|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
|  | | | | |
| KIIT(sponsor).png | | | | |
|  | | | | |
|  | | | | |
| A minor project report on | | | | |
|  | | | | |
| **Smart Parking System** | | | | |
|  | | | | |
|  | | | | |
| submitted in partial fulfillment of the requirements for the degree of | | | | |
|  | | | | |
| B. Tech | | | | |
| In | | | | |
| Electronics and Electrical Engineering | | | | |
|  | | | | |
| By | | | | |
|  | | | | |
| **SANGLAP KUNDU**  **SAPNA GOYAL**  **SOMAPRITA CHAKRABORTY**  **TANMOY BISWAS**  **TUSHAR SINGH** | | **1707046**  **1707050**  **1707070**  **1707079**  **1707080** | |
|  | | | |
| under the guidance of | | | |
|  | | | |
| **Prof. SATYADEEP DAS** | | | |
|  | | | |
| |  | | --- | | School of Electronics Engineering | | **KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY**  (Deemed to be University)  BHUBANESWAR | |  | | JUNE 2020 | | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | | |
|  | | | | |
| **CERTIFICATE**  This is to certify that the project report entitled **“SMART PARKING SYSTEM”** submitted by   |  |  | | --- | --- | | **SANGLAP KUNDU**  **SAPNA GOYAL**  **SOMAPRITA CHAKRABORTY**  **TANMOY BISWAS**  **TUSHAR SINGH** | **1707046**  **1707050**  **1707070**  **1707079**  **1707080** |   in partial fulfilment of the requirements for the award of the **Degree of Bachelor of Technology** in **Electronics and Electrical Engineering**  is a bona fide record of the work carried out under my guidance and supervision at School of Electronics Engineering, KIIT (Deemed to be University). | | | | |
|  | Signature of Supervisor :  Prof. Satyadeep Das  School of Electronics Engineering  KIIT (Deemed to be University) | |
|  | | | | |
|  | | | | |

ACKNOWLEDGEMENTS

We feel immense pleasure and feel privileged in expressing our deepest and most sincere gratitude to our supervisor **Professor Satyadeep Das**, for his excellent guidance throughout our project work. His kindness, dedication, hard work and attention to detail have been a great inspiration to us. Our heartfelt thanks to you sir for the unlimited support and patience shown to us. We would particularly like to thank him for all his help in patiently and carefully correcting all our manuscripts.

We are also very thankful to **Professor Suman Roy** B.tech project coordinator (E&E), Associate Dean Professor **Dr. Amlan Datta** and **Professor Dr. Arun Kumar Ray**, Dean (School Of Electronics) for their support and suggestions during our course of the project work in the final year of our undergraduate course.

**SANGLAP KUNDU(1707046)**

**SAPNA GOYAL (1707050)**

**SOMAPRITA CHAKRABORTY(1707070)**

**TANMOY BISWAS(1707079)**

**TUSHAR SINGH(1707080)**

**ABSTRACT**

In recent times the concept of smart cities have gained great popularity. Thanks to the evolution of Internet of things the idea of smart city now seems to be achievable. Consistent efforts are being made in the field of IoT in order to maximize the productivity and reliability of urban infrastructure. Problems such as, traffic congestion, limited car parking facilities and road safety are being addressed by IoT. In this paper, we present an IoT based cloud integrated smart parking system. The proposed Smart Parking system consists of an on-site deployment of an IoT module that is used to monitor and signalize the state of availability of each single parking space. A mobile application is also provided that allows an end user to check the availability of parking space and book a parking slot accordingly. The paper also describes a high-level view of the system architecture.

