

AN IoT BASED SMART OUTDOOR PARKING SYSTEM

Submitted in partial fulfilment of the requirements for the award of
Bachelor of Engineering Degree in
Electronics and Communication Engineering

By

SAREDDU NAGA YUGANDHAR PRASAD (38130195)

MARADA NANDA KISHORE (38130129)

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

**SCHOOL OF ELECTRICAL AND ELECTRONICS
ENGINEERING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade “A” by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH 2022





BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **SAREDDU NAGA YUGANDHAR PRASAD (38130195)** and **MARADA NANDA KISHORE (38130129)** who carried out the project entitled “ **AN IoT BASED SMART OUTDOOR PARKING SYSTEM**” under our supervision from December 2021 to March 2022.

Internal Guide
(Mrs.T. VIJAYASHREE)

Dr. T. RAVI, M.E., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on _____

Name: _____

Signature: _____

Internal Examiner

External Examiner

DECLARATION

We **S.N.YUGANDHAR PRASAD** (38130195) and **M.NANDA KISHORE** (38130129) hereby declare that the Project Report entitled “ **AN IoT BASED SMART OUTDOOR PARKING SYSTEM**” done by us under the guidance of **Mrs.T.VIJAYASHREE** at Sathyabama Institute of Science and Technology, Chennai- 600119 is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

1.

2.

SIGNATURE

OF

CANDIDATES

DATE :

PLACE :

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to Board of Management of SATHYABAMA for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr.N.M.Nandhita, Dean, School of Electrical and Electronics Engineering** and **Dr.T.RAVI, Head of the Department, Dept. of Electrical and Electronics Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would also like to express our sincere and deep sense of gratitude to our internal guide **Mrs.T.VIJAYASHREE** for her valuable guidance and suggestions during the progressive reviews for our project.

ABSTRACT

The Internet of Things (IoT) is a large technical field that is actively being researched and implemented. Several Internet-of-Things (IoT)-based technologies offer extra benefits while lowering labour costs. As the world's population expands, so does the demand for transportation. As a result, the dependency on parking autos expands. People visiting public facilities in large cities, such as malls, parks, temples, and theatres, are finding it difficult to locate parking. Despite the existence of various established systems, the bulk of parking places are still controlled by hand. The only way to travel about in most smart cities is to use a well-organized indoor parking system. Regulations for automobile parking in the open air have yet to be created and implemented. Traditional parking methods are far too inefficient and inconvenient for metropolitan areas where it is difficult to find available spaces. This might result in a lot of traffic, small collisions, and public accidents. As a result, a smart outdoor parking system based on IoT (Internet of Things) and weighbridgeload sensors may be built for the organised, well-timed, flexible, easy, and safe parking of automobiles in public spaces.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
1	EMBEDDED SYSTEM	7
	1.1 Overview of embedded systems	8
	1.1.1 Block diagram of an embedded system	9
	1.1.2 Embedded system software	11
	1.2 INTERNET OF THINGS	14
	1.2.1 EXAMPLE OF AN IoT SYSTEM	18
2	LITERATURE SURVEY	20
	2.1 ADVANCE SMART CAR PARKING SYSTEM	20
	2.2 ANDROID BASED CAR PARKING SYSTEM	22
	2.3 PARKING MONITORING SYSTEM BASED ON MFS	23
3	PROJECT DISCRIPTION	25
	3.1 EXIXSTING SYSTEM	25
	3.2 PROPOSED SYSTEM	25
	3.3 BLOCK DIAGRAM	26
	3.4 HARDWARE REQUIREMENTS	26
	3.5 SOFTWARE REQUIREMENTS	26
4	HARDWARE DESCRIPTION	27
	4.1 ARDUINO UNO	28
	4.2 LOAD CELL	32
	4.3 ULTASONIC SENSOR	36
	4.4 LED	42
	4.5 BUZZER	49
	4.6 ESP 8266 WIFI MODULE	55
5	SOFTWARE DESCRIPTION	60
	5.1 ARDUINO IDE	61
	5.2 EMBEDDED C	63
6	CONCLUSION	68

