

A Project Report

On

PARKING INFORMATION SYSTEM

Submitted in partial fulfillment of the requirements

for the award of the degree of

Bachelor of Technology

In

Electronics & Communication Engineering

Manish Kumar

1422031017

Rajat Tyagi

1422031023

Rocky Bhushan

1422031024

Upendra Kumar

1422031028

Mr. Raj Gopal Mishra
(Project Guide)



Prof. Vivek Kumar Chopra
(H.O.D-EC/EE)



Ghaziabad

Affiliated to

Dr. A.P.J Abdul Kalam Technical University, Lucknow

CANDIDATE’S DECLARATION

We, **Manish Kumar, Rajat Tyagi, Rocky bhushan** and **Upendra kumar** hereby declare that the work which is being presented in this project entitled “**Parking Information System**”, in partial fulfillment of the requirement for the award of the degree of “Bachelor of Technology in Electronics & Communication Engineering”, submitted to Dr. A.P.J. Abdul Kalam Technical University Lucknow, is an authentic record of our work carried out during the period from (Jan 2018 to April 2018) under the guidance of **Mr. Raj Gopal Mishra**.

The work reported in this dissertation has not been submitted by us for award of any other degree or diploma.

1. Manish Kumar (1422031017)

2. Rajat Tyagi (1422031023)

3. Rocky Bhushan (1422031024)

4. Upendra kumar (1422031028)

Date:

Place: Ghaziabad

ACKNOWLEDGEMENT

With a deep sense of gratitude, we wish to express our sincere thanks to our supervisor, **Mr. Raj Gopal Mishra**, for his immense help in planning and executing the work in time. The confidence and dynamism with which he guided the work requires no elaboration. His valuable suggestions as final words during the course of work are greatly acknowledged. What we know today about the process of project, we learned from **Mr. Raj Gopal Mishra**.

Our sincere thanks to **Mrs. Anuja Gupta** (project coordinator) for providing us constant encouragement. Our Special thanks to **Dr. Vihang Garg** (Vice Chairman), **Dr. Dhiraj Gupta** (Director) and **Prof. Vivek kumar Chopra** (Head of department, Department of Electronics and communication) for extending timely help in carrying out our important pieces of work. The cooperation we received from other faculty members of our department is gratefully acknowledged. We will be failing in our duty if we do not mention the laboratory staff and administrative staff of this department for their timely help.

Finally, we would like to thank all whose direct and indirect support helped us in completing our project in time.

1. Manish Kumar
2. Rajat Tyagi
3. Rocky Bhushan
4. Upendra kumar

CERTIFICATE

This is to certify that the project report (NEC-851) entitled “**Parking Information System**” submitted by Manish kumar(1422031017), Rajat Tyagi(1422031023), Rocky Bhushan(1422031024) and Upendra kumar(1422031028) is a bona-fide record of the work carried out by them at HIET, Ghaziabad under my supervision. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

Date:

Mr. Raj Gopal Mishra
Asst. Professor

Mrs Anuja Gupta
Project Coordinator

Prof. Vivek kumar Chopra
H.O.D(EC/EE)

ABSTRACT

The primary objective of this work is to design a parking guidance system to reliably detect entering/exiting vehicles to a parking garage in a cost-efficient manner. Existing solutions (inductive loops, RFID based systems, and video image processors) at shopping malls, universities, airports etc., are expensive due to high installation and maintenance costs. There is a need for a parking guidance system that is reliable, accurate, and cost-effective. The proposed parking guidance system is designed to optimize the use of parking spaces and to reduce wait times. Based on a literature review we identify that the ultrasonic sensor is suitable to detect an entering/exiting vehicle. Initial experiments were performed to test the sensor using an Arduino based embedded system.

CONTENTS

<i>Candidate's declaration</i>	<i>i</i>
<i>Acknowledgement</i>	<i>ii</i>
<i>Certificate</i>	<i>iii</i>
<i>Abstract</i>	<i>iv</i>
<i>List of Figures</i>	<i>ix-x</i>
<i>List of Tables</i>	<i>xi</i>

CHAPTER-1 INTRODUCTION

1.1 Motivation for Parking Information System	2
1.2 Proposed Embedded System	2
1.2.1 Overview	2
1.2.2 Advantages and disadvantages	3
1.3 Schematics	4
1.3.1 Display Domain Schematic	4
1.3.2 Security Domain Schematic	5
1.4 Novelty and Contributions	6
1.5 Organization of Thesis	6
1.6 Chapter Summary	6

CHAPTER-2 THEORY AND WORKING

2.1 Working	7
2.1.1 Display domain	7
2.1.2 Security domain	7
2.2 Block diagram	8
2.3 Flow Chart	9
2.3.1 Security Domain Logic	9

2.4 Chapter Summary	10
---------------------	----

CHAPTER-3 MODULES

3.1 Arduino	11
3.1.1 Arduino Uno	11
3.1.2 Arduino Nano	12
3.2 PIS Modules	13
3.2.1 RFID MFRC522	13
3.2.2 ESP8266	15
3.3 Chapter Summary	17

CHAPTER-4 COMPONENTS AND SENSORS

4.1 Stepper Motor	18
4.1.1 Features	19
4.1.2 Application	19
4.2 ULN2003A	20
4.2.1 Darlington Transistor	21
4.2.2 Application	21
4.3 Stepper Motor and Arduino Nano Interfacing with ULN2003A	22
4.4 LD33V Voltage Regulator	23
4.5 Ultrasonic Sensor HC-SR04	24
4.5.1 HC-SR04 Parameters	25
4.5.2 Timing Diagram	26
4.5.3 Arduino Uno and HC-SR04 Interfacing	27
4.5 Chapter Summary	27

CHAPTER-5 POWER SUPPLY

5.1 Methods to power up the Arduino	28
5.1.1 Using USB Cable	28
5.1.2 Using DC Adapter	29
5.1.3 Using 9V Battery	30
5.2 Arduino Cable	31
5.2.1 Arduino Uno Cable	31
5.2.2 Arduino Nano Cable	33
5.3 Modules and Components power supply of PIS	35
5.4 Chapter Summary	35

CHAPTER-6 SOFTWARE

6.1 Different Languages and tool used in PIS	36
6.1.1 Arduino IDE	36
6.1.2 HTML and CSS	38
6.1.3 PHP	39
6.2 Online Free Platforms	40
6.2.1 ThingSpeak	41
6.2.2 000WebhostApp.com	44
6.2.3 AppsGeyser	46
6.3 Our Website	51
6.4 Chapter Summary	54

CHAPTER 7 PROGRAMMING

7.1 Arduino IDE Algorithms	55
----------------------------	----

7.1.1 Display Domain Algorithm	56
7.1.2 Security Domain Algorithm	57
7.2 Library Used	57
7.3 Chapter Summary	58
<u>CHAPTER 8 CONCLUSION AND FUTURE SCOPE</u>	
8.1 Conclusion	59
8.2 Future Scope	59
8.2.1 OTA (Over The Air Technology) Updates in esp32	61
8.3 Chapter Summary	63
<u>REFERENCES</u>	64
<u>APPENDIX</u>	66

LIST OF FIGURES

Serial No	Title	Page No
Figure 1.1	Preliminary CO2 Estimation by CUTR	3
Figure 1.2	Display Domain Schematic	4
Figure 1.3	Security Domain Schematic	5
Figure 2.1	Block diagram of parking information system	8
Figure 2.2	Security domain logic flow chart	9
Figure 3.1	Arduino Uno	12
Figure 3.2	Arduino Nano	13
Figure 3.3	RFID MFRC522	13
Figure 3.4	ESP8266	15
Figure 3.5	ESP8266EX block diagram	16
Figure 4.1	Stepper Motor 28BYJ-48	18
Figure 4.2(a)	ULN2003A Pins	21
Figure 4.2(b)	UL2003A IC	21
Figure 4.3	ULN2003A Inside Circuit	22
Figure 4.4	LD33V	23
Figure 4.5	HC-SR04	24
Figure 4.6	Timing Diagram	26
Figure 5.1	Uno power up by power bank	29
Figure 5.2	Uno Power up by DC Adapter	29
Figure 5.3	Arduino Uno Barrel plug	30
Figure 5.4	Arduino Power Up by 9V Battery	31
Figure 5.5(a)	Arduino Uno Cable	32
Figure 5.5(b)	USB type B to type A Cable	32
Figure 5.6(a)	Mini USB A and B Plug	33
Figure 5.6(b)	Mini USB B to USB type A	33

Figure 5.7	USB Plug Pins	34
Figure 6.1	Arduino IDE in which AnalogReadSerial Example is written	37
Figure 6.2	Home Page of ParkingInfo.000webhostapp.com our website	38
Figure 6.3	My Channel Screenshot	40
Figure 6.4	New Channel Entry Screenshot	41
Figure 6.5	Chart of Parking Slot 1	43
Figure 6.6	AppsGeyser Homepage	46
Figure 6.7	AppsGeyser Create App	47
Figure 6.8	AppsGeyser Create Smart URL App	48
Figure 6.9	AppsGeyser App Name	49
Figure 6.10	AppsGeyser Click on Create	50
Figure 6.11	ParkingInfo Home Page	51
Figure 6.12	ParkingInfo Area Section	52
Figure 6.13	ParkingInfo Ghaziabad Area	52
Figure 6.14	ParkingInfo HIET Parking Area	53
Figure 8.1	Flowchart of the logic to detect parking space availability	60
Figure 8.2	Block Diagram Showing web server and device connectivity	61
Figure 8.3	Block Diagram of Connectivity	62
Figure 8.4	OTA Workflow	63

LIST OF TABLES

Serial No	Title	Page No
Table 3.1	Communication overview for ISO/IEC 14443 A/MIFARE	14
Table 3.2	Pin Configuration	14
Table 3.3	Clarification of RC522 Pins	15
Table 3.4	ESP8266 pins clarification	17
Table 4.1	Absolute Maximum Ratings	20
Table 4.2(a)	ULN and Nano Connection	22
Table 4.2(b)	ULN and Stepper Connection	23
Table 4.3	Electric Parameters	25
Table 4.4	HC-SR04 and Arduino Uno Pin Connection	27
Table 5.1	USB Cable Parameter	32
Table 5.2	USB Plug Pins Signal Table	34
Table 5.3	Modules and Components Power supply table	35

CHAPTER 1

INTRODUCTION

The parking Information system is a component of Intelligent Transportation Systems (ITS) that have the capability of informing drivers of parking availability. Parking Guidance System has been called the GPS of the Parking World. It has become evident that to ensure the highest possible occupancy, optimal space utilization and provide a stress-free pleasurable environment for parking customers, it is necessary to manage the movement of vehicles within these facilities. When it comes to parking, co-ordination is key.

Parking a car is becoming a challenge to drivers. Drivers are frustrated while waiting for a long time in a queue or will just drive through the entire parking lot but still unable to find a parking spot. While searching for a vacant parking spot, fuel is lost, and time is wasted; this also increases the traffic due to slow moving vehicles that are driving around the parking garage. Further it contributes to green house gas emissions (CO₂ emissions). A study from Boston University states that more than 30% of the drivers take around 7.8 minutes to park their vehicle. Parking spaces are necessary to park the vehicle, and the need is increasing day by day as there are more cars on the road than the number of parking spaces. The only way to optimize the use of parking space and reduce the parking wait time is to install a proper parking information system. For this reason, our team decided to improve the parking experience by making a parking information system. The objective of this project is to established a modern and proper parking system, which is reliable and secure. This thesis focuses on designing of an embedded system that detects vehicles when they enter/exit the parking garage. This system is part of an overall parking guidance system that can facilitate the drivers to know the available number of spaces in a parking lot/garage before entering via an android app or website. This system is also focused to improve the security of vehicles over parking spot. We wanted to make a project which can overcome the problem of searching of empty parking area. That's why, we made this project. We have used the appropriate and cheap sensors to detect the vehicle and we also used the wifi module and RFID reader and Writer. Each and every components of parking information system which is being used is very cheap and effective compare with others sensors. Nowadays there is a need of this type of project. These types of project will overcome the many parking problems and it will also save the fuel, money and resources.

1.1 Motivation for Parking Information System

As per the report there are more than 10 million cars in India and Delhi NCR has 84 lakhs car till 2015, but now it is 2018, so we can imagine there are lots of car in Delhi NCR. More cars means more traffic problems and more parking problems. Thus, there is a need to develop a system to detect the location of parking spot and tell to the user. Hence, we had to make a project to overcome the problem of parking and pollution. That's why the parking information system has been made. With the help of this system the Fuel and time can be saved by informing users about the space availability well in advance. With a reliable parking information system, we can reduce the time taken by user to park and decrease CO₂ emissions, thus protecting the environment. Security is also a measure concern for an owner of a vehicle. So, to improve the security of that parking area and spot, barrier and RFID are being used. This will overcome the cases of motor vehicle theft and increase the reliability of a system. According to the CUTR (Center for urban transportation research at the university of south Florida) surveys, if the parking time is reduced by 2 minutes, the CO₂ emissions would be reduced from the three garages to 155 metric tons (53% annual reduction). Figure 1.1 is the preliminary estimation done by CUTR. This primarily motivated us to research a parking information system that is reliable, efficient, and economical.

1.2 Proposed Embedded System

This section discusses the embedded system researched in this project. An overview of the embedded system is given. The advantages and disadvantages are also discussed.

1.2.1 Overview

The idea of using a readymade embedded platform led us to choose Arduino, an open source platform that has easy to use hardware and software. We selected Arduino UNO because of its excellent features and low price. Ultrasonic sensor HC-SR04 is the sensor used for detecting vehicles. A Wi-Fi module ESP8266 is also used to upload the data into server. ESP8266 is a wifi-module. This module is very cheap and not much complex to use. ESP8266 works on a AT Commands. In this, MFRC522 and Stepper are also used. MFRC522 is a RFID Module and Stepper motor is a type of motor which rotates in steps. The Parking information system is separated in two domains, that are Display domain and Security domain. Each domain has been made for its different functionality in a project. Display domain and Security domain have been explained in detail in chapter 2: theory and working.

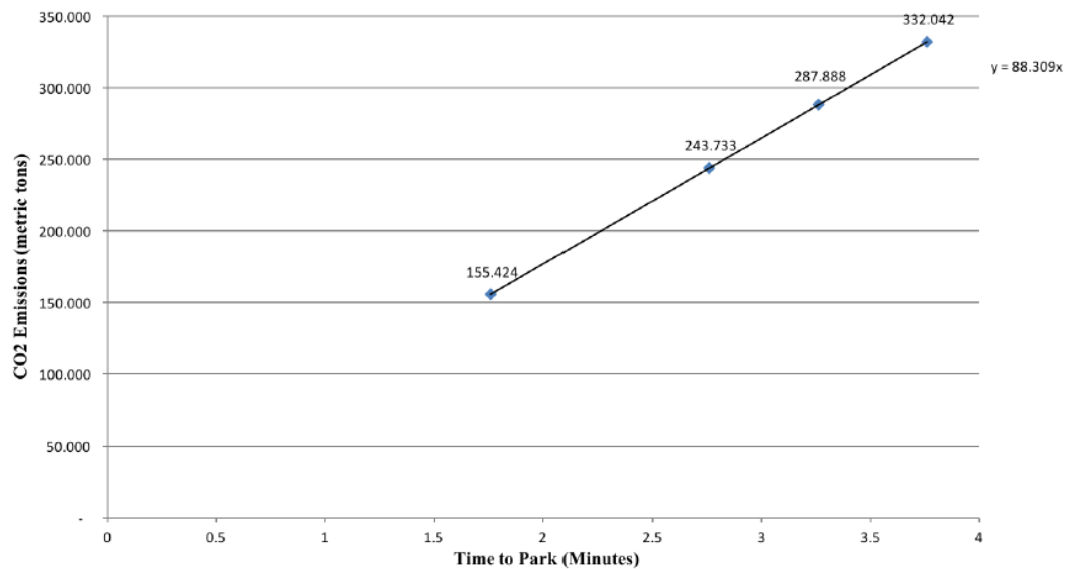


Figure 1.1: Preliminary CO2 Estimation by CUTR

1.2.2 Advantages and disadvantages

This section lists the advantages and disadvantages of the proposed embedded system.

1. Advantages

- Consumes less power
- Very simple to operate
- Easy to install in the garage and any parking spot
- Cost efficient

2. Disadvantages

- Detects people as a car
- Detects animal or anything as a car

These can be overcome by further extending the design as discussed in future scope section. The demerits of parking information system is very less comparison with its merits. The few disadvantages of parking information system will be sorted out soon.