TABLE OF CONTENTS

Name :Page

CHAPTER 1 – INTRODUCTION 5

* 1. BACKGROUND 5
  2. NEED FOR IOT-CLOUD INTEGRATION 7

CHAPTER 2 - BACKGROUND THEORY OF MICROCONTROLLER 9

2.1 HISTORY OF MICROCONTROLLER 9

2.2 OVERVIEW OF NODEMCU 11

2.3 OVERVIEW OF BLYNK 15

2.4 OVERVIEW OF ULTRASONIC SENSOR 16

CHAPTER 3 – DESIGN OF SMART PARKING SYSTEM 18

3.1 CIRCUIT DESIGN 18

3.2 CODING FOR THE ULTRASONIC SENSOR 19

3.3 CODE EXPLANATION 22

3.4 PROJECT EXPLANATION 22

CONCLUSION 23

PLANNING AND PROJECT MANAGEMENT 24

REFERENCE 25

**CHAPTER 1**

**INTRODUCTION**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* 1. **BACKGROUND**

The concept of Internet of Things (IoT) started with things with identity communication devices. The devices could be tracked, controlled or monitored using remote computers connected through Internet. IoT extends the use of Internet providing the communication, and thus inter-network of the devices and physical objects, or ‘Things’. The two prominent words in IoT are “internet” and “things”. Internet means a vast global network of connected servers, computers, tablets and mobiles using the internationally used protocols and connecting systems. Internet enables sending, receiving, or communicating of information. Thing in English has number of uses and meanings. Dictionary meaning of ‘Thing’ is a term used to reference to a physical object, an action or idea, situation or activity, in case when we do not wish to be precise. IoT, in general consists of inter-network of the devices and physical objects, number of objects can gather the data at remote locations and communicate to units managing, acquiring, organizing and analyzing the data in the processes and services. It provides a vision where things (wearable, watch, alarm clock, home devices, surrounding objects with) become smart and behave alive through sensing, computing and communicating by embedded small devices which interact with remote objects or persons through connectivity. The scalable and robust nature of Cloud computing is allowing developers to create and host their applications on it. Cloud acts as a perfect partner for IoT as it acts as a platform where all the sensor data can be stored and accessed from remote locations. These factors gave rise to the amalgamation of both technologies thus leading to the formation of a new technology called Cloud of Things(CoT) . In CoT the things(nodes) could be accessed, monitored and controlled from any remote location through the cloud. Due to high scalability in cloud any number of node could be added or removed from the IoT system on a real time basis. In simple terms IoT can be explained in form of an equation stating:  
The ideal of creating a Smart City is now becoming possible with the emergence of the Internet of Things. One of the key issues that smart cities relate to are car parking facilities and traffic management systems. In present day cities finding an available parking spot is always difficult for drivers, and it tends to become harder with ever increasing number of private car users. This situation can be seen as an opportunity for smart cities to undertake actions in order enhance the efficiency their parking resources thus leading to reduction in searching times, traffic congestion and road accidents. Problems pertaining to parking and traffic congestion can be solved if the drivers can be informed in advance about the availability of parking spaces at and around their intended destination. Recent advances in creating low-cost, low-power embedded systems are helping developers to build new applications for Internet of Things. Followed by the developments in sensor technology, many modern cities have opted for deploying various IoT based systems in and around the cities for the purpose of monitoring. A recent survey performed by the International Parking Institute reflects an increase in number of innovative ideas related to parking systems. At present there are certain parking systems that claim to citizens of delivering real time information about available parking spaces. Such systems require efficient sensors to be deployed in the parking areas for monitoring the occupancy as well as quick data processing units in order to gain practical insights from data collected over various sources. The smart parking system that we propose is implemented using a mobile application that is connected to the cloud. The system helps a user know the availability of parking spaces on a real time basis.

