AVR, ATMEGA16)

The ATmega16 is a low-control CMOS 8-bit microcontroller in view of the AVR improved RISC engineering. By executing effective guidelines in a solitary clock cycle, the ATmega16 accomplishes throughputs moving toward 1 MIPS for every MHz enabling the framework architect to upgrade control utilization as opposed to preparing speed.

The AVR core consolidates a rich guideline set with 32 universally useful working registers. All the 32 registers are specifically associated with the Arithmetic Logic Unit (ALU), enabling two free registers to be gotten to in one single guideline executed in one clock cycle. The subsequent design is more code productive while accomplishing throughputs up to ten times quicker than regular CISC microcontrollers. The ATmega16 gives the accompanying highlights: 16 Kbytes of In-System Programmable Flash Program memory with Read-While-Write capacities, 512 bytes EEPROM, 1 Kbyte SRAM, 32 universally useful I/O lines, 32 broadly useful working registers, a JTAG interface for limit filter, On-chip Debugging backing and programming, three adaptable Timer/Counters with think about modes, Internal and External Interrupts, a serial programmable USART, a byte situated Two-wire Serial Interface, a 8-channel, 10-bit ADC with discretionary differential information organize with programmable pick up (TQFP bundle just), a programmable Watchdog Timer with Internal Oscillator, a SPI serial port, and six programming selectable power sparing modes.

The Idle mode stops the CPU while permitting the USART, Two-wire interface, A/D Converter, SRAM, Timer/Counters, SPI port, and intrude on framework to hold working. The Power-down mode spares the sign up substance however solidifies the Oscillator, debilitating all other chip capacities till the point that the following External Interrupt or Hardware Reset. In Power-spare mode, the Asynchronous Timer maintains on running, enabling the customer to keep up a clock base while anything stays of the system is resting. The ADC Noise Reduction mode stops the CPU and all I/O modules with the exception of Asynchronous Timer and ADC, to restriction exchanging clamour amid ADC changes. In Standby mode, the gem/resonator Oscillator is going for walks whilst whatever is left of the machine is dozing. This lets in speedy start-up joined with low-control utilization. In Extended Standby mode, both the principal Oscillator and the Asynchronous Timer maintain on running.
The gadget is made utilizing Atmel's high thickness non-unstable memory innovation. The on-chip ISP Flash enables the program memory to be reconstructed in-framework through a

SPI serial interface, by an ordinary non-unpredictable memory developer, or by an On-chip Boot program running on the AVR center. The boot program can utilize any interface to download the application program in the Application Flash memory. Programming in the Boot Flash area will keep on running while the Application Flash segment is refreshed, giving genuine Read-While-Write task. By consolidating a 8-bit RISC CPU with In-System Self-Programmable Flash on a solid chip, the Atmel ATmega16 is a capable microcontroller that gives a very adaptable and practical answer for some installed control applications. The ATmega16 AVR is upheld with a full suite of program and framework improvement devices including: C compilers, large scale constructing agents, program debugger/test systems, in-circuit emulators, and assessment packs.

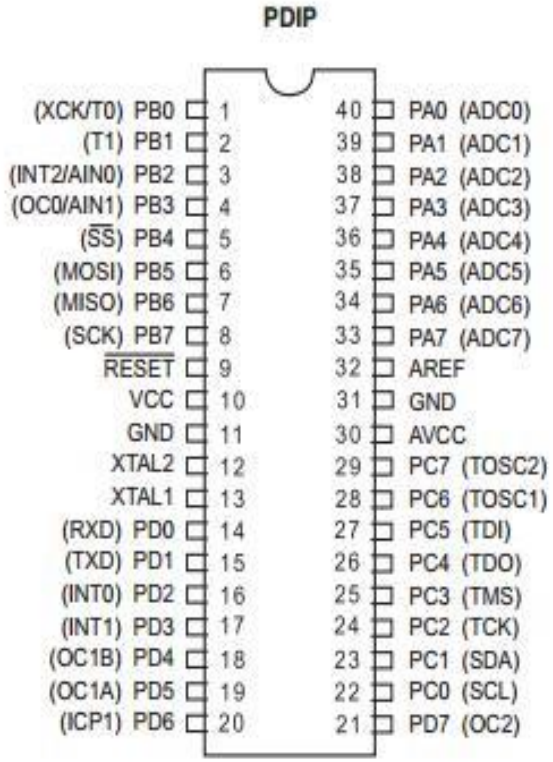


Figure 2.1: Pinout of ATmega16 [18]

PIN DESCRIPTIONS:

VCC –"Digital supply voltage."

GND -Ground.

Port A (PA7..PA0) - Port A act as the analog inputs to the A/D Converter. Port A additionally functions as an 8-bit bi-directional I/O port, if the A/D Converter is now not utilized. Pins can supply interior pull-up resistors (selected for each bit).

The Port A output buffers have symmetrical power characteristics with each 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 interior pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is now not running.

Port B (PB7..PB0) - Port B is 8-bit bi-directional I/O port with interior draw up resistors (chose for each piece). The Port B yield cradles have symmetrical drive qualities with both high sink and source ability. As data sources, Port B sticks that are remotely pulled low will source current if the draw up resistors are actuated. The Port B pins are tri-expressed when a reset condition ends up dynamic, regardless of whether the clock isn't running.

Port C (PC7..PC0) – Port C is an 8-bit bi-directional I/O port with inside draw up resistors (decided for each piece). The Port C yield underpins have symmetrical drive traits with both high sink and source limit. As data sources, Port C sticks that are remotely pulled low will source current if the draw up resistors are instituted. The Port C pins are tri-communicated when a reset condition winds up unique, paying little heed to whether the clock isn't running. If the JTAG interface is enabled, the draw up resistors on pins PC5 (TDI), PC3(TMS) and PC2(TCK) will be started paying little heed to whether a reset happens. Port C in like manner serves the components of the JTAG interface.

Port D (PD7..PD0) - Port D is a 8-bit bi-directional I/O port with inner draw up resistors (chose for each piece). The Port D yield cradles have symmetrical drive qualities with both high sink and source capacity. As information sources, Port D sticks that are remotely pulled low will source current if the draw up resistors are actuated. The Port D pins are tri-expressed when a reset condition ends up dynamic, regardless of whether the clock isn't running. Port D likewise serves the elements of different extraordinary highlights of the ATmega16**.**

RESET - Reset Input. A low level on this pin for longer than the minimum pulse length will produce a reset, regardless of whether the clock isn't 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 stick for Port A and the A/D Converter. It ought to be remotely associated with VCC, regardless of whether the ADC isn't utilized. In the event that the ADC is utilized, it ought to be associated with VCC through a low-pass channel.

AREF -AREF is the analog reference pin for the A/D Converter.

2.3.2 L293D MOTOR DRIVER IC

It goes about as double H-bridge engine driver IC. Engine drivers are current speakers with a low-current control flag and create a higher-current flag. This higher current flag is critical to drive the engines.

L293D has two inbuilt H-bridge circuits. In its normal mode, two DC motors can be driven together, in forward and reverse direction both. The operations of two motors can be controlled. By input logic at pins 2, 7 and 10, 15. Input logic 00 or 11 will stop the motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise.

Empower pins 1 and 9 (relating to the two engines) ought to be high for engines to begin. At the point when an enable input is high. Associated driver gets empowered. Coming about the yields wind up dynamic and work in stage with their sources of info. Same occurs with the empower input is low, that driver is incapacitated.

Its working is based on the concept of H-bridge. H-bridge allows the voltage to flow in both direction. As we know voltage requires to change its direction to be able to rotate the motor in both directions, Hence H-bridge IC are ideal for running a DC motor. In a single l293d chip there two h-Bridge circuit in the IC which can rotate two dc motor independently.

L293D has four input pins, pin 2, 7 to the left and pin 15, 10 to the right as shown. Left input pins will alter the rotation of motor linked throughout left facet and proper enter for motor on the proper hand side. The motors are turned around on the groundwork of the inputs provided throughout the input pins as logic 0 or logic 1.

VCC is the voltage that it requirements for its own internal operation 5v; L293D won't utilize this voltage for driving the engine. For driving the engines it has a different arrangement to give engine supply VSS (V supply). L293d will utilize this to drive the engine. It implies on the off chance that you need to work an engine at 9V then we have to give a Supply of 9V crosswise over VSS Motor supply

The max voltage for VSS supply is 36V. It can supply a max current of 600mA/ channel because it can drive motors Up to 36V that’s why you can drive big motors with this L293D. Motor driver is a current amplifier who carries a low-current signal from the microcontroller and output as a higher current signal who will control and drive a motor.

There are 16 pins sticking out of this IC and functionality of each pin before implementing this in a circuit is as follows:

1. Pin 1 and Pin 9 are "enable" pins. They ought to be associated with +5V for the drivers to work. In the event that they pulled low (GND), at that point the outputs will be off paying little heed to the information states, ceasing the engines. On the off chance that you have two extra sticks in your microcontroller, interface these pins to the microcontroller, or simply associate them to controlled positive 5 Volts.
2. Pin 4, 5, 12, 13 are ground pins which should be connected to the ground of the microcontroller.
3. Pin 2, 7, 10, 15 are logic input pins. These are control pins and should be connected to microcontroller pins. Pin2 & Pin7 control the first motor; Pin 10 & Pin 15 control the second motor (right).
4. Pin 3, 6, 11, 14 are output pins. Pin 3 and Pin 6 to the first motor, Pin 11 and Pin 14 to second motor
5. Pin 16 empowers the IC and it would be connected to +5Volts regulated.

A quadruple half H-bridge bidirectional motor driver IC that can drive current of up to 600mA with voltage range of 4.5 to 36 volts. They have separate bridge enable option and are suitable to drive small DC-Geared motors, bipolar stepper motor etc. The specifications are as follows:

- Supply Voltage: 4.5V to 36V
- Output current capability per driver: 600mA
- Pulsed Current: 1.2A Per Driver
- Package: 16-pin PDIP

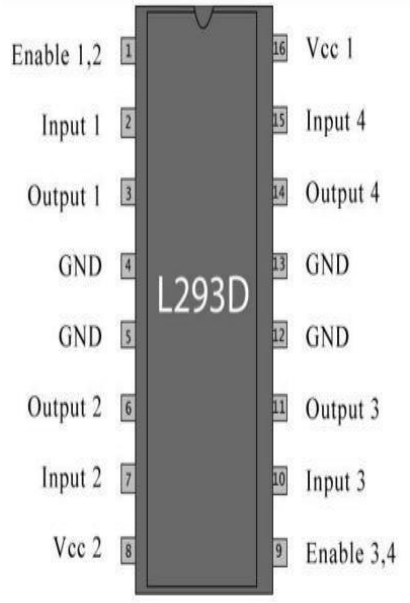


Figure 2.2: Pinout of L293D [20]

Table 2.1: L293D Pin Functions [20]

Pin No	Function	Name
1	Enable pin 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 Motors; 9-12V	Vcc_2
9	Enable pin 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	Input2 for Motor 1	Input 4
16	Supply voltage; 5V	Vcc

2.3.3 INFRARED (IR) SENSOR

IR sensor is a type of active proximity sensors. It emits near infrared energy and measures whether any significant amount of IR light is returned. Infrared radiation is part of the electromagnetic spectrum, which includes radiowaves, microwaves, visible light, and ultraviolet light, as well as gamma rays and X-rays. The IR range falls between the visible portion of the spectrum and the radio waves. IR wavelengths are usually expressed in microns, with IR spectrum extending from 0.7 to 1000 microns. Because every object (except black body) reflects an optimum amount of IR energy at a specific point along IR band, the reflected energy comes from an object and reaches the IR sensor through its optical system, which focuses the energy onto one or more photosensitive detectors. The detector then converts the IR energy into an electrical signal.

Infrared Transmitter is a light emitting diode (LED) which produces infrared radiations. Henceforth, they are called IR LED's. Despite the fact that an IR LED resembles an ordinary LED, the radiation produced by it is undetectable to the human eye.

Infrared receiver are likewise called as infrared sensors as they identify the radiation from an IR transmitter. IR recipients come as photodiodes and phototransistors. Infrared Photodiodes are not the same as expected photograph diodes as they identify just infrared radiation Diverse kinds of IR exist in light of the wavelength, voltage, bundle, and so on. At the point when utilized as a part of an infrared transmitter – beneficiary mix, the wavelength of the recipient should coordinate with that of the transmitter.

Object Detection using IR light:

The basic concept is to infrared light through IR-LEDs, which is then reflected by any object in front of the sensor.