1.3 Schematics

1.3.1 Display Domain Schematics

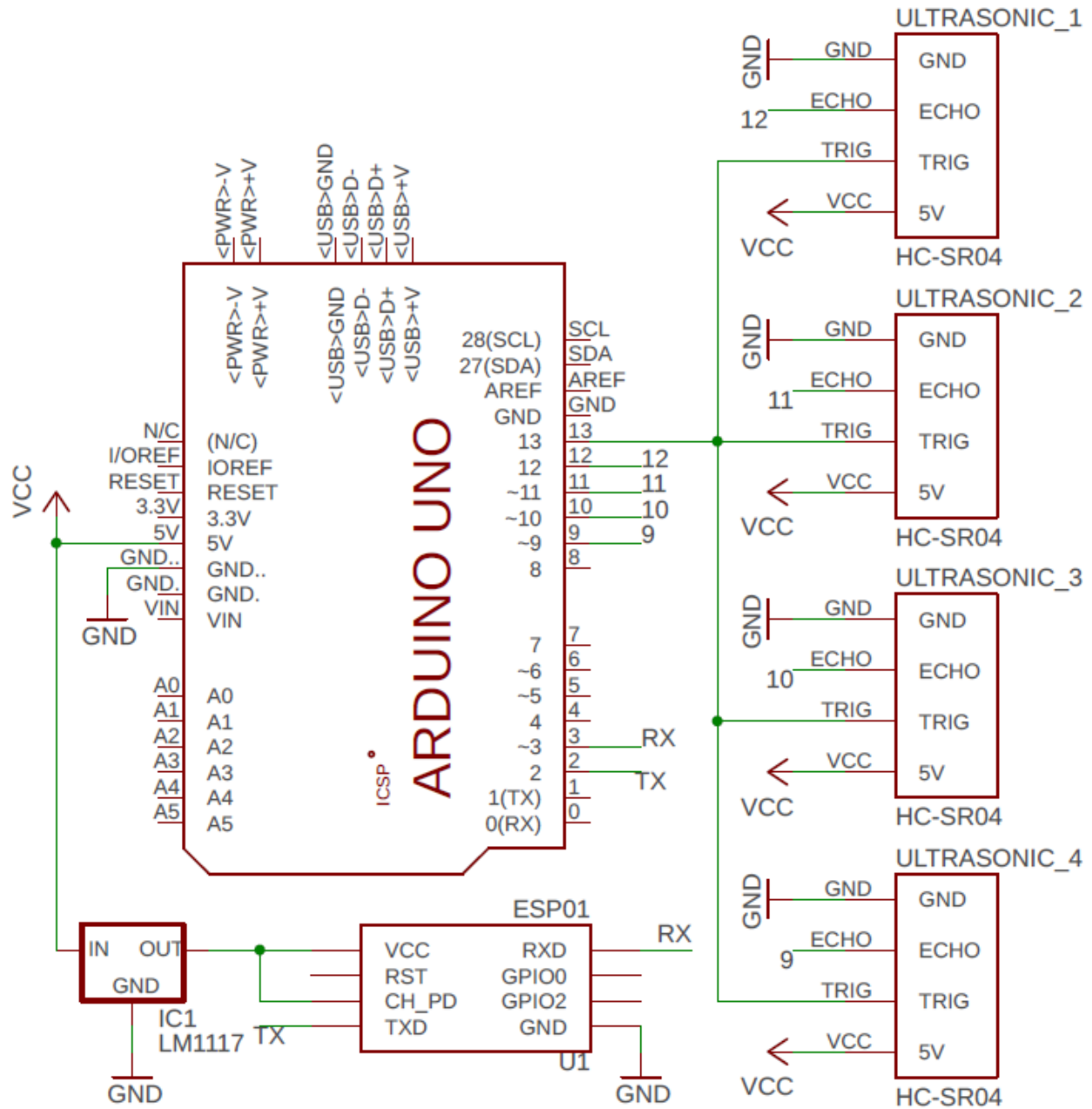


Figure 1.2 Display Domain Schematic

1.3.2 Security Domain Schematic

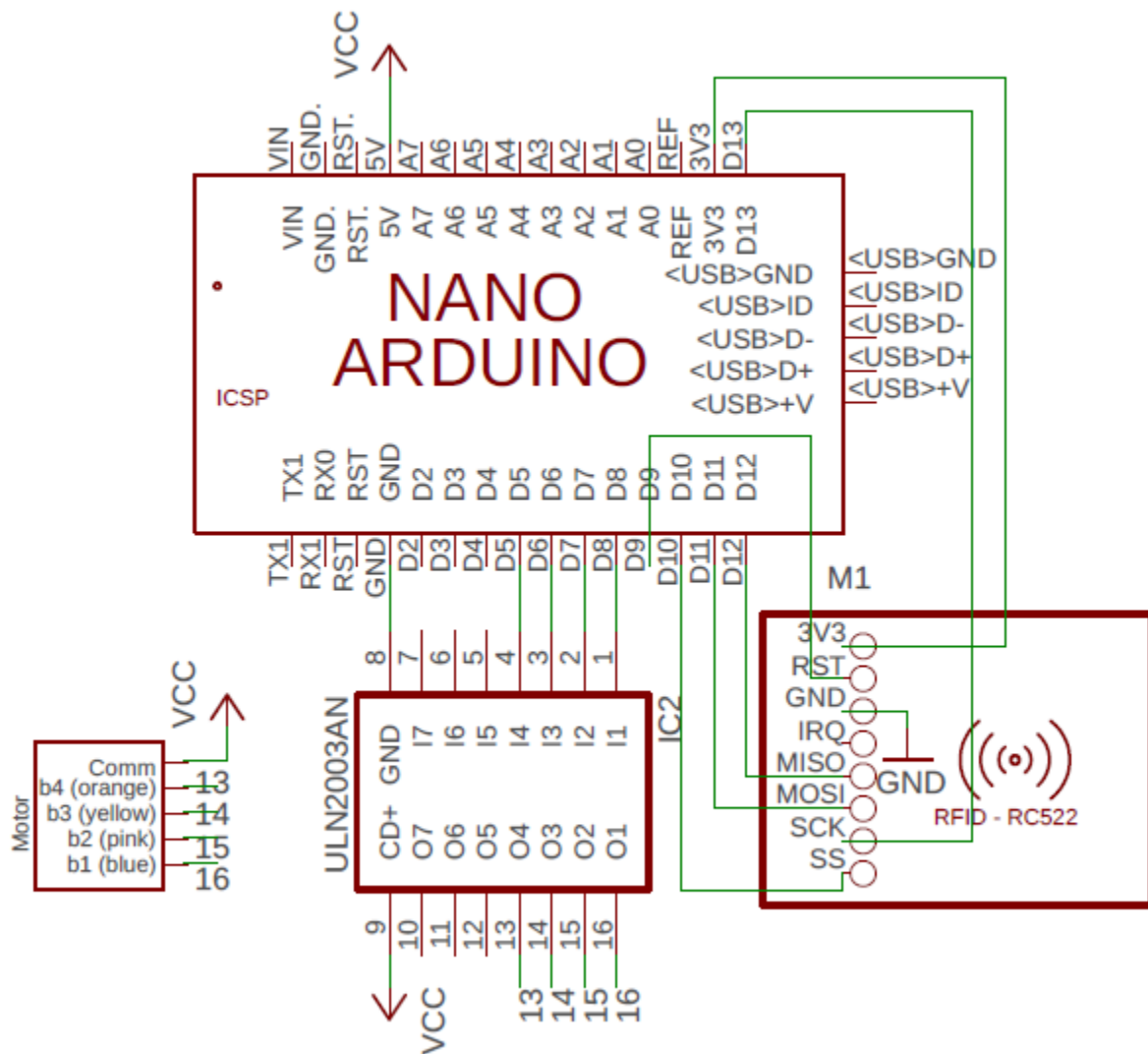


Figure 1.3 Security Domain Schematic

The block diagram of Parking Information System is shown in next chapter in figure 2.1. both the schematics are showing each and every proper connection of Parking Information System Prototype(Model).

1.4 Novelty and Contributions

The proposed parking information system presents a unique method of vehicle detection using an ultrasonic sensor. Although researchers have used ultrasonic sensor to detect empty parking spaces, to best of our knowledge, no prior researcher have used to upload that data on a server in real time. We also show the logic involved in detecting the vehicle using flowcharts. The system is cheaper than any other available parking technologies. We built a working prototype and that prototype can be implemented on any other parking spot. We survey extensively the possible sensors and compare them in detail. This is useful for future researchers working on this problem.

1.5 Organization of Thesis

The rest of the thesis is organized as follows. Chapter 2 presents a theory and working on PIS and chapter 3 is based on the modules which are being used in PIS. Chapter 4 is based on components and sensors. All of these components are explained in details. Chapter 5 is on power supply of PIS prototype and chapter 6 is based on software which are being used in PIS. Chapter 7 is on programming languages and chapter 8 is the last chapter of this thesis which is based on conclusion and future scope.

1.6 Chapter Summary

In this chapter, the challenges involved in parking a vehicle in a high demand parking lot are explained. The motivation behind this project is also discussed. The overview of the embedded system is presented.

CHAPTER 2

THEORY AND WORKING

This chapter is based on the working of parking information system and its parts which can be improved by using some new techniques and factors. Parking information system is divided in two domains 1.) Display domain 2.) Security domain.

In display domain, we are displaying the parking information on a mobile application and website. In security domain, the parking spot which is being displayed is secured with the help of barriers and RFID. So, whole parking information system has two important domains. One is to automate the barrier of parking system and overcome motor vehicle theft.

2.1 Working

There are two microcontrollers modules in parking information system. One is Arduino UNO and second is Arduino Nano. One is used for display domain and other is for security domain. Both the domains are part of project.

2.1.1 Display domain

In this domain the main components are Arduino Uno, Esp8266, ultrasonic sensor and IoT platform. The main objective of this domain is to show the vacant and occupied parking spot on our own website that is parkinginfo.000webhostapp.com and Android application. Arduino uno controls the ultrasonic sensor HC-SR04 by its two important pins 1.) Trigger 2.) Echo. Uno receives the data from Ultrasonic sensor HC-SR04. Firstly, Ultrasonic sensor data is calibrated and uploaded on our website via Esp8266 wifi module. Each and every parking slot information is updated on website is updated in every 15 seconds. Vacant slot is shown as green and occupied as red color.

2.1.2 Security domain

In this domain the components which we used are Arduino nano, RFID MFRC522 and Stepper motor. The main objective of security domain is to established anti vehicle theft parking area or automate parking. In this user can only exit via an exit gate which will be opened only when RFID will match. Each and every user will have their own unique RFID which will be provided to user when owner of vehicle(user) will enter through the entry gate. This will improve the reliability towards user because motor vehicle theft is very big problem in nowadays. Unique ID which is provided to user is very simple to match with RFID Reader RC-522.

2.2 Block Diagram

As we discussed, there are two subsystems in parking information system that are display domain and security domain. Figure 2.1 is showing a block diagram of parking information system consisting all important components the controller module of display domain is Arduino Uno. In security domain it is Arduino Nano. As you can see in the figure 2.1 the parking spot has two slots. Slot 1 and slot 2, each slot is for one single car. Real time value of each slot is captured by ultrasonic sensor and sent to Arduino Uno. Uno uploaded that value to website via Esp8266.

Where, in security domain RFID MFRC522 provide unique ID to user at entry of parking area and match that unique ID at Exit of parking area. Stepper motor BYJ48-28 is used to control the barrier at exit in parking area. If unique ID of user has been matched, then the barrier at exit in parking area will be opened automatically with the help of stepper motor BYJ48-28.

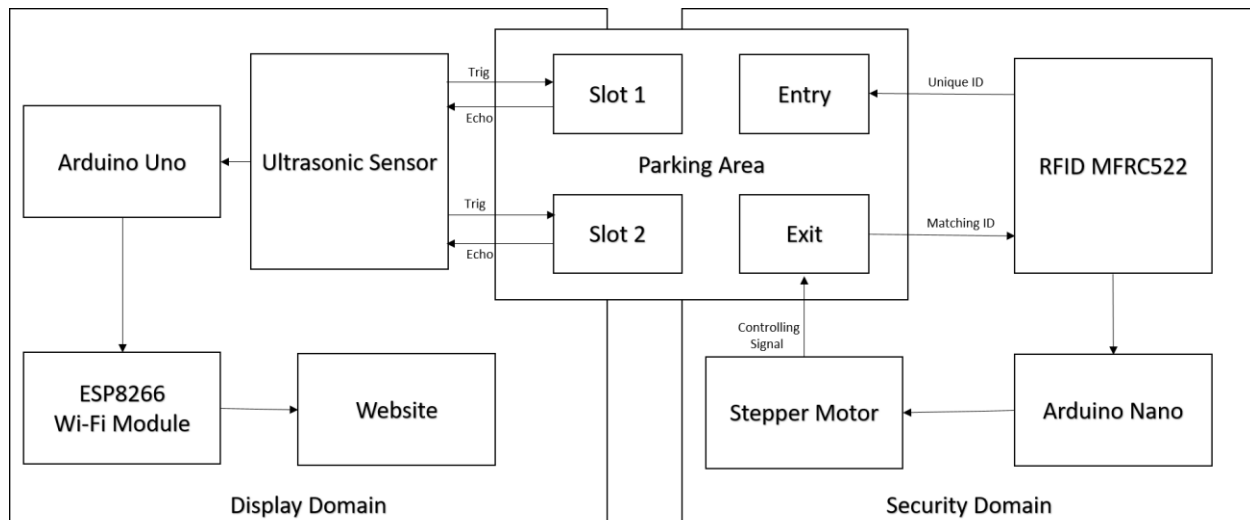


Figure 2.1 Block Diagram of Parking Information System

Each and every component as shown in figure 2.1 is explained in detail in chapter 3 components. The power supply of each component is not shown in the block diagram but power supply is also discussed in chapter 3 components. As you can see in the block diagram there is no need of any website or internet connection in security domain. we could add the new feature in online security by showing the entry and exit of time of user on website.

2.3 Flow Chart

The working and functioning of any logical system is complicated or complex to understand without the using of flow chart. As we know, there are many elements in parking information system and, to understand the working of this project, block diagram and flow chart are must. Security subsystem is works on a logic in which a barrier will open only when the RFID will match. So, flow chart of security system logic is shown in figure 2.2 and discussed in 2.3.1 Security System Logic.

2.3.1 Security Domain Logic

The flow chart is quite simple, if the ID will match then barrier will open. if the will not match then barrier will remain close. The opened time duration of barrier is 10 seconds. After 10 seconds the barrier will close automatically. Arduino Nano is a controlling body of a security system which is a subsystem of parking information system. The delay of 10 seconds for stepper motor is provides by Arduino Nano with the help of Arduino software inbuilt function delay().

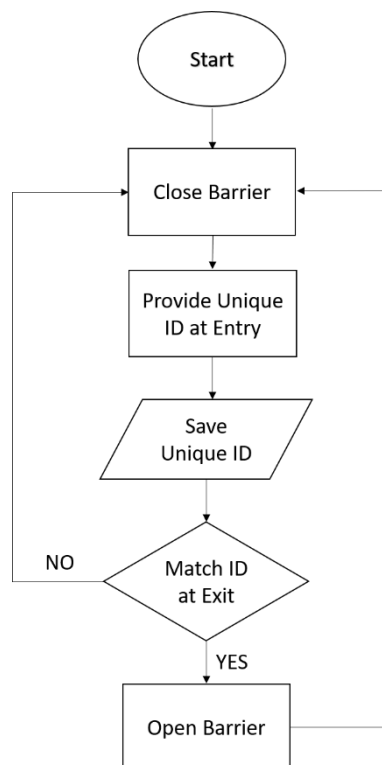


Figure 2.2 Security Domain Logic Flow Chart

2.4 Chapter Summary

In this chapter, the working of parking information system and its subsystems are explained. The block diagram and flow chart of this project are shown and described. Each elements which is used in this project is shown in block diagram. The explanation of each element is explained in next chapter that is chapter 3 components.

CHAPTER 3

MODULES

Components are the basic parts of any project. Each component has their own property and obligation. There are many components and modules are used to make parking information system as proper working system. We know the names of components and modules which we used in this project but now we have to explain each module and component individually. In this chapter we will study each components and modules of parking information system in detail.

3.1 Arduino

Arduino is an open source computer hardware and software company, project, and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical and digital world. We have used Arduino Uno and Nano in our project prototype.

3.1.1 Arduino Uno

Arduino UNO is the microcontroller board used in this project. It has easy to use hardware for prototyping, low cost and has simple software development interface. It contains ATmega328 microcontroller chip with a clock speed of 16 MHz, which accelerates the system response time. ATmega328 is a low power AVR 8-bit microcontroller with 2 KB SRAM. It has 23 programmable I/O lines of which Arduino UNO provides 12 digital I/O pins (2 - 13) and 6 analog I/O pins (A0 - A5) as shown in Figure 3.1. The ATmega328 on the Arduino Uno comes preprogrammed with a bootloader that allows to upload new code to it without the use of an external hardware programmer. As we know, Arduino Uno is used in Display domain of parking information system. So, it is a main controlling body of display domain. It communicates using the original STK500 protocol. The Uno also differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. The Arduino UNO is generally considered the most user-friendly and popular board, with boards being sold worldwide for less than 5\$. Arduino use many types of microcontrollers of AVR. We have also used Arduino as programmer to program Arduino Nano in our project that is parking information system. We took Arduino Uno as master and Arduino Nano as Slave and via the help of ICSP(In Circuit System Programming) we programmed the Arduino Nano through Arduino Uno.



Figure 3.1 Arduino Uno

3.1.2 Arduino Nano

The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328 (Arduino Nano 3.x) or ATmega168 (Arduino Nano 2.x). It has more or less the same functionality of the Arduino Duemilanove, but in a different package. It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard one.

The Arduino Nano can be powered via the Mini-B USB connection, 6-20V unregulated external power supply (pin 30), or 5V regulated external power supply (pin 27). The power source is automatically selected to the highest voltage source. There are two types of microcontroller are used in Nano, that are ATmega168 and ATmega328 with flash memory 1KB and 2KB respectively. We could have used Arduino Uno or any other Arduino type instead of Arduino Nano but it would have been very costly comparison with Arduino Nano. Like any other Arduino, Arduino Nano also has a ICSP pins for programming and Master Slave Configuration.



Figure 3.2 Arduino Nano

3.2 PIS Modules

There are two significant modules in parking information system. Those are RFID module MFRC522 and Wi-Fi Module ESP8266.

3.2.1 RFID MFRC522

The RFID-RC522 module comes with eight pins. In the following, the pin layout is shown as recommended by the documentation of the MFRC522 library. The RFID-RC522 module runs with 3.3V. Therefore, the module's 3.3V pin must be connected to the Arduino's 3.3V. The module might get damaged, if it is accidentally connected to the Arduino's 5V pin. The complete pin layout is shown by the table 3.1. the controller which is used in RC522 is a MIFARE series chip of NXP Semiconductors. The MIFARE name covers proprietary technologies based upon various levels of the ISO/IEC 14443 Type A 13.56 MHz contactless smart card standard.