* 1. **NEED FOR IOT-CLOUD INTEGRATION**

Cloud computing and IoT have witnessed large evolution. Both the technologies have their advantages, however several mutual advantages can be foreseen from their integration. On one hand, IoT can address its technological constraints such as storage, processing and energy by leveraging the unlimited capabilities and resources of Cloud. On the other hand, Cloud can also extend its reach to deal with real world entities in a more distributed and dynamic fashion by the use of IoT. Basically, the Cloud acts as an intermediate between things and applications, in order to hide all the complexities and functionalities necessary for running the application. Below are some of the factors that led to the amalgamation of Cloud and IoT. *Storage capacity*: IoT comprises of a large number of information sources (things), which produce huge amounts of non-structured or semi-structured data. As a result IoT requires collecting, accessing, processing, visualizing and sharing large amounts of data. Cloud provides unlimited, low-cost, and on-demand storage capacity, thus making it the best and most cost effective solution to deal with data generated by IoT. The data stored on the Cloud can be accessed and visualized from anywhere through standard APIs. *Computation power*: The devices being used under IoT have limited processing capabilities. Data collected from various sensors is usually transmitted to more powerful nodes where its aggregation and processing can be done. The computation needs of IoT can be addressed by the use of unlimited processing capabilities and on-demand model of Cloud. With the help of cloud computing, IoT systems could perform real-time processing of data thus facilitating highly responsive applications. *Communication resources*. The basic functionality of IoT is to make IP-enabled devices communicate with one another through dedicated set of hardware. Cloud computing offers cheap and effective ways of connecting, tracking, and managing devices from anywhere over the internet. By the use of built-in applications IoT systems could monitor and control things on a real-time basis through remote locations. *Scalability*: Cloud provides a scalable approach towards IoT. It allows increase or decrease in resources in a dynamic fashion. Any number of “things” could be added or subtracted from the system when cloud integration is provided. The cloud allocates resources in accordance with the requirements of things and applications.

**CHAPTER 2**

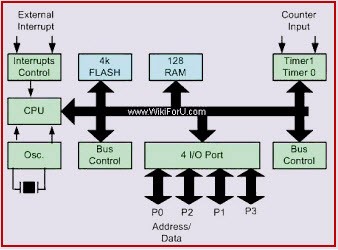
**BACKGROUND THEORY OF MICROCONTROLLER**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

This chapter provides background information regarding microcontroller and IOT.

**2.1 HISTORY OF MICROCONTROLLER**

A microcontroller (MCU for microcontroller unit) is a small computer on a single metal-oxide-semiconductor (MOS) integrated circuit (IC) chip. In modern terminology, it is similar to, but less sophisticated than, a system on a chip (SoC); a SoC may include a microcontroller as one of its components. A microcontroller contains one or more CPUs (processor cores) along with memory and programmable input/output peripherals. Program memory in the form of ferroelectric RAM, NOR flash or OTP ROM is also often included on chip, as well as a small amount of RAM. Microcontrollers are designed for embedded applications, in contrast to the microprocessors used in personal computers or other general purpose applications consisting of various discrete chips.



**FIGURE 2.1 ARCHITECTURE OF MICROCONTROLLER**

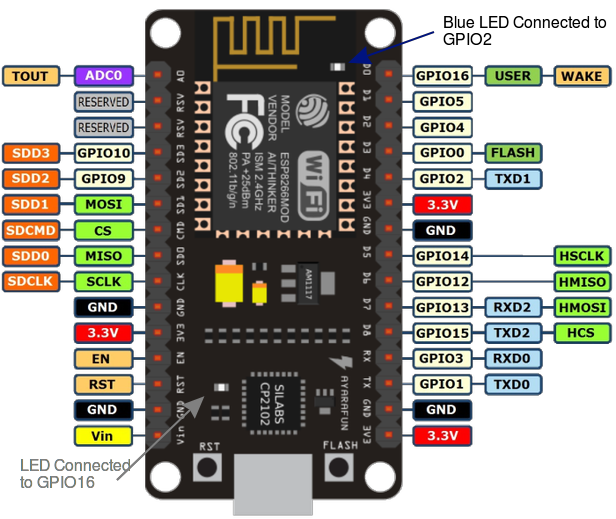
Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, appliances, power tools, toys and other embedded systems. By reducing the size and cost compared to a design that uses a separate microprocessor, memory, and input/output devices, microcontrollers make it economical to digitally control even more devices and processes. Mixed signal microcontrollers are common, integrating analog components needed to control non-digital electronic systems. In the context of the internet of things, microcontrollers are an economical and popular means of data collection, sensing and actuating the physical world as edge devices.