CHAPTER-1

INTRODUCTION

As quote says" why walk when drive " that is when journey on wheels started and which has created hard ongoing problem to park vehicles into parking slots.in urban cities as population is going on increasing the production of vehicles has also increased but parking space available has become congested to park N number of vehicles out of that people are finding it a great problem to search the available space during peak hours and festive times which consumes lot of energy and waste valuable time which is a major drawback, in order to replace the above issue .Internet of things was first introduced in 1999 at auto ID center and first used by Kevin Aston. As evolving this latest burning technology, it promises to connect all our surrounding things to a network and communicating with each other with less human involvement. Still internet of things is in beginning stage and there is no common architecture exists till today. There is lot of researches and implementations are currently being going on in all the respective areas. Thus, there is no guidelines or boundaries exists to define the definition of internet of things. So, depending on the context, application the internet of things has different definitions. Shortly it is defined as the things present in the physical world or in an environment are attached with sensors or with any embedded systems and made connected to network via wired or wireless connections. These connected devises are called as smart devices or smart objects. And it consists of smart machines which communicating interacting with other machines, environment, objects etc. And also, it incorporates to connect any two machines, machine to human and vice versa etc. IOT technology grows in various fields of smart applications but we have not yet found boundary constraints of this technology. Some smart applications which it has implementing currently such as on smart grids, smart lighting,

smart energy, smart city, smart health etc. This is broadly classified into three categories such as sensing, processing and connectivity. Whereas sensing includes sensing the speed of vehicles and humans or any objects (accelerometer), sensing of temperature, pressure etc. And these can be processing by using some processors such as network processor, hybrid processor MCU/MPU etc. And the devices are connected by using some technologies called GPS, Wi-Fi, RFID etc.

1.1 EMBEDDED SYSTEM:

An embedded system is a combination of computer hardware and software, either fixed in capability or programmable, designed for a specific function or functions within a larger system. Industrial machines, agricultural and process industry devices, automobiles, medical equipment, cameras, household appliances, airplanes, vending machines and toys, as well as mobile devices, are possible locations for an embedded system.

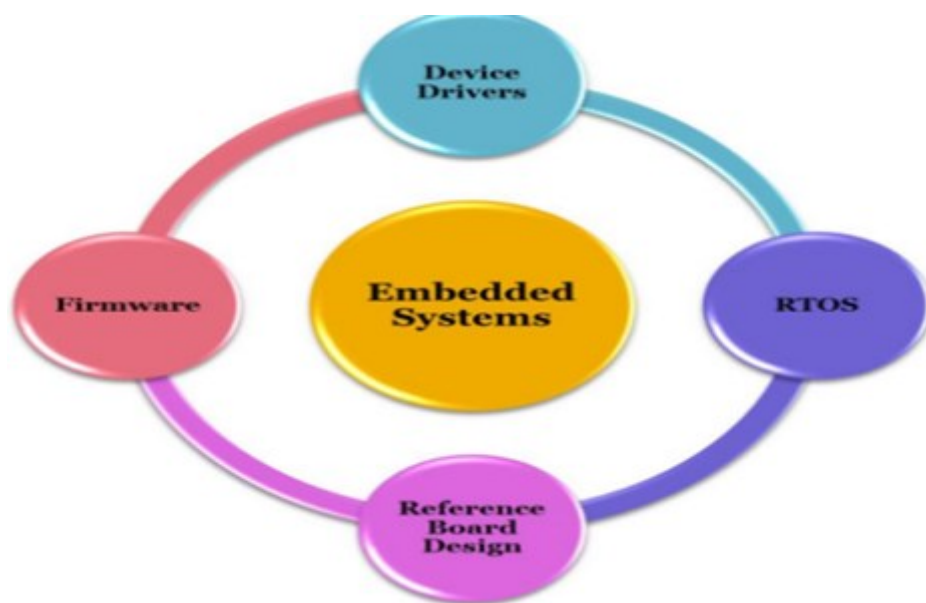
Embedded systems are computing systems, but they can range from having no user interface (UI) -- for example, on devices in which the system is designed to perform a single task -- to complex graphical user interfaces (GUIs), such as in mobile devices. User interfaces can include buttons, LEDs, touchscreen sensing and more. Some systems use remote user interfaces as well.