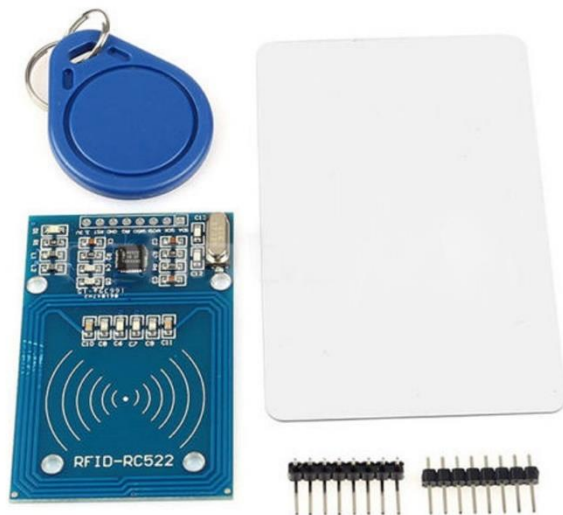


Figure 3.3 RFID MFRC522

The MFRC522's internal transmitter is able to drive a reader/writer antenna designed to communicate with ISO/IEC 14443 A/MIFARE cards and transponders without additional active circuitry. The receiver module provides a robust and efficient implementation for demodulating and decoding signals from ISO/IEC 14443 A/MIFARE compatible cards and transponders. The digital module manages the complete ISO/IEC 14443 A

framing and error detection (parity and CRC) functionality. The MFRC522 image is shown in figure 3.3.

As we know, communication between card and reader is depends upon types of bit encoding, modulation, bit length and subcarrier frequency. So, we should know about it. Table 3.1 is showing a communication overview of RFID card and reader.

Table 3.1 Communication overview for ISO/IEC 14443 A/MIFARE reader/writer

Communication direction	Signal type	Transfer speed			
		106 kBd	212 kBd	424 kBd	848 kBd
Reader to card (send data from the MFRC522 to a card)	reader side modulation	100 % ASK	100 % ASK	100 % ASK	100 % ASK
	bit encoding	modified Miller encoding	modified Miller encoding	modified Miller encoding	modified Miller encoding
	bit length	128 (13.56 μ s)	64 (13.56 μ s)	32 (13.56 μ s)	16 (13.56 μ s)
Card to reader (MFRC522 receives data from a card)	card side modulation	subcarrier load modulation	subcarrier load modulation	subcarrier load modulation	subcarrier load modulation
	subcarrier frequency	13.56 MHz / 16	13.56 MHz / 16	13.56 MHz / 16	13.56 MHz / 16
	bit encoding	Manchester encoding	BPSK	BPSK	BPSK

The pin connection of RC522 with Arduino Uno is fixed. Because Arduino Uno can operate RC522 via I²C Protocol only. The pins of I²C Bus are MISO, MOSI, SDA, SCK, GND, VCC. The proper pin connection with Arduino Uno is shown in table 3.2. I²C uses only two bidirectional open-drain lines, Serial Data Line (SDA) and Serial Clock Line (SCL), pulled up with resistors. Typical voltages used are +5 V or +3.3 V, although systems with other voltages are permitted. Arduino Uno's 13,11,12 pins are SCK, MOSI and MISO respectively. Most of the Atmel atmega series microcontrollers have an I²C bus pins. Each pin has its own functionality and uses. The explanation of each pin is given on Next page. The pins that are shown in table 3.2 are used for I²C, SPI and ICSP Communication between two controllers and modules. The MOSI and MISO signals are usually stable (at their reception points) for the half cycle until the next clock transition. SPI master and slave devices may well sample data at different points in that half cycle. Clarification of RC522 pins is shown in Table 3.3 on next page.

Table 3.2 Pin Configuration

RFID RC522 PIN	ARDUINO NANO PIN
SDA	10
SCK	13
MOSI	11
MISO	12
IRQ	UNUSED
GND	GND
RST	9
3.3V	3.3V

Table 3.3 Clarification of RC522 Pins

Pins	Clarification
SDA	I2C-bus serial data line input/output
SCK	Serial Clock signal which is an output from Master
MOSI	Master Output Slave Input.
MISO	Master Input Slave Output
IRQ	Interrupt request output: indicates an interrupt event.
RST	Reset Pin
VCC	3.3 Input Voltage
GND	Ground

The MFRC522 can act either as a slave receiver or slave transmitter in Standard mode, Fast mode and High-speed mode. As shown in table 3.3 IRQ is an interrupt request output: indicates an interrupt event but in our project, there is no need to use IRQ. Thus, IRQ is unused and disconnected. RFID MFRC522 is operates on 3.3 Volt. If the input voltage is greater than 3.3 or 5 volts then RC522 can be damaged. The range of a RFID Reader can be increased by using a antenna. RFID was first invented by, Charles Walton Born in 1921. he is best known as the first patent holder for the RFID (radio frequency identification) device. Many individuals contributed to the invention of the RFID, but Walton was awarded ten patents in all for various RFID-related devices, including his key 1973 design for a "Portable radio frequency emitting identifier".

3.2.2 ESP8266

The ESP8266 is a low-cost Wi-Fi module with full TCP/IP stack and microcontroller capability produced by Shanghai-based Chinese manufacturer, Espressif Systems. ESP8266 is shown in figure 3.4. ESP8266 Module has ESP8266EX Chip. The chip first came to the attention of western makers in August 2014 with the **ESP-01** module, made by a third-party manufacturer, Ai-Thinker. This small module allows microcontrollers to connect to a Wi-

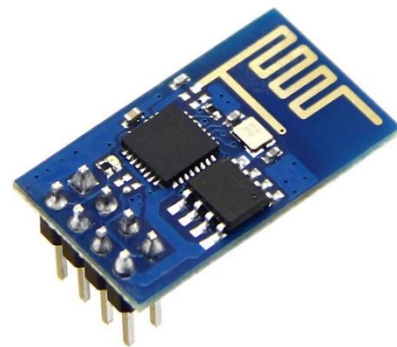


Figure 3.4 ESP8266

Fi network and make simple TCP/IP connections using Hayes-style commands. However, at the time there was almost no English-language documentation on the chip and the commands it accepted. The very low price and the fact that there were very few external components on the module which suggested that it could eventually be very inexpensive in volume, attracted many hackers to explore the module, chip, and the software on it, as well as to translate the Chinese documentation. There are 8 pins in ESP8266 includes two GPIO pins as shown in Figure 3.4.

The Clarification of ESP8266 Wi-Fi module pins is shown in Table 3.4 Pins clarification. The block diagram of ESP8266EX is given below in figure 3.5 ESP8266EX Block diagram.

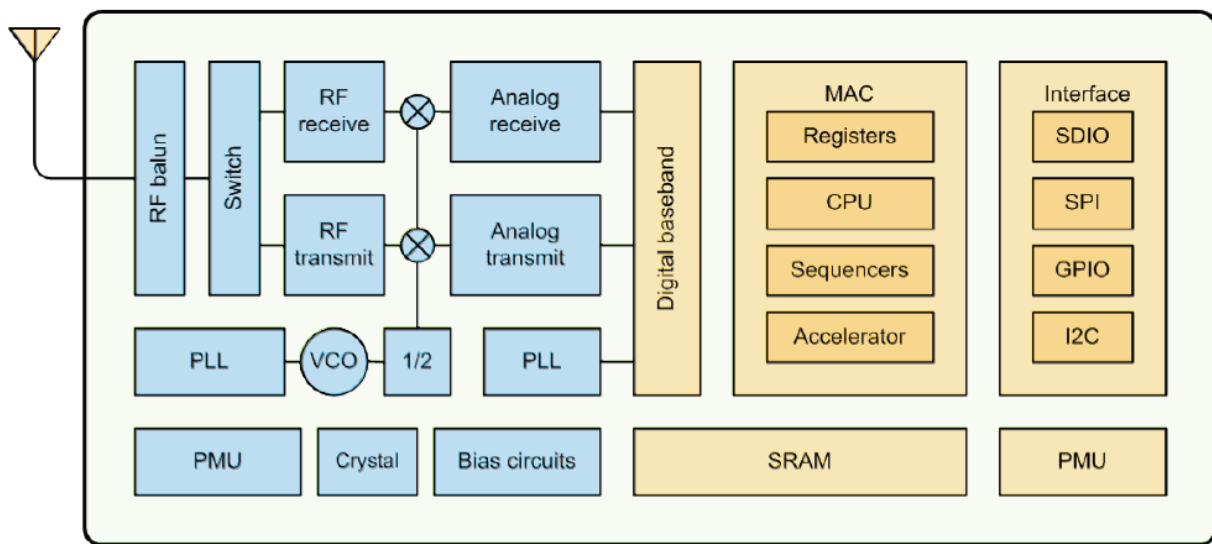


Figure 3.5 ESP8266EX Block diagram.

ESP8266EX offers a complete and self-contained Wi-Fi networking solution; it can be used to host the application or to offload Wi-Fi networking functions from another application processor. When ESP8266EX hosts the application, it boots up directly from an external flash. It has integrated cache to improve the performance of the system in such applications. Alternately, serving as a Wi-Fi adapter, wireless internet access can be added to any micro controller based design with simple connectivity (SPI/SDIO or I2C/UART interface). ESP8266EX is among the most integrated Wi-Fi chip in the industry; it integrates the antenna switches, RF balun, power amplifier, low noise receive amplifier, filters, power management modules, it requires minimal external circuitry, and the entire solution, including front-end module, is designed to occupy minimal PCB area. ESP8266EX also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor, with on-chip SRAM, besides the WiFi functionalities. ESP8266EX is often

integrated with external sensors and other application specific devices through its GPIOs; sample codes for such applications are provided in the software development kit (SDK).

Table 3.4 ESP8266 pins Clarification

ESP8266 PIN	Clarification
Tx	Transmitter
GND	Ground
Rx	Receiver
CH_PD	Chip Enable
GPIO1	General purpose pin 1
GPIO2	General purpose pin 2
RST	Reset
3.3V	3.3V

3.3 Chapter Summary

In this chapter we discussed about Arduino and modules that are used in PIS. As we discussed, we used Arduino Uno and Nano microcontroller modules and for security we used RFID MFRC522. For to internet access using wireless frequency we have used ESP8266 Wi-Fi Module. All the modules are discussed in detail in this chapter. Pin configuration and figures of each module are shown in table and figures. These modules are the pillar of PIS. Each module has its own significance and impact on functioning of PIS in appropriate way. These modules have chosen to be part of PIS because, these are the most suitable and fitting modules for PIS in best of our knowledge.

CHAPTER 4

COMPONENTS AND SENSORS

In this chapter we are going to study the components and sensors that are used in parking information system. As we know, we already discussed about modules. In this chapter we will discuss about an IC ULN2003, Stepper motor and the most important part of our project, Ultrasonic sensor HC-SR04. Let's start with stepper motor.

4.1 Stepper Motor

A **stepper motor** or **step motor** or **stepping motor** is a brushless DC electric motor that divides a full rotation into a number of equal steps. The motor's position can then be commanded to move and hold at one of these steps without any position sensor for feedback (an open-loop controller), as long as the motor is carefully sized to the application in respect to torque and speed. The shaft or spindle of a stepper motor rotates in discrete step increments when electrical command pulses are applied to it in the proper sequence. The motors rotation has several direct relationships to

these applied input pulses. The sequence of the applied pulses is directly related to the direction of motor shafts rotation. The speed of the motor shafts rotation is directly related to the frequency of the input pulses and the length of rotation is directly related to the number of input pulses applied. One of the most significant advantages of a stepper motor is its ability to be accurately controlled in an open loop system.



Figure 4.1 Stepper Motor 28BYJ-48

Open loop control means no feedback

information about position is needed. This type of control eliminates the need for expensive sensing and feedback devices such as optical encoders. Your position is known simply by keeping track of the input step pulses.

4.1.1 Features

- The rotation angle of the 1. motor is proportional to the input pulse.
- The motor has full torque at standstill (if the windings are energized).
- Precise positioning and repeatability of movement since good stepper motors have an accuracy of – 5% of a step and this error is non cumulative from one step to the next.
- Excellent response to starting/stopping/reversing.
- very reliable since there are no contact brushes in the motor. Therefore, the life of the motor is simply dependent on the life of the bearing.
- The motors response to digital input pulses provides open-loop control, making the motor simpler and less costly to control.
- It is possible to achieve very low speed synchronous rotation with a load that is directly coupled to the shaft.
- A wide range of rotational speeds can be realized as the speed is proportional to the frequency of the input pulses.

4.1.1 Applications

Computer controlled stepper motors are a type of motion-control positioning system. They are typically digitally controlled as part of an open loop system for use in holding or positioning applications. In the field of lasers and optics they are frequently used in precision positioning equipment such as linear actuators, linear stages, rotation stages, goniometers, and mirror mounts. Other uses are in packaging machinery, and positioning of valve pilot stages for fluid control systems. Commercially, stepper motors are used in floppy disk drives, flatbed scanners, computer printers, plotters, slot machines, image scanners, compact disc drives, intelligent lighting, camera lenses, CNC machines and, more recently, in 3D printers. Stepper motors are used inside medical scanners, samplers, and also found inside digital dental photography, fluid pumps, respirators and blood analysis machinery. Stepper motors are used in cameras for automatic digital camera focus and zoom functions. Stepper motors are also used in automotive gauges and machine tooling automated production equipment's. Stepper motors are similar to switched reluctance motors. We have used stepper motor because of its precise positioning and repeatability of movement since good stepper motors have an accuracy of 3 to 5 percent of a step and this error is non cumulative from one step to the next. This is a feature of stepper motor.

4.2 ULN2003A

The **ULN2003A** is an IC of array of seven NPN Darlington transistors capable of 500 mA, 50 V output. It features common-cathode fly back diodes for switching inductive loads. It can come in PDIP, SOIC, SOP or TSSOP packaging. In the same family are ULN2002A, ULN2004A, as well as ULQ2003A and ULQ2004A, designed for different logic input levels. The UNL2003A is also similar to the ULN2001A (4 inputs) and the ULN2801A, ULN2802A, ULN2803A, ULN2804A and ULN2805A, only differing in logic input levels (TTL, CMOS, PMOS) and number of in/outputs (4/7/8). We are using a ULN2003A of PDIP packaging.

The ULN2003 is known for its high-current, high-voltage capacity as shown in table 4.1. The

Table 4.1 Absolute Maximum Ratings

Symbol	Parameter	Value	Unit
V_o	Output Voltage	50	V
V_{in}	Input Voltage	30	V
I_c	Continuous Collector Current	500	mA
I_b	Continuous Base Current	25	mA
T_{amb}	Operating Ambient Temperature Range	-20 to 85	°C
T_{stg}	Storage Temperature Range	-55 to 150	°C
T_j	Junction Temperature	150	°C

drivers can be paralleled for even higher current output. Even further, stacking one chip on top of another, both electrically and physically, has been done. Generally, it can also be used for interfacing with a stepper motor, where the motor requires high ratings which cannot be provided by other interfacing devices. The ULN2001A, ULN2002A, ULN2003 and ULN2004A are high voltage, high current darlington arrays each containing seven open collector darlington pairs with common emitters. Each channel rated at 500mA and can withstand peak currents of 600mA. Suppression diodes are included for inductive load driving and the inputs are pinned opposite the outputs to simplify board layout. ULN2003A is a DIP type IC. The body (housing) of a DIP containing an IC chip is usually made from molded plastic or ceramic. The hermetic nature of a ceramic housing is preferred for extremely high reliability devices.

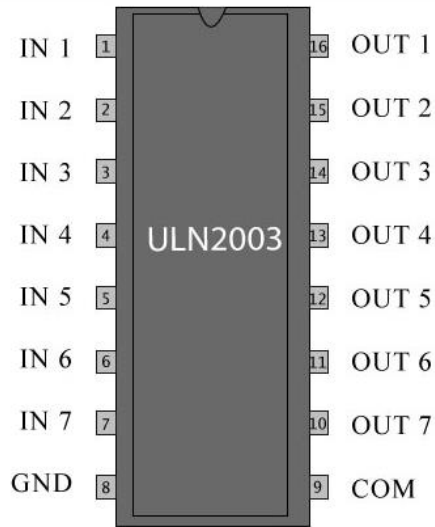


Figure 4.2 (a) ULN2003A Pins



Figure 4.2 (b) ULN2003A IC

4.2.1 Darlington Transistor

A Darlington transistor (also known as Darlington pair) achieves very high current amplification by connecting two bipolar transistors in direct DC coupling so the current amplified by the first transistor is amplified further by the second one. The resultant current gain is the product of those of the two component transistors:

$$\beta_{total} \approx \beta_1 \cdot \beta_2$$

The seven Darlington pairs in ULN2003 can operate independently except the common cathode diodes that connect to their respective collectors.

The inside circuit of ULN2003A is shown in figure 4.2. As you can see in the figure 4.2 all the diodes cathodes are common.

4.2.2 Application

Typical usage of the ULN2003A is in driver circuits for relays, lamp and LED displays, stepper motors, logic buffers and line drivers. We are using this IC to operate our Stepper motor to control barrier.