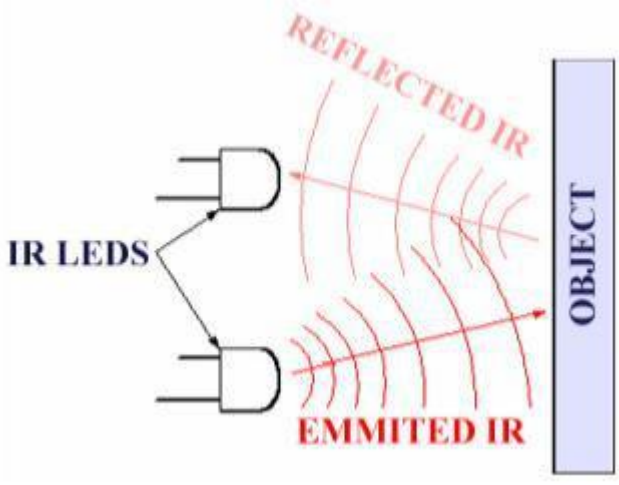


Figure 2.3: IR sensor

2.3.4 LCD DISPLAY

LCD (Liquid Crystal Display) screen is an electronic show module and locate an extensive variety of utilizations. A 16x2 LCD show is exceptionally fundamental module and is generally utilized as a part of different gadgets and circuits. These modules are favoured more than seven portions and other multi fragment LEDs. The reasons being: LCDs are efficient; effectively programmable; have no impediment of showing extraordinary and even custom characters (not at all like in seven sections).

A 16x2 LCD implies it can show 16 characters for each line and there are 2 such lines. In this LCD each character is shown in 5x7 pixel lattice. This LCD has two registers, to be specific, Command and Data.

The charge enlist stores the order directions given to the LCD. An order is a guideline given to LCD to complete a predefined assignment like instating it, clearing its screen, setting the cursor position, controlling presentation and so forth. The information enlist stores the information to be shown on the LCD. The information is the ASCII estimation of the character to be shown on the LCD. Snap to take in more about inner structure of a LCD. 8-Data pins carries 8-bit data or command from an external unit such as microcontroller. All the LCD's performs the same functions (display characters numbers special characters ASCII characters etc).

All LCDs have:

Eight(8) Data pins,VCC (Apply 5v here),GND (Ground this pin),RS (Register select),RW (read - write),EN (Enable),V0 (Set LCD contrast)

V0 (Set LCD contrast)
Set LCD contrast here. Best way is to use variable resistor such as potentiometer. Output of the potentiometer is connected to this pin. Rotate the potentiometer knob forward and backward to adjust the LCD contrast.

RS(Register select)
There are two registers in every LCD

Command Register:

When we send commands to LCD these commands go to Command register and are processed there.

When RS=0 Command Register is selected.

Data Register:

When we send Data to LCD it goes to data register and is processed there.

When RS=1 Data Register is selected.

RW(Read - Write)

When RW=1, we want to read data from LCD.

When RW=0, we want to write to LCD.

EN(Enable signal) When you select the register(Command and Data) and set RW(read - write) now its time to execute the instruction. By instruction it means that 8-bit data or 8-bit command present on Data lines of LCD.
This requires an extra voltage push to execute the instruction and EN (enable) signal is used for this purpose. Usually we make it EN=0 and when we want to exectue the instruction we make it high EN=1 for some milli seconds. After this we again make it ground EN=0.

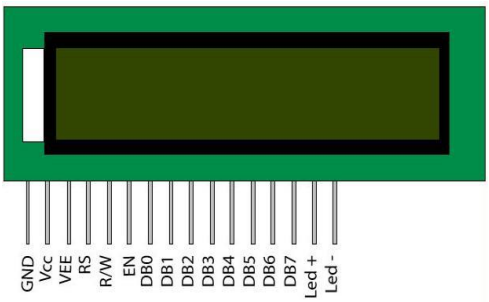


Figure 2.4: LCD Display [19]

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

Table 2.2: LCD Functions [19]

2.3.5 DC GEAR MOTOR

The DC Gear motor, comprising of a DC electric powered motor and a gearbox, is at the coronary heart of quite a few electrical and electronic applications.

The two sorts of motors that we are probably to use in robotic journey are DC motors and RC servo motors. The most frequent motor for robotics is DC equipment motor, which works with the aid of gearing down a quickly DC motor to make the motor flip at a slower speed and gives the motor a greater torque appropriate for robot locomotion.

A dc tools motor is basically a normal dc motor with an exclusive tools box attached to the output shaft. Your robot electrical drive circuitry can control the dc gear motor to rotate the wheels of your robotic for locomotion.
You can get a dc motor without a gear head, but typically these are too fast(around 15,000 RPM).For a robot to move at realistic price you have to gear down a DC motor to about 30 to eighty RPM. When you gear down a DC motor, you get a slower pace and lots of torque.

The outer body of the tools head is made of high density plastic however it is quite handy to open as only screws are used to connect the outer and the internal structure. The important purpose in the back of this could be to lubricate tools head from time to time. The plastic body has a threading through which nut can be easily established and vice versa from the gearhead. The rear view of the geared motor is comparable to the DC motor and it has two wires soldered to it.

These tools assemblies are exceedingly lubricated with grease so as to keep away from any sort of wear and tear due to frictional forces. The top phase of the gear head is linked to rotating shaft and has one equipment that allows the rotation. A sturdy round imprint shows the presence of the gear that rotates the gear at the higher portion.

The gears are essentially in shape of a small sprocket however in view that they are now not connected via a chain, they can be termed as duplex gears in phrases of a second cog arrangement coaxially over the base. Among the three gears, two

are exactly identical whilst the 1/3 one is greater in terms of the number of teeth at the higher layer of the duplex gear. The third equipment is connected to the tools at the higher portion of the gear head.

2.3.6 4X3 KEYPAD

A simple 12 button keypad for consumer input. The buttons are setup in a matrix format. This allows a microcontroller to 'scan' the 7 output pins to see which of the 12 buttons is being pressed.

A Matrix keypad is the most generally used input gadget in many of the application areas like digital circuits, phone communications, calculators, ATMs, and so on. A matrix keypad consists of a set of push button or switches which are arranged in a matrix structure of rows and columns. These keypads are accessible in configurations like 3×4 and 4×4 based on the utility it is applied.

Matrix keypad can be connected to the microcontroller in numerous ways or techniques, but the fundamental logic is same as making the columns as input and the rows as output. So, in order to detect the key pressed from the keypad, the row lines have to be made low one by one and to read the columns.

Table 2.3: Corresponding row and Column for each key [21]

Data Output				Data Input			Key Pressed
R1	R2	R3	R4	C1	C2	C3	
0	1	1	1	0	1	1	1
0	1	1	1	1	0	1	2
0	1	1	1	1	1	0	3
1	0	1	1	0	1	1	4
1	0	1	1	1	0	1	5
1	0	1	1	1	1	0	6
1	1	0	1	0	1	1	7
1	1	0	1	1	0	1	8
1	1	0	1	1	1	0	9
1	1	1	0	0	1	1	*
1	1	1	0	1	0	1	0
1	1	1	0	1	1	0	#

2.3.7 AVR PROGRAMMER

AVR Programmer is used to burn code written on AVR Studio or CV AVR into the microcontroller. When we build our code in AVR Studio or CV AVR, it is converted into a hex file. This hex file is burned from computer to hardware microcontroller using AVR Programmer.

Features of USBasp:

- Operates on multiple platforms for example Linux, Mac OS X and Windows.
- No special other controllers, components or equipments required.
- Planned: “Serial Interface to Target” (e.g. for debugging).
- 10 pin ISP interface.
- Permits you to write or read the microcontroller EEPROM, firmware, fuse bits and lock bits etc.

These AVR programmers are based on Thomas Fischl's USBasp design and connect to your computer's USB port.

To transfer the Hex file to controller we require software to access USBASP. At the first time when we connect programmer to the pc our programmer will be detected as usbasp and we have to provide a proper path for drivers to be installed.

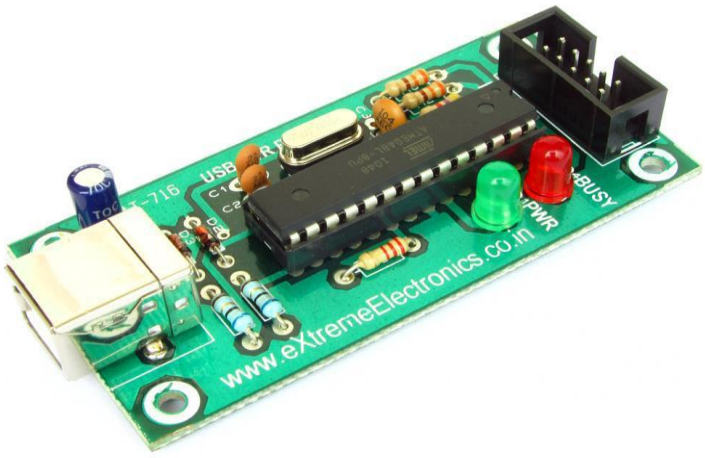


Figure 2.6: AVR Programmer

CHAPTER 3

DESCRIPTION OF WORK DONE

We have divided our project into three phases i.e. conceptualization and planning phase, software implementation phase and hardware implementation phase. In first phase we have planned the layout of our car automated car parking area. In second phase we have implemented our whole project in software. For software implementation we are using Atmel AVR Studio (for writing code in embedded C) and ISIS professional proteus (for circuit simulation). And finally in fourth phase we have successful implemented our project work on hardware.

Timeline of project:

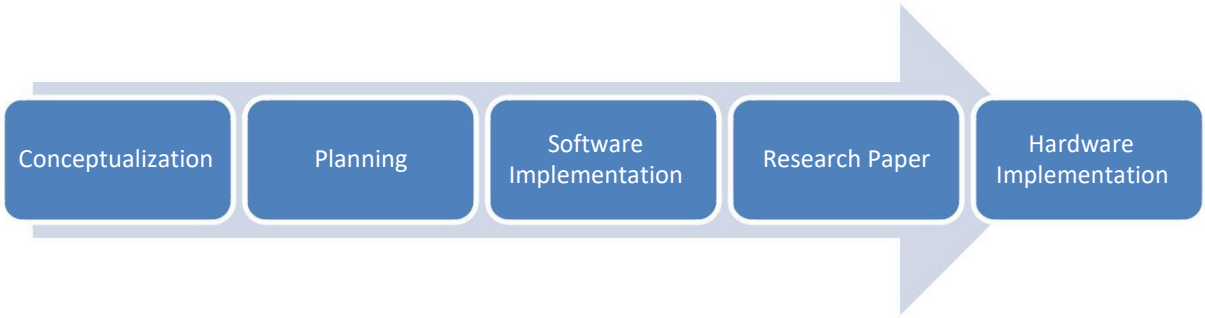


Figure 3.1: Project Timeline

Our project is divided into two major parts. First part-to build a smart and automated parking lot and the Second part-to design the smart and automated car. The first step towards our work was to design a parking area keeping in mind the proper movements of car like turn-around gaps and reversing space etc. Keeping all these things in mind the parking area was aesthetically designed where there is sufficient space for parking, turning, reversing and the parking entry point. Minimization of space has been done in such a way that there should be no glitches in functioning and safety of parked cars.

Next step was to select the primary key of car which we found that it had to be the car registration number through which the car will be registered in the parking. We also had to make the registration convenient so we used the LCD Display for the output and a simple 4X3 Keypad for Input. Both these components united with the microcontroller together made

Entry Gate area most simplified and convenient for this area. Management of slot numbers allocated is also done in this segment by code.

Finally carving a way out for the car to be parked at specified slot number was done. The principle used was to follow a black trajectory and indication marks which were sensed by the IR sensors for the making a way for parking of car. The marks signify indications to skip, turn and stop which is mutually coordinated with the slot number allotted for car parking. In this way we automated the car as well as parking system.

3.1 SOFTWARE IMPLEMENTATION

To implement our project on software grounds we used AVR Studio 4 and Proteus 7.8. Here we simulated our automated car parking model which gave us the desired results.