The origins of both the microprocessor and the microcontroller can be traced back to the invention of the MOSFET (metal-oxide-semiconductor field-effect transistor), also known as the MOS transistor.It was invented by Mohamed M. Atalla and Dawon Kahng at Bell Labs in 1959, and first demonstrated in 1960. The same year, Atalla proposed the concept of the MOS integrated circuit, which was an integrated circuit chip fabricated from MOSFETs. By 1964, MOS chips had reached higher transistor density and lower manufacturing costs than bipolar chips. MOS chips further increased in complexity at a rate predicted by Moore's law, leading to large-scale integration (LSI) with hundreds of transistors on a single MOS chip by the late 1960s. The application of MOS LSI chips to computing was the basis for the first microprocessors, as engineers began recognizing that a complete computer processor could be contained on a single MOS LSI chip.

The first multi-chip microprocessors, the Four-Phase Systems AL1 in 1969 and the Garrett AiResearch MP944 in 1970, were developed with multiple MOS LSI chips. The first single-chip microprocessor was the Intel 4004, released on a single MOS LSI chip in 1971. It was developed by Federico Faggin, using his silicon-gate MOS technology, along with Intel engineers Marcian Hoff and Stan Mazor, and Busicom engineer Masatoshi Shima. It was followed by the 4-bit Intel 4040, the 8-bit Intel 8008, and the 8-bit Intel 8080. All of these processors required several external chips to implement a working system, including memory and peripheral interface chips. As a result, the total system cost was several hundred (1970s US) dollars, making it impossible to economically computerize small appliances. MOS Technology introduced sub-$100 microprocessors, the 6501 and 6502, with the chief aim of addressing this economic obstacle, but these microprocessors still required external support, memory, and peripheral chips which kept the total system cost in the hundreds of dollars.

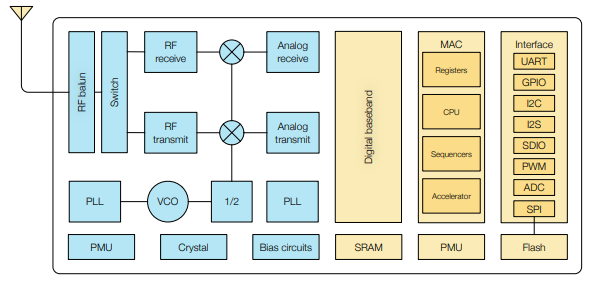
**2.2 OVERVIEW OF NODEMCU**

NodeMCU is a low-cost open source IoT platform. It initially included firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which was based on the ESP-12 module. Later, support for the ESP32 32-bit MCU was added.



**FIGURE 2.2 : NODEMCU PIN DIAGRAM**

NodeMCU is an open source firmware for which open source prototyping board designs are available. The name "NodeMCU" combines "node" and "MCU" (micro-controller unit). The term "NodeMCU" strictly speaking refers to the firmware rather than the associated development kits.



**FIGURE 2.3 : ARCHITECTURE OF NODEMCU**

Both the firmware and prototyping board designs are open source.

The firmware uses the Lua scripting language. The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS. Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented

NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to the NodeMCU project, enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

NodeMCU provides access to the GPIO (General Purpose Input/Output)

|  |  |
| --- | --- |
| I/O index | ESP8266 pin |
| 0 | GPIO16 |
| 1 | GPIO5 |
| 2 | GPIO4 |
| 3 | GPIO0 |
| 4 | GPIO2 |
| 5 | GPIO14 |
| 6 | GPIO12 |
| 7 | GPIO13 |
| 8 | GPIO15 |
| 9 | GPIO3 |
| 10 | GPIO1 |
| 11 | GPIO9 |
| 12 | GPIO10 |

**TABLE 2.1 GPIO TABLE OF NODEMCU**

**2.3 OVERVIEW OF BLYNK**

Blynk is a Platform with IOS and Android apps to control Arduino, Raspberry Pi and the likes over the Internet. It’s a digital dashboard where you can build a graphic interface for your project by simply dragging and dropping widgets.