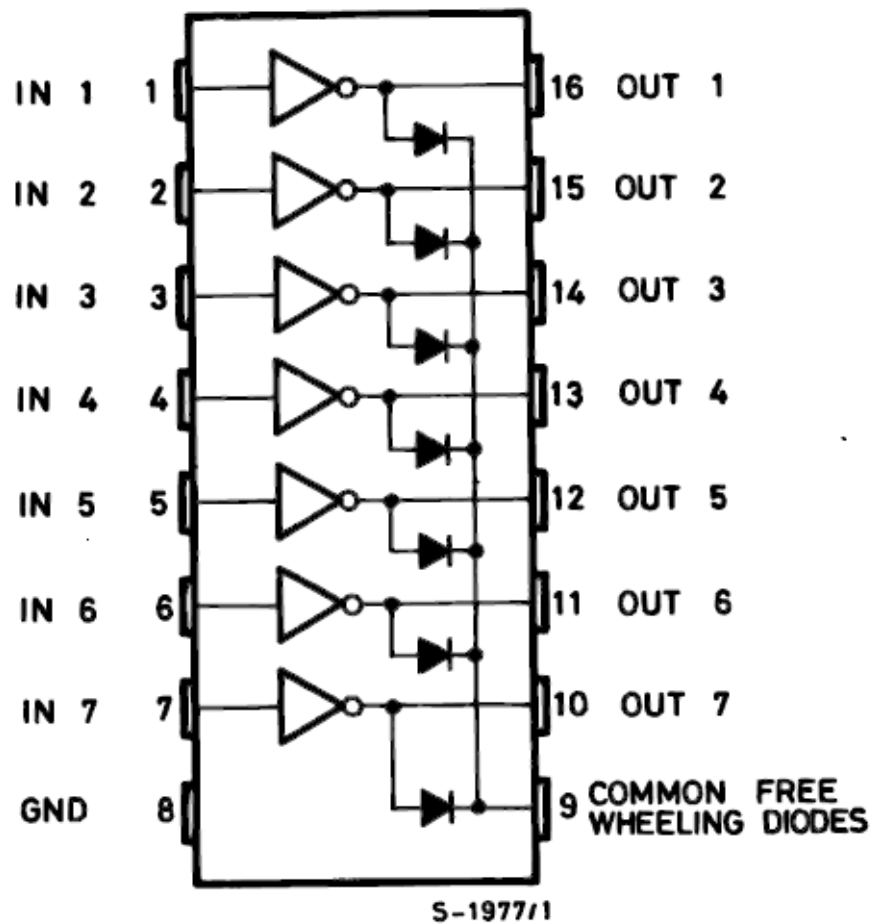


Figure 4.3 ULN2003A Inside Circuit

4.3 Stepper Motor and Arduino Nano Interfacing with ULN2003A

ULN2003A is used to operate stepper motor via Arduino Nano. The table that is showing a connection of ULN2003A with Nano and Stepper motor is on table 4.2(a) and (b) respectively. There are 16 pins in ULN2003A. As you can see in table 4.1(a) pin number 1,2,3 and 4 are connected with D8,D7,D6 and D5 of Arduino Nano. Pin number 8 and 9 of

Table 4.2 (a) ULN and Nano Connection

ULN2003A	ARDUINO NANO
IN1,1	D8
IN2,2	D7
IN3,3	D6
IN4,4	D5
GND,8	GND
COM,9	5V

ULN2003A are ground and Common(Vcc). ULN2003A can operate on more than 5 volt. As we

discussed in the Theory of ULN200A(4.1 ULN2003A). It is known for its high-current, high-voltage capacity as shown in table 4.1. But, we are providing only 5 V which is common to the Red wire of Stepper Motor as shown in Table 4.2 (b) ULN and Stepper Connection. We can also use external power supply but stepper motor 28BYJ-48 can easily

Table 4.2 (b) ULN and Stepper Connection

ULN2003A	STEPPER MOTOR
OUT1,16	BLUE
OUT2,15	PINK
OUT3,14	YELLOW
OUT4,12	ORANGE
COM,9	RED

operate in 5V. So, there is no need to provide external power supply. If we had used two stepper motor or Another type of stepper motor, then more voltage would have been required. We rotate stepper motor at 90degree to open and close the barrier. The stepper motor is discussed in next topic in detail and the programming of Arduino Uno and Nano is discussed in Chapter 7: Programming. In next chapter the topic of discussion is Power Supply.

4.4 LD33V Voltage Regulator

We have used 3.3 V voltage regulator to provide power supply to ESP8266. As we know ESP8266 is very sensitive and works on Low power ratings. Thus, it is necessary to provide 3.3v power supply. So, that can operate perfectly and faultlessly. Figure 4.4 is showing a LD33V Voltage

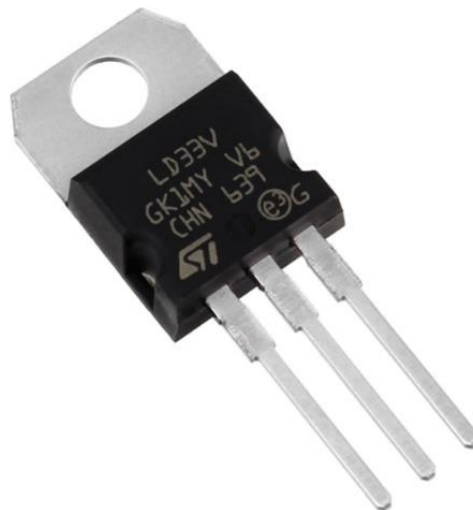


Figure 4.4 LD33V

regulator. This is the basic LD1117V33 voltage regulator, a low drop positive regulator with a 3.3V fixed output voltage. This fixed regulator provides a great amount of stability and protection

for your project and with its on ship trimming this regulator is able to reach an output voltage tolerance within $\pm 1\%$. Each one of these voltage regulators can output a max current of 800mA. As you can see in figure 3.4 there are 3 pins in LD33V, that are GND, INPUT and OUTPUT. Output is connected to input VCC of ESP8266. Input is connected to 5v pin of Arduino UNO. And ground is connected to GND of Arduino Uno. We could have also used Arduino's 3.3V pin, and we could have connected that pin to 3.3V input pin of ESP8266 but, current of Arduino's 3.3V pin is insufficient or not enough to provide it. That's why, we have chosen LD33V.

4.5 Ultrasonic Sensor HC-SR04

This sensor is the most significant component of PIS. This helps us to detect any object or vehicle via its sonar wave. We are using four ultrasonic sensors in our PIS Prototype. Each sensor is for each parking slot. Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules include ultrasonic transmitters, receiver and control circuit. HC-SR04 is shown in figure 3.5 HC-SR04. Its stable



Figure 4.5 HC-SR04

performance and high ranging accuracy make it a popular module in electronic market. Compared to the Sharp IR ranging module, HC-SR04 is more inexpensive than it. But it has the same ranging accuracy and longer ranging distance.

The Basic principle of working is given in following points

- It uses IO trigger for at least 10us high level signal.

- The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
- IF the signal back, through high level, time of high output IO duration is the time from sending

Table 4.3 Electric Parameters

Parameters	Value
Working Voltage	5V
Working Current	15mA
Working Frequency	40kHz
Maximum Range	4m
Minimum Range	2cm
Measuring Angle	15°
Trigger Input Signal	10μs
Echo Output signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm

ultrasonic to returning.

- $Test\ distance = \frac{High\ Level\ Time \times Velocity\ of\ Sound}{2}$
- Velocity of sound is 1,235 km/h at 20° C.

4.5.1 HC-SR04 Parameters

The Electric parameters of HC-SR04 are shown in table 4.3 Electric parameters. We provided 5V via Arduino Uno 5v pin. There are four HC-SR04. Ground and VCC of HC-SR04 are common with GND and 5V pins of Arduino Uno.

As you can see in above table, the working voltage of HC-SR04 is 5V which is fed by Arduino Uno 5V pin. The working frequency of HC-SR04 is 40 KHz and its measuring angle is 15 degrees. These parameters are well suited to use in our PIS Prototype. The dimension of HC-SR04 is also quite suitable to us.

The principle of ultrasonic distance measurement used the already-known air spreading velocity, measuring the time from launch to reflection when it encountered obstacle, and then calculate the

distance between the transmitter and the obstacle according to the time and the velocity. Thus, the principle of ultrasonic distance measurement is the same with radar. A common use of ultrasound is in underwater range finding this use is also called Sonar. An ultrasonic pulse is generated in a particular direction. If there is an object in the path of this pulse, part or all of the pulse will be reflected back to the transmitter as an echo and can be detected through the receiver path. By measuring the difference in time between the pulse being transmitted and the echo being received, it is possible to determine the distance. This is same as Ultrasonic sensor working in air, as we discussed in starting of this chapter. The measured travel time of Sonar pulses in water is strongly

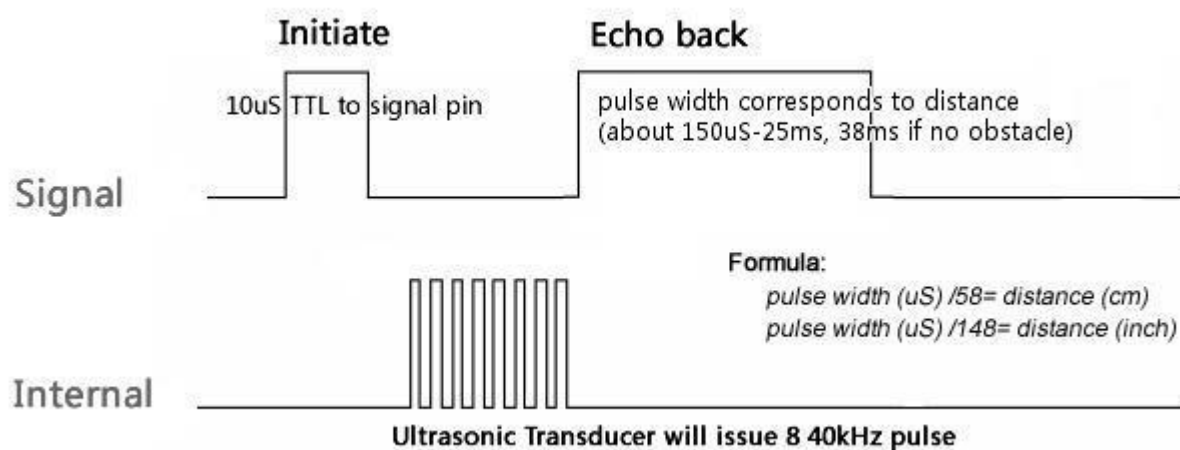


Figure 4.6 Timing Diagram

dependent on the temperature and the salinity of the water. Ranging in water varies from about hundreds to thousands of meters, but can be performed with centimeters to meters accuracy.

4.5.2 Timing Diagram.

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion. You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: $\text{uS} / 58 = \text{centimeters}$ or $\text{uS} / 148 = \text{inch}$; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.

A short ultrasonic pulse is transmitted at the time 0, reflected by an object. The sensor receives this signal and converts it to an electric signal. The next pulse can be transmitted when the echo is

faded away. This time period is called cycle period. The recommend cycle period should be no less than 50ms. If a $10\mu\text{s}$ width trigger pulse is sent to the signal pin, the Ultrasonic module will output eight 40kHz ultrasonic signal and detect the echo back. The measured distance is proportional to the echo pulse width and can be calculated by the formula above. If no obstacle is detected, the output pin will give a 38ms high level signal. Now, we have known, how to find the distance by using that formula. We used same formula in Arduino programming to find the exact value of distance of object in cm. after that we uploaded that sensor value to our website via ESP8266 Wi-Fi Module.

4.5.3 Arduino Uno and HC-SR04 Interfacing

As we know there are four pins in HC-SR04 that are VCC, GND, TRIG and ECHO. We are using four HC-SR04 in PIS Prototype. The pin connection table is given below.

Table 4.4 HC-SR04 and Arduino Uno Pin Connection

HC-SR04	ARDUINO UNO
Trigger pin common	13
Echo pin sensor 1	12
Echo pin sensor 2	11
Echo pin sensor 3	10
Echo pin sensor 4	9
VCC Common	5V
GND Common	GND

4.6 Chapter Summary

In this chapter we have discussed about components and Ultrasonic sensor, which are used in our project. The components in which we have discussed are IC ULN2003A, stepper motor and LD33V Voltage regulator. The sensor which we have used is ultrasonic sensor with model name HC-SR04. All these components and sensor are integral part of PIS. The most significant element in PIS is HC-SR04. Its range is quite good and suitable to our project. All of these components goods and well suited to PIS. The power consumptions of these components are also optimum.

CHAPTER 5

POWER SUPPLY

In any electrical or electronic project, the power supply is the most important topic of concern, because the project or system can only work by power. Without power supply each and every project is useless and unworkable. We know power supply is an electrical device that supplies electric power to an electrical load. The primary function of a power supply is to convert electric current from a source to the correct voltage, current, and frequency to power the load. This chapter is for discussion of power supply techniques and methods which is best suited to our project. In this we will also study the cables and wires which are very useful to feed and upload program in Arduino Uno and Nano.

5.1 Methods to power up the Arduino

The Arduino Uno and Nano has undergone many revisions, and hence the power supply circuit has evolved to an almost foolproof design. In this topic, we will learn about the four different ways in which we can power up the Arduino Uno. While making projects, it is necessary to know the following techniques, since there are instances when flexibility with regards to the power supply is required. All below methods can also applicable on Arduino Nano.

5.1.1 Using USB Cable

The USB port of the Arduino Uno can be connected to a desktop/laptop. If the connection is enumerated, i.e. the computer recognizes the device, the current supplied to the board is 500mA at 5V. If the connection is not enumerated, 100mA is supplied at 5V. where, USB Enumeration is the process by which the host computer identifies a device to load the appropriate driver and learns the capabilities of the device. We can also use Power Bank instead of Laptop USB port. Because laptop is not easy to carry compare with power bank. Figure 5.1 showing a Arduino Uno power up via Power bank. As we know A Power Bank is a mobile charger can be charged in advance for later phones, tablets or laptops to recharge. A Power Bank is a handy gadget that you can charge your electric devices without a power socket. While we use power bank to power up Arduino we should know the power ratings of that power bank. Most of the phone charger power bank are 5 volts but the current of those power banks could be large.



Figure 5.1 Uno Power up by power bank

5.1.2 Using DC Adapter

The barrel connector can be supplied an input of 7-12V. This is regulated to 5V by the onboard voltage regulator, and the board is powered on. It must be a DC adapter (i.e. it has to put out DC, not AC) and it should be between 7V and 12V DC. It must be rated for a minimum of 250mA



Figure 5.2 Uno Power up by DC Adapter

current output, although you will likely want something more like 500mA or 1A output, as it gives you the current necessary to power a servo or twenty LEDs if you want to must be rated for a minimum of 250mA current output, although you will likely want something more like 500mA or 1A output, as it gives you the current necessary to power a servo or twenty LEDs if you want to. It must have a 2.1mm power plug on the Arduino end, and the plug must be "centre positive", that is, the middle pin of the plug has to be the +

connection. The power up via DC Adapter is shown in figure 5.2 Below. The current rating of DC Adapter should be desirable regarding input current rating of Arduino Uno. As we discussed before, Arduino Uno has a voltage regulator for 9V to 12V power supply via barrel plug. Figure

5.3 is showing a barrel plug or jack of Arduino Uno for DC adapter power supply. If the barrel connector and an AC-DC adapter are being used to power up the Arduino, make sure that the

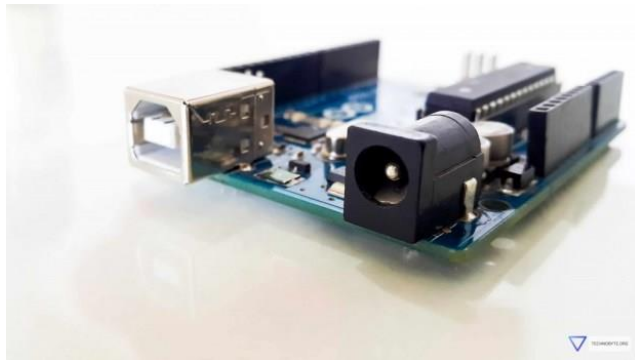


Figure 5.3 Arduino Uno barrel plug

output of the adapter is between 7-12V. Although the rated input can exceed to as much as 20V, it is safe to stay within the recommended range to protect the voltage regulator from excessive heating. Also, see to it that the GND and Vin pins are not shorted. The next and last method to power up Arduino is providing 5V and GND via battery. Use voltage regulator to convert 9V battery power

into 5V. we can also connect 9V battery to Vin pin of Arduino Uno directly without any use regulator. Depending on your application you can input 12V too but make sure the current values stay around 500mA.

5.1.3 Using 9V Battery

As we discussed before, it is possible to power up the Arduino using the 5V and GND pins, provided that the input given is steady and regulated 5V. The 5V pin bypasses the voltage regulator and all the safety measures present on the Arduino Uno, so if the input exceeds 5V (5.5 is the maximum upper limit), the board can be damaged. It is generally advised to avoid powering up the Arduino Uno using this method. The most common type of nine-volt battery is commonly referred to simply as 9-volt, although there are less common nine-volt batteries of different sizes. Codes for the usual size include PP3 (for size and voltage, any technology), 6LR61 (IEC code for alkaline batteries), and in Japan 006P. The PP3 size battery is 48.5 mm × 26.5 mm × 17.5 mm or 1.91 in × 1.04 in × 0.69 in. Both terminals are at one end and their centers are ½inch (12.7 mm) apart. Figure 5.4 is showing an Arduino power up by 9 volt battery. The battery has both terminals in a snap connector on one end. The smaller circular (male) terminal is positive, and the larger hexagonal or octagonal (female) terminal is the negative contact. The connectors on the battery are the same as on the connector itself; the smaller one connects to the larger one and vice versa. An LED can be used to check the incoming power from battery at regulator

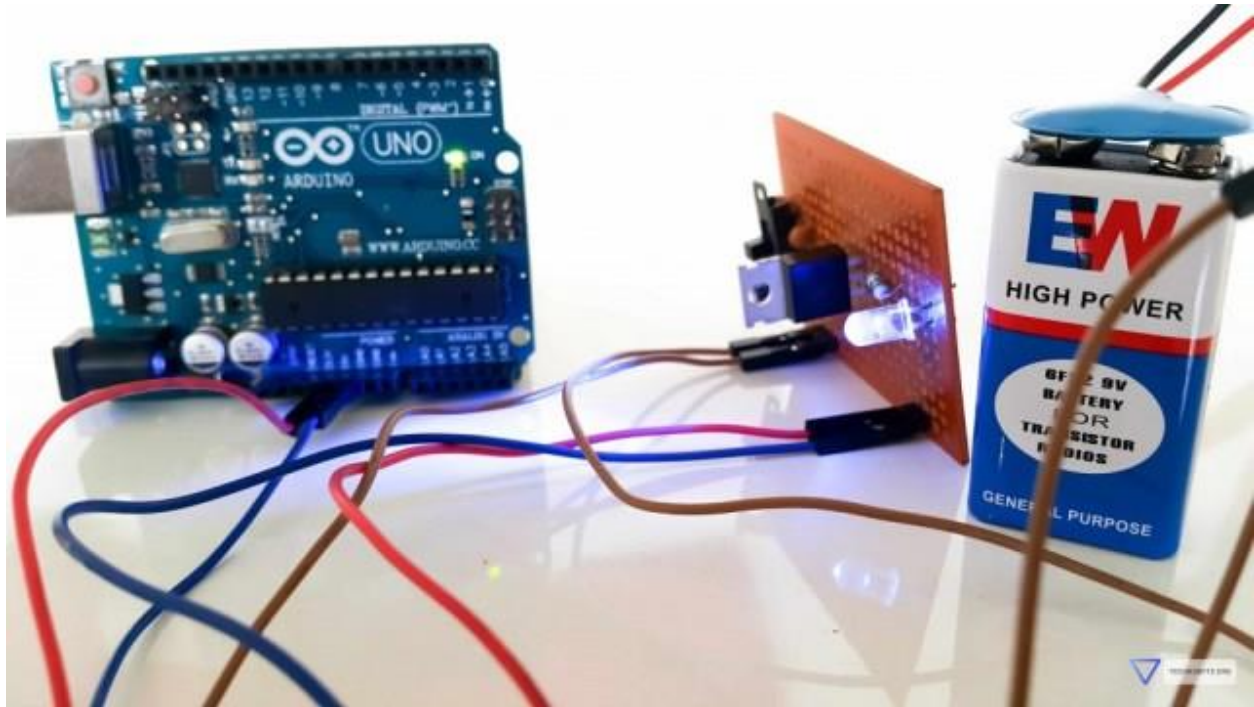


Figure 5.4 Arduino Power Up by 9V Battery

5.2 Arduino Cable

Arduino Uno and Nano are programmed via an USB cable. As we know, USB stands for Universal Serial Bus. In this section of this chapter we are going to discuss about USB cables which is used to program Arduino modules.

