VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELGAUM-590 014

Automatic Vehicle Parking System

A Project report submitted to Visvesvaraya Technological University
in partial fulfillment of the award of degree of
Bachelor of Engineering in Electronics & Communication Engineering

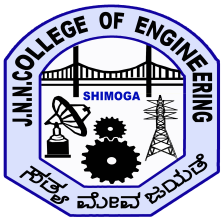
Submitted by

Akshay kumar:	4JN12EC008
Khushboo:	4JN12EC035
Niha firdouse:	4JN12EC048
Piyush kumar mishra:	4JN12EC051

Under the guidance of

Prof. M D Sunil

Professor, Department of Electronics & Communication Engineering



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
JAWAHARLAL NEHRU NATIONAL COLLEGE OF ENGINEERING
SHIMOGA-577204

June - 2016

JAWAHARLAL NEHRU NATIONAL COLLEGE OF ENGINEERING
SHIMOGA-577204

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Certificate

This is to certify that the Project work entitled "AUTOMATIC VEHICLE PARKING SYSTEM" is a bonafide work carried

by

Akshay Kumar: 4JN12EC008
Khushboo: 4JN12EC035
Niha Firdouse: 4JN12EC048
Piyush Kumar Mishra: 4JN12EC051

in partial fulfillment of the award of the degree of Bachelor of Engineering in Electronics & Communication Engineering of Visvesvaraya Technological University, Belgaum, during the year 2013. It is certified that all corrections / suggestions indicated during internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work.

.....	.....	.....
Prof. M D Sunil	Prof. H.K. Harish	Dr. R. Srinivasa Rao Kunte
Asst. Professor, Guide	HOD	Principal

External Viva

Name of the Student :

USN :

Name and Signature of Examiners with date

1.

2.

Acknowledgements

I would like to acknowledge the help and encouragement given by various people during the course of this project.

I would like to express my sincere gratitude to Prof. **H. K. Harish,** Associate Professor and Head, Department of Electronics & Communication Engineering, J. N. N. College of Engineering, Shimoga, for his kind support, guidance and encouragement throughout the course of this work.

I am deeply indebted and very grateful to the invaluable guidance given by project guide **Prof. M D Sunil,** Professor during this project work.

I am thankful to our beloved principal **Dr. R. Srinivasa Rao Kunte** for providing excellent academic climate.

I would like to thank all the teaching and non-teaching staff of Dept. of E&CE for their kind co-operation during the course of the work. The support provided by the college and departmental library is greatly acknowledged.

Finally, I am thankful to my parents and friends, who helped me in one way or the other throughout my project work.

Student Name.

Abstract

In the modern world, where parking-space has become a very big problem and it is a very crucial necessity to avoid the wastage of space like big companies and shopping malls etc. In places where more than 100 vehicles need to be parked, this system proves to be useful in reducing wastage of time. Nowadays many companies are looking for driverless car and some of the companies already started working on that robotic car. Due to that, the standard parking system requires to park these type of robotic vehicle. The aim of our project ”Automatic Vehicle Parking System” is to implement the automatic parking of a vehicle. It involves the parking as well as the intimation to the vehicle owner that vehicle has been parked. When the owner of the vehicle reaches at the entrance of the parking section, the owner gets off the vehicle. After getting to know about the vacant slot displayed in the LCD, the owner types the vacant slot number. The vehicle is then directed to the free slot. A message is then sent to the owner regarding its safe parking. If the slot is not available the owner of vehicle return or handover the vehicle for parking in that case owner will knows about his/her parking position. In this project, we work on both robotic and non robotic vehicle. For robotic vehicle, it drives automatically to the assigned slot , while for non robotic it drive by the staff of parking section. Here we have a display board i.e. LCD, which shows the information about vacant slots at the entrance of parking gate. This vacancy is sensed by IR Module which is placed in the parking slot. The vacant slot is displayed according to the priority concept. Once the vacant slot is known then the code for it is typed in the mobile of the owner which is then decoded using the DTMF decoder. The DTMF signals from the mobile is decoded using the decoder IC HT9170 which decodes it into its corresponding binary codes. This is then given to the ATMega328P microcontroller which then performs the action according to the program.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 General introduction

The aim of our project ”Automatic Vehicle Parking System” is to implement the automatic parking of a vehicle. The project has two sections: *parking section*, which is in the area of parking and the *vehicle section*, which is installed in the robotic vehicle intended to park. When a vehicle reaches the entrance of the parking section, the owner gets off the vehicle. Seeing the vacant slots displayed on the LCD, he presses the vacant slot number through the mobile phone. If the vehicle is robotic, it is moved to that assigned slot automatically. Otherwise a person from the parking section has to drive the vehicle to the assigned slot. The owner in both the cases is intimated through the message saying ”Vehicle Parked Safely”.

1.2 Problem Statement

Due to the increase in the number of vehicles on the road, traffic problems are bound to exist. This is due to the fact that the current transportation infrastructure and car parking facility developed are unable to cope with the influx of vehicles on the road. To address the above mentioned problems, the smart car parking system has been developed. With the implementation of the parking system, a person can easily locate and secure a vacant parking space at any car park deemed convenient to them. In this car parking system, there are two sections, one is allocating a slot for parking and another is vehicle system. The present design deals with a display board which shows a vacancy slot for a car together with a message displaying, car parked safely after parking.

1.3 Methodology

In this project we have a LCD board, which shows the current status of our parking area. It indicates the vacant and busy slots present in our parking section. This vacancy is sensed by the IR Reflectance Module which is placed in each parking slot. It senses the vehicle when parked in particular slot to know that the particular parking slot is busy and unavailable. Once the vacant slot is known then the code for it is pressed in the mobile of the owner which is then decoded using the DTMF decoder. The DTMF signals from the mobile is decoded into binary codes and sends it to the micro controller used in both parking and vehicle section. Motor driver is used to drive the wheels automatically using Dc motors. In this project, GSM module is used to intimate the owner through the message saying ”Vehicle Parked Safely”.

1.4 Scope of the Project

This system is effectively in use in most of the European countries and many of the American states. This design is mainly comprised of low manual operation as well as efficient equipment which can be installed in any of the commercial, industrial, apartments, institutions/universities ,etc. Hence it is a low cost apparatus as it mainly uses a micro controller which is programmable, which is easy to install in any of the above places mentioned. This project can be used in companies and shopping malls. There are two companies which are working on automatic vehicle. First one is Google Self-Driving Car which is developed by Google X as part of its project to develop technology for mainly electric cars. And, second one is Chinese hi-tech firm Baidu which has unveiled a plan to let driverless vehicles range freely around an entire city. The five-year plan will see the autonomous cars, vans and buses slowly introduced to the eastern city of Wuhu.Initially no passengers will be carried by the vehicles as the technology to control them is refined via journeys along designated test zones.Eventually the test areas will be expanded and passengers will be able to use the vehicles. So, based on these ideas we have developed a vehicle which automatically moves to its assigned slot in the parking area.

1.5 Limitations

This project is efficient for robotic vehicle while for non robotic vehicle it needs staff to park the vehicle from parking gate to the parking slot.
Here we are following distance based parking system so we have to program every robotic vehicle. When a vehicle is being directed to the parking slot, we assume that there are no obstacles on the way.

1.6 Organization of The report

This project description is divided into 4 chapters. Chapter 1 gives the general introduction about the project, problem statement,methodology,scope of the project,limitations and organization of the report. Chapter 2 contains the information about the theoretical background, basic block diagram and hardware requirements such as power supply, ATmega328p microcontroller, IR senor, LCD display, GSM module, DTMF decoder, motor driver, Dc motors. Chapter 3 details about the design, implementation, circuit diagram, description, flowchart and algorithm of the overall project. Chapter 4 shows the result of this project and discuss about the future scope of this project. The software programs of overall project are included in Appendix A.

Chapter 2

Theoretical Background

The basic block diagram and its explanations are presented in section 2.1. In section 2.2, hardware requirements are described.