Embedded system hardware (microprocessor-based, microcontroller-based)

Embedded system hardware can be microprocessor- or microcontroller-based. In either case, an integrated circuit is at the heart of the product

that is generally designed to carry out computation for real-time operations. Microprocessors are visually indistinguishable from microcontrollers, but while the microprocessor only implements a central processing unit (CPU) and, thus, requires the addition of other components such as memory chips, microcontrollers are designed as self-contained systems.

Microcontrollers include not only a CPU, but also memory and peripherals such as flash memory, RAM or serial communication ports. Because microcontrollers tend to implement full (if relatively low computer power) systems, they are frequently put to use on more complex tasks. For example, microcontrollers are used in the operations of vehicles, robots, medical devices and home appliances, among others. At the higher end of microcontroller capability, the term system on a chip (SoC) is often used, although there's no exact delineation in terms of RAM, clock speed and so on.



The embedded market was estimated to be in excess of \$140 billion in 2013, with many analysts projecting a market larger than \$20 billion by 2020. Manufacturers of chips for embedded systems include many mainstays of the computer world, such as Apple, IBM, Intel and Texas Instruments, as well as numerous other companies less familiar to those outside the field. Arm has been a highly influential vendor in this space. The company began as an outgrowth of Acorn, a U.K. maker of early PCs. Arm chips, produced under license by other companies, are based on the reduced instruction set computer (RISC) architecture and are often used in mobile phones; they remain the most widely deployed SoC in the embedded world, with billions of units fielded.



Embedded system software

A typical industrial microcontroller is unsophisticated compared to the typical enterprise desktop computer and generally depends on a simpler, less-memory-intensive program environment. The simplest devices run on bare metal and are programmed directly using the chip CPU's machine code language.

Often, embedded systems use operating systems or language platforms tailored to embedded use, particularly where real-time operating environments must be served. At higher levels of chip capability, such as those found in SoCs, designers have increasingly decided the systems are generally fast enough and the tasks tolerant of slight variations in reaction time that near-real-time approaches are suitable. In these instances, stripped-down versions of the Linux operating system are commonly deployed, although other operating systems have been pared down to run on embedded systems, including Embedded Java and Windows IoT (formerly Windows Embedded).

Generally, storage of programs and operating systems on embedded devices make use of either flash or rewritable flash memory.

Embedded firmware

The firmware on embedded systems, referred to as embedded firmware, is specific software written into the memory of a device that serves the purpose of ROM, but can be updated more easily. Firmware can be stored in non-volatile memory devices including ROM, programmable ROM, erasable PROM or flash memory. Embedded firmware is used to

control various device and system functions, for example, telling the device how to communicate with other devices, perform specific functions and provide input and output functionality.

The delineation between the terms embedded firmware and embedded software are blurring, but embedded software often refers to the only code running on a piece of hardware, while firmware can also refer to the chip that houses a device's basic input/output system (BIOS) or Unified Extensible Firmware Interface (UEFI), which connect software and a system's operating system.

Embedded systems vs. VLSI

Very-large-scale integration, or VLSI, is a term that describes the complexity of an integrated circuit. VLSI is the process of embedding hundreds of thousands of transistors into a chip, whereas LSI (large-scale integration) microchips contain thousands of transistors, MSI (medium-scale integration) contain hundreds of transistors, and SSI (small-scale integration) contain tens of transistors. ULSI, or ultra-large-scale integration, refers to placing millions of transistors on a chip.

VLSI circuits are common features of embedded systems. Many ICs in embedded systems are VLSI, and the use of the VLSI acronym has largely fallen out of favor.

Debugging embedded systems

One area where embedded systems part ways with the operating systems and development environments of other, larger-scale computers is in the area of debugging. While programmers working with desktop computer environments have systems that can run both the code being developed and separate debugger applications that monitor the actions of the development code as it is executed, embedded system programmers generally cannot.

Some programming languages run on microcontrollers with enough efficiency that rudimentary interactive debugging is available directly on the chip. Additionally, processors often have CPU debuggers that can be controlled -- and, thus, control program execution -- via a JTAG or similar debugging port.

In many instances, however, programmers of embedded systems need tools that attach a separate debugging system to the target system via a serial or other port. In this scenario, the programmer can see the source code on the screen of a conventional personal computer just as would be the case in the debugging of software on a desktop computer. A separate, frequently used approach is to run software on a PC that emulates the physical chip in software, thus making it possible to debug the performance of the software as if it were running on an actual, physical chip.