5.2.1 Arduino Uno Cable

Arduino Uno is programmed via USB type B cable. USB Type B connectors, officially referred to as Standard-B connectors, are square in shape with either a slight rounding or large square protrusion on the top, depending on the USB version. USB Type-B connectors are supported in every USB version, including USB 3.0, USB 2.0, and USB 1.1. A second type of "B" connector, called Powered-B, also exists but only in USB 3.0. A male USB Type B connector is called a plug while a female connector is called either a receptacle (as used in this article) or port. The Universal Serial Bus was developed to simplify and improve the interface between personal computers and peripheral devices, when compared with previously existing standard or ad-hoc proprietary interfaces As you can see in above figures. Arduino Uno cable is an USB type B to

USB type A. USB type A is connected to Computer and type B is connected to Arduino Uno. The parameter table of USB Cable is given in Table 5.1.

USB type A and B have 4 Pins in Bus, but USB Type C has 24 Pins hence its speed is superfast



Figure 5.5 (a) Arduino Uno Cable



Figure 5.5 (b) USB type B to type A Cable

compare with type A and type B. Arduino Uno has only type B port. If we talk about Arduino Nano then it has micro-USB type B and type A port.

Neither USB 1.0 nor 1.1 specified a design for any connector smaller than the standard type A or

Table 5.1 USB Cable Parameter

Parameter	Value
Signal	5 V DC
Maximum Voltage	5 V
Maximum Current	0.5 A(USB 2.0) and 0.9 A(USB 3.0)
Length	2-5 meter
Pins	4 (type A and B)

type B. Though many designs for a miniaturised type B connector appeared on many peripherals, conformance to the USB 1.x standard was fudged by treating peripherals that had miniature connectors as though they had a tethered connection (that is: no plug or socket at the peripheral end). There was no known miniature type A connector until USB 2.0 (rev 1.01) introduced one. A USB device may consist of several logical sub-devices that are referred to as device functions. A composite device may provide several functions.

5.2.2 Arduino Nano Cable

The **Micro- USB** connectors (Micro-A and Micro-B) are the smallest connectors provided by the USB standard. They have been designed for small devices such as mobile phones, smartphones and portable media players. Compared to Mini-USB connectors, they have the same width but are thinner and more robust, having been designed to withstand 10,000 insertion and removal cycles.



Table 5.6 (a) Mini USB A and B Plug

Figure 5.6 (b) Mini USB B to USB type A

The Micro-USB connector has been chosen by some of the major cell phone companies in the world. Alcatel, Atmel, BlackBerry, Emblaze, Huawei, LG, Motorola, NEC, Nokia, HTC, Qualcomm, Samsung, Sony Ericsson and Texas Instruments) to become the standard connector that from June 2011 (initially planned for 2010) is present on all the new models of mobile phones in the world. It was the European Union to ask for it in order to reduce the electronic pollution (the existence of dozens of different shippers obliges those who want to change cell phone to throw away also the relative charger, being in most cases incompatible with the new phone); without counting that each battery charger gives tension and current different outgoing. The choice to standardize the connector would also solve this problem because the micro-USB socket, being normally connected to a PC, provides 5 V and 500 mA (but there are also more powerful chargers, from 800 mA, 1000 mA and 1200 mA that they provide more power to charge faster, but any mobile phone with a Micro-USB connector can still recharge with the standard values supplied by the PC, ie 5 V and 500 mA). Arduino Nano is very small in size compare with Uno. So, Mini USB is suitable for Nano. Pins and other parameters of Mini USB are similar like USB. The latest and

new generation USB cable is USB type C. The speed of USB type C is very high. Hence, the transfer rate is very good. There are 24 pins in USB type C. The pinout of USB type A and type B is shown in below. The connectors the USB committee specifies support a number of USB's

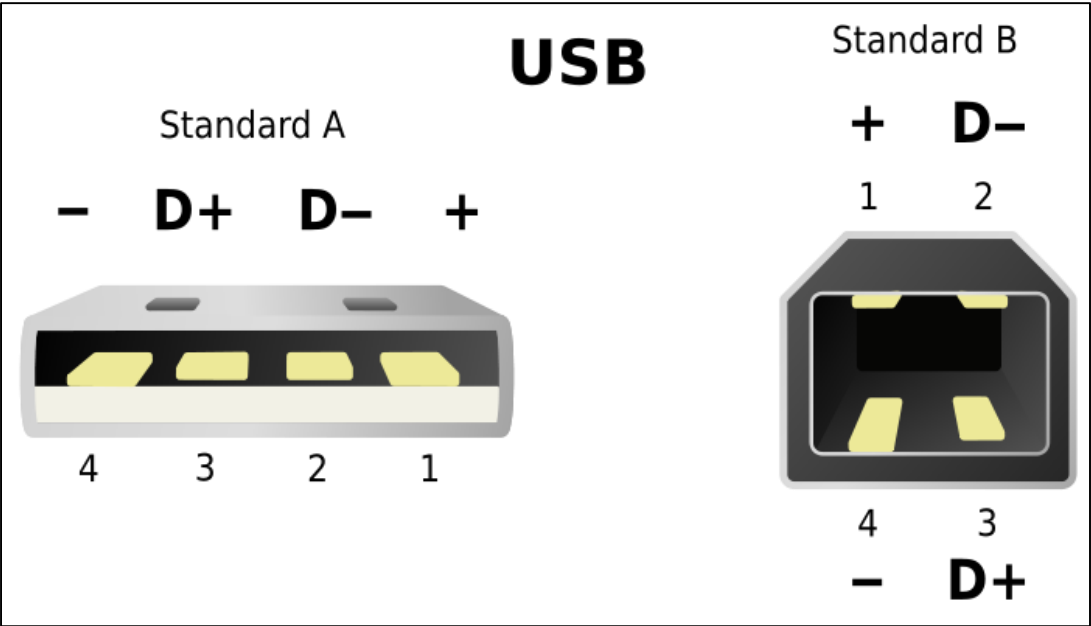


Figure 5.7 USB Plug Pins

underlying goals, and reflect lessons learned from the many connectors the computer industry has used. The female connector mounted on the host or device is called the receptacle, and the male connector attached to the cable is called the plug.

Table 5.2 USB Plug Pins Signal Table

USB extension cable By design, it is difficult to insert a USB plug into its receptacle incorrectly. The USB specification requires that the cable plug and receptacle be marked so the user can recognize the proper orientation. The type-C plug is reversible. USB cables and small USB devices are

Pin	Signal
Pin 1	V _{BUS} (+5 V)
Pin 2	Data -
Pin 3	Data +
Pin 4	Ground

held in place by the gripping force from the receptacle, with no screws, clips, or thumb-turns as other connectors use. The USB 1.1 standard specifies that a standard cable can have a maximum length of 5 metres (16 ft 5 in) with devices operating at full speed (12 Mbit/s), and a maximum length of 3 metres (9 ft 10 in) with devices operating at low speed (1.5 Mbit/s). USB 2.0 provides for a maximum cable length of 5 meters (16 ft 5 in) for devices running at high speed (480 Mbit/s). The USB 3.0 standard does not directly specify a maximum cable length.

5.3 Modules and Components Power Supply of PIS

PIS has some components and modules. So, it is essential to discuss upon the method to power up them. Basically all modules are powered by Arduino Uno and Nano. There is no need to provide external power supply to those modules via any adapter or cable. Stepper motor 28BYJ-48 can be powered by external supply but we did not provide external power to it because it is not essential. The table of power supply of different modules via Arduino is given below.

Table 5.3 Modules and Components Power supply table

Module	Voltage By
RFID MFRC522	3.3 V Arduino Nano
ESP8266	3.3 V Arduino Uno
HC-SR04	5 V Arduino Uno
Stepper Motor and ULN2003A	5 V Arduino Nano

5.4 Chapter Summary

In this chapter we have discussed about different power supplies that can be used to power up the Arduino Nano and Uno. We have also discussed about modules powered by Arduino Nano and Uno, 5 V and 3.3 V pins. As we discussed Arduino Uno and Nano are programmed via USB cable, where Uno has USB type B port and Nano has Mini USB B port in their board. After next chapter of discussion is Chapter 6 Software.

CHAPTER 6

SOFTWARE

In this chapter we are going to discuss about the software's which are used to develop proper PIS. Software is also very important part of any embedded system or IOT based project. In simple way, we can say if hardware is a body then software is a soul of that body. So, the software and languages which are used in PIS are discussed well in this chapter. The language in which Arduino hardware is programmed is Arduino software (IDE). The website of PIS is made by using HTML, CSS and PHP for frontend and backend work of our website. In this chapter we will discuss about some websites which provide free hosting and platform for IOT Projects.

6.1 Different Languages and tools used in PIS

In PIS, Software part we have used different languages and also markup language to make our website (parkininfo.000webhostapp.com).

6.1.1 Arduino IDE

The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in the programming language Java. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. A program written with the IDE for Arduino is called a sketch. Sketches are saved on the development computer as text files with the file extension .ino. Arduino Software (IDE) pre-1.0 saved sketches with the extension .pde.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into

a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

A minimal Arduino C/C++ program consist of only two functions:

- `setup()`: This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
- `loop()`: After `setup()` has been called, function `loop()` is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.

Most Arduino boards contain a light-emitting diode (LED) and a load resistor connected between pin 13 and ground, which is a convenient feature for many tests and program functions. A typical program for a beginning Arduino programmer blinks a LED repeatedly.

There are too many inbuilt functions are available in Arduino software. Some of the functions are `pinMode()`, `digitalWrite()`, and `delay()`, which are provided by the internal libraries included in the IDE environment. The program is usually loaded in the Arduino by the manufacturer. We can

The image is a screenshot of the Arduino IDE interface. At the top, the title bar reads "AnalogReadSerial | Arduino 1.8.4". Below the title bar is a menu bar with "File", "Edit", "Sketch", "Tools", and "Help". A toolbar with various icons is located below the menu bar. The main text area displays the code for the "AnalogReadSerial" example. The code includes a multi-line comment explaining the sketch's purpose: reading an analog input on pin 0 and printing the result to the Serial Monitor. It also provides a URL to the Arduino website for more information. The code defines a `setup()` function to initialize serial communication at 9600 baud and a `loop()` function that reads the analog value from pin 0, prints it, and delays for 1 millisecond. The status bar at the bottom right indicates "Arduino Nano, ATmega328P on COM4".

```
/*
  AnalogReadSerial

  Reads an analog input on pin 0, prints the result to the Serial Monitor.
  Graphical representation is available using Serial Plotter (Tools > Serial Plotter menu).
  Attach the center pin of a potentiometer to pin A0, and the outside pins to +5V and ground.

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/AnalogReadSerial
*/

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);        // delay in between reads for stability
}
```

Figure 6.1 Arduino IDE in which AnalogReadSerial Example is written

also make our own library to make short code and more effective code.

6.1.2 HTML and CSS

We used HTML and CSS to make our own website in which we can upload our PIS data to show publicly. Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. Where, Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications. Home page of parkinginfo.000webhostapp.com is shown in figure 6.2. We show parking data in Area section of our website. All the functions of our website are discussed in detail in this chapters 6.2 part. All the work of front end and presentation layer are done by the help of HTML and CSS.

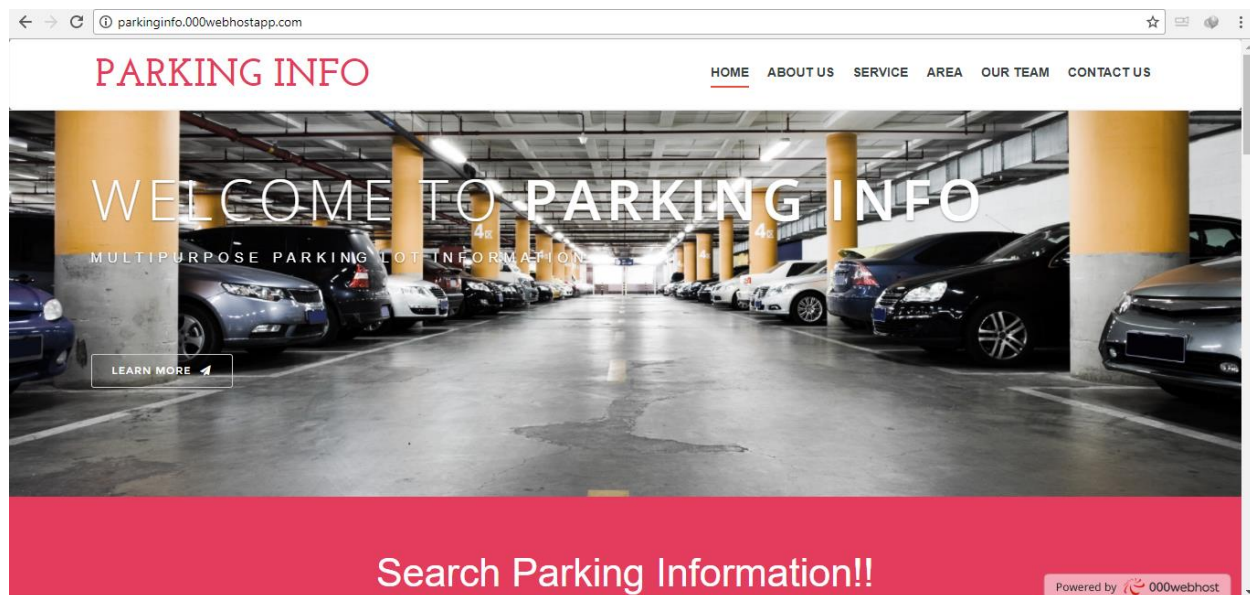


Figure 6.2 Home page of ParkingInfo.000webhostapp.com our website

6.1.3 PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. It is originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Group. PHP originally stood for Personal Home Page, but it now stands for the recursive acronym PHP: Hypertext Preprocessor.

PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems, and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications. We have used PHP to do our back ends work in PIS website. The main work of PHP is done in Area section of our website where, we are showing the parking data. The PHP code is shown in next chapter, that is programming. The figure 7.11 is showing the program of PHP. We are using the logic in which when the value of Ultrasonic sensor is less or equal to 5cm then the slot should be shown full otherwise the slot should be shown available. This logic could not have been implemented if we did not use PHP in our website. PHP has played a very important role in our website to show the slots in Area section of our website. We fetching the data of ultrasonic sensors in every 15 seconds from ThingSpeak. It is easy to understand, first ESP8266 Uploading the data on ThingSpeak in every 15 seconds and by the using of PHP we are fetching that data to our website to control the slot on the basis of ultrasonic sensor value. In HTML and CSS we cannot apply logics or conditions but in PHP we can apply it. As we know HTML and CSS are the markup language unlike as PHP language.

We embedded the PHP code in our HTML code to do the website work of parking information system in Area section where we are showing the availability and unavailability of parking slot. The color of the parking slots is also change accordance with the availability of parking slots. PHP stores integers in a platform-dependent range, either a 64-bit or 32-bit signed integer equivalent to the C-language long type. Unsigned integers are converted to signed values in certain situations; this behavior is different from other programming languages. Integer variables can be assigned using decimal, octal, hexadecimal and binary notations.

6.2 Online Free Platforms

There are too many online platforms which provides free hosting, data bases and many other online services. Now, we are going to discuss about those platforms who helped us to make perfect PIS by using that open source platforms.

6.2.1 ThingSpeak

ThingSpeak is an open source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP protocol over the Internet or via a Local Area Network. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates. We are using API of thingspeak to upload our data on our website. In this, first we have to create ID and after that its channel. Each channel has its own API Key. In order to send data to ThingSpeak using an Arduino®, you need an Arduino with network connectivity either onboard or with a shield. We have an [official library](#) for ThingSpeak and we require [Arduino 1.6.x](#) or above running on Windows®, MAC OS X®, and Linux®. This library needs to be installed and used by the Arduino device in order to send data to ThingSpeak using one of our examples. [ThingSpeak](#) requires a user account and a channel. A channel is where you send data and where ThingSpeak stores data. Each channel has up to 8 data fields, location fields, and a status field. You can send data every 15 seconds to ThingSpeak, but most applications work well every minute.

The step by step procedure to create a channel in ThingSpeak is given below.