3.1.1 SMART PARKING AREA/LOT:

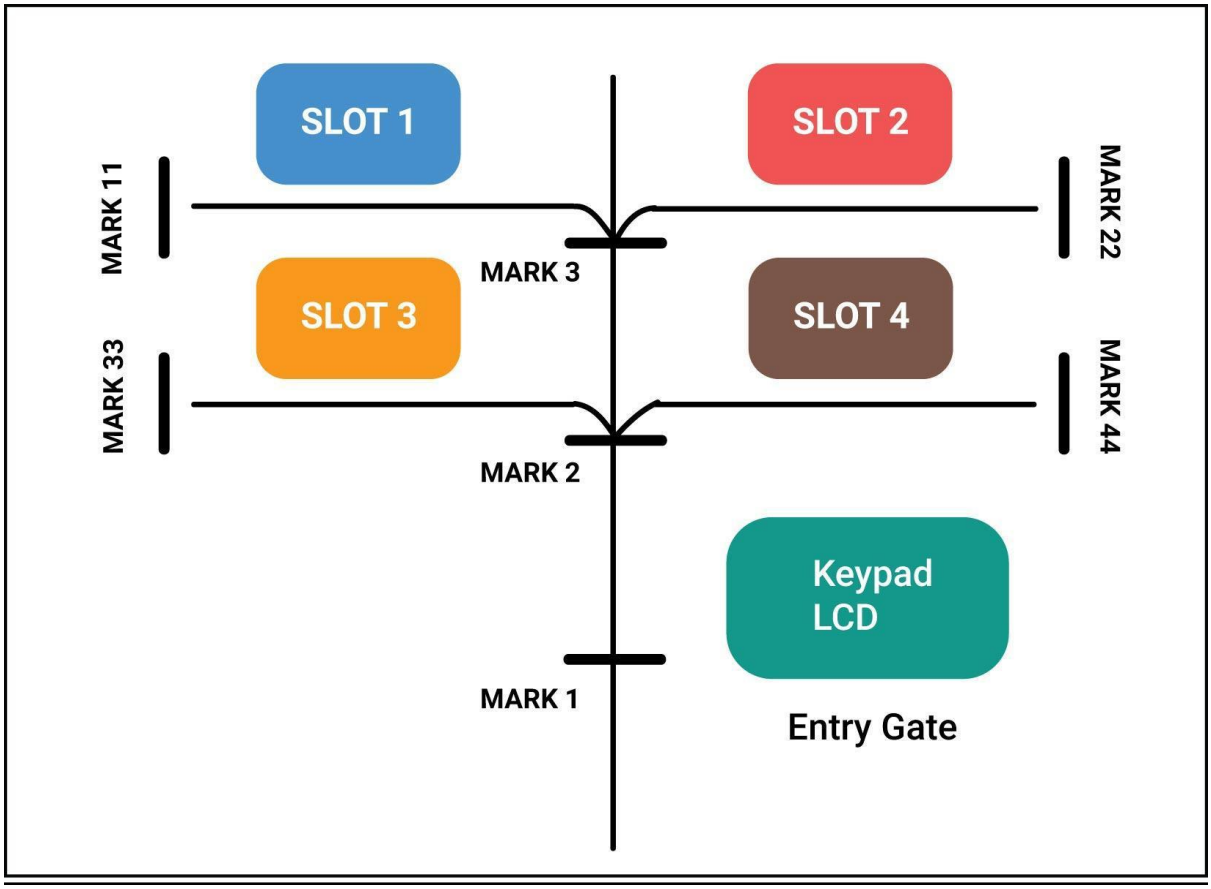


Figure 3.2: Layout of 'smart parking area/lot'

Description of "Smart Parking Area":

Figure 3.2 illustrates layout of our smart parking lot. We have designed our parking area for four cars but we can extend our layout further for as many number of cars as required keeping the basic concept same. When the car comes to the parking area, it will reach the entrance of parking lot which is represented by Mark 1. Here the person driving the car comes out of his/her car and types in his car registration no. as the unique identification of his car. As he enters his car number, an available slot will be allocated to his car. Now the driver has to just enter the allocated slot number into his car and then he can leave from the parking area. With the help of set of four sensors embedded on the car, the car will move further over black line and will be parked onto the allocated slot. For example if the car has to be parked at slot number 1, then the car will move from Mark 1 to Mark 3, then it will rotate by 90 degree towards left and finally it will move further till Mark 11. As it reaches Mark 11, the controller will come to know that the car is successfully parked at slot 1.

ALGORITHM:

START

Step1: Car comes to Smart parking area.

Step 2: Car stops at entry gate (i.e. Mark 1).

Step 3: Car driver enters details like his phone number and car registration number on the keypad setup at the entrance.

Step 4: Controller of parking lot saves details and allocates an empty slot to the car which is displayed on LCD screen (if no slot available "parking area full!" is displayed).

Step 5: Driver enters the allocated slot on keypad embedded on smart car.

Step 6: Smart car moves inside parking area and a clock timer will start for that car.

Step 7: If allocated slot is 3, car moves forward till 'Mark 2', after which it will take a left turn and again move forward till 'Mark 33' is encountered.

Step 8: If allocated slot is 4, car moves forward till 'Mark 2', after which it will take a right turn and again move forward till 'Mark 44' is encountered.

Step 9: If allocated slot is 1, car moves forward till 'Mark 3', after which it will take a left turn and again move forward till 'Mark 11' is encountered.

Step 10: If allocated slot is 2, car moves forward till 'Mark 3', after which it will take a right turn and again move forward till 'Mark 22' is encountered.

Step 11: Once car is parked, a success message is send to the car owner/driver with an OTP.

Step 12: When the car driver comes back to his car, he will entry car number and valid OTP at exit gate, and his car will be automatically be de-parked from parking area to the entry gate.

Step 13: At this moment clock timer will stop and the time period will be recorded. Using this time interval, parking fees will be calculated and the car driver can take the car back after paying the fees.

STOP

FLOW OF CONTROL:

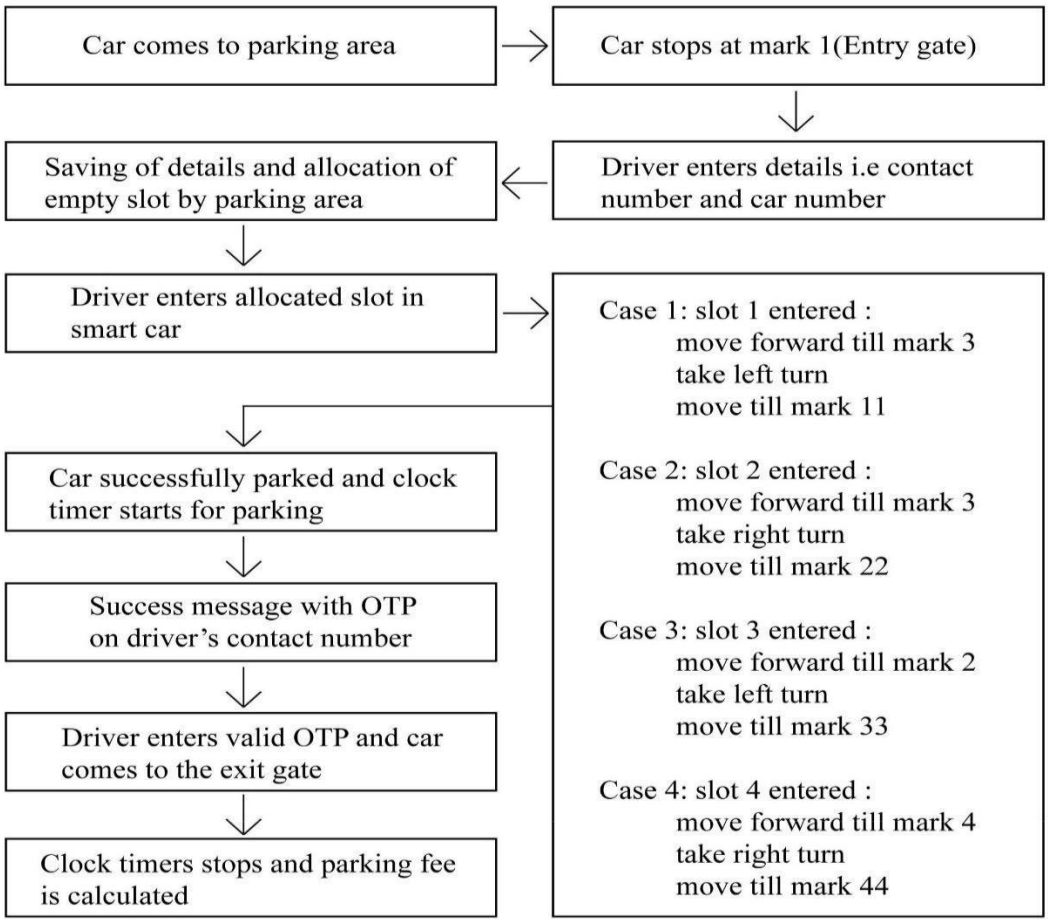


Figure 3.3: Flow Control of 'smart parking lot'

PROGRAM:

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

```
voidlcdcmd(unsigned char a)
{PORTD=a;PORTC=0b00000100;
        _delay_ms(100);
        PORTC=0b00000000;
        _delay_ms(100);
}
voidlcddata(unsigned char a)
{PORTD=a; PORTC=0b00000101;
        _delay_ms(100);
        PORTC=0b00000001;
        _delay_ms(100);
}
void refresh()
{               lcdcmd(0x01);lcdcmd(0x80);
                lcddata('E');lcddata('n');lcddata('t');lcddata('e');lcddata('r');lcddata(' ');
                lcddata('c');lcddata('a');lcddata('r');lcddata(' ');
                lcddata('n');lcddata('o');lcddata('.');lcddata(':');lcddata(' ');
}
void warning()
{               lcdcmd(0x01);lcdcmd(0x80);
                lcddata('I');lcddata('n');lcddata('v');lcddata('a');lcddata('l');lcddata('i');
                lcddata('d');lcddata(' ');
                lcddata('c');lcddata('a');lcddata('r');lcddata(' ');
                lcddata('n');lcddata('o');lcddata('.');lcddata('.');lcddata('.');
                _delay_ms(500);
                refresh();
}
voidparkingfull()
{               lcdcmd(0x01);lcdcmd(0x80);
                lcddata('P');lcddata('a');lcddata('r');lcddata('k');lcddata('i');lcddata('n');
                lcddata('g');lcddata(' ');
                lcddata('i');lcddata('s');lcddata(' ');
                lcddata('f');lcddata('u');lcddata('l');lcddata('l');lcddata('!');

}
void success(unsigned char a)
{               lcdcmd(0x01);lcdcmd(0x80);
                lcddata('S');lcddata('l');lcddata('o');lcddata('t');//lcddata(':');
                lcddata(a);
                lcddata(' ');
                lcddata('a');lcddata('l');lcddata('l');lcddata('o');lcddata('t');lcddata('e');
                lcddata('d');
                _delay_ms(5000);
}
void main()
{intcarcount=0;int a[4][4];int count=0,i=0,j=0;
DDRB = 0x00; PORTB = 0x0F; DDRA=0x0F; PORTA=0X00; DDRC=0xFF; DDRD=0xFF;
lcdcmd(0x38);           // Initialize LCD
lcdcmd(0x0E); lcdcmd(0x01);lcdcmd(0x80);
        refresh();
```

```
        for(i=0;i<4;i++){
                for(j=0;j<4;j++){
                a[i][j]=0;
                }}
while (1) //loop key check forever
  { if(carcount<4){
//first column
      PORTA = 0b00001110;
      _delay_ms(50);
       if (!(PINB & 0x01)){ lcddata('1');a[carcount][count]=1;count++;}
       if (!(PINB & 0x02)){lcddata('4');a[carcount][count]=4;count++;}
       if (!(PINB & 0x04)){lcddata('7');a[carcount][count]=7;count++;}
if (!(PINB & 0x08)) {refresh();
                for(j=0;j<4;j++){
                a[carcount][j]=0;}
                count=0;}
      //second column PORTA
       =0b00001101 ;
     _delay_ms(50);
       if (!(PINB & 0x01)){lcddata('2');a[carcount][count]=2;count++;}
       if (!(PINB & 0x02)){lcddata('5');a[carcount][count]=5;count++;}
       if (!(PINB & 0x04)){lcddata('8');a[carcount][count]=8;count++;}
       if (!(PINB & 0x08)){lcddata('0');a[carcount][count]=0;count++;}
       //third column
       PORTA = 0b00001011;
       _delay_ms(50);
       if (!(PINB & 0x01)){lcddata('3');a[carcount][count]=3;count++;}
       if (!(PINB & 0x02)){lcddata('6');a[carcount][count]=6;count++;}
       if (!(PINB & 0x04)){lcddata('9');a[carcount][count]=9;count++;}
if (!(PINB & 0x08)){//lcddata('#');
                if(count>4||count<3){
                        warning();
                        count=0;
                        }
                        else{
                        carcount++;
                        success(carcount+48);
                        refresh();
                        }
        }
}
        else{
                parkingfull();
                }
        }
}
```