Broadly speaking, embedded systems have received more attention to testing and debugging because a great number of devices using embedded

controls are designed for use in situations where safety and reliability are top priorities.

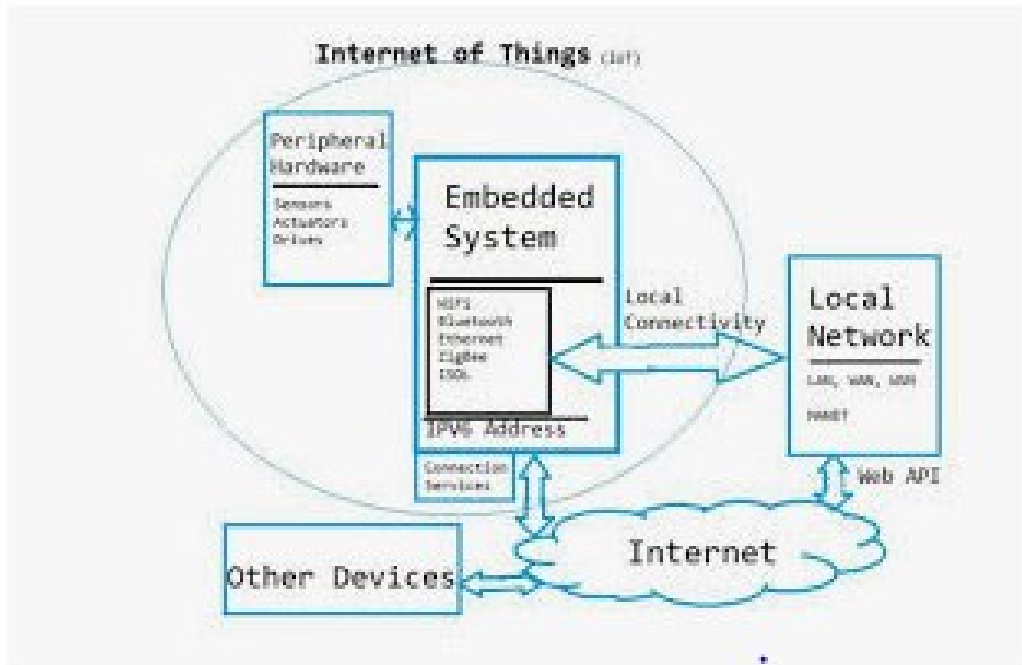
The internet of things builds on an embedded systems base

While some embedded systems can be relatively simple, a growing number either supplant human decision-making or offer capabilities beyond what a human could provide. For instance, some aviation systems, including those used in drones, are able to integrate sensor data and act upon that information faster than a human could, permitting new kinds of operating features.

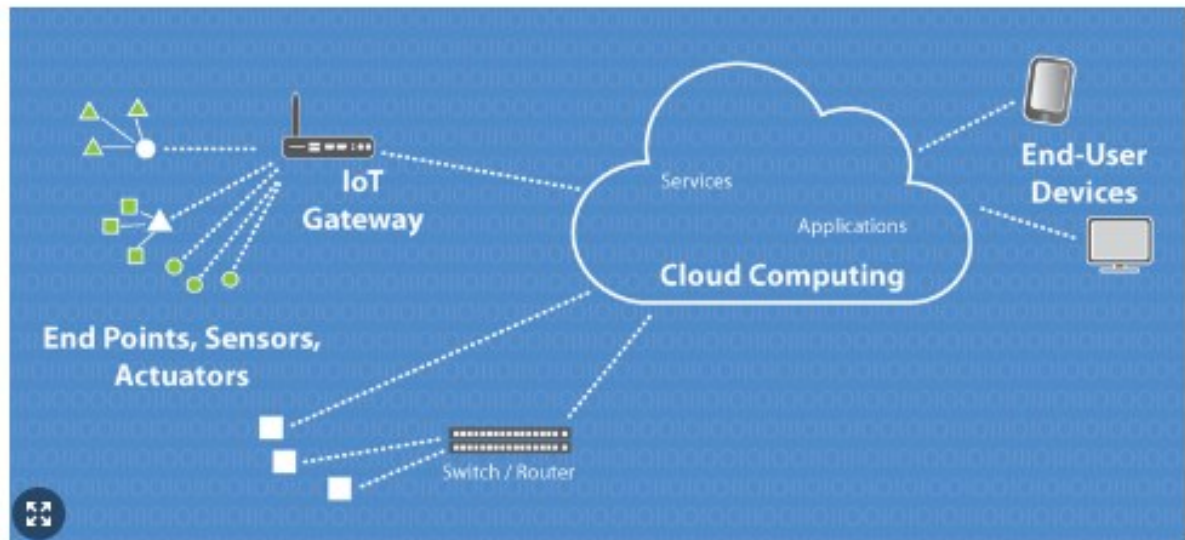
1.2 INTERNET OF THINGS:

- The term Internet of Things generally refers to scenarios where network connectivity and computing capability extends to objects, sensors and everyday items not normally considered computers, allowing these devices to generate, exchange and consume data with minimal human intervention. There is, however, no single, universal definition.
- Enabling Technologies: The concept of combining computers, sensors, and networks to monitor and control devices has existed for decades. The recent confluence of several technology market trends, however, is bringing the Internet of Things closer to widespread reality. These include Ubiquitous Connectivity, Widespread Adoption of IP-based Networking, Computing Economics, Miniaturization, Advances in Data.
- Connectivity Models: IoT implementations use different technical communications models, each with its own characteristics. Four common communications models described by the Internet

Architecture Board include: Device-to-Device, Device-to-Cloud, Device-to-Gateway, and Back-End Data-Sharing. These models highlight the flexibility in the ways that IoT devices can connect and provide value to the user.



IoT devices are implemented using both hardware and software components. Dedicated hardware components are used to implement the interface with the physical world, and to perform tasks which are more computationally complex. Microcontrollers are used to execute software that interprets inputs and controls the system. This module discusses the roles of both the hardware and software components in the system. The functions of common hardware components are described and the interface between the software and hardware through the microcontroller is explained. IoT devices often use an operating system to support the interaction between the software and the microcontroller. We will define the role of an operating system in an IoT device and how an IoT operating system differs from a standard one.

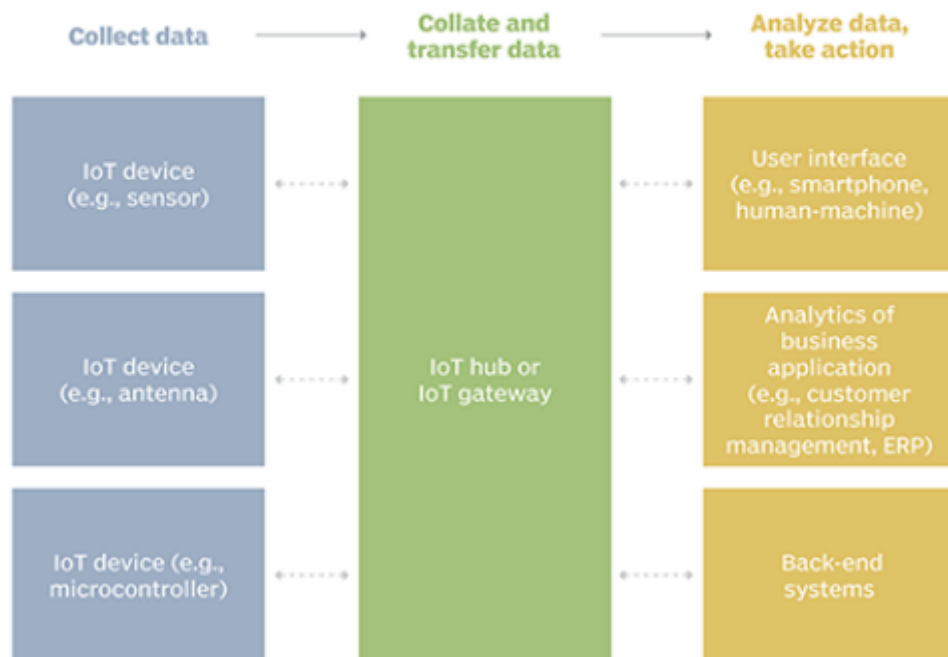


How IoT works

An IoT ecosystem consists of web-enabled smart devices that use embedded processors, sensors and communication hardware to collect, send and act on data they acquire from their environments. IoT devices share the sensor data they collect by connecting to an IoT gateway or other edge device where data is either sent to the cloud to be analyzed or analyzed locally. Sometimes, these devices communicate with other related devices and act on the information they get from one another. The devices do most of the work without human intervention, although people can interact with the devices -- for instance, to set them up, give them instructions or access the data.