- Sign up for new User Account – https://thingspeak.com/users/sign_up
- Create a new Channel by selecting Channels, My Channels, and then New Channel
- Click **Channels** > **MyChannels** as shown in figure below

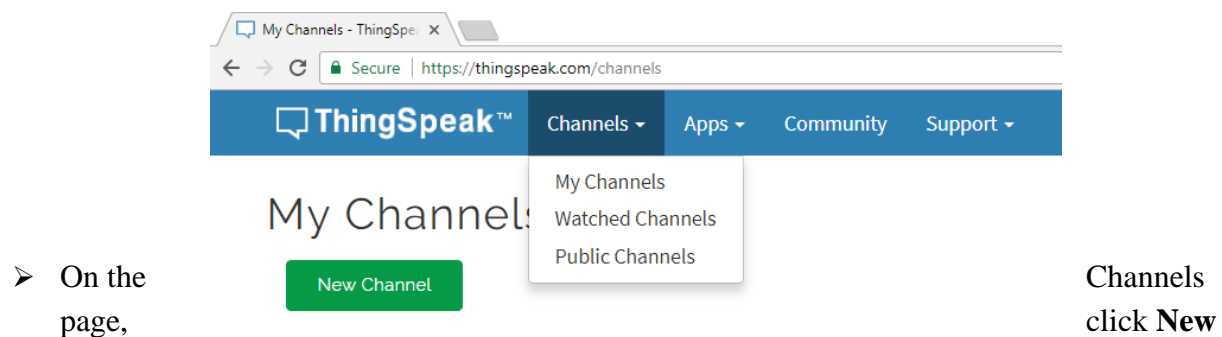


Figure 6.3 My Channel Screenshot

Channel as shown in figure 6.3

- Check the boxes next to Fields 1–3. Enter these channel setting values:

ThingSpeak™

New Channel

Name

Description

Field 1 ☒

Field 2 ☐

Field 3 ☐

Field 4 ☐

Field 5 ☐

Field 6 ☐

Field 7 ☐

Field 8 ☐

Metadata

Tags
(Tags are comma separated)

Link to External Site

Elevation

Show Location ☐

Latitude

Longitude

Show Video ☐
☒ YouTube
☐ Vimeo

Video URL

Show Status ☐

Save Channel

Help

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

- Channel Name:** Enter a unique name for the ThingSpeak channel.
- Description:** Enter a description of the ThingSpeak channel.
- Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- Tags:** Enter keywords that identify the channel. Separate tags with commas.
- Latitude:** Specify the position of the sensor or thing that collects data in decimal degrees. For example, the latitude of the city of London is 51.5072.
- Longitude:** Specify the position of the sensor or thing that collects data in decimal degrees. For example, the longitude of the city of London is -0.1275.
- Elevation:** Specify the position of the sensor or thing that collects data in meters. For example, the elevation of the city of London is 35.052.
- Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- Video URL:** If you have a YouTube™ or Vimeo® video that displays your channel information, specify the full path of the video URL.

Using the Channel

You can get data into a channel from a device, website, or another ThingsSpeak channel. You can then visualize data and transform it using [ThingSpeak Apps](#).

See [Tutorial: ThingSpeak and MATLAB](#) for an example of measuring dew point from a weather station that acquires data from an Arduino® device.

[Learn More](#)

Figure 6.4 New Channel Entry Screenshot

- Fill all the necessary detail of New Channel as shown in above figure 6.4.
- Click **Save Channel** at the bottom of the settings.

➤ The Channel has been created.

Some of the important tabs & points of thingspeak are given below.

- **Private View:** This tab displays information about your channel that only you can see.
- **Public View:** If you choose to make your channel publicly available, use this tab to display selected fields and channel visualizations.
- **Channel Settings:** This tab shows all the channel options you set at creation. You can edit, clear, or delete the channel from this tab.
- **Sharing:** This tab shows channel sharing options. You can set a channel as private, shared with everyone (public), or shared with specific users.
- **API Keys:** This tab displays your channel API keys. Use the keys to read from and write to your channel.
- **Data Import/Export:** This tab enables you to import and export channel data.

We have been showing 4 parking slots in our PIS Prototype but we have been using 4 channel of thingspeak instead of 4 fields of single channel. We have been using this because each channel of thingspeak takes delay of 15 seconds to upload the data. if we had chosen 4 fields of single channel, it would have taken 1 minute to upload a value of single sensor. That's why we have chosen 4 separate channel for each individual ultrasonic HC-SR04 Sensor. There are several notable reasons why we have chosen the ThingSpeak API over other available IoT APIs. This section will discuss the strengths and weaknesses of ThingSpeak in comparison to a handful of its direct competitors, including Carriots, SmartObject, Skynet, and Sensorthings. Each API offers a similar service to ThingSpeak, although with some nuances. For instance, Carriots seems to mainly focus on business and industry applications. Hence, the API is not open source and charges for parts of its services. On the other hand, other alternatives like Skynet, SensorThings and SmartObject are all open source, and can be changed and adapted to the will and needs of the user just like ThingSpeak. These open source APIs are quite similar in terms of functionality, each with their own strengths. The API describes and prescribes the expected behavior (a specification) while the library is an actual implementation of this set of rules. A single API can have multiple implementations (or none, being abstract) in the form of different libraries that share the same programming interface.

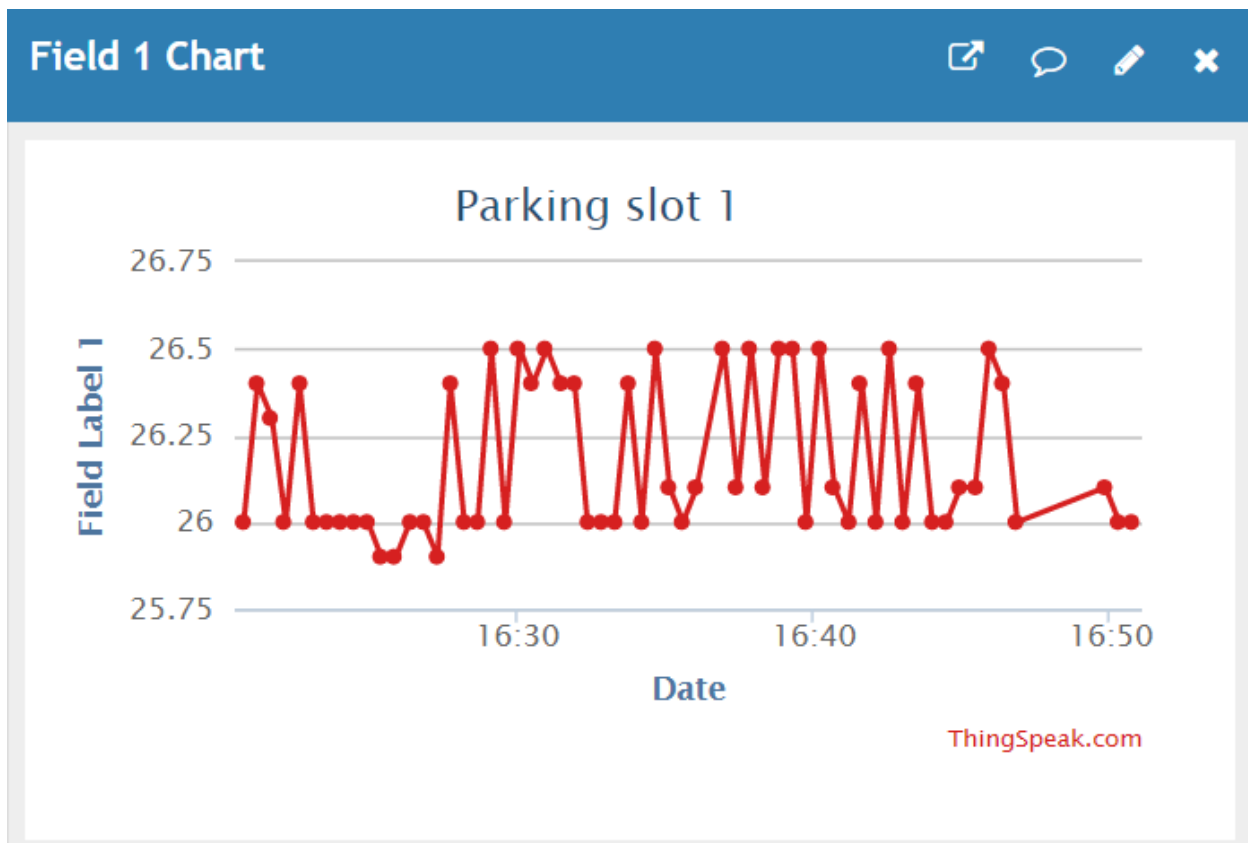


Figure 6.5 Chart of Parking Slot 1

As an example, the Skynet API is most tested and used with Arduino boards, therefore there is a lot of user generated programs and support for using this particular board. On the other hand, SmartObject is not as user-friendly as Skynet or ThingSpeak, and one really needs to be knowledgeable of programming in order to get started with this API. It is therefore not recommended for beginners or those who just wish to experiment on a very basic level. Finally, the SensorThings API on the surface appears to operate under similar principles as ThingSpeak, such as using HTTP requests to transfer data. Yet, the API has split its data model into two parts; sensing and tasking. Hence, there is a distinctive profile depending on if you are using the API to simply gather sensor data, or to communicate with an actuator. This would be most useful when troubleshooting issues, or if you wish to use the API for either sensing or tasking separately. Similarly however, all platforms have the potential to be compatible with almost any open or custom hardware, as long as it has connectivity to the Internet and can talk to the aforementioned APIs.

6.2.2 000webhostapp.com

000webhostapp is a website which provides free website hosting. Our is made with the help of 000webhostapp.com. The step by step procedure to make website via 000webhostapp is given below.

- Login to 000webhost members area <https://www.000webhost.com/cpanel-login>
- 000webhost members area was built having user experience in mind. Therefore, it's extremely easy to navigate to find tools needed to configure websites.
- List of your websites is presented you after login. At any time you can click on top “+” to create new website. Follow on screen instructions to create new website.
- Website is shown as a card with a website name. “Manage” button will provide tools to control your website. “Info” button will show basic website information
- Using navigation buttons at the top of the members area you can quickly jump to different website management sections:
 - **Build website** - Start building your website here. Access easy to use website builder. One-click WordPress install. Upload files with web file manager.
 - **Set web address** - Change website domain name at anytime. Choose new premium domain name or add domain name you already own.
 - **Upload files** - Access file manager. Upload files using web browser.
 - **Manage database** - Place to manage your databases. Create, access, change password for MySQL databases
 - **Manage emails** - Easy to use email forwarders. Email forwarders helps to redirect your domain emails to mailbox you already own.
 - **Settings** - Change website password, php version and other settings in this section. Create cron jobs, redirects and password protected directories here.
- Creating a new website has never been easier. Choose from 2 powerful tools and start a professionally looking website with just a few mouse clicks:
 - This powerful drag and drop website builder is perfect if you are not familiar with content management systems or coding. Just choose a preferred template and start filling your website with content.

- Using our custom built app installer it will take just a few minutes to install popular apps like WordPress, Joomla, Moodle, etc.
- Besides these options, you can code your own website and use already mentioned methods to upload files to the server.
- After signing up to 000webhost.com cloud hosting you will receive default website name such as my-name.**000webhostapp.com**. This is very convenient - website is instantly online.
- Website building can be started immediately. This name will be reserved only for you. During your website building process you might think of domain name or you might already have a domain name.
- Setting your domain name instead of my-name.000webhostapp.com is a common intension.
- When you are ready with your website next step is to offer beautiful looking email address and make your clients trust your service. Create email forwarder
- What is email forwarding? Email forwarding is a service provided by 000webhost for free hosting clients without any charge.
- It means an email sent to one address can be re-delivered to another without being stored on 000webhost mail servers.
- What is the difference between regular email and an email forwarding?
- Regular email, and forwarding email accounts, are two different account types. You can only have email forwarder in free hosting account.
- Moving or transferring website can be painful and overwhelming for many users. But not on 000webhost! You can use 2 different tools to upload your website files:
 - **Web File Manager** - Using web based file manager you can upload and manage your website files. It's fast and super easy to use. Remember you can upload zip files and extract them.
 - **FTP Access** - 000webhost supports File Transfer Protocol (FTP), therefore you can use an FTP client to upload your website files.
- Most important task for your website is to be found. Creating content is by far the most influential step. Some tips for the start.
 - **Create unique and meaningful content** - Search engines will quickly catch interesting content you are creating. Be unique - it works

- Start Blog** - Blog is very good way to start creating unique content
- Collect subscribers** - No matter if you are offering products or services start collecting email subscribers. You will soon be able to communicate with your customers directly.
- Landing page** - Your home page should introduce products or services you are providing in simple manner.

- 000webhost is the oldest free hosting provider on the net. Since 2006 inception big community of webmasters have gathered and are willing to help you in community support forums.

6.2.3 AppsGeyser

We have made our Android application via using AppsGeyser, this platform is open source and easy to use. An Android browser app is an app that is created for a specific topic It is set up with pre-set links that allow your user to navigate various resources without leaving your application. Browser apps are increasing in popularity because they meet all the user requirements. A browser app allows the user to navigate off to various website without leave the app. It also increases developers' revenues because user levels are high. Statistics have shown that users tend to stay within the app for longer periods of time and visit the app more often.

A browser app is easy to create in 8 simple steps:

- First visit appsgeyser.com and then click on create app as shown below

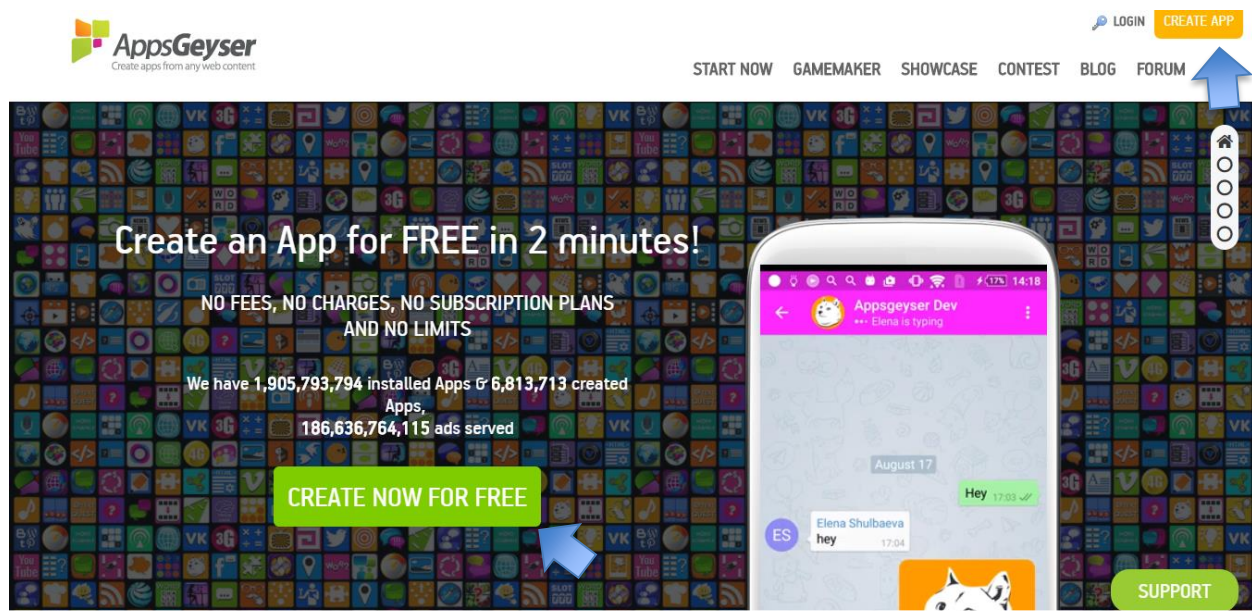


Figure 6.6 AppsGeyser Homepage

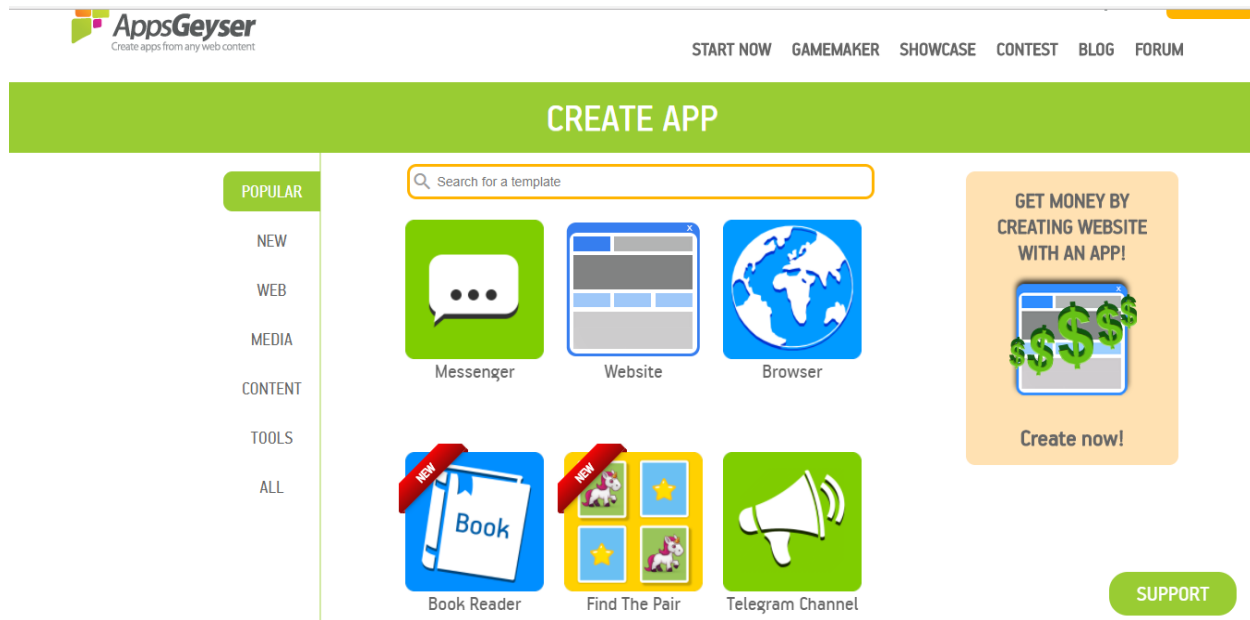


Figure 6.7 AppsGeyser Create App

- Scroll Down and Choose more option according to your topic or your website as shown in figure 6.7
- Click on Website to make android application of your website like we did. The figure 6.7 is showing the website tab.
- The next step is to write URL of your website as shown in figure 6.8. we wrote the URL of our website to make our android application of PIS.
- To write a URL of your website to create your own android application you must have any website, which has been made, related to your project




Figure 6.8 AppsGeyser Create Smart URL App

- Choose the colors and theme of your application which you want to make as shown in above figure 6.8. You can also check the preview of application by click on review button or refresh review.
- There are so many themes and templates are available at AppsGeyser you can choose according to your desires and needs.