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, vizualize it and do many other cool things.

There are three major components in the platform:

* **Blynk App** - allows to you create amazing interfaces for your projects using various widgets we provide.
* **Blynk Server** - responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your private Blynk server locally. It’s open-source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.
* **Blynk Libraries** - for all the popular hardware platforms - enable communication with the server and process all the incoming and outcoming commands.

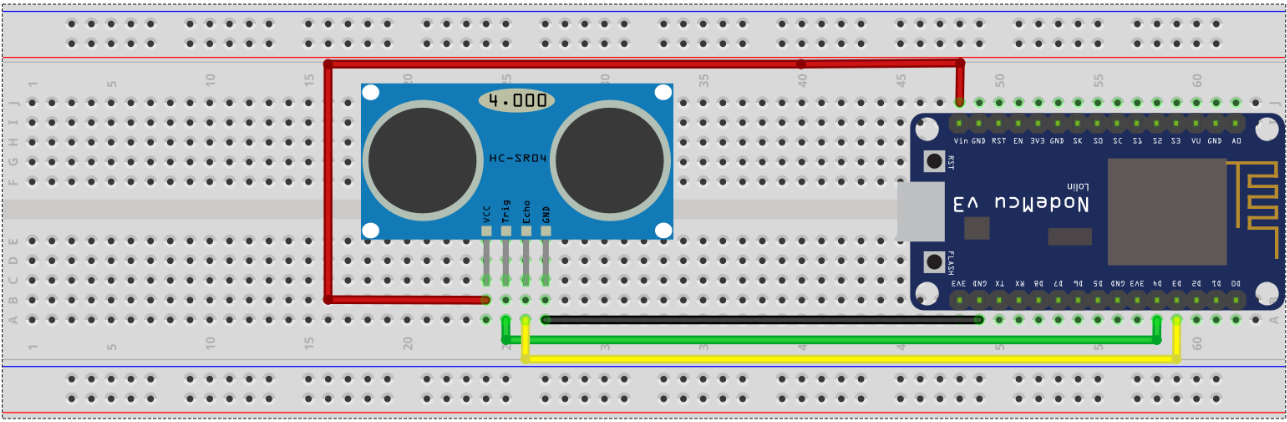
**2.4 OVERVIEW OF ULTRASONIC SENSOR**

The Ultrasonic sensor emits an ultrasound at 40 000 Hz which travels through the air and if there is an object or obstacle on its path It will bounce back to the module. Considering the travel time and the speed of the sound you can calculate the distance



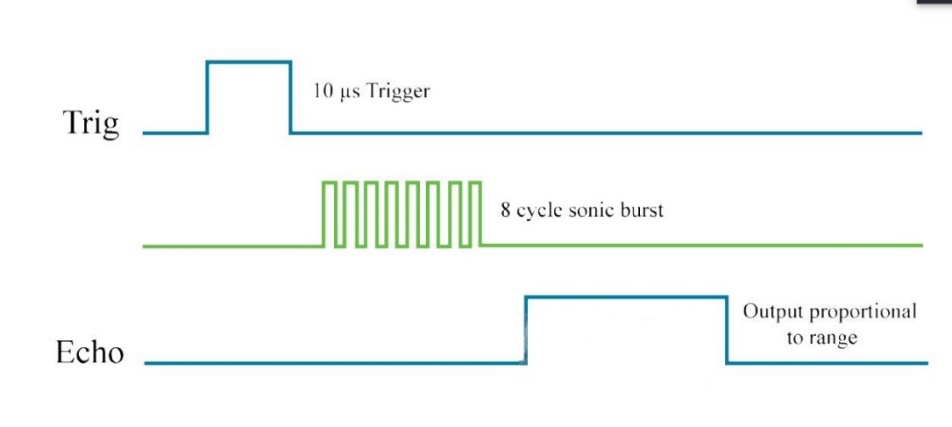
**FIGURE 2.4 : WORKING OF ULTRASONIC SENSOR**

The HC-SR04 Ultrasonic Module has 4 pins, Ground, VCC, Trig and Echo. The Ground and the VCC pins of the module needs to be connected to the Ground and the 5 volts pins on the NodeMCU Board respectively and the trig and echo pins to any Digital I/O pin on the NodeMCU Board.