The connectivity, networking and communication protocols used with these web-enabled devices largely depend on the specific IoT applications deployed .

Example of an IoT system



Benefits of IoT

The internet of things offers a number of benefits to organizations, enabling them to:

- monitor their overall business processes;
- improve the customer experience;
- save time and money;
- enhance employee productivity;
- integrate and adapt business models;
- make better business decisions; and

- generate more revenue.

IoT encourages companies to rethink the ways they approach their businesses, industries and markets and gives them the tools to improve their business strategies.

Consumer and enterprise IoT applications

There are numerous real-world applications of the internet of things, ranging from consumer IoT and enterprise IoT to manufacturing and industrial IoT (IIoT). IoT applications span numerous verticals, including automotive, telco, energy and more.

In the consumer segment, for example, smart homes that are equipped with smart thermostats, smart appliances and connected heating, lighting and electronic devices can be controlled remotely via computers, smartphones or other mobile devices.

Wearable devices with sensors and software can collect and analyze user data, sending messages to other technologies about the users with the aim of making users' lives easier and more comfortable. Wearable devices are also used for public safety -- for example, improving first responders' response times during emergencies by providing optimized routes to a location or by tracking construction workers' or firefighters' vital signs at life-threatening sites.

In healthcare, IoT offers many benefits, including the ability to monitor patients more closely to use the data that's generated and analyze it. Hospitals often use IoT systems to complete tasks such as inventory management, for both pharmaceuticals and medical instruments.

CHAPTER-2

LITERATURE SURVEY

ADVANCED CAR PARKING SYSTEM USING ARDUINO

This paper explains the architecture and design of Arduino based car parking system. Authorization of driver or user is the basic rule used to park a vehicle in a parking place. Authorization card will be given to each user, which carries the vehicle number or other details. If the user is authorized and space is available in the parking, then the parking gate will open and the user is allowed to park the vehicle in parking place else the user is not allowed even the user is authorized person. If car is allowed to park, then mobile notification will be sent to user about parking. It solves the parking issue in urban areas, also provides security to a vehicle and an unauthorized user is not allowed to enter into a parking place. It helps to park vehicle in multfloored parking also as it will display which floor has free space.

SMART URBAN PARKING DETECTION SYSTEM

Nowadays in an urban area, looking for available parking spots, is not only a waste of time and energy but also it causes more traffic and air pollution. Therefore, finding available parking spots is becoming more challenging on daily life. This paper proposes a novel solution by providing motorists with real-time information about the availability of

spots in parking spaces through a smartphone application. Ultrasonic sensors and raspberry pi were used at the gates of parking zones to detect cars and calculate the number of available parking spots. Client web server was used to accommodate multiple users in an android platform. Drivers can see available parking spots in real time on a map, they also can get navigate to nearest available parking spots using the mobile application. The parking system is designed for different types of parking zone (such as open space car parking and multilevel car parking); the cost of deploying the system is inexpensive due to the low demand for Raspberry pi single board computer as a local server and ultrasonic sensors for each entrance and exit gates compared to those that require a sensor for each parking spot.

ANDROID BASED SMART CAR PARKING SYSTEM

The purpose of this system is to computerize the parking space reservation. Its talk about undertaking which introduces a miniature model of car parking that can direct and manage the number of cars that can be parked in given space at any given time based on availability of parking space after doing the registration by user using android application on his smart phone. Automated parking is a strategy for parking and leaving cars utilizing detecting device i.e., sensors. The entering to or leaving from parking lot is also commanded by an android based application. This provide users to book parking spaces online in advance for given location and then park the vehicle with minimal fees. We have concentrated on some current systems and it shows that the

current systems are not totally automated and require a certain level of human interference and communication with the system. The difference between