2.1 System Design

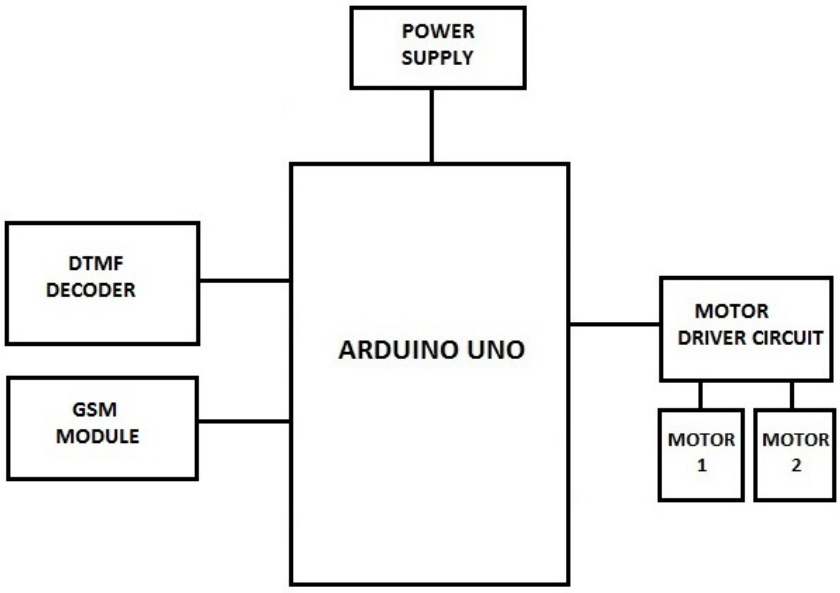


Figure 2.1: Vehicle Section

The block diagram consists of two sections, parking section and the vehicle section. Power supply is required for the working of various components. In this project we have a display board, which shows the information about vacant slots in the parking area. Display system used here is a 16X2 LCD. Liquid Crystal Displays consumes less power and it is cheaply available. This vacancy is sensed by IR Reflectance Module which is placed in the parking slot. The vacant slot is displayed according to the priority concept. Once the

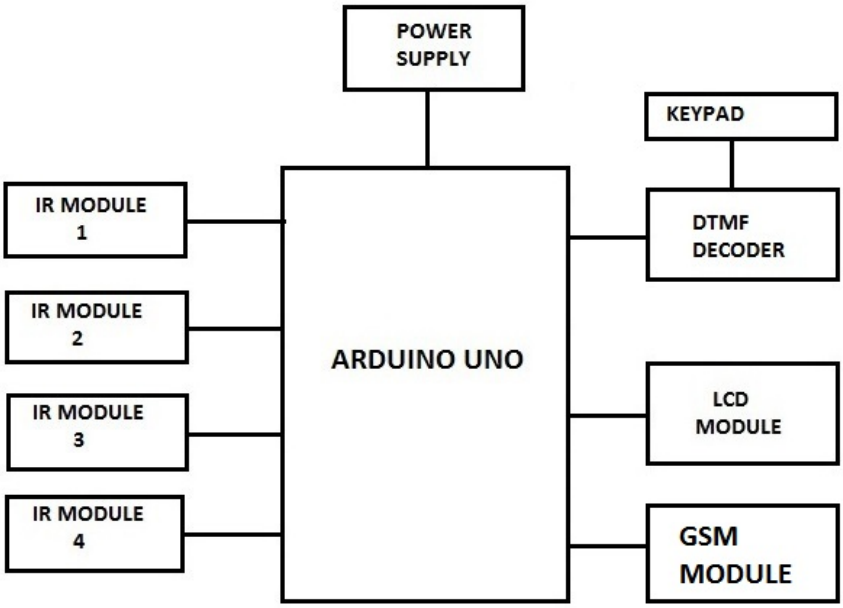


Figure 2.2: Parking Section

vacant slot is known then the code for it is typed in the mobile of the owner which is then decoded using the DTMF decoder. The DTMF signals from the mobile is decoded using the decoder IC HT9170.Here the owner presses the code for entrance in his mobile. This DTMF signal from the mobile is then sent to the IC HT9170 which decodes it into its corresponding binary codes. This is then given to the micro controller which then performs the entrance action according to the program. As we know that the DTMF decoded signals are then given to the micro controller placed inside the vehicle, it has a major role in our project. The micro controller used here is ATMega328P. The vehicle is controlled by the micro controller. The forward and the backward movement of the vehicle is controlled by the DTMF code. After pressing the code the vehicle then moves automatically, using the motor fixed to it , according to the in-built measured distance of each slot and then the vehicle is parked at the vacant slot. After vehicle is parked in the slot it intimates the owner of the vehicle through message saying ”Vhicle Parked Safely” using GSM module. A GSM modem is a wireless modem that works with a GSM wireless network like a GSM mobile phone. A GSM modem requires a SIM card from a wireless carrier in order to operate. Here we assume that some vehicle has to be parked in some slot, it becomes difficult to search the vacant slots in the parking area.So, this system deals with this difficulty. This project can be used in companies, shopping malls and apartments.

2.2 Hardware Requirement

In our project, we have used 3x3 sq. feet ply board for parking section which consists of four equal slots of area 22.5x20 sq. cm and we have designed a robotic vehicle which is of area 18x15 sq. cm. Following are the major components which we have used in our project.

2.2.1 Power Supply

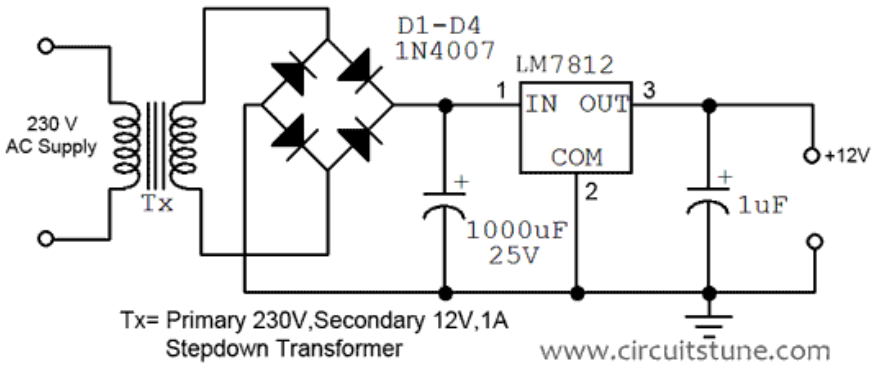


Figure 2.3: Power Supply

The power supply circuits built using filters, rectifiers, and then voltage regulators. Starting with an ac voltage, a steady dc voltage is obtained by rectifying the ac voltage, then filtering to a dc level, and finally, regulating to obtain a desired fixed dc voltage. The regulation is usually obtained from an IC voltage regulator unit, which takes a dc voltage and provides a somewhat lower dc voltage, which remains the same even if the input dc voltage varies, or the output load connected to the dc voltage changes. The block diagram of power supply is shown in Figure 2.3 and its waveforms are shown in Figure 2.4.

2.2.2 ARDUINO ATmega328P

The Uno is a micro controller board based on the ATmega328P as shown in Figure 2.5. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the micro controller; simply connect it to

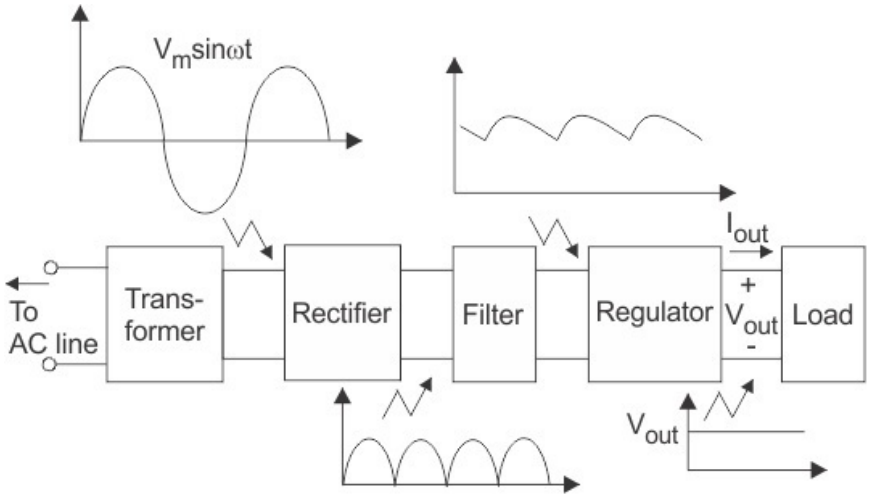


Figure 2.4: Waveforms

a computer with a USB cable or power it with AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again. "Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform.