PROTEUS SIMULATION:

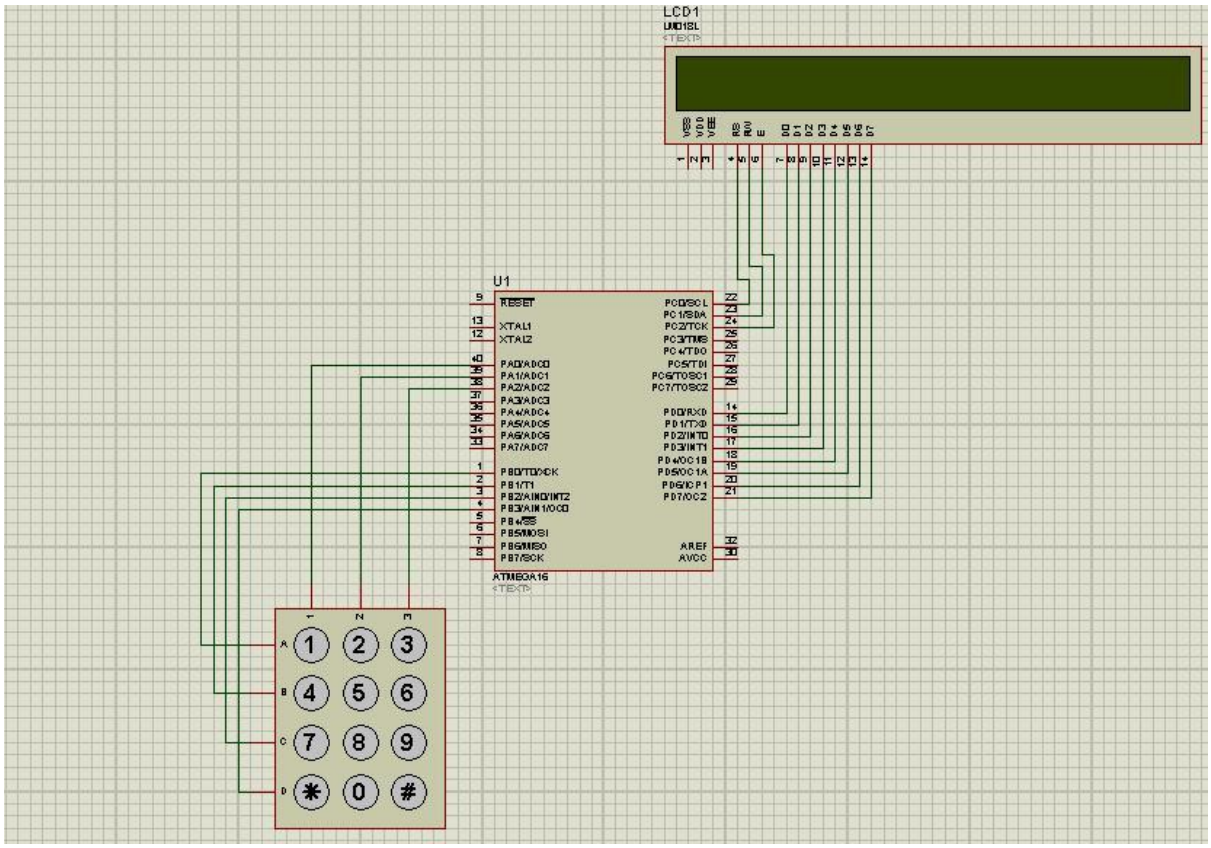


Figure 3.4: Proteus simulation of 'smart parking area'

Working of Smart Parking Area:

Above figure shows proteus simulation of smart parking area. It consists of three components i.e. AVR Microcontroller, 4X3 Keypad and LCD. When the car arrives at entry gate, the LCD initially displays "Enter Car Number". Using keypad the car owner will enter his car's unique registration no. and a vacant slot will be allocated to him which will be displayed on the LCD. If parking area is full, then a warning message will be displayed on LCD "Parking is full!" While entering car registration number if user makes any mistake then he can rest the LCD and re-enter his car registration number by pressing "*". Also once the user has entered his car number, he has to press "#" to get empty slot from the controller.

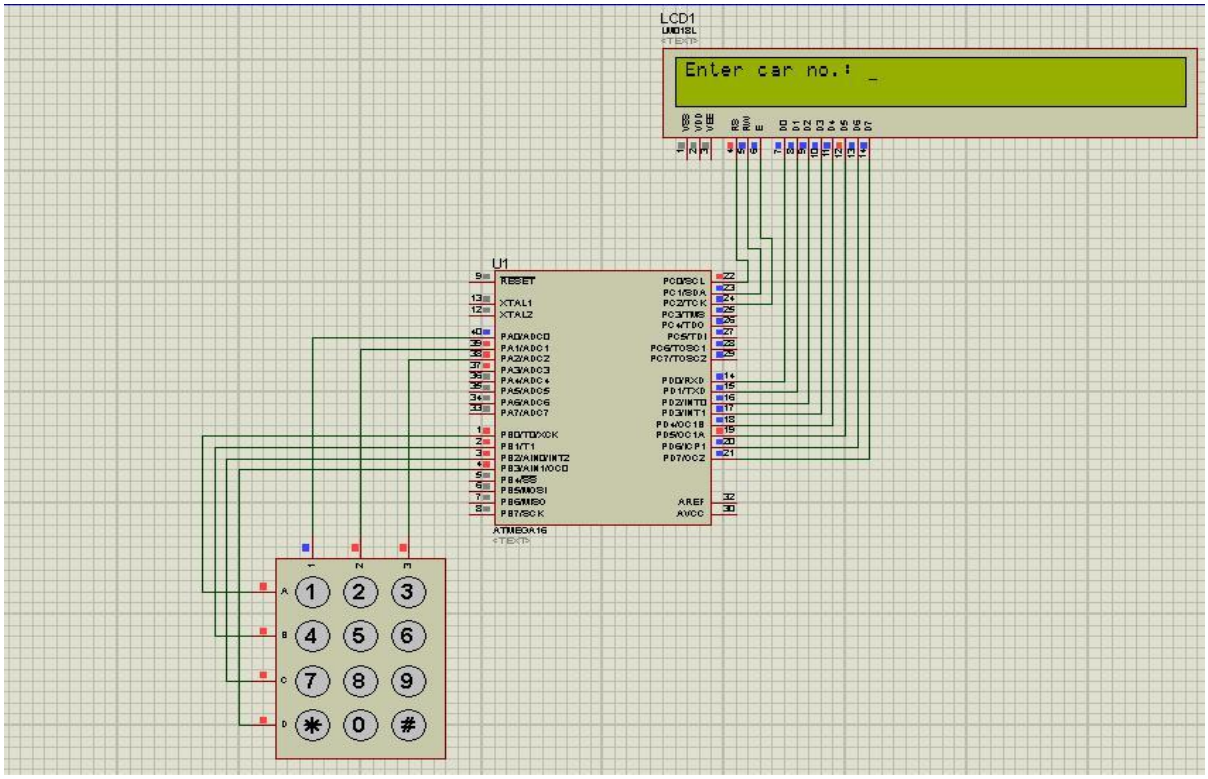


Figure 3.5: Proteus simulation of smart parking area-asking to enter car no.

Figure 3.5 shows the LCD displaying “Enter car no.:” at entry gate of parking area/lot.

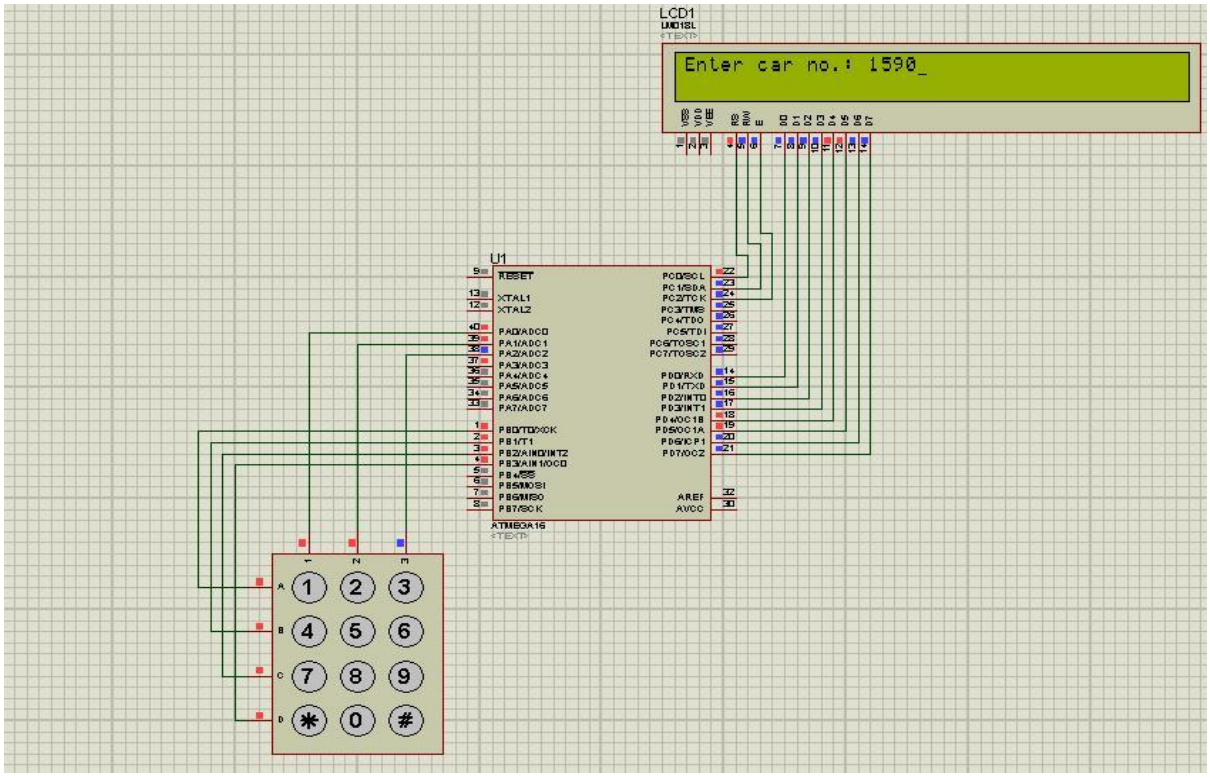


Figure 3.6: Proteus simulation of smart parking area/lot-user entering his car no.

Figure 3.6 shows the user entering his car registration number at entry gate using keypad.

In this case user has entered 1590 as his car registration number.

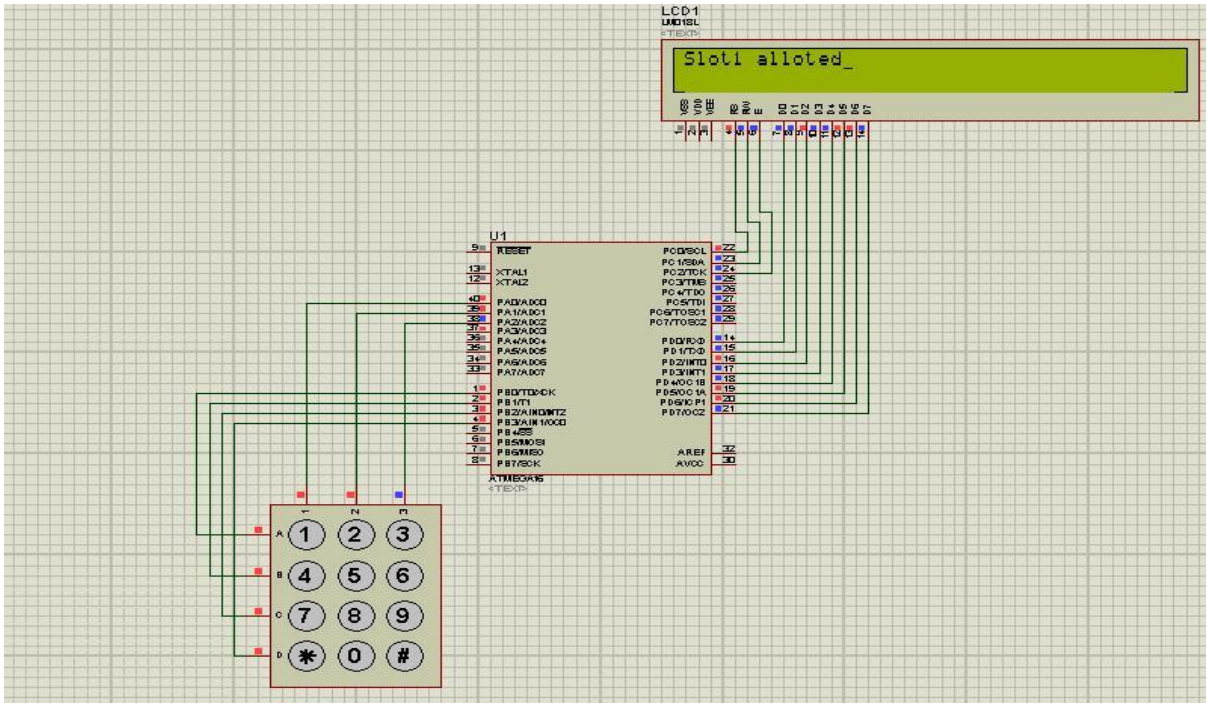


Figure 3.7: Proteus simulation of smart parking area-slot alloted

Figure 3.7 shows the allotment of an empty slot once the user press "#" from keypad.