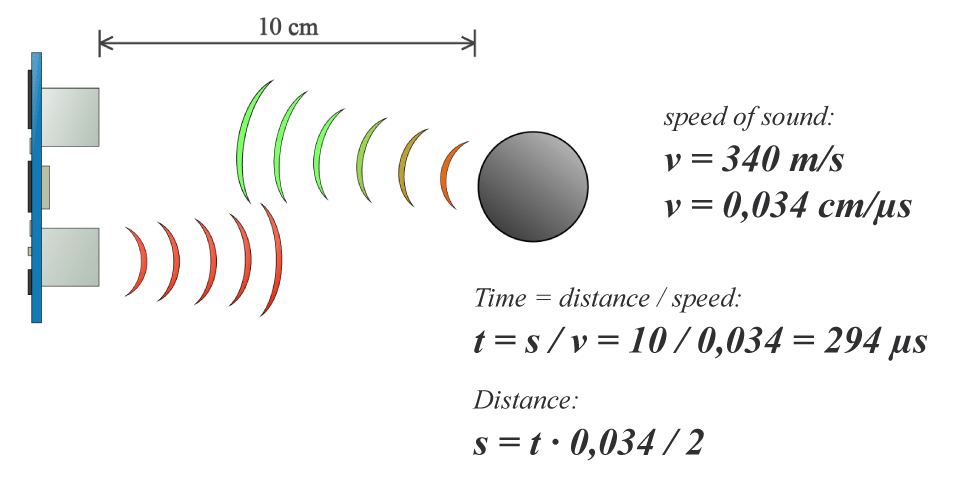
**FIGURE 2.5 : CONNECTION DIAGRAM OF HC-SR04 WITH NODEMCU**

In order to generate the ultrasound you need to set the Trig on a High State for 10 µs. That will send out an 8 cycle sonic burst which will travel at the speed sound and it will be received in the Echo pin. The Echo pin will output the time In microseconds the sound wave travelled.



**FIGURE 2.6 :WORKING OF TRIG AND ECHO PIN**

For example, if the object is 10 cm away from the sensor, and the speed of the sound is 340 m/s or 0.034 cm/µs the sound wave will need to travel about 294 u seconds. But what you will get from the Echo pin will be double that number because the sound wave needs to travel forward and bounce backward.  So in order to get the distance in cm we need to multiply the received travel time value from the echo pin by 0.034 and divide it by 2.



**FIGURE 2.7 : CALCULATION OF DISTANCE USING ULTRASONIC SENSOR**

**CHAPTER 3**

**DESIGN OF SMART PARKING SYSTEM**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

In this chaper, we would be designing the Smart Parking System using NodeMCU , Ultrasound sensor and Blynk.