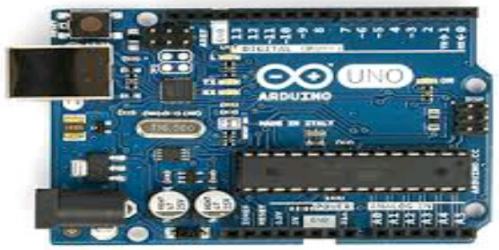


Figure 2.5: Arduino ATmega328P

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by

plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The features of Aurduino ATmega328P has been described in Figure 2.6. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. The power pins are as follows: VIN is the input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. 5 volt is the regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Figure 2.6: Features of Arduino ATmega328P

The Atmega328 has 32 KB of flash memory for storing code (of which 0,5 KB is used for the bootloader). It has also 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each

pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions: o Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. TThese pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip . o External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
o PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function. o SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. o LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off. The Uno has 6 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality: o I 2C: 4 (SDA) and 5 (SCL). Support I2C (TWI) communication using the Wire library. There are a couple of other pins on the board: o AREF. Reference voltage for the analog inputs. Used with analogReference(). o Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board. See also the mapping between Arduino pins and Atmega328 ports.

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an *.inf file is required.. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-toserial chip and USB connection

to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. To use the SPI communication, please see the ATmega328 datasheet.

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data. The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

For overcurrent protection, the Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16”), not an even multiple of the 100 mil spacing of the other pins.

2.2.3 IR TX-RX

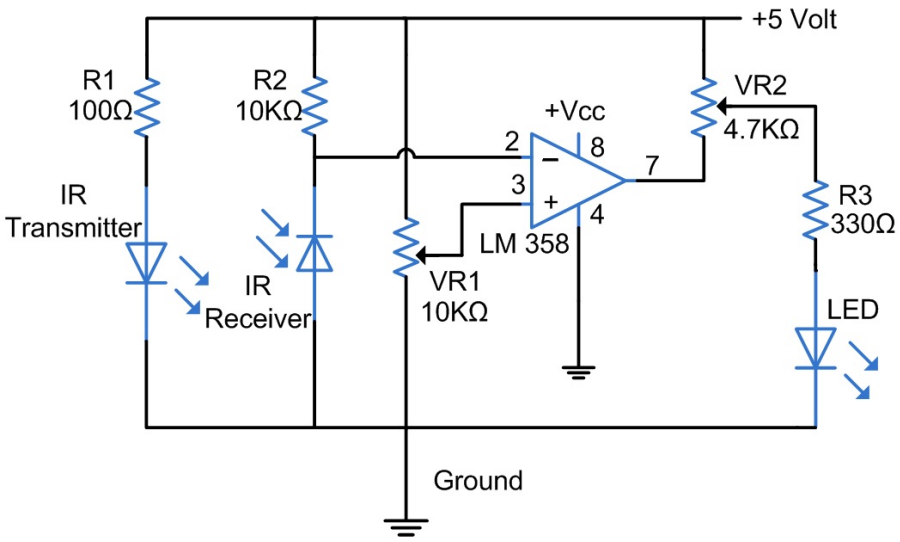


Figure 2.7: IR Tx-Rx

Infrared transmitter is one type of LED which emits infrared rays generally called as IR Transmitter. Similarly IR Receiver is used to receive the IR rays transmitted by the IR transmitter. One important point is both IR transmitter and receiver should be placed straight line to each other as shown in Figure 2.7.

The transmitted signal is given to IR transmitter whenever the signal is high, the IR transmitter LED is conducting it passes the IR rays to the receiver. The IR receiver is connected with comparator. The comparator is constructed with LM 358 operational amplifier. In the comparator circuit the reference voltage is given to inverting input terminal. The non inverting input terminal is connected IR receiver. When interrupt the IR rays between the IR transmitter and receiver, the IR receiver is not conducting. So the

comparator non inverting input terminal voltage is higher then inverting input. Now the comparator output is in the range of +5V. This voltage is given to micro controller or PC and led so led will glow.

When IR transmitter passes the rays to receiver, the IR receiver is conducting due to that non inverting input voltage is lower than inverting input. Now the comparator output is GND so the output is given to micro controller or PC. This circuit is mainly used to for counting application, intruder detector etc.

2.2.4 LCD DISPLAY

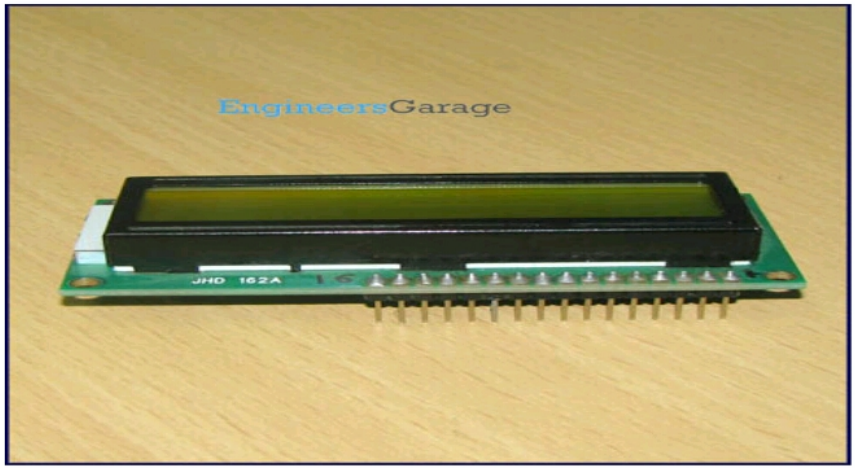


Figure 2.8: LCD display

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display as shown in Figure 2.8, is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special evencustom characters (unlike in seven segments), animations and so on.

A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen,

setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD. Click to learn more about internal structure of a LCD.

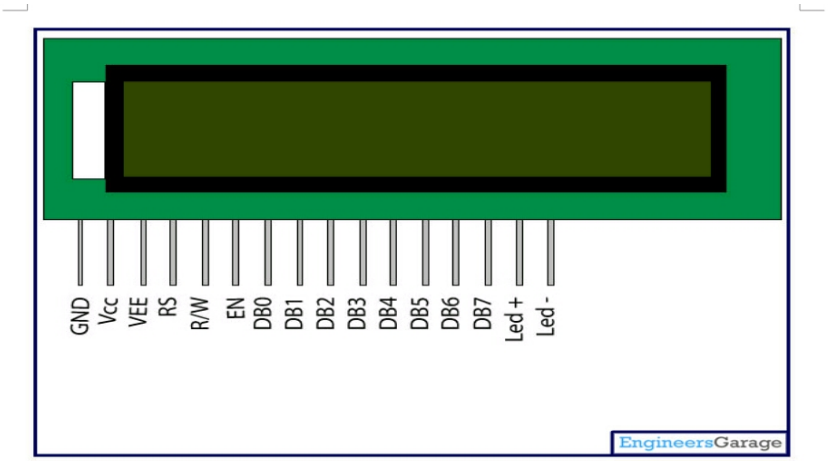


Figure 2.9: Pin diagram of LCD display

Pin Description:

The name and functions of each pin of the 16x2 LCD module(shown in Figure 2.9) is given below.

- Pin1(Vss):Ground pin of the LCD module.
- Pin2(Vcc): Power to LCD module (+5V supply is given to this pin)
- Pin3(VEE):Contrast adjustment pin. This is done by connecting the ends of a 10K potentimeter to +5V and ground and then connecting the slider pin to the VEE pin. The voltage at the VEE pin defines the contrast. The normal setting is between 0.4 and 0.9V.
- Pin4(RS):Register select pin.The JHD162A has two registers namely command register and data register. Logic HIGH at RS pin selects data register and logic LOW at RS pin selects command register. If we make the RS pin HIGH and feed an input to the data lines (DB0 to DB7), this input will be treated as data to display on LCD screen. If we make the RS pin LOW and feed an input to the data lines, then this

will be treated as a command (a command to be written to LCD controller - like positioning cursor or clear screen or scroll).