In this case slot 1 has been allocated to the user for parking of his car.

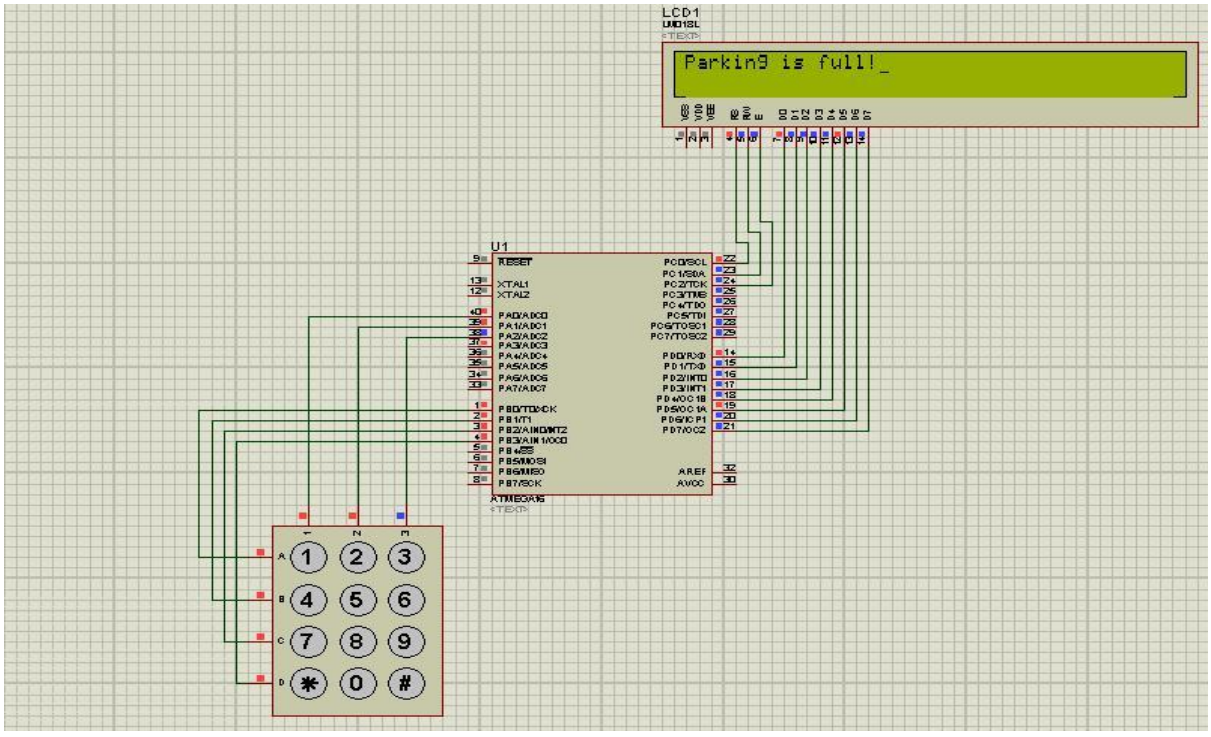


Figure 3.8: Proteus simulation of smart parking area-parking full message displayed

3.1.2 SMART CAR

Description of "Smart Car":

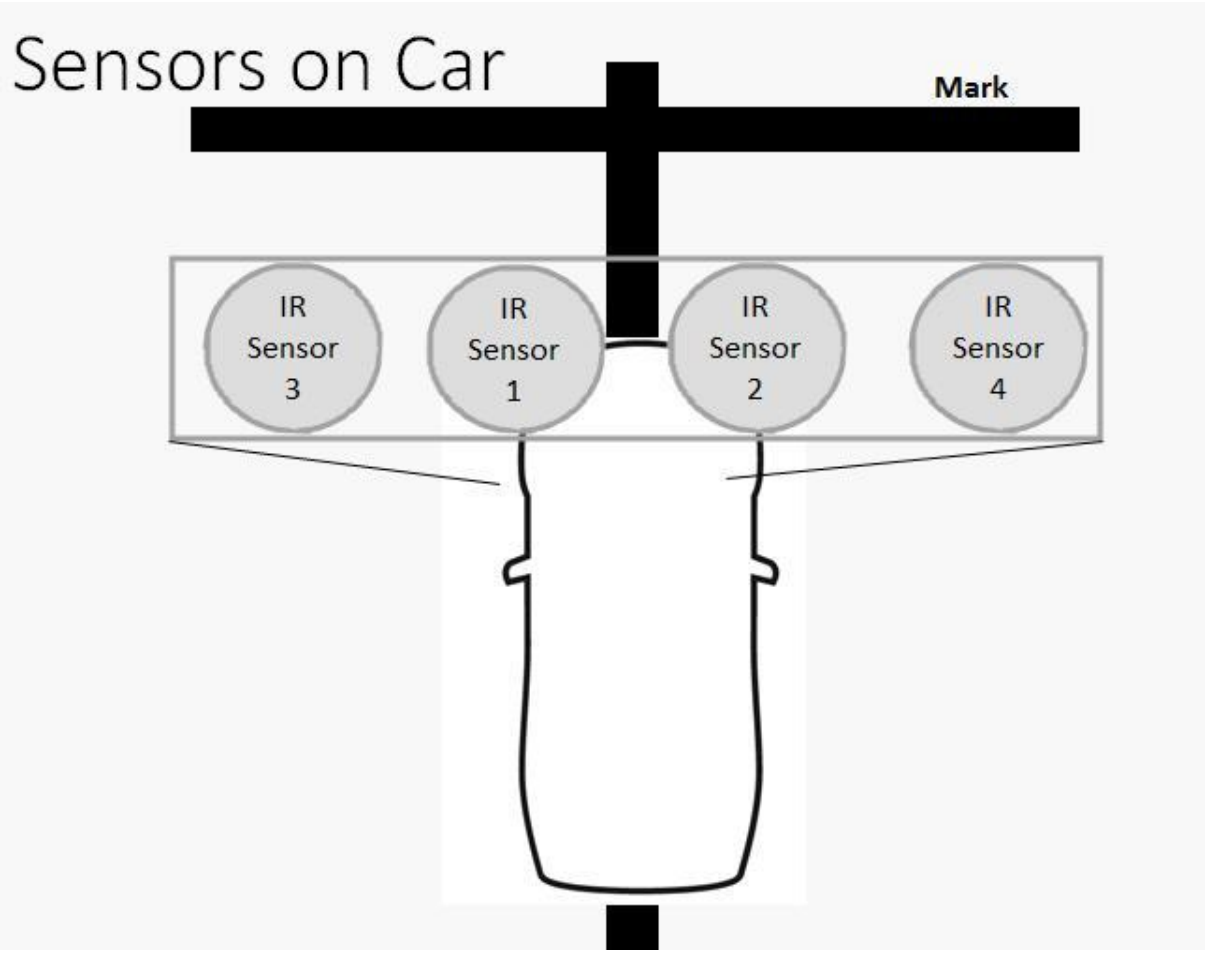


Figure 3.9: Design of "Smart Car"

Figure 3.9 illustrates the design of Smart Car. Our smart car is similar to any other car with the only difference that it has a set of four IR Sensors embedded at its bumper. The inner two sensors are used to control the car so that the car always moves over the black path while the outer two sensors are used to detect any "Mark" being present at the parking area.

ALGORITHM:

START

Step1: Car moving over black track, all four sensors are ON.

Step 2: If Sensor 1 goes OFF, that means car has lost the track and take a move towards right, so to bring it back on track turn OFF the right motor till Sensor 1 goes ON again.

Step 3: If Sensor 2 goes OFF, that means car has lost the track and take a move towards left, so to bring it back on track turn OFF the left motor till Sensor 2 goes ON again.

Step 4: If Sensor 3 and Sensor 4 both turns OFF, that means a 'Mark' is encountered. Depending upon the Mark number and the allocated slot number process the movement of car and park it accordingly.

Step 5: Once 'Mark 11/22/33/44' is encountered, it means that car is successfully parked so send a success message to car owner/driver.

STOP

FLOW OF CONTROL:

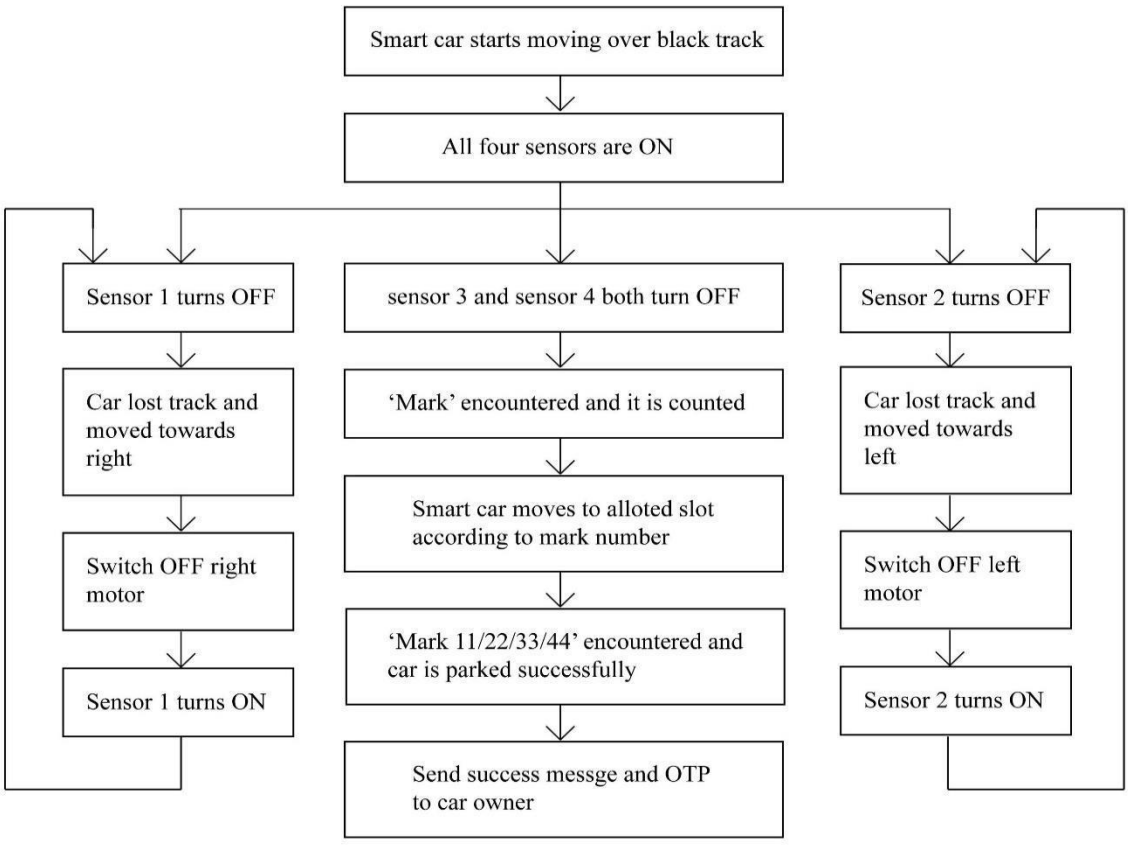


Figure 3.10: Flow of Control of 'automated car'