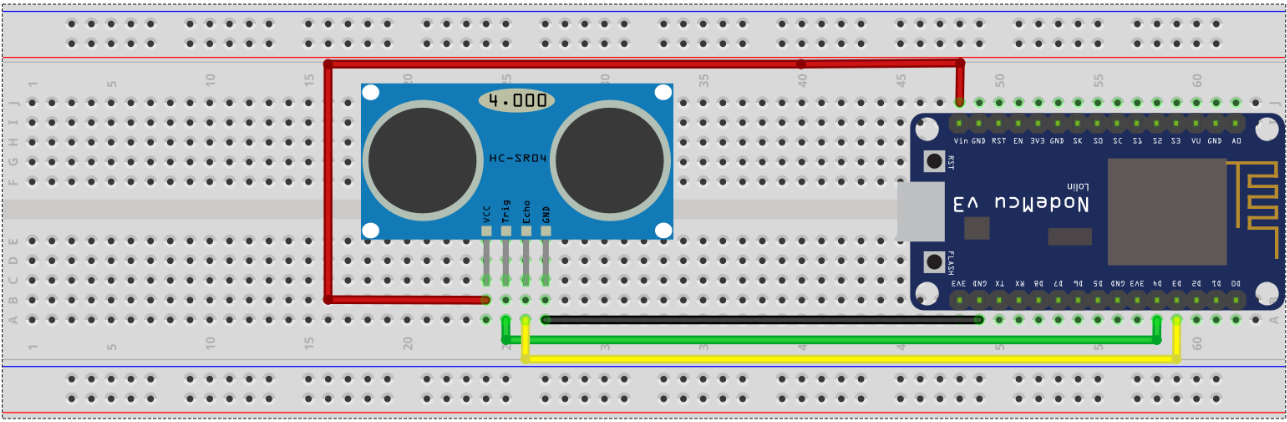
**3.1 CIRCUIT DESIGN**

The **TRIGGER** pin was connected to GPO-2(D4)

The **ECHO** pin was connected to GPO-0(D3)

The Vcc of Ultrasound Sensor (HC-SR04) is connected to Vin of NodeMCU

The Ground pin of Ultrasound Sensor (HC-SR04) was connected to Ground pin of NodeMCU



**FIGURE 3.1 : CONNECTION DIAGRAM OF HC-SR04 WITH NODEMCU**

|  |  |
| --- | --- |
| **WIRE** | **COLOR** |
| Vcc | Red |
| Ground | Black |
| Echo | Yellow |
| Trigger | Green |

**TABLE 3.1 : COLOR REFERENCE FOR WIRES**

**3.2 CODING FOR THE ULTRASOUND SENSOR**

This project was done using two NodeMCU as reference

Code for the first NodeMCU :

#define TRIGGER 2 // D4 pin of Node MCU

#define ECHO 0 // D3 pin of Node-MCU

#define BLYNK\_PRINT Serial

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h> // Blynk Library

char auth[] = "m0u\_DLuAWUf9DbaXBJvnLw0kEbyUBZVm";

// Your WiFi credentials.

// Set password to "" for open networks.

char ssid[] = "SSID";

char pass[] = "password";

void setup() {

Serial.begin (9600);

Blynk.begin(auth, ssid, pass); // Taking WiFi credentials.

pinMode(TRIGGER, OUTPUT); // trigger throws a signal to the object

pinMode(ECHO, INPUT); // takes the returning signal

}

void loop() {

long duration, distance;

digitalWrite(TRIGGER, LOW);

delayMicroseconds(2);

digitalWrite(TRIGGER, HIGH);

delayMicroseconds(10);

digitalWrite(TRIGGER, LOW);

duration = pulseIn(ECHO, HIGH);

distance = (duration/2) / 29.1; // Calculating distance

if (distance <=50) {

Blynk.virtualWrite(V0, "Full"); // Printing that space is full when the distance of a object less or 50 cm

}

else {

Blynk.virtualWrite(V0,"Vacant"); // Print vacant when there is no car

} // V0 is a virtual pin assigned by blynk platform

}

Code for 2nd NodeMCU : #define TRIGGER 2 // D4 pin of Node MCU

#define ECHO 0 // D3 pin of Node-MCU

#define BLYNK\_PRINT Serial

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h> // Blynk Library

char auth[] = "XhiikDQv4hvRByK3KgvAHhYYbzEGeO7T";

// Your WiFi credentials.