- Pin5(R/W): Read/Write modes. This pin is used for selecting between read and write modes. Logic HIGH at this pin activates read mode and logic LOW at this pin activates write mode.
- Pin6(E): This pin is meant for enabling the LCD module. A HIGH to LOW signal at this pin will enable the module.
- Pin7(DB0) to Pin14(DB7): These are data pins. The commands and data are fed to the LCD module though these pins.
- Pin15(LED+): Anode of the back light LED. When operated on 5V, a 560 ohm resistor should be connected in series to this pin. In arduino based projects the back light LED can be powered from the 3.3V source on the arduino board.
- Pin16(LED-): Cathode of the back light LED.

Interfacing 16x2 LCD to Arduino uno

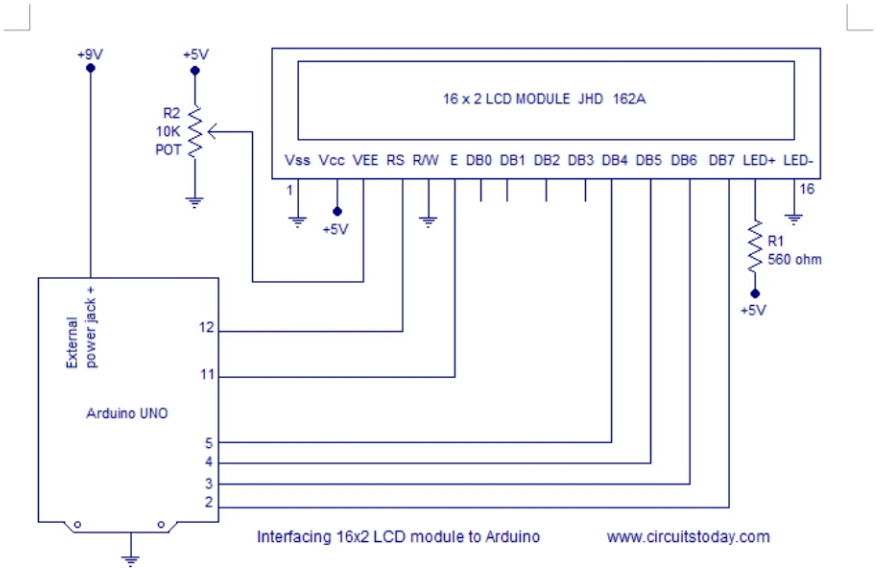


Figure 2.10: Interfacing 16x2 LCD to Arduino uno

LCD modules form a very important part in many arduino based embedded system designs. So the knowledge on interfacing LCD module to arduino is very essential in

designing embedded systems. This section of the article is about interfacing an Arduino to 16x2 LCD which is shown in Figure 2.10. JHD162A is the LCD module used here. JHD162A is a 16x2 LCD module based on the HD44780 driver from Hitachi. The JHD162A has 16 pins and can be operated in 4-bit mode (using only 4 data lines) or 8-bit mode (using all 8 data lines). Here we are using the LCD module in 4-bit mode. First, I will show you how to display a plain text messages on the LCD module using arduino and then I have designed a useful project using LCD and arduino - a digital thermometer. Before going in to the details of the project, let's have a look at the JHD162A LCD module.

RS pin of the LCD module is connected to digital pin 12 of the arduino. R/W pin of the LCD is grounded. Enable pin of the LCD module is connected to digital pin 11 of the arduino. In this project, the LCD module and arduino are interfaced in the 4-bit mode. This means only four of the digital input lines(DB4 to DB7) of the LCD are used. This method is very simple, requires less connections and you can almost utilize the full potential of the LCD module. Digital lines DB4, DB5, DB6 and DB7 are interfaced to digital pins 5, 4, 3 and 2 of the Arduino. The 10K potentiometer is used for adjusting the contrast of the display. 560 ohm resistor R1 limits the current through the back light LED. The arduino can be powered through the external power jack provided on the board. +5V required in some other parts of the circuit can be tapped from the 5V source on the arduino board. The arduino can be also powered from the PC through the USB port.

2.2.5 GSM NETWORK

The GSM shield by Arduino is used to send/ receive messages and make/receive calls just like a mobile phone by using a SIM card by a network provider. We can do this by plugging the GSM shield into the Arduino board and then plugging in a SIM card from an operator that offers GPRS coverage. The shield employs the use of a radio modem by SIMComm. We can communicate easily with the shield using the AT commands. The GSM library contains many methods of communication with the shield. This GSM Modem can work with any GSM network operator SIM card just like a mobile phone with its own unique phone number. Advantage of using this modem will be that its RS232 port can be used to communicate and develop embedded applications. Applications like SMS Control, data

transfer, remote control and logging can be developed easily using this. The modem can either be connected to PC serial port directly or to any microcontroller through MAX232. It can be used to send/receive SMS and make/receive voice calls. It can also be used in GPRS mode to connect to internet and run many applications for data logging and control. In GPRS mode you can also connect to any remote FTP server and upload files for data logging This GSM modem is a highly flexible plug and play quad band SIM900A GSM modem for direct and easy integration to RS232 applications. It Supports features like Voice, SMS, Data/Fax, GPRS and integrated TCP/IP stack. To be connected to a cellular network, the shield requires a SIM card provided by a network provider. Most recent revision of the board makes the connection of the shield with the Arduino Uno board by connecting its TX to pin 0 of Arduino and pin 1 of Arduino to RX of shield. This is an ultra compact and reliable wireless module.

The SIM900A is a complete Dual-band GSM/GPRS solution in a SMT module which can be embedded in the customer applications allowing you to benefit from small dimensions and cost-effective solutions. Featuring an industry-standard interface, the SIM900A delivers GSM/GPRS 900/1800MHz performance for voice, SMS, Data, and Fax in a small form factor and with low power consumption. With a tiny configuration of 24mm x24mm x 3 mm, SIM900A can fit in almost all the space requirements in your applications, especially for slim and compact demand of design.

- **Features:**

  1.Dual-Band 900/ 1800 MHz

  2.GPRS multi-slot class 10/8

  3.GPRS mobile station class B

  4.Compliant to GSM phase 2/2+

  5.Class 4 (2 W @900 MHz)

  6.Class 1 (1 W @ 1800MHz)

  7.Dimensions: 24*24*3 mm Weight: 3.4g

  8.Control via AT commands (GSM 07.07, 07.05 and SIMCOM enhanced AT Commands)

  8.SIM application toolkit

  9.Supply voltage range: 3.1- 4.8V

10.Low power consumption: 1.5mA(sleep mode)

11.Operation temperature: -40 C to +85C

- **Specifications for Fax:**

  1.Group 3, class 1 Specifications for Data

  2.GPRS class 10: Max. 85.6 kbps (downlink)

  3.PBCCH support Coding schemes CS 1, 2, 3, 4

  4.CSD up to 14.4 kbps USSD

  5.Non transparent mode PPP-stack

- **Specifications for SMS via GSM/GPRS:**

  1.Point to point MO and MT

  2.SMS cell broadcast

  3.Text and PDU mode

- **Software features:**

  1.0710 MUX protocol

  2.Embedded TCP/UDP protocol FTP/HTTP

- **Enhanced version has following features:** 1.FOTA

  2.MMS

  3.Embedded AT

- **Specifications for Voice:**

  1.Tricodec

  2.Half rate (HR)

  3.Full rate (FR)

  4.Enhanced Full rate (EFR)

  5.Hands-free operation (Echo suppression)

  6.AMR

  7.Half rate (HR)

  8.Full rate (FR)

- **Compatibility:**

  AT cellular command interface It can communicate with controllers via AT

commands (GSM 07.07, 07.05 and SIMCOM enhanced AT Commands).

- **Power Requirements:**

  The board should be powered with an external power supply that can provide current between 700mA and 1000mA. Powering an Arduino and the GSM shield from a USB connection is not recommended, as USB cannot provide the required current when the modem is in heavy use. So instead we have to use 12V adapter. The modem can pull up to 2A of current at peak usage, which can occur during data transmission.

- **Applications:**

  1.SMS based Remote Control and Alerts

  2.Security Applications

  3.Sensor Monitoring

  4.GPRS Mode Remote Data logging

- **On Board Indicators:**

  The shield contains a number of status LEDS: ON: It shows that the shield is getting power and is switched on. NET: This LED blinks when the modem is communicating with the radio network.