CREATE SMART URL APP

Enter URL to create mobile version of your Website

 Refresh preview

APP SETTINGS


APP NAME

PIS Official

Tip: If you add word "Indian" before your app name, it will make your app more popular in India

NEXT

Having troubles with this form? Follow simple [guide](#).



Preview may not work for some templates. Your actual app can look differently on your Android device.

Figure 6.9 AppsGeyser App Name

- Write your android application name as shown in above figure 6.9.
- After that, click on next as shown in above image.
- Write description and choose icon.
- In last, click on Create as shown in figure on next page that is figure 6.10.

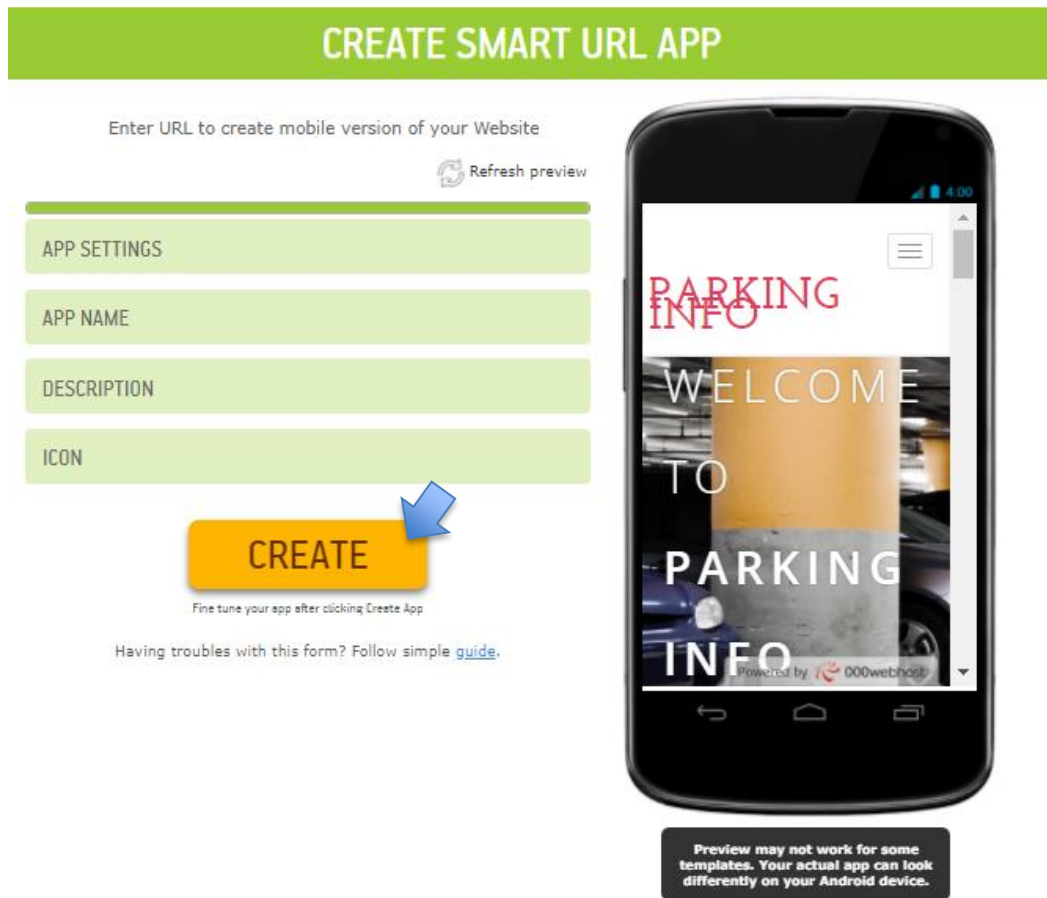


Figure 6.10 AppsGeyser Click on Create

- Now, your website has been created. We created our website by this same method. This is very simple and short procedure to create your own website in free.

We can also make android application of Messenger, media player, radio player, music app, browser, store, blog, mobile TV etc. there are so many templates and themes are available in AppsGeyser to use. Nowadays, Android application is very useful. Each and every person has android phone except some windows and apple users. Our website users can easily access our website by the use of this android application. If we had not made an android application, then it would have taken so much time to open PIS website. But now it is very easy to open it. This is the benefit of an android application. Our next topic is Our Website. The tabs that are available in our website is same as in our android application.

6.3 Our Website

Now, it is time to discuss about our website. In this sub topic we are going to show the different parts of our website with the help of some screenshots and theory. The home page of parkinginfo.000webhostapp.com is shown in figure 6.2. there are 6 tabs in our website Home, About us, Service, Area, Our Team and Contact us. Home page is simple, it's made via HTML and CSS. About us contains information and motives of maker of this website. The most important part of our website is Area Section. Below is the step by step procedure to find your parking location.

- Go to the Area section by clicking on Area as shown in below.

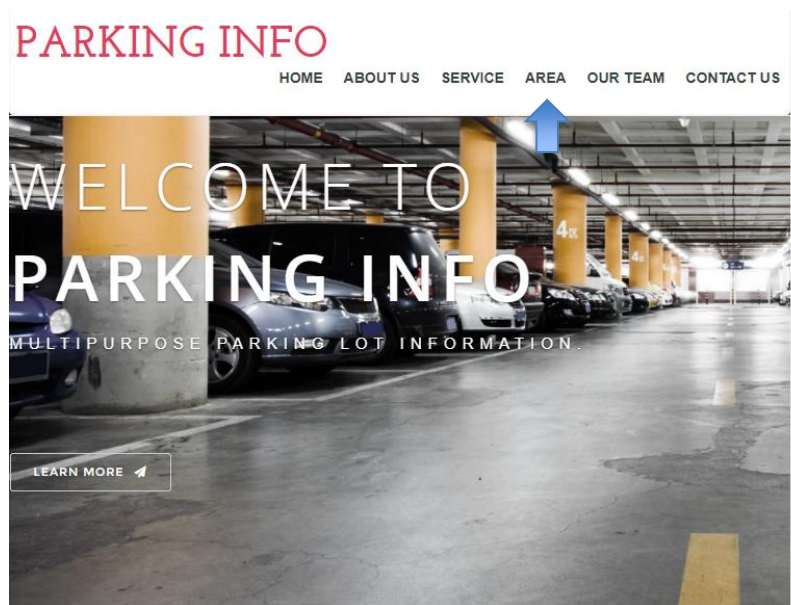


Figure 6.11 ParkingInfo Home Page.

- Then select the state and district in which you have to park the vehicle or see the vehicle.
- When the details are filled then click on Submit as shown in figure on next page.
- There would be also a option of search to find the location fast and easily.

Parking Information!!
Area

State
Uttar Pradesh ▾

District
Ghaziabad ▾

Submit

Figure 6.12 ParkingInfo Area Section

- We selected Ghaziabad as you can see in the figure. In Ghaziabad you have to choose the parking area which is shown in the figure below. We have only made the parking area prototype of HIET Ghaziabad. Hence we show it in the below image.

Parking Information!!
Ghaziabad Area

EAST GHAZIABAD	HIET	AKGEC	IMS	ABES	JSS
WEST GHAZIABAD	RKGIT	BBDIT	SGIT	ITS	IDEAL
NORTH GHAZIABAD	IPEM	INMANTEC	IP	KIET	JAIPURIA
SOUTH GHAZIABAD	MAIT	Sunderdeep	Vivekanand	MGM	JK

Figure 6.13 ParkingInfo Ghaziabad Area

- Click on HIET of row EAST GHAZIABAD then you will see the parking status of HIET parking. As shown in the figure on the next page.



Figure 6.14 ParkingInfo HIET Parking Area

- Above figure is showing the parking status of HIET. As we know, we have made the prototype of Parking Information System, so these slots are showing the status of Full and Available of parking slot of our model. If the slot is full then it will show full on slot with red color and if, it is available then it will show available with green color as you can see in the above figure.
- We are showing four slots because we have used four ultrasonic sensors HC-SR04. Each sensor is Showing each slot data. It we take only 15 seconds to update the slot status. The 15 seconds delay is come from ThingSpeak. If we don't use ThingSpeak then there will be no delay but we are using ThingSpeak because it is open source and very easy to use.
- So, this is our website. Which is made with the help of our guide and our team members.

6.4 Chapter Summary

In this chapter we have discussed about the software which are being used in parking information system. We have discussed about the different types of programming language which is used in our project. We have also discussed about some open source platform for IOT, web hosting and android application development. In last we have discussed on our website by showing some figures and explaining them in detail.

CHAPTER 7

PROGRAMMING

In this chapter we are going to show, the algorithms which have been used to program the Arduino Uno and Nano, and we will also show some basic Code which have been used in PIS.

7.1 Arduino IDE Algorithms

There are two main programs of PIS, one is of Display domain and another one is of Security domain. The Algorithms of each domain are given in next pages. As we discussed in previous chapters, display domain is playing an important role to upload the data of Ultrasonic Sensor HC-SR04 on our website by using Wi-Fi module ESP8266. So, the program of display domain is based on two components, ESP8266 and HC-SR04. We have used library for ESP8266 to communicate with ESP8266 via Arduino Uno. The library name is SoftwareSerial. It is used for Serial Communication. We used Serial Monitor of Arduino IDE to write AT Commands. We have been using four ultrasonic sensors, so their program is containing functions of four ultrasonic sensors and one ESP8266. In security domain, there are 2 main components MFRC522 and Stepper Motor. We have used libraries for MFRC522 and Stepper motor like we used for ESP8266. The libraries are MFRC522, SPI and Conscious. Here, MFRC522 and SPI are used for RFID MFRC522. For stepper motor we have used Conscious. Conscious is quite unique name for Stepper Motor library. The reason behind this is that, Conscious has been Created by us. The functions which are available in Conscious.h are added by the help of different Stepper motor library. We created this library because the library which was being used before conscious, had a pin numbers which were already occupied by MFRC522 pins. Those pins were saved in default. We had to change the pins by creating Conscious.h. that's why we created conscious.h. we named it conscious because it's quite unique and different. There are two main functions of conscious which are used in Security Domain program, Clockwise and Anti Clockwise. `moveDegreesCW()` and `moveDegreesCCW()` are those functions. We are rotating Stepper motor on 90 degrees. With the help PHP program, we have been receiving the data from ThingSpeak which has been displaying on our website in Area section. All these codes are the soul of PIS which have been controlling the body (Hardware). The analysis and study of algorithms is a discipline of computer science, and is often practiced abstractly without the use of a specific programming language or implementation. That's why we have written the algorithm of both domains in our thesis.

7.1.1 Display Domain Algorithm

The algorithm of display domain code in step by step procedure is given below:

Step 1 :

//ultrasonic sensor measures the distance on parking

distance1 = slot1_distance;

distance2 = slot2_distance;

distance3 = slot3_distance;

distance4 = slot4_distance;

Step 2:

// ultrasonic sensor data is uploaded to server via wifi module esp8266

Get.string1 = distance1;

Get.string2 = distance2;

Get.string3 = distance3;

Get.string4 = distance4;

Step 4 :

/* calibration of ultrasonic sensor data on the basis of distance on server and graphically represent on website. */

If (ultrasonic sensor data < 5)

{

Then display, parking slot is “occupied” on website.

Occupied slot is represented by red color.

}

else

{

Then display, parking slot is” empty” on website.

Empty slot is represented by green color.

}

Step 5 : Stop

7.1.2 Security Domain Algorithm

The algorithm of security domain code in step by step procedure is given below:

Step 1 :

Start

Step 2 :

Initially barrier is closed.

Step 3 :

RFID or Unique Id is provided to new car.

Step 4 :

Save Unique Id to server database.

Step 5 :

Unique Id is match at exit gate.

Step 6 :

If Unique is match, then barrier will open. Otherwise close.

Step 7 :

Stop

7.2 Library Used

In this sub topic we are going to show the Libraries and their codes, which are used in PIS Arduino Coding. There are total 4 libraries, which is used in Arduino IDE Codes. In display domain system we used SoftwareSerial library to establish communication between Arduino Uno and ESP8266 by using serial monitor of Arduino IDE. In security domain, there are 3 libraries, name as SPI, MFRC522 and Conscious. Uses and some functions which are used in programs are given on next page.

➤ SoftwareSerial

This library is used in Display Domain code. This is used to establish Serial communication between Arduino Uno and ESP8266 by using Serial Monitor of Arduino IDE. The function which is used in program is `ESP8266.begin()`.

➤ SPI

Serial peripheral interface(SPI) is a synchronous serial data protocol used by microcontrollers for communicating with one or more peripheral devices quickly over short distances. It can also be used communication between two microcontrollers. With an SPI connection, there is always one master device (Usually a microcontroller) which controls the peripheral devices. Typically, there are three lines common to all the devices.

➤ MFRC522

It is used to establish the communication between Arduino Uno and MFRC522. There are so many functions available in the library. Which helped us to minimize the code. The functions which are used are 1) `mfr522.pcd_init()` 2) `mfr522.PICC_IsNewCardPresent()`, 3) `mfr522.PICC_ReadCardSerial()` 4) `mfr522.PICC_HaltA()` 5) `mfr522.PCD_StopCrypto1()`;

➤ StepperMotor

This library is used to control stepper motor. As we know, we could make program without using these libraries but we used these libraries because by using this the program size is reduced. The functions which are used in programs are

- 1) `stepper.setRpm()`
- 2) `stepper.moveDegreesCW(Clockwise);`
- 3) `stepper.moveDegreesCCW(CounterClockwise);`

7.3 Chapter Summary

In this chapter, we have shown the codes or programs of Arduino IDE and PHP, and we also shown the programs of libraries like, SoftwareSerial, SPI, MFRC522 and Conscious. These libraries have extension .h but their code is written in C++. As we discussed before, we have also used HTML and CSS but their code is not shown in this Chapter. The reason behind this is that they are the markup languages. In markup language, there is no need of logic unlike any other programming language. Hence, it is not necessary to show the Code of HTML and CSS. Except that, all the important programs are shown in this chapter.

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

We draw conclusion and outline the future work. Future work proposes vehicle space availability feature, which can help the driver to know location of the available parking spot after entering the garage. We also propose installation of the system on each floor of Richard Beard Garage to determine the number of parking spots available on each floor.

8.1 Conclusion

The vehicle count feature monitors overall count of the parking lot in real time. This system can help the drivers to know the available number of parking spaces before entering the garage. The test results show that the system is accurate in the field. Moreover, the proposed parking guidance system is easy to install because of sensor's small size and low cost. Initially, the challenging task was to select a reliable sensor for this application, which was overcome by a thorough review of sensor technologies. Compared to other technologies, this system does not require installation of sensors into pavements and maintenance. Also, the system works in any weather condition. For this reason, it is suitable for all kinds of parking areas. In the future, the Arduino processor can be replaced by a PCB design using ATmega328 processor that can further reduce the cost of the system.

8.2 Future Scope

The vehicle space availability indicator can be an add-on feature to the system. It is simple to indicate if there is a vehicle or not in a particular parking spot using the logic developed to detect a vehicle. This feature needs some additional logic to glow red or green lights. The processor glows red to indicate an occupied spot and green to indicate an available spot. Figure 5.1: Flowchart of the logic to detect parking space availability Figure 5.1 shows the flowchart of logic involved in indicating spots available. A single processor can monitor three parking spots using three ultrasonic sensors, three green lights, and three red lights. Initially, the processor calculates obstacle distance by triggering pulses to all three sensors. Then, it checks for an obstacle by triggering more pulses. When a vehicle occupies the parking space, the obstacle height is calculated by subtracting new obstacle distance from the maximum range. The 50obstacle height

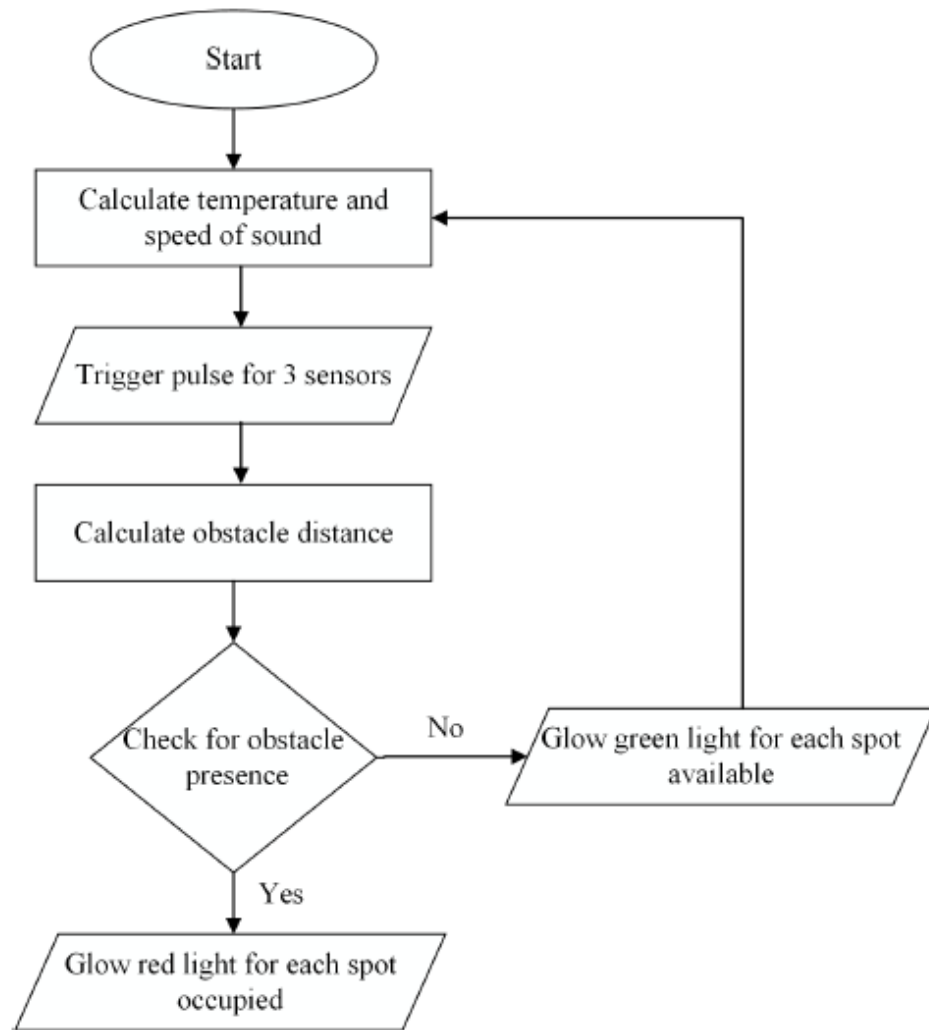


Figure 8.1 Flowchart of the logic to detect parking space availability

is compared with the minimum threshold height of a vehicle. If the obstacle height is less than minimum obstacle height, then it is an empty spot/available spot, and a green light glows. If greater, then it is an occupied spot and a red light glows. The loop repeats for every two milliseconds to check for an obstacle. Number of available spots on each floor can be known by installing the system at each entrance/exit of each floor. If the system is installed at each floor, it can give us number of cars that have exited and entered each floor. This way, an overall count of each floor can be maintained.