// Set password to "" for open networks.

char ssid[] = "SSID";

char pass[] = "password";

void setup() {

Serial.begin (9600);

Blynk.begin(auth, ssid, pass); // Taking WiFi credentials.

pinMode(TRIGGER, OUTPUT); // trigger throws a signal to the object

pinMode(ECHO, INPUT); // takes the returning signal

}

void loop() {

long duration, distance;

digitalWrite(TRIGGER, LOW);

delayMicroseconds(2);

digitalWrite(TRIGGER, HIGH);

delayMicroseconds(10);

digitalWrite(TRIGGER, LOW);

duration = pulseIn(ECHO, HIGH);

distance = (duration/2) / 29.1; // Calculating distance

if (distance <=50) {

Blynk.virtualWrite(V1, "Full"); // Printing that space is full when the distance of a object less or 50 cm

}

else {

Blynk.virtualWrite(V1,"Vacant"); // Print vacant when there is no car

} // V1 is a virtual pin assigned by blynk platform

}

**3.3 CODE EXPLANATION**

* The libraries of ESP8266 and Blynk were imported
* Authentication token was defined
* SSID and password were defined
* pinMode() was used to define the TRIGGER and ECHO pin
* Blynk.begin() was used to pass the Authentication ID, SSID and Password
* digitalWrite() was used to make the TRIGGER pin high and low respectively for a certain duration of time using delayMicroseconds()
* pulseIn() was used to receive signal from the object
* Blynk.virtualWrite() is used to define the virtual pins of Blynk software

**3.4 PROJECT EXPLANATION**

The Ultrasonic sensor emits an ultrasound at 40 000 Hz which travels through the air and if there is an object or obstacle on its path It will bounce back to the module. The whole thing works on the principle of “Doppler effect”.

The Doppler Effect is the change in frequency of a wave in relation to an observer who is moving relative to the wave source.

The TRIGGER pin emits a sound wave towards the object.

The ECHO pin receives the rebound wave.

The time taken for the wave to rebound is calculated.

The distance is calculated by the formula s=vt , where v is the speed of sound

If the distance is larger than a certain margin, then the parking spot is considered to be vacant.

If the distance is less than the margin then, the parking spot is considered to be filled.

In either case, the data is sent over to the Blynk server and the user can check if that parking spot is empty or filled.

Since, all the microcontroller(NodeMCU) are acting autonomously, data of various spots can be placed at a single place.

**CONCLUSION**

The concept of Smart Cities have always been a dream for humanity. Since the past couple of years large advancements have been made in making smart cities a reality. The growth of Internet of Things and Cloud technologies have give rise to new possibilities in terms of smart cities. Smart parking facilities and traffic management systems have always been at the core of constructing smart cities. In this paper, we address the issue of parking and present an IoT based Cloud integrated smart parking system. The system that we propose provides real time information regarding availability of parking slots in a parking area. The efforts made in this paper are indented to improve the parking facilities of a city and thereby aiming to enhance the quality of life of its people.

**PLANNING AND PROJECT MANAGEMENT**

**Table :** Showing details about project planning and management

|  |  |  |
| --- | --- | --- |
| Activity | Starting week | Number of weeks |
| Literature review | 1st week of March | 1 |
| Finalising problem | 2nd week of March | 1 |
| Getting familiarised with the required software and tool | 3rd week of March | 1 |
| Finalising the project | 4th week of March | 1 |
| Report Making | 1st week of April | 1 |

**REFERENCE**

**[1] Wikipedia website,** <https://en.wikipedia.org/wiki/NodeMCU>

**[2] Kafle, V. P., Fukushima, Y., & Harai, H. (2015, April). ID-based communication for realizing IoT and M2M in future heterogeneous mobile networks. In Recent Advances in Internet of Things (RIoT), 2015 International Conference on (pp. 1-6). IEEE.**

**[3] Sarkar, C., Uttama Nambi SN, A., Prasad, R., Rahim, A., Neisse, R., & Baldini, G. (2012). DIAT: A Scalable Distributed Architecture for IoT.**

**[4] Arrow website,** <https://www.arrow.com/en/research-and-events/articles/ultrasonic-sensors-how-they-work-and-how-to-use-them-with-arduino#:~:text=Ultrasonic%20sensors%20work%20by%20emitting,return%20after%20hitting%20an%20object>

**[5] Howtomechatronics website,** <https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/>

**[6] Blynk website,** <https://docs.blynk.cc/>