- **NETWORK LED:**

  The Network LED indicates the various states of the GSM module i.e. POWER ON, NETWORK REGISTERATION and GPRS CONNECTIVITY. When the modem is powered up, this NETWORK LED will blink every second. After the Modem registers in the network (it takes 10-60 seconds), this LED will blink in step of 3 seconds at slow rate. At this stage we can start using the modem for our application. This shows that the modem is registered with the network. AT Commands for using the shield

- **Checking The Operantion And Cconnection Of GSM Shield:**

  AT Press ENTER this would print OK which signifies of working connection and operation of the GSM shield.

  MAKING A VOICE CALL: ATD+(country code)mobile number; Press ENTER.

  DISCONNECTING THE ACTIVE CALL: ATH Press ENTER.

RECEIVING THE CALL: ATA Press ENTER.

SENDING A MESSAGE: For sending SMS in text Mode: AT+CMGF=1 Press ENTER AT+CMGS=”mobile number” Press ENTER Once the AT commands is given’ ¿’ prompt will be displayed on the screen. Type the message to be sent via SMS. After this, Press CTRL+Z to send the SMS. If the SMS sending is successful, ”OK” will be displayed along with the message number.

RECEIVING A MESSAGE: For reading SMS in the text mode: AT+CMGF = 1 Press ENTER AT+CMGR = num. Number (num.) is the message index number stored in the SIM card. For new SMS, URC will be received on the screen as + CMTI: SM ’num’. After this AT+CMGR=1 Press ENTER This displays the message on the screen along with sender details, number and timing too.

- **How to Interface the GSM shield with ARDUINO UNO**

  1.First we connect our Arduino Uno to the Computer or Laptop to see which COM port will be used to burn the program from the computer or laptop. This also provides power to the Arduino Uno.

  2.Next we supply power to the GSM shield (supply only 12V to the GSM shield from the power jack using the adapter) which is going to be used for our program

  3.For GSM programs, only 2 pins, RX and TX are to be used mainly. So we require only these two pins of the Arduino Uno. These pins are pins 0 and 1 of the Arduino Uno

  4.Next burn the required program in The Arduino Uno using the software

  5.Then connect the GSM shield to Arduino such that RX, TX of the shield is connected to the TX, RX of the Arduino Uno.

  6.Your interfacing is completed

2.2.6 DTMF MODULE(HT9170)

HT9170 is the series of Dual Tone Multi Frequency (DTMF) receivers. They employ digital counting techniques to detect and decode the 16 DTMF tones into 4 bit output code.

The pin diagram is shown in Figure 2.11. HT9170 series receivers do not require any

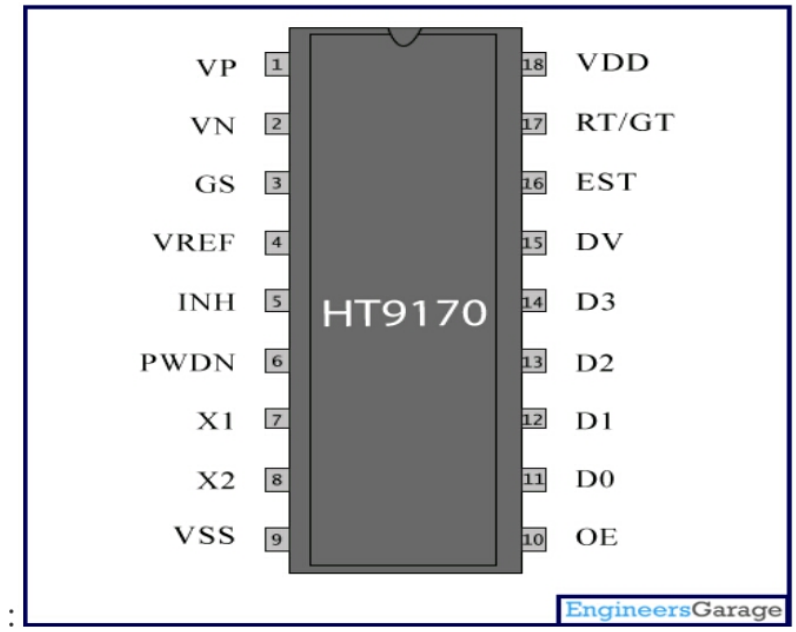


Figure 2.11: DTMF decoder

external filters as they use highly accurate switched capacitor filters for filtering low and high frequency signals from the DTMF tones. They also support power down (PWDN) and inhibit (INH) modes. PWDN mode is used to power off the crystal, while INH mode to inhibit the A, B, C D DTMF tones. The clock is provided by a 3.58 MHz crystal.

In simple terms, HT9170 IC detects and decodes the 16 DTMF tones into 4 bit output. In case the tones are not detected, the four output bits remain low. The DV pin goes high on detection of a valid tone.

Interfacing of DTMF Decoder to Arduino uno

The interfacing of DTMF circuit is shown in Figure 2.12 The input is given by a mobile phone with a 3.5mm jack which provides the DTMF signals to MT8870 decoder. I used USBasp to program arduino u directly program it using usb it is better to use usbasp if you are building the circuit on breadboardUse the code to program arduino it uses hex codes for numbers pressed on dialpad of mobile

The Hex Codes For Keypress Are As Follows

1 = 0x01

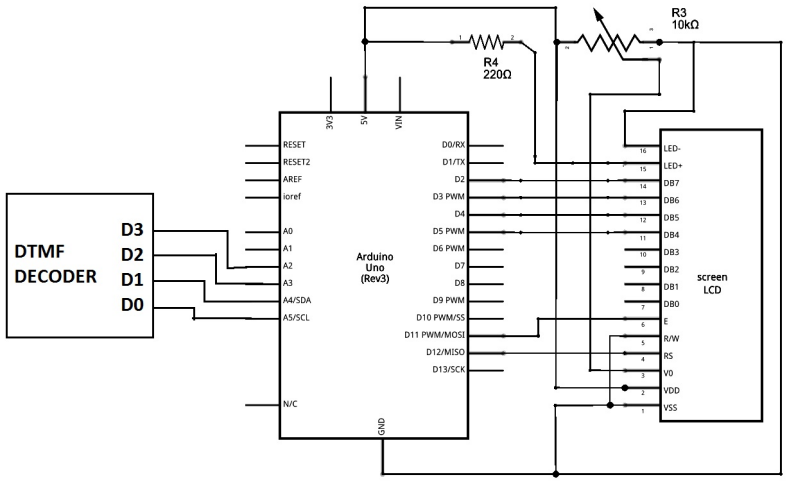


Figure 2.12: Interfacing DTMF decoder

2 = 0x02

3 = 0x03

4 = 0x04

5 = 0x05

6 = 0x06

7 = 0x07

8 = 0x08

9 = 0x09

0 = 0x0A

= 0x0C

2.2.7 MOTOR DRIVER(L293D)

L293D is a typical Motor driver or Motor Driver IC which allows DC motor to drive on either direction. L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that you can control two DC motorwith a single L293D IC. Dual H-bridge Motor Driver integrated circuit(IC).

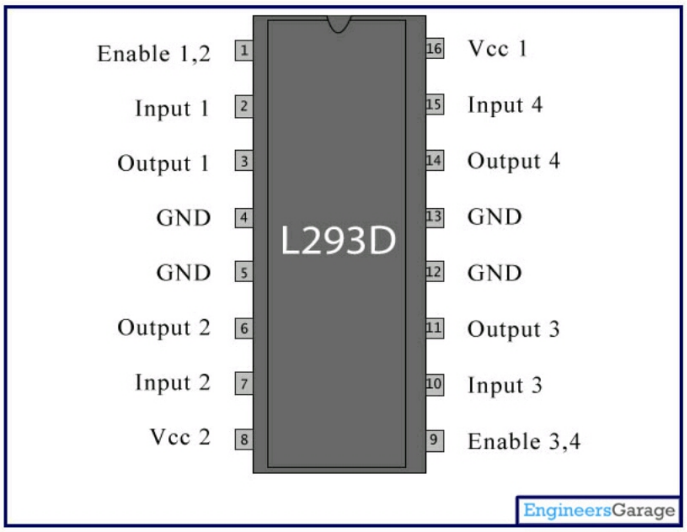


Figure 2.13: Pin Diagram of L293D

The l293d can drive small and quiet big motors as well, check the Voltage Specification at the end of this page for more info.

The L293D works on the concept of typical H-bridge, a circuit which allows the high voltage to be flown in either direction. In a single L293D IC there are two H-bridge circuits which can rotate two DC motors independently. Due to its size and voltage requirement, it is frequently used in robotics applications for controlling DC motors, including in Arduino projects. The L293D is also a key component in larger 'motor driver' boards available premade for hobbyists.