8.2.1 OTA(Over The Air Technology) Updates in esp32

This is a follow-up to our previous blog post, “Over-the-air updating an ESP32”. We’ve implemented a new version of our OTA example with various improvements:

- In the initial version, you had to “push” a firmware update to a specific device. Now, the device periodically connects to a web server to check if an update is available and if yes, downloads and installs it. This makes it easier to update a large number of devices automatically.
- The initial version used plain HTTP to transfer the firmware image. We now use HTTPS with certificate pinning to make sure the device only downloads the firmware from a trusted server.

The graphic shows a typical scenario:

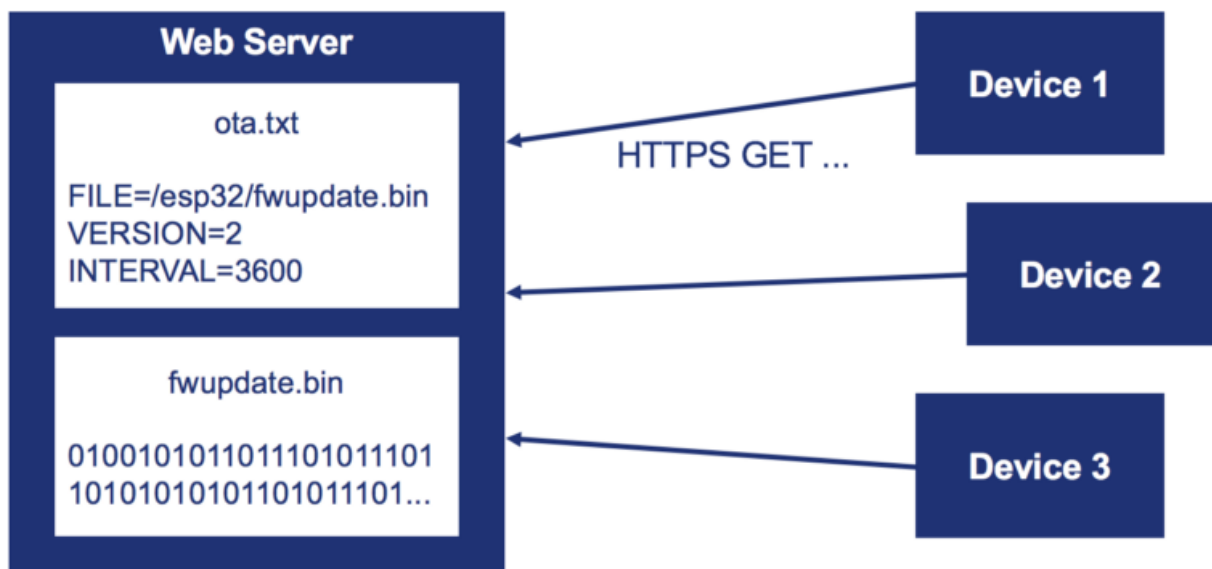


Figure 8.2 Block Diagram Showing Web Server and Device Connectivity

The web server provides two files. The first one is a text file with meta information on the available update (where to find it and which version it has) and the polling interval that the devices should use to check for new updates. We use a simple integer to identify the version (1, 2, 3 and so on). The second file is the actual firmware binary. If a device detects that the firmware version on the

server is different from the version it's running, it downloads and installs the firmware.

We're working on another follow-up post to show how this mechanism can be integrated with the secure bootloader infrastructure of the ESP32. In this post, we ignore that aspect. We also ignore the fact that you may want to authenticate the client to access the firmware.

The code in the new example is split into multiple modules to make it easier to understand and to integrate into a real application. For example, there may be other parts of the application that want to perform HTTP requests. Now they can, because the code for HTTP requests is in its own separate module. Or you may want to update the device via Bluetooth or via an USART. In this case, you can simply re-use the IAP module.

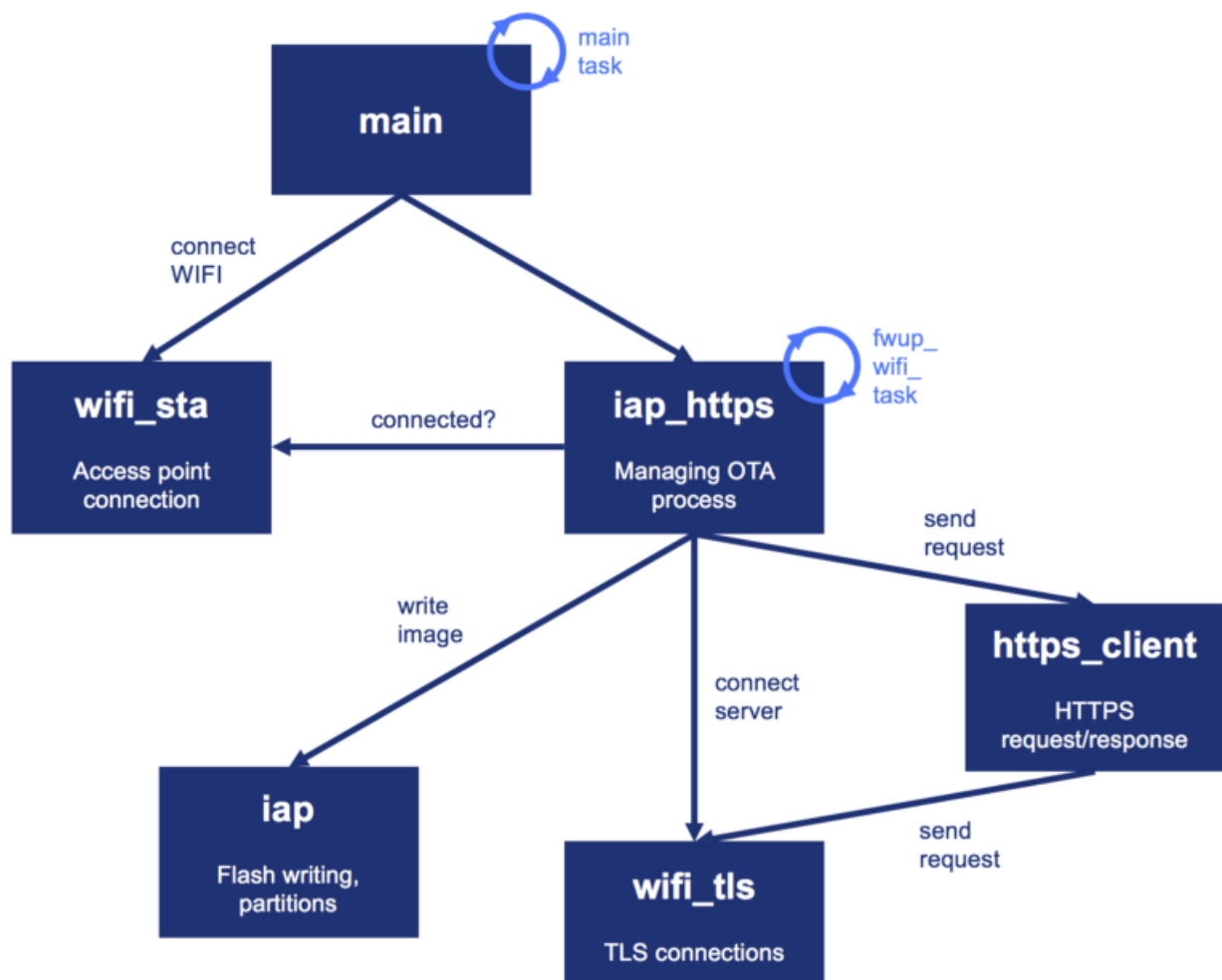


Figure 8.3 Block Diagram of Connectivity

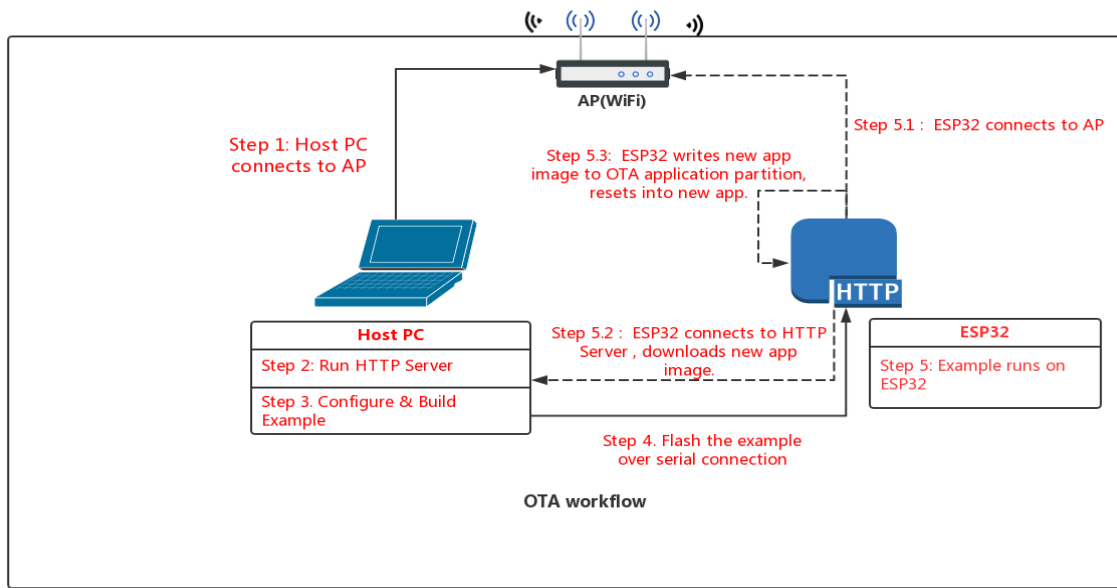


Figure 8.4 OTA Workflow

8.3 Chapter Summary

In this chapter we have discussed about the Future scope and conclusion of parking information system and we also discussed about the new ESP32 Update that is OTA(Over the update technology). OTA is a new technology it will be very helpful in future project of IOT(Internet Of Things).

REFERENCES

- [1] Axis Communications. AXIS Q1614 network camera data sheet. http://www.axis.com/files/manuals/um_q1614_58675r1_en_1407.pdf.
- [2] Elec Freaks. Ultrasonic ranging module HC - SR04. <http://www.electroschematics.com/wp-content/uploads/2013/07/HC-SR04-datasheet-version-2.pdf>.
- [3] E. Polycarpou, L. Lambrinos, and E. Protopapadakis. Smart parking solutions for urban areas. In World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a, pages 1–6, June 2013.
- [4] Y. Geng and C.G. Cassandras. A new “smart parking” system infrastructure and implementation. *Procedia-Social and Behavioral Sciences*, 54:1278–1287, 2012.
- [5] R. Vishnubhotla, P.S. Rao, A. Ladha, S. Kadiyala, A. Narmada, B. Ronanki, and S. Illapakurthi. Zigbee based multi-level parking vacancy monitoring system. In Electro/Information Technology (EIT), 2010 IEEE International Conference on, pages 1–4, May 2010.
- [6] G. Revathi and V.R.S. Dhulipala. Smart parking systems and sensors: A survey. In Computing, Communication and Applications (ICCCA), 2012 International Conference on, pages 1–5, Feb 2012.
- [7] S. V. Reve and S. Choudhri. Management of car parking system using wireless sensor network. *International Journal of Emerging Technology and Advanced Engineering*, 2:232–268, July 2012.
- [8] ThingSpeak Official website, <https://thingspeak.com/>
- [9] Ultrasonic Sensor DataSheet <http://www.micropik.com/PDF/HCSR04.pdf>
- [10] Stepper Blog <http://www.tigoe.net/pcomp/code/circuits/motors/stepper-motors/>
- [11] ULN2003 Data Sheet <http://www.geeetech.com/Documents/ULN2003%20datasheet.pdf>
- [12] Espressif Systems, ESP8266EX DataSheet <http://bbs.espressif.com/>
- [13] M. Patil and N. V. Bhonge. Parking guidance and information system using RFID and Zigbee. *International Journal of Engineering Research and Technology (IJERT)*, 2:2490–2493, April 2013.
- [14] A. Kianpisheh, N. Mustaffa, P. Limtrairut, and P. Keikhosrokiani. Smart parking system (SPS) architecture using ultrasonic detector. *International Journal of Software Engineering and Its*

Applications, 6(3):55–58, 2012.

[15] W. Atmadja, J. Yosafat, R.A. Setiawan, and I.I. Irendy. Parking guidance system based on real time operating system. In Industrial Automation, Information and Communications Technology (IAICT), 2014 International Conference on, pages 5–8, Aug 2014.

[16] S. Shaheen, C.J. Rodier, and C. Kemmerer. Smart parking management field test: A bay area rapid transit (bart) district parking demonstration. Institute of Transportation Studies, 2005.

[17] M.Y. Aalsalem, W.Z. Khan, and K.M. Dhabbah. An automated vehicle parking monitoring and management system using ANPR cameras. In Advanced Communication Technology (ICACT), 2015 17th International Conference on, pages 706–710, July 2015.

APPENDIX

- **ESP8266 AT Command Control Code**

```
#include <SoftwareSerial.h>

SoftwareSerial esp8266(2,3); // connect pin 2 of arduino to Tx of ESP8266 and pin
3 of arduino to Rx of ESP8266.

// This means that you need to connect the TX line from the esp to the Arduino's
pin2 // and the RX line from the esp to the Arduino's pin 3

void setup()
{
  Serial.begin(9600);
  esp8266.begin(9600); // your esp's baud rate might be different
}

void loop()
{
  if(esp8266.available()) // check if the esp is sending a message
  {
    while(esp8266.available())
    {
      // The esp has data so display its output to the serial window
      char c = esp8266.read(); // read the next character.
      Serial.write(c);
    }
  }

  if(Serial.available())
  {
    // the following delay is required because otherwise the arduino will read the first
    letter of the command but not the rest
```

```

// In other words without the delay if you use AT+RST, for example, the Arduino
will read the letter A send it, then read the rest and send it
// but we want to send everything at the same time.
delay(1000);
String command="";
while(Serial.available()) // read the command character by character
{
// read one character
command+=(char)Serial.read();
}
esp8266.println(command); // send the read character to the esp8266
}
}

```

- **Four Ultrasonic Sensor Code**

```

#define trigpin1 5
#define trigpin2 6
#define trigpin3 7
#define trigpin4 8
#define echopin1 12
#define echopin2 11
#define echopin3 10
#define echopin4 9
void setup() {
pinMode(echopin1,INPUT);
pinMode(echopin2,INPUT);
pinMode(echopin3,INPUT);

```

```

pinMode(echopin4,INPUT);
pinMode(trigpin1,OUTPUT);
pinMode(trigpin2,OUTPUT);
pinMode(trigpin3,OUTPUT);
pinMode(trigpin4,OUTPUT);
Serial.begin(115200);

}

void loop() {
long duration1;
long duration2;
long duration3;
long duration4;
float distance1;
float distance2;
float distance3;
float distance4;
digitalWrite(trigpin1,LOW);
delayMicroseconds(100);
digitalWrite(trigpin1,HIGH);
delayMicroseconds(100);
digitalWrite(trigpin1,LOW);
duration1=pulseIn(echopin1,HIGH);
digitalWrite(trigpin2,LOW);
delayMicroseconds(100);
digitalWrite(trigpin2,HIGH);
delayMicroseconds(100);

```



```

digitalWrite(trigpin2,LOW);
duration2=pulseIn(echopin2,HIGH);
digitalWrite(trigpin3,LOW);
delayMicroseconds(100);
digitalWrite(trigpin3,HIGH);
delayMicroseconds(100);
digitalWrite(trigpin3,LOW);
duration3=pulseIn(echopin3,HIGH);
digitalWrite(trigpin4,LOW);
delayMicroseconds(100);
digitalWrite(trigpin4,HIGH);
delayMicroseconds(100);
digitalWrite(trigpin4,LOW);
duration4=pulseIn(echopin4,HIGH);
distance1=(duration1*0.034)/2;
distance2=(duration2*0.034)/2;
distance3=(duration3*0.034)/2;
distance4=(duration4*0.034)/2;
//digitalWrite(trigpin,HIGH);
//delayMicroseconds(100);
//digitalWrite(trigpin,LOW);

//digitalWrite(trigpin,HIGH);
//delayMicroseconds(100);
//digitalWrite(trigpin,LOW);

```

```
//digitalWrite(trigpin,HIGH);  
//delayMicroseconds(100);  
//digitalWrite(trigpin,LOW);
```

```
Serial.print(distance1);  
Serial.println(" cm1");  
delay(1000);  
Serial.print(distance2);  
Serial.println(" cm2");  
delay(1000);  
Serial.print(distance3);  
Serial.println(" cm3");  
delay(1000);  
Serial.print(distance4);  
Serial.println(" cm4");  
delay(1000);  
}
```