PROGRAM:

```
#include<avr/io.h>              //Header file
#include<util/delay.h>          //Header file
#define sensor1 PA0             //Left Sensor
#define sensor2 PA1             //Right sensor
#define sensor3 PA2
#define sensor4 PA3
voidsw();
void park();
voidparkToSlot();
int mark=0, slotNumber=0, temp_counter=0;

voidlcd_cmd(unsigned char a)
{
        PORTD=0b11110000 & a;
        PORTD|= 0<<0 | 0<<1 | 1<<2;
        _delay_ms(50);
        PORTD&=0b11110000;
        a=a<<4;
        PORTD= 0b11110000 & a;
        PORTD|= 0<<0 | 0<<1 | 1<<2;
        _delay_ms(50);
        PORTD&=0b11110000;
}

voidlcd_data(unsigned char b)
{
        PORTD =0b11110000 & b;
        PORTD|= 1<<0 | 0<<1 | 1<<2;
        _delay_ms(50);
        PORTD&=0b11110001
        b=b<<4;
        PORTD= 0b11110000 & b;
        PORTD|= 1<<0 | 0<<1 | 1<<2;
        _delay_ms(50);
        PORTD&=0b11110001;
}

void refresh()
{
        lcd_cmd(0x01);lcd_cmd(0x80);//lcd_cmd(0x01);lcd_cmd(0x80);
        lcd_data('E');lcd_data('n');lcd_data('t');lcd_data('e');lcd_data('r');lcd_data(' ');
        lcd_data('s');lcd_data('l');lcd_data('o');lcd_data('t');lcd_data(' ');
        lcd_data('n');lcd_data('o');lcd_data('.');lcd_data(':');lcd_data(' ');
}

voiderrorMessage()
{
        lcd_cmd(0x01);lcd_cmd(0x80);
```

```
        lcd_data('I');lcd_data('n');lcd_data('v');lcd_data('a');lcd_data('l');lcd_data('i');
.       lcd_data('d');lcd_data(' ');
        lcd_data('s');lcd_data('l');lcd_data('o');lcd_data('t');lcd_data(' ');
        lcd_data('n');lcd_data('o');lcd_data('.');
        _delay_ms(2000);
}

void display()
{
lcd_cmd(0x01);lcd_cmd(0x80);
        lcd_data('E');lcd_data('n');lcd_data('t');lcd_data('e');lcd_data('r');lcd_data('e');
.       lcd_data('d');lcd_data(' ');
        lcd_data('s');lcd_data('l');lcd_data('o');lcd_data('t');lcd_data(':');lcd_data(' ');
lcd_data(slotNumber+48);
        _delay_ms(1000);
        lcd_cmd(0x01);lcd_cmd(0x80);
        lcd_data('P');lcd_data('r');lcd_data('e');lcd_data('s');lcd_data('s');lcd_data('
        '); lcd_data('#');lcd_data(' ');
        lcd_data('t');lcd_data('o');lcd_data(' ');
        lcd_data('p');lcd_data('a');lcd_data('r');lcd_data('k');
        _delay_ms(1000);
        park();
}

voiddisplay_car_parked()
{
        lcd_cmd(0x01);lcd_cmd(0x80);
        lcd_data('C');lcd_data('a');lcd_data('r');lcd_data(' ');
        lcd_data('p');lcd_data('a');lcd_data('r');lcd_data('k');lcd_data('e');lcd_data('d');
        lcd_data('!');
        _delay_ms(1000);
}

int main(void)
{
DDRB=0xFF;
DDRA=0xF0;
DDRC=0b00000111;
PORTC=0b11110000;
DDRD =0b11111111;
lcd_cmd(0x02); //IMPORTANT
lcd_cmd(0x28); lcd_cmd(0x0E); lcd_cmd(0x01); lcd_cmd(0x80);
refresh();

while(1)
{PINA=0x0F;
}
}
void sw1()
{
```

```
        if(bit_is_clear(PINA,sensor2))
        PORTB=0b00000000;
        else
        PORTB=0b00000100;
}
void sw2()
{
        if(bit_is_clear(PINA,sensor2))
        PORTB=0b00000001;
        else
        PORTB=0b00000101;
}
voidsw()
{
        if(bit_is_clear(PINA,sensor1))
        sw1();
        else
        sw2();
        if(bit_is_clear(PINA,sensor3) &&bit_is_clear(PINA,sensor4))
        {if(slotNumber==0)
                {park();
                }
                if(slotNumber!=0)
                {parkToSlot();
                }
        } // if(bit_is_clear(PINA,sensor3)&&bit_is_clear(PINA,sensor4))
}

void park() // to stop the car, take slot no. & enter it into car
keypad {
        if(mark==0 &&bit_is_clear(PINA,sensor3) &&bit_is_clear(PINA,sensor4)
        &&slotNumber==0) //Then stop the car for allocation of slot no. &entrying slot no. in
        car
        {PORTB=0b00000000; // stop the car
        }
        PORTC=0b00000110;
        _delay_ms(50);
        if(!(PINC  &  0x10)){lcd_data('1');slotNumber=1;}
        if(!(PINC & 0x20)){lcd_data('4');slotNumber=4;}
        if(!(PINC & 0x40)){lcd_data('7');slotNumber=-1;} // We are making project for 4
        slots so if slot no. entered is greater than 4, make it -1 & display error afterwards.
        if(!(PINC & 0x80)){refresh();slotNumber=0;} // If user enter *, then clear or refresh
        the screen.

        PORTC=0b00000101;
        _delay_ms(50);
        if(!(PINC & 0x10)){lcd_data('2');slotNumber=2;}
        if(!(PINC & 0x20)){lcd_data('5');slotNumber=-1;}
        if(!(PINC & 0x40)){lcd_data('8');slotNumber=-1;}
        if(!(PINC & 0x80)){lcd_data('0');slotNumber=-1;}
```

```
	PORTC=0b00000011;
	_delay_ms(50);
	if(!(PINC & 0x10)){lcd_data('3');slotNumber=3;}
	if(!(PINC & 0x20)){lcd_data('6');slotNumber=-1;}
	if(!(PINC & 0x40)){lcd_data('9');slotNumber==-1;}
	if(!(PINC & 0x80)){parkToSlot();} // If user enter #, then call the function to
	move the CAR to the slot

	if(slotNumber==-1)
	{
	slotNumber=0;
	errorMessage();
	refresh();
	}
	if(slotNumber>=1 &&slotNumber<=4)
	display();
}// park()
voidparkToSlot() // to park car to the allocated slot
{
	if(mark==0 &&bit_is_clear(PINA,sensor3) &&bit_is_clear(PINA,sensor4)
	&&slotNumber!=0) // i.e. driver has entered the slot
	{mark=1;
		PINA=0x0F;
		sw();//move car forward
	}
	if(mark!=0 &&bit_is_clear(PINA,sensor3) &&bit_is_clear(PINA,sensor4)
	&&slotNumber!=0) // i.e. car has reached second mark
	{
		if(slotNumber==3) // move left i.e. to slot 3
		{mark=2;
			// stop the car & rotate the car by 90 degree anti-clockwise(i.e. towards
			left) and then move forward
			PORTB=0b00000000;
			for(int i=1;i<=100000;i++) { // for left side rotatin stop left motor
			& start right motor
			PORTB=0b00000101;
			}
			PORTB=0b00000000;
			PINA=0x0F;
			mark=33;
			sw();

		}//if(slotNumber==3)

		if(mark==33 &&bit_is_clear(PINA,sensor3) &&bit_is_clear(PINA,sensor4))
		{display_car_parked();
		}

		if(slotNumber==4) // move right i.e. to slot 4
```

```
{mark=2;
        // stop the car & rotate the car by 90 degree clockwise(i.e. towards
        right) and then move forward
        PORTB=0b00000000;
        for(int i=1;i<=100000;i++){// for right side rotatin stop right motor &
        start left motor
        PORTB=0b00000001;
        }
        PORTB=0b00000000;
        PINA=0x0F;
        mark=44;
        sw();

}//if(slotNumber==4)

if(mark==44 &&bit_is_clear(PINA,sensor3) &&bit_is_clear(PINA,sensor4))
{display_car_parked();
}

if(slotNumber==1) // move forward to mark 3 and then to left i.e. to slot 1
{mark=2;
        if(temp_counter==0) // then move forward as it is mark 2
        {temp_counter=1;
                PINA=0x0F;
                sw();
        } //if(temp_counter==0)
        if(temp_counter==1) // then stop as it is mark 3
        {mark=3; // stop the car & rotate the car by 90 degree anti-
                //clockwise(i.e. towards left) and then move forward
                PORTB=0b00000000;
                for(int i=1;i<=100000;i++){// for left side rotatin stop
                left motor & start right motor
                PORTB=0b00000101;
                }
                PORTB=0b00000000;
                PINA=0x0F;
                mark=11;
                sw();
        }//if(temp_counter==1)
}//if(slotNumber==1)

if(mark==11 &&bit_is_clear(PINA,sensor3) &&bit_is_clear(PINA,sensor4))
{display_car_parked();
}

if(slotNumber==2) // move forward to mark 3 and then to right i.e. to slot
2 {
        mark=2;
        if(temp_counter==0) // then move forward as it is mark 2
        {
```

```
                    temp_counter=1;
                    PINA=0x0F;
                    sw();//move car forward (do not stop, rather move forward & then stop
                    on mark 3)
                    } //if(temp_counter==0)

                    if(temp_counter==1) // then stop as it is mark 3
                    {
                    mark=3;
                    // stop the car & rotate the car by 90 degree clockwise(i.e. towards
                    right) and then move forward
                    PORTB=0b00000000;
                    for(int i=1;i<=100000;i++){// for right side rotatin stop right motor &
                    start left motor
                    PORTB=0b00000101;
                    }
                    PORTB=0b00000000;
                    PINA=0x0F;
                    mark=22;
                    sw();
                    }//if(temp_counter==1)
                }//if(slotNumber==2)

                if(mark==22 &&bit_is_clear(PINA,sensor3) &&bit_is_clear(PINA,sensor4))
                {display_car_parked();
                }
        }
}// parkToSlot()
```

PROTEUS SIMULATION:

Figure 3.11 shows the proteus simulation of smart car. It consists of AVR Microcontroller, Keypad, LCD display, L293D Motor Driver, 2 Motors and four switches. These four switches represent four sensors which are embedded on front bumper of car. All these components combines together to form a nonholonomic car which corresponds to an original car.

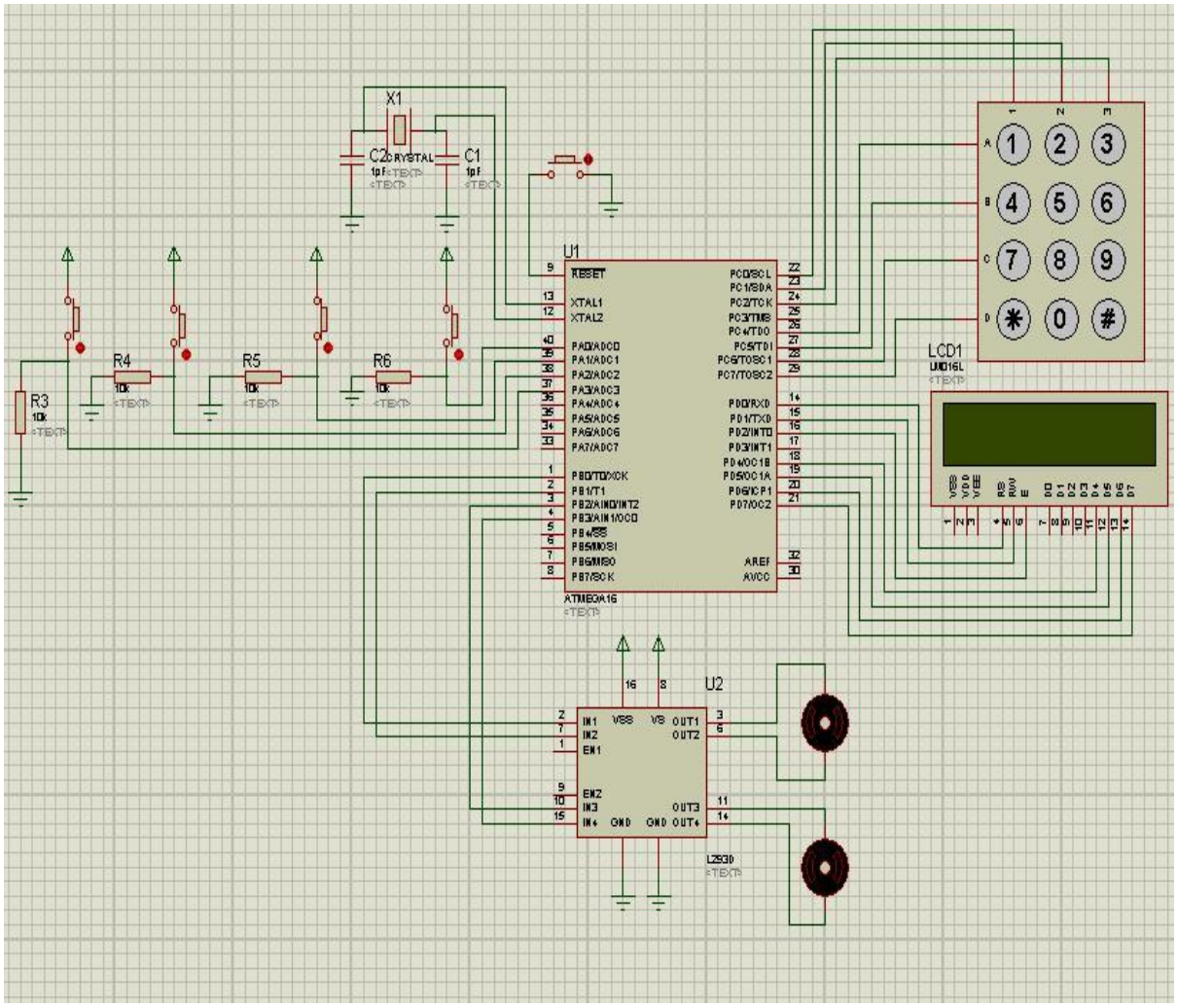


Figure 3.11: Proteus simulation of 'smart car'

Working of "Smart Car":

Since the car will be all over a Black Path, and the two sensors (sensor 1 and 2) will be anticipated simply out of the way where the surface of parking area is light shaded (white surface), so now whenever the car will move around then every one of the four sensors will be ON. In the event of car taking a left, at that point sensor 2nd will consequently be over the dark way, and as dark shading will absorb every single IR radiation of nothing will be detected by beneficiary and sensor 2nd will stop. As sensor 2 gets stopped, code burnt on the microcontroller will control the car to return back to the track i.e. to move somewhat towards right and for this controller will turn OFF right wheel for quite a while and just left sided wheel will pivot, which takes the car towards right. At the point when the car will return over the way, sensor 2 will again turn ON and the two wheels will begin pivoting/moving once more.

Table 3.1: Working of Sensor 1 and Sensor 2 of car

Sensor 1	Sensor 2	Movement of car
OFF	OFF	Is Not Possible
OFF	ON	Car has lost the path and moved towards right, to bring it back on path switch off left motor & switch on right motor
ON	OFF	Car has lost its path and has moved towards left, to bring it back on path switch off right motor & switch on left motor
ON	ON	Car is moving over black path

Sensor third and fourth are to recognize Marks show on parking lot. At the point when car moves both these sensors ON in light of the fact that car will be on the white surface. At whatever point any Mark is encountered, since marks are of dark shading, they will turn OFF both the sensors, i.e. third and fourth.

Table 3.2: Working of Sensor 3 and Sensor 4 of car

Sensor 3	Sensor 4	Movement of car
OFF	OFF	Car has reached to a "Horizontal Mark" (stop at the mark till the next command to move is given)
OFF	ON	Not Possible
ON	OFF	Not Possible
ON	ON	Car is moving over black track

3.3 HARDWARE IMPLEMENATATION

Our next and final step was to implement our idea on hardware so that this automated parking system can be physically realised.

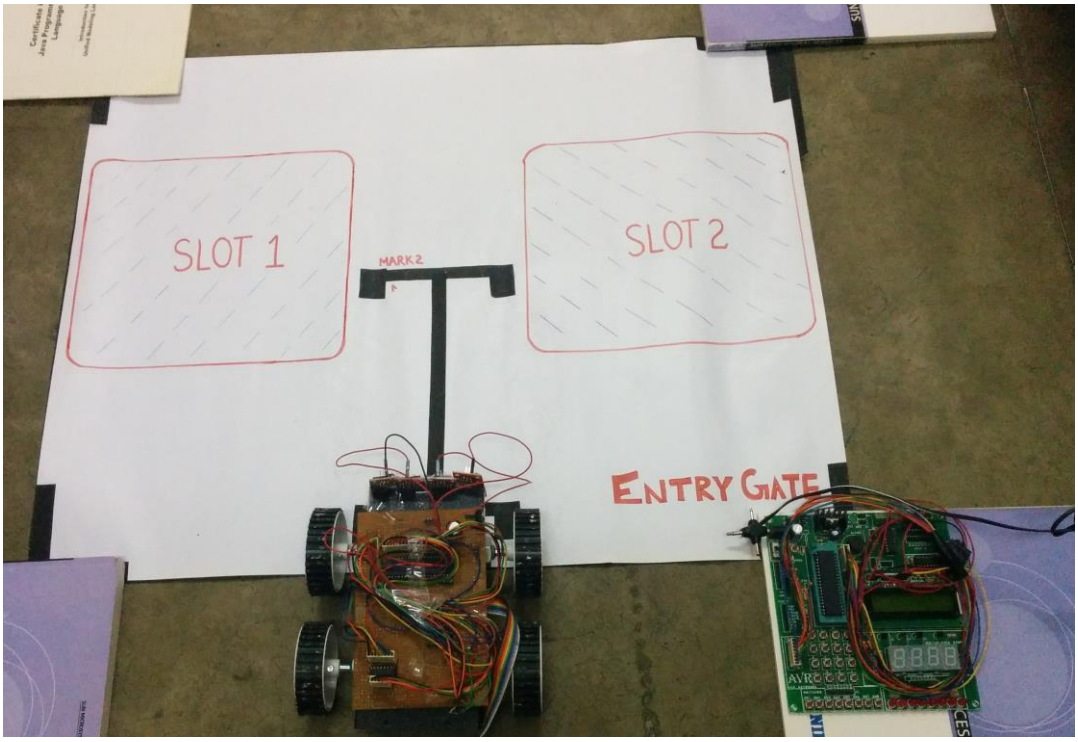


Figure 3.13: Photo of smart parking area with smart car

Figure 3.13 represents layout of our whole project i.e. smart parking area with smart car. For keeping our project simple we have used only two slots in our hardware implementation instead of using four slots as we did for software implementation.

3.3.1 SMART PARKING AREA

The main part of Smart Parking Area is its control board which is present at entry gate of parking system. This control board consists of LCD display, AVR Microcontroller and Keypad interface as shown in figure 3.14.

Figure 3.14: Photo of smart parking area control board (at Entry Gate)

3.3.2 SMART CAR

Above Figure shows the design of Smart Car which consists of car chassis, IR sensors, car wheels and PCB. PCB consists of AVR Microcontroller, L293D motor driver and L7805 voltage regulator.

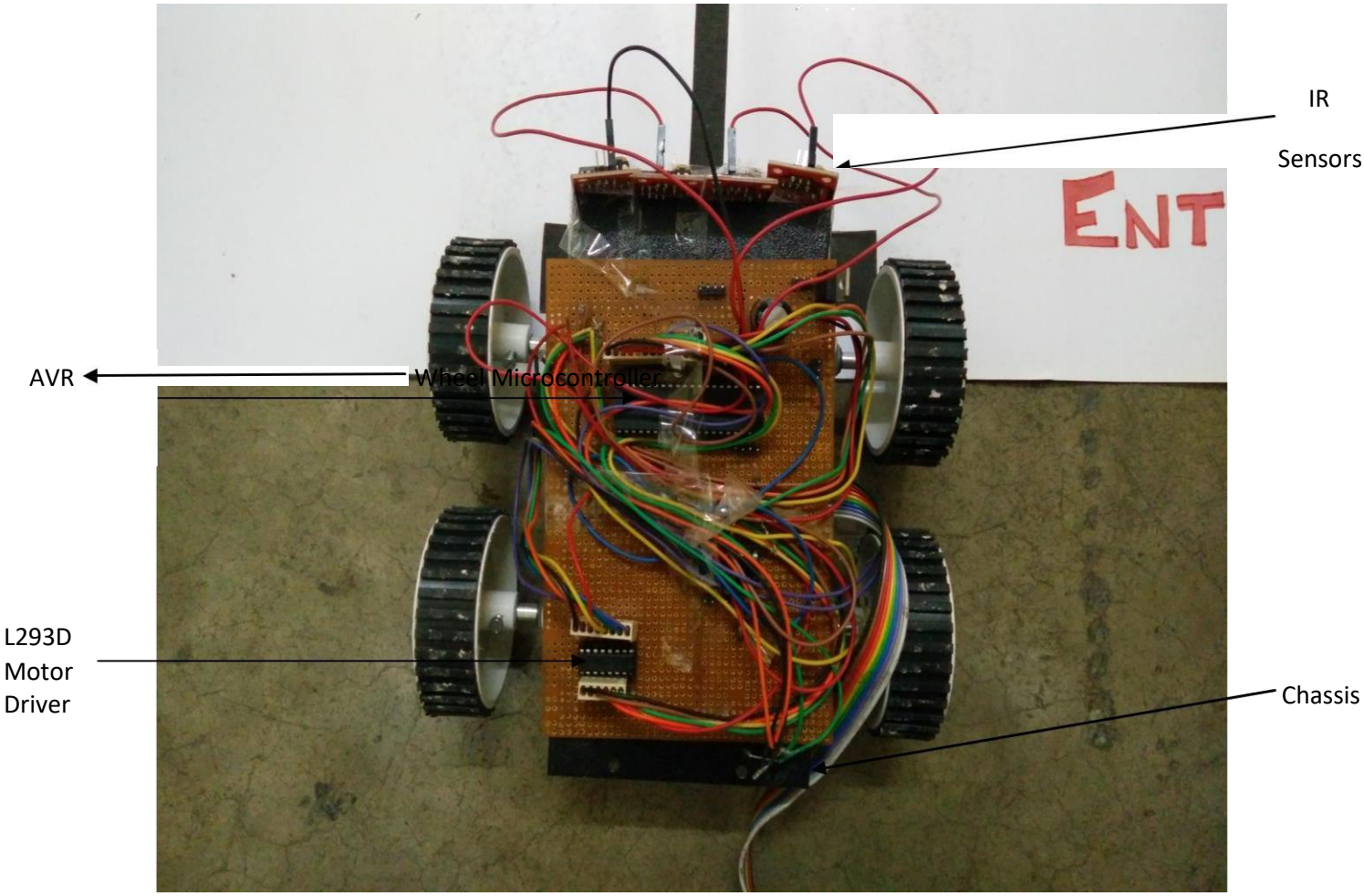


Figure 3.15: Photo of smart car (nonholonomic car)

3.3.3 WORKING OF AUTOMATED CAR PARKING SYSTEM

Firstly as we can see in above figure, the LCD at control board present at entrance of smart parking lot displays “Enter Car No.:”. Now the driver will enter his car registration no. on the control board using the 4X3 keypad as shown in figure 3.16. Once the information has been entered, empty slot will be given to the car. In figure 3.17 slot 1 has been allocated to him.
In case no slot is available for parking, “Parking is full!” will be displayed on the screen at board. The case is shown in figure 3.18.

Figure 3.16: Driver entering his car registration number

Figure 3.17: A Vacant slot is allocated to the user (here slot 1 is allocated)

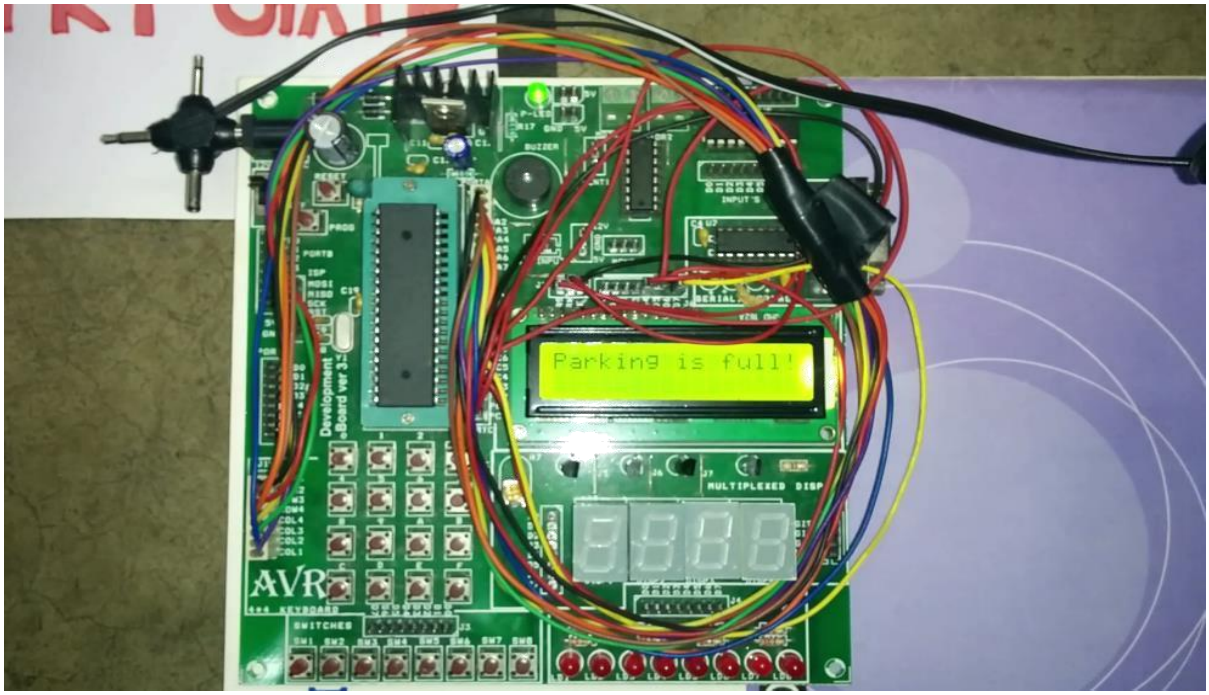


Figure 3.18: When parking area is full, “Parking is full!” is displayed

Once allotment of slot is done, the car will automatically move to the allocated slot with the help of set of four sensors embedded at front of the chassis. Figure 3.19 illustrates the movement of the robot car over the black trajectory while figure 3.20 shows the car being parked at the alloted slot i.e. slot 1.