Working

There are two drive pins on L293D(as shown in figure 2.13). Pin 1 (left H-bridge) and pin 9 (right H-bridge). To turn ON the corresponding motor, pin 1 or 9 need to be set to HIGH. If either pin 1 or pin 9 goes low then the motor in the corresponding section will go OFF (high impedance). These inputs (1 and 9) are the ones that should be used to control motor START/STOP and motor speed under PWM, since there would be high impedance output during low semiperiod of PWM, it would not provoke overload of the L293D when the motor is turning. Thus, PWM or motor ON/OFF control should never be input to pins 2, 7, 15, 10, which should only be used to control direction (Clockwise - CounterClockwise).

The direction-defining four Input pins for the L293D are pin 2 and 7 on the left and pin 15 and 10 on the right as shown on the pin diagram. Left input pins will determine the rotation of motor connected on the left side and right input for motor on the right hand

side. The motors are rotated on the basis of the inputs provided at the input pins as LOGIC 1 or LOGIC 0.

L293D Logic Table

Assuming a motor connected on left side output pins (pin 3,6).

1. Pin 2 = Logic 1 and Pin 7 = Logic 0 — Clockwise Direction
2. Pin 2 = Logic 0 and Pin 7 = Logic 1 — Anticlockwise Direction
3. Pin 2 = Logic 0 and Pin 7 = Logic 0 — Brake (this is not high impedance) (force stop rotation using electric brake = same voltage both pins of the motor = overload while the motor is still running)
4. Pin 2 = Logic 1 and Pin 7 = Logic 1 — Brake (this is not high impedance) (force stop rotation using electric brake = same voltage both pins of the motor = overload while the motor is still running)

In a similar way, the motor can be operated across input pins 15 and 10 to control the direction of the motor attached to the H-bridge's right side. Using pins 2 and 7 (15 and 10) to determine motor START/STOP or PWM duties it's dangerous, since there wouldn't be high impedance outputs: Current would flow back during the low semiperiod of PWM when the motor is turning. For on/off purposes or PWM speed control , pins 1 and 9 should be used.

Voltage specification

The voltage (Vcc) needed to for its own working is 5V but L293d will not use that Voltage to drive DC Motors. That means you should provide that voltage (36V maximum) to drive the motors. A maximum current of 600mA per output is allowed.

2.2.8 DC MOTOR

100RPM Centre Shaft Economy Series DC Motor is high quality low cost DC geared motor. It has steel gears and pinions to ensure longer life and better wear and tear properties. The gears are fixed on hardened steel spindles polished to a mirror finish. The output shaft rotates in a plastic bushing. The whole assembly is covered with a plastic ring. Gearbox is sealed and lubricated with lithium grease and require no maintenance. The motor is

screwed to the gear box from inside.The inside view of DC motor is shown in figure 2.14.

Figure 2.14: Inside view of Dc motor

Although motor gives 100 RPM at 12V but motor runs smoothly from 4V to 12V and gives wide range of RPM, and torque. Tables below gives fairly good idea of the motor's performance in terms of RPM and no load current as a function of voltage and stall torque, stall current as a function of voltage.

For compatible wheels refer to Wheels and Accessories product category.

You can also mount this motor on the chassis using Motor Mount for Centre Shaft Economy Series DC Motor(shown in figure 2.15.). For adding Position Encoder, refer to Encoder Kit for Centre Shaft Economy Series DC Motor

Figure 2.15: DC motor

Specifications

- DC supply: 4 to 12V
- RPM: 100 at 12V
- Total length: 46mm
- Motor diameter: 36mm
- Motor length: 25mm
- Brush type: Precious metal
- Gear head diameter: 37mm
- Gear head length: 21mm
- Output shaft: Centred
- Shaft diameter: 6mm
- Shaft length: 22mm
- Gear assembly: Spur
- Motor weight: 100gms

2.3 Summary

This chapter consists of block diagram of both, parking and vehicle sections. And it also describes about the system design and hardware requirements for this project.

Chapter 3

Implementation

This chapter provides a detailed discussion of the implementation phase of the automatic vehicle parking system.

3.1 Circuit Description

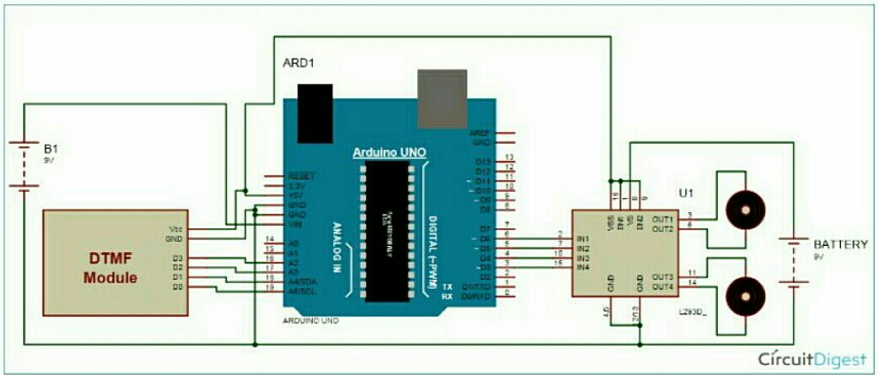


Figure 3.1: Vehicle Section

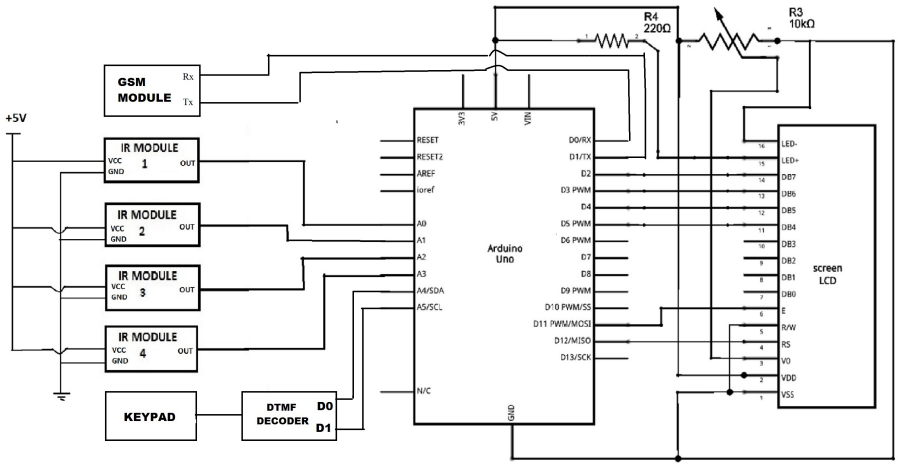


Figure 3.2: Parking Section

Our project consist of two sections, vehicle section and parking section shown in figure 3.2. The power supply is needed to run the circuit of both section, here required supply is 5-12 v.

At the parking section, a 16x2 LCD display is provided at the entrance of the parking gate, which displays the availability of the vacant slots, the LCD output pin is connected to the digital pin of Arduino. IR module is used to sense the vacant slots , here the IR module is connected to the analog pin of Arduino. From the figure 3.2 we can see the DTMF decoder is also connected to the analog pin of Arduino which is used to assign the slot. We use the GSM module to send the messege to the owner of the vehicle, the RX and TX pin of GSM module is connected to the TX and RX pin of arduino respectively.

The IR module sense the vacant slot and transmits it to Arduino, where the Arduino sends the signal to the LCD display, when the vehicle owner enters at the parking gate, he sees the vacant slot on LCD and chooses the slot for parking, the staff of the parking lot assigns that slot by pressing the slot number in keypad which is connected to the DTMF decoder HT9170, after the vehicle reaches to the assigned slot, the GSM module in the parking area sends the message to the owner that the vehicle is parked safely.

At the vehicle section, the motor driver L293D is connected to the digital pin of Arduino which drives the motor present in robotic vehicle, we can see in figure 3.1, the DTMF decoder HT9170 is connected to the analog pin of Arduino . when the owner of the vehicle comes to the entrance of the parking gate, he sees the vacant slot displayed on the LCD, he presses the vacant slot number in his mobile, then the DTMF decoder present in vehicle section decodes the code into binary and sends it to the Arduino. Using the motor driver the robotic vehicle moves to that assigned slot automatically.

3.2 Implementation

3.2.1 Software Used

The Arduino Uno can be programmed with the Arduino software (download). Select ”Arduino Uno w/ ATmega328” from the Tools ¿ Board menu (according to the microcontroller on your board). For details, see the reference and tutorials. The

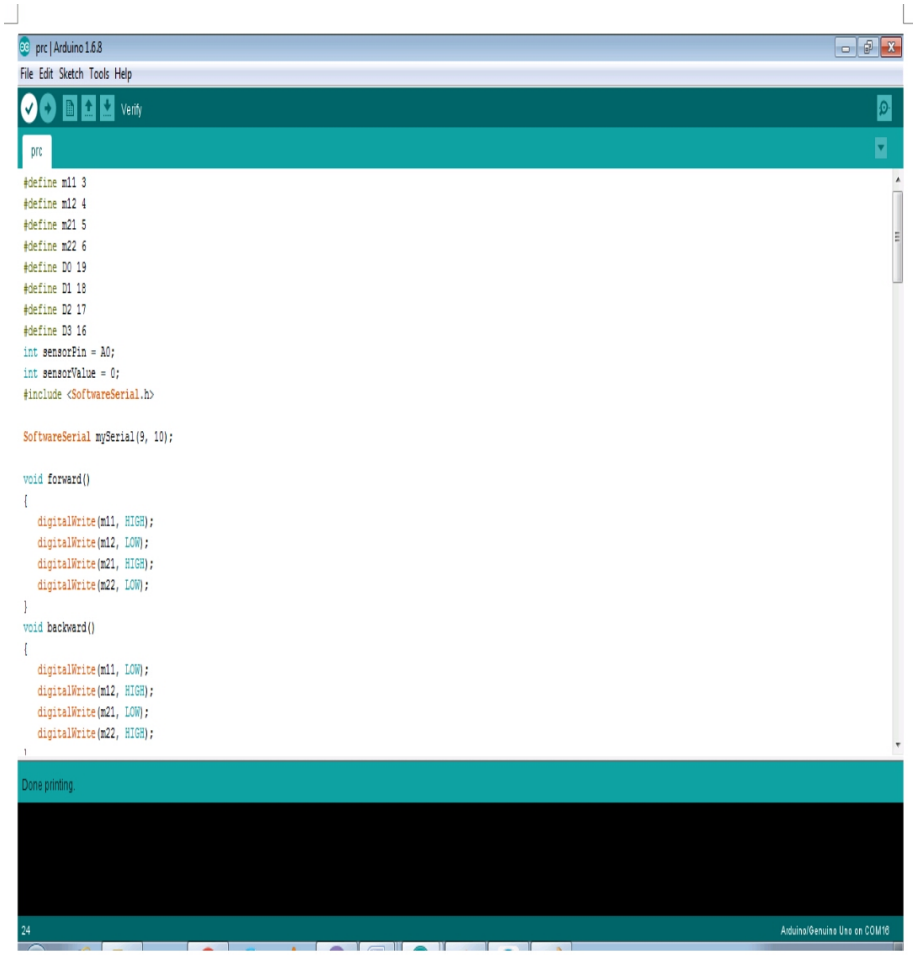


Figure 3.3: Arduino software

ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files). You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details. The ATmega8U2 firmware source code is available . The ATmega8U2 is loaded with a DFU bootloader, which can be activated by connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader).

3.2.2 Flowchart

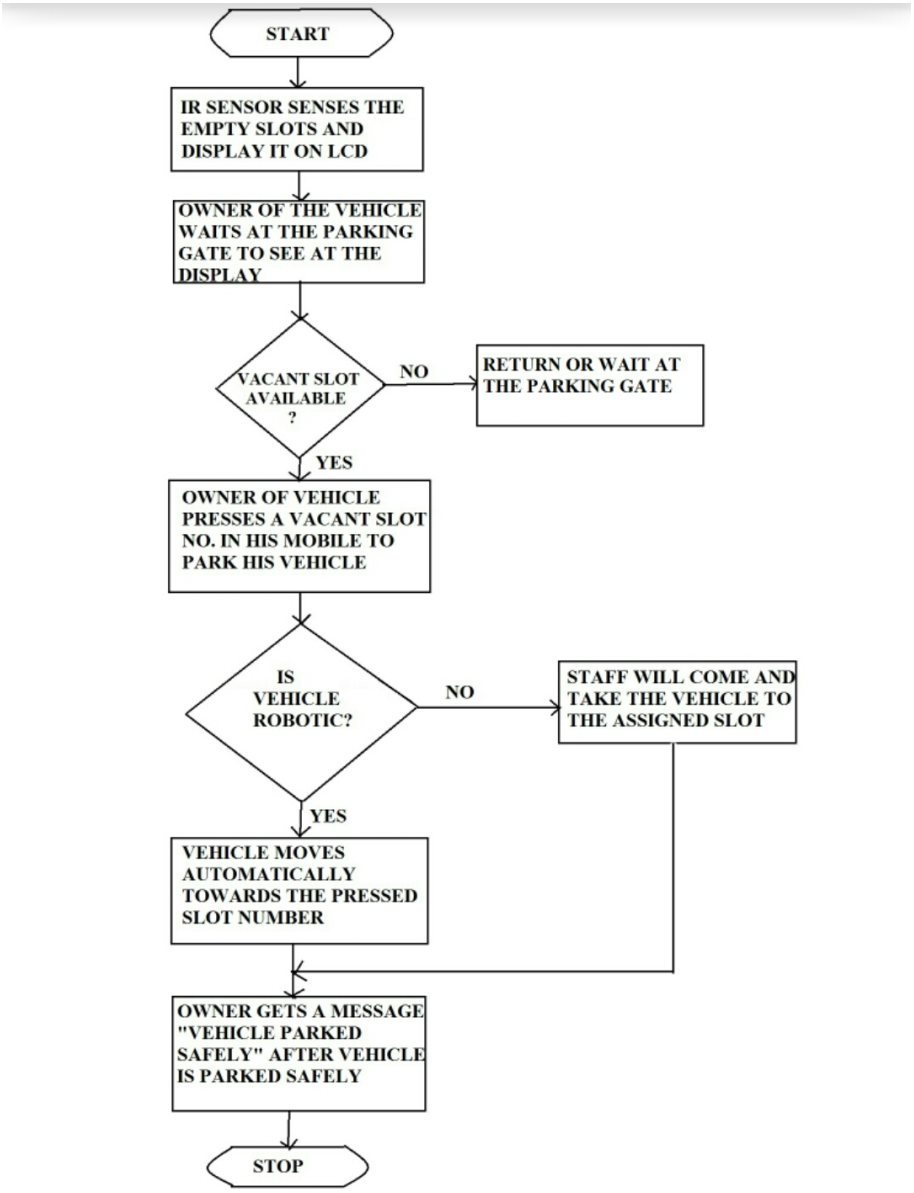


Figure 3.4: Flowchart

The algorithm for this process is as follows:

Step1: Start

Step2: IR sensors senses the vacant slots and displays it on the LCD display at the gate

Step3: Owner of the vehicle waits at the gate of the parking lot to see the vacant slots displayed

Step4: If any slot is vacant then goto step5 or wait at entrance or leave the parking lot

Step5:Owner of the vehicle presses a vacant slot where he wants to park his vehicle

Step6: If the vehicle is robotic then goto step7 else goto step8

step7: Vehicle moves automatically towards the pressed slot number.

Step8: A staff will come and take the vehicle at the assigned slot

Step9: Owner gets a message ”vehicle parked safely” after the vehicle is parked at the assigned slot

Step10: Stop

3.3 Summary

This chapter consists of circuit diagram and describe about its operations in both the sections. It describes the software used for our project and its implementations. This section shows the flowchart of the overall system.

Chapter 4

Results and Discussions

This chapter provides the results of the automatic vehicle parking system implemented using microcontroller. The automatic vehicle parking system algorithm has been simulated using Arduino programming language. The complete system is implemented using arduino software on windows 7 ultimate, intel core i3 processor.

4.1 Results

The vacant slots are displayed on the LCD display.
The owner presses the vacant slot number in his mobile.
If the vehicle is robotic, it will move and park itself to the assigned slot.
If it ois non robotic vehicle then parking is done by the staff in the parking section.
After vehicle reaches to assigned slot , the message is sent to the owner of the vehicle that ”Vehicle parked safely”.