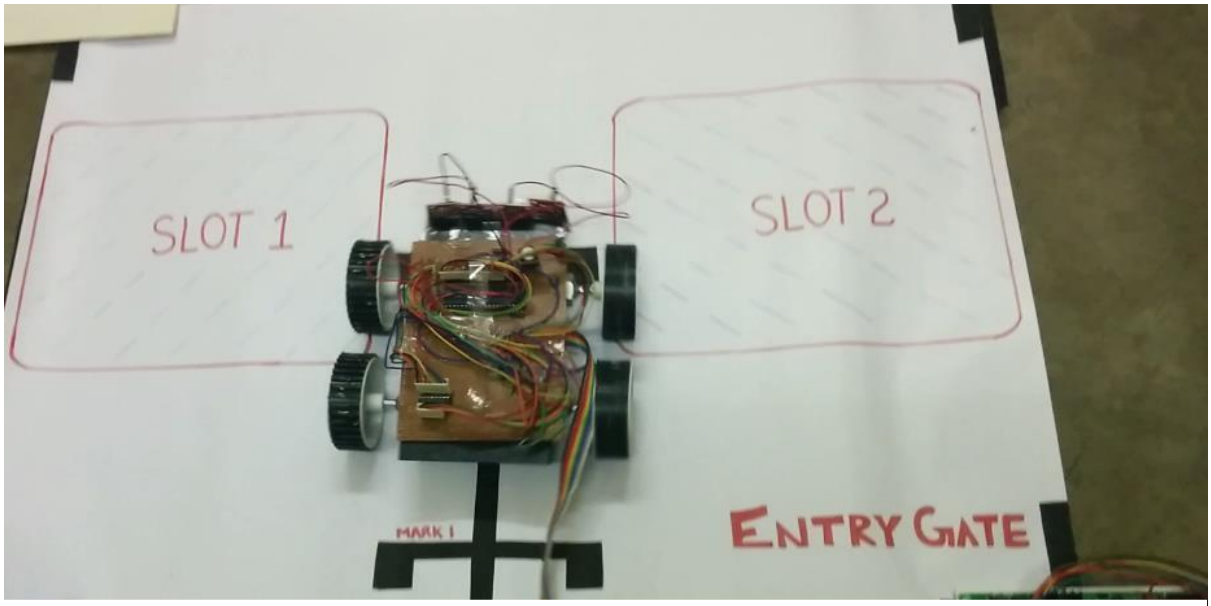


Figure 3.19: Car getting parked at parking area

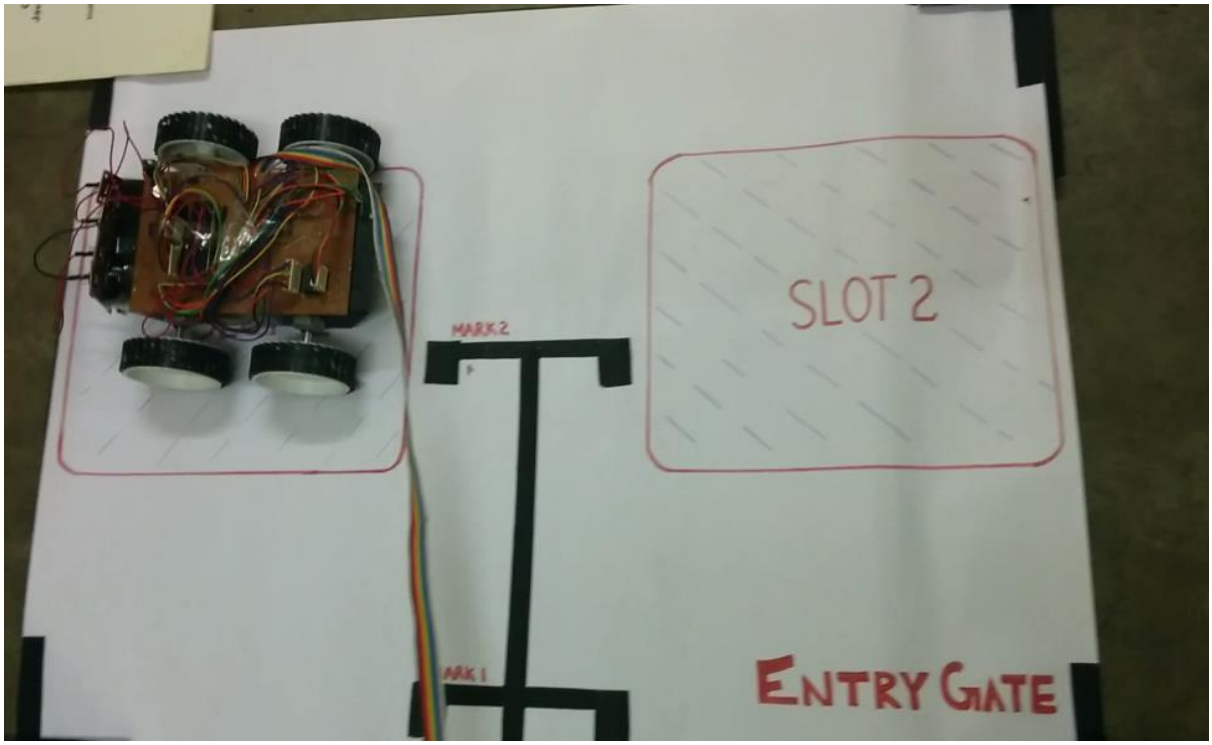


Figure 3.20: Car being parked at slot 1

CONCLUSION

This report proposes a thought of auto parking system as opposed to manual parking which was subject to the auto owner/driver. This thought has been executed utilizing sensors and microcontrollers. This kind of computerized parking model spares time, space and vitality as well as gives an insightful method for stopping which decreases issues like auto crashes and mischances which may emerge because of human mistake and carelessness. This thought is separated into two modules, first module is planning of parking area and second module is outlining of designed car which can move inside that parking area.
The outline and convention utilized by us can be made "Standard" in future with the goal that every last parking lot and automated car utilizes the same measures. The Big organizations like Ford, Hyundai, Volvo, BMW and Google are chipping away at manufacturing driverless automobiles. In the near future autos will proceed to the streets with no driver operating it. So the concept of giving an automated car parking lot for these autos will be an enhancement to these new up and coming advancements since it will give better parking facilities and better parking space to those driverless autos.

This report is divided into three chapters. First chapter gives introduction of our project with its benefits and disadvantages. Second chapter gives review of our project with brief introduction of the software and the hardware components being used in our project. Third chapter explains in details the work done by us i.e. the software implementation and the hardware implementation of our project.

“Our Software and hardware implementation are both successful and through this we can say that this system is purely a realisable concept and the future will be surely headed towards it.”

FUTURE WORK

We can add more highlights to make our model more User friendly and secure. As a matter of first importance we can add a GSM module to our brilliant parking zone. When the car reaches the entrance of lot, the car owner along with his/her car registration number will enter his/her cell phone number too. When the car gets parked to the allotted slot, a SMS will be send to the car owner in which along with allotted slot number, entry time and a random number will also be send. This random number will act like OTP (one time password) which the car owner has to enter when he comes back to take back his car at exit gate. The car will be given to the person only if he enters correct OTP corresponding to his car registration number. This will increase security to our system.

Also a timer will start at the time when the car enters the parking lot and that timer stops as soon as the car will come out of the parking area. The time between the entry and exit of the car can be calculated, which will help in further calculation of parking fee.

REFERENCES

[1] M. M. Rashid, A. Musa, M. Ataur Rahman, and N. Farahana, A. Farhana, "Automatic Parking Management System and Parking Fee Collection Based on Number Plate Recognition," *International Journal of Machine Learning and Computing, Vol. 2, No. 2, April 2012.*

[2] K. Jiang School of Manufacturing & Mechanical Engineering, The University of Birmingham B 15 2TT UK

and L. D. Seneviratne Department of Mechanical Engineering, King's College London WC2R 2LS UK, "A sensor guided autonomous parking system for nonholonomic mobile robots," *Proceedings of the 1999 IEEE*

International Conference on Robotics & Automatlon Detroit, Michigan May 199.

[3] Zhi-Long Wang, Chih-Hsiung Yang, Tong-Yi Guo, "The Design of An Autonomous Parallel Parking Neuro-Fuzzy Controller for A Car-like Mobile Robot," *SICE Annual Conference 2010 August 18-21, 2010, The Grand Hotel, Taipei, Taiwan.*

[4] Prof. D. J. Bonde , Rohit S. Shende, Ketan S. Gaikwad, Akshay S. Kedari,Amol U. Bhokre, "Automated Car Parking System Commanded by Android Application," *D. J. Bonde et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (3) , 2014, 3001-3004*

[5] T. Bhanusri1 K.Prabhakara Rao2, "Advanced Car Parking System with GSM Supported Slot Messenger," *IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) e-ISSN: 2278-2834,p- ISSN: 2278-8735.Volume 10, Issue 1, Ver. II (Jan - Feb. 2015), PP 14-18*

[6] Shihong Qin, Xiangling Yao, "An intelligent parking system based on GSM module", *Appl. Math. Inf. Sci. 7, No. 1L, 55-59 (2013)*

[7] I.E. Paromtchik and C. Laugier, "Autonomous Parallel Parking of a Nonholonomic Vehicle", *IEEE Intelligent Vehicles Symposium, NJ*

[8] F. Gbmez-Bravo, F. Cuesta, A. Ollera, "Autonomous parking and navigation by using soft-computing techniques"

[9] A Tutorial and Elementary Trajectory Model for the Differential Steering System of Robot Wheel Actuators by G.W. Lucas

[10] Daniel Howley, Technology Reporter (February 13, 2016), “Self-Driving Cars Are Coming Soon to a Highway Near You,” Yahoo News. Retrieved from “https://in.news.yahoo.com/self-driving-cars-are-coming-soon-to-a-highway-221223190.html”

[11] A. Albagul, K. Alsharef, M. Saad, Y. Abujeela, “Design and Fabrication of an Automated Multi-level Car Parking System,” Manufacturing Engineering, Automatic Control

[12] Automated car parking system, https://www.youtube.com/watch?v=QZLfv8u2pyU

[13]Smart Car Parking Lot Management System,

https://www.youtube.com/watch?v=56AiTPYecTM

[14] Automatic Car Parking Indicator System,

http://www.projectsof8051.com/automatic-car-parking-indicator-system/

[15]Automated parking system, https://en.wikipedia.org/wiki/Automated_parking_system

[16]Figure of automatic parking of car, http://krishnaparkinfracon.com/wp-content/uploads/2012/05/multilevel-car-parking-systems.jpg

[17]Figure of manual parking of car, http://www.myparkingsign.com/blog/wp-content/uploads/Bad-parking-2.jpg

[18] Pin out of Atmega 16, http://logic-bratsk.ru/radio/micro/atmel_pic/at_dip40.htm

[19] LCD datasheet,

http://www.engineersgarage.com/electronic-components/16x2-lcd-module-datasheet

[20] L293D datasheet,

http://www.engineersgarage.com/electronic-components/l293d-motor-driver-ic

[21] 4X3 keypad datasheet,

http://microforbetterlive.blogspot.in/2009/05/access-4x3-keypad.html