4.2 Conclusion

1.The project has been successfully implemented to park the vehicle automatically and to intimate the parking information to the user.
2.In this project two sections-Parking and Vehicle sections were implemented. The IR reflectance module gives the information regarding the vacant slot.
3.The basic component in this project is the microcontroller which controls all the actions performed by various devices.
4.LCD is provided in this project in order to intimate the user about the vacant slots. Since this project is a demonstration unit, only four slots are assumed for parking.

4.3 Future Scope

1. At present we have implemented only for four slots. We can increase the number of slots as well as the number of floors in order to utilize the parking space to its maximum.

2. RFID system can be used for authorized people only and can be used for more safer parking system.

3.The vibration sensor can be used for safer parking

Appendix A

Code Listing

A.1 Codes for automatic vehicle parking

A.1.1 For parking section

```
int sensorPin1 = A0;//initialize the arduino input pin for ir
sensor int sensorPin2 = A1; //initialize the arduino input pin for
ir sensor int sensorPin3 = A2; //initialize the arduino input pin
for ir sensor int sensorPin4 = A3; //initialize the arduino input
pin for ir sensor

int sensorValue1 = 0; //set the initial value zero int
sensorValue2 = 0; //set the initial value zero int sensorValue3 =
0; //set the initial value zero int sensorValue4 = 0; //set the
initial value zero #define D0 19 //define dtmf input pin #define
D1 18//define dtmf input pin

#include <LiquidCrystal.h>//include the library code for LCD
#include <SoftwareSerial.h>include the library code for GSM

SoftwareSerial mySerial(9, 10);//initialize the library with  the
number of interface pins.

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //initialize the library
with  the number of interface pins.
```

```
void SendSms() { mySerial.println("AT+CMGF=1");    //Sets the GSM
Module in Text Mode delay(1000);  // Delay of 1000 milli seconds
or 1 second mySerial.println("AT+CMGS=\"+918197975954\"\r"); //
Replace x with mobile number delay(1000);
mySerial.println("Vehicle parked safely");// The SMS text you want
to send
delay(100);
mySerial.println((char)26);// ASCII code of
CTRL+Z
delay(1000); }


void setup() {
pinMode(D0, INPUT);
pinMode(D1, INPUT);
Serial.begin(9600);
lcd.begin(16, 2);//setup the LCD the number of
columns and rows
mySerial.begin(9600);    // Setting the baud rate
of Serial Monitor (Arduino)
delay(100);

}

void loop() { int temp1=digitalRead(D0);
int
temp2=digitalRead(D1);

  sensorValue1 = analogRead(sensorPin1);
//Serial.println(sensorValue1);
//delay(1000);
  if(sensorValue1<5)
  {
```

```
  lcd.setCursor(0,0);
  lcd.print("A-01");
  }
  if(sensorValue1>500)
  {
  lcd.setCursor(0,0);
  lcd.print("NA-1");
}
  sensorValue2 = analogRead(sensorPin2);
  Serial.println(sensorValue2);
delay(1000);
  if(sensorValue2<550)
  {

  lcd.setCursor(8,0);
  lcd.print("A-02");
  }
  if(sensorValue2>700)
  {
  lcd.setCursor(8,0);
  lcd.print("NA-2");
  }
sensorValue3 = analogRead(sensorPin3);
//  Serial.println(sensorValue);
//  delay(100);
  if(sensorValue3<5)
  {

  lcd.setCursor(0,1);
  lcd.print("A-03");
  }
  if(sensorValue3>500)
```

```
  {
  lcd.setCursor(0,1);
  lcd.print("NA-3");
  }

  sensorValue4 = analogRead(sensorPin4);
//Serial.println(sensorValue4);
//delay(100);
  if(sensorValue4<6)
  {

  lcd.setCursor(8,1);
  lcd.print("A-04");
  }
  if(sensorValue4>500)
  {
  lcd.setCursor(8,1);
  lcd.print("NA-4");
  }
  if(temp1==0 && temp2==1 && sensorValue1>500 )
  {
   SendSms();
   loop();
  }
   if(temp1==1 && temp2==0 && sensorValue2>500 )
  {
   SendSms();
   loop();
  }
   if(temp1==1 && temp2==1 && sensorValue3>500 )
  {
   SendSms();
   loop();
```

```
  }
   if(temp1==0 && temp2==0 && sensorValue4>500 )
  {
   SendSms();
   loop();
  }
  }
```

A.1.2 For Vehicle Section

```
#define m11 3//Define motor 1 interfacing pin
#define m12//Define
motor 1 interfacing pin
#define m21 5// Define motor 2 interfacing
pin
#define m22 6// Define motor 2interfacing pin
#define D0
19//Define dtmf interfacing pin
#define D1 18//Define dtmf
interfacing pin
#define D2 17//Define dtmf interfacing pin
#define
D3 16//Define dtmf interfacing pin void forward() {
   digitalWrite(m11, HIGH);
   digitalWrite(m12, LOW);
   digitalWrite(m21, HIGH);
   digitalWrite(m22, LOW);
} void backward() {
   digitalWrite(m11, LOW);
   digitalWrite(m12, HIGH);
   digitalWrite(m21, LOW);
   digitalWrite(m22, HIGH);
} void left() {
```

```
   digitalWrite(m11, HIGH);
   digitalWrite(m12, LOW);
   digitalWrite(m21, LOW);
   digitalWrite(m22, LOW);
} void right() {
   digitalWrite(m11, LOW);
   digitalWrite(m12, LOW);
   digitalWrite(m21, HIGH);
   digitalWrite(m22, LOW);
} void Stop() {
   digitalWrite(m11, LOW);
   digitalWrite(m12, LOW);
   digitalWrite(m21, LOW);
   digitalWrite(m22, LOW);
} void setup() {
  pinMode(D0, INPUT);
  pinMode(D1, INPUT);
  pinMode(D2, INPUT);
  pinMode(D3, INPUT);
  pinMode(m11, OUTPUT);
  pinMode(m12, OUTPUT);
  pinMode(m21, OUTPUT);
  pinMode(m22, OUTPUT);
} void loop() {
  int temp1=digitalRead(D0);
  int temp2=digitalRead(D1);
  int temp3=digitalRead(D2);
  int temp4=digitalRead(D3);
 //for slot1
  if(temp1==1 && temp2==0 && temp3==0 && temp4==0)
  {
  digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH); delay(6000);
```

```
   digitalWrite(m11, LOW);
   digitalWrite(m21, LOW);

   exit(0);
  }
//for slot2
   if(temp1==0 && temp2==1 && temp3==0 && temp4==0)
  {
  digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH); delay(2400);
    digitalWrite(m21, HIGH);
   digitalWrite(m11, LOW);delay(2250);
   digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH); delay(1000);
   digitalWrite(m21, LOW);
   digitalWrite(m11, HIGH);delay(2250);
   digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH);delay(2400);
   digitalWrite(m11, LOW);
   digitalWrite(m21, LOW);
   exit(0);
  }
//for slot3
  if(temp1==1 && temp2==1 && temp3==0 && temp4==0)
  {
  digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH); delay(2400);
    digitalWrite(m21, HIGH);
   digitalWrite(m11, LOW);
   );delay(2250);
   digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH); delay(2400);
   digitalWrite(m21, LOW);
```

```
   digitalWrite(m11, HIGH);delay(2250);
   digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH);delay(2400);
   digitalWrite(m11, LOW);
   digitalWrite(m21, LOW);
   exit(0);
  }
//for slot4
   if(temp1==0 && temp2==0 && temp3==1 && temp4==0)
  {
  digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH); delay(2400);
    digitalWrite(m21, HIGH);
   digitalWrite(m11, LOW);delay(2250);
   digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH); delay(4600);
   digitalWrite(m21, LOW);
   digitalWrite(m11, HIGH);delay(2300);
   digitalWrite(m11, HIGH);
   digitalWrite(m21, HIGH);delay(2400);
   digitalWrite(m11, LOW);
   digitalWrite(m21, LOW);
   exit(0);
  }
//for stop
  if(temp1==1 && temp2==0 && temp3==1 && temp4==0)
  {
    Stop();
  }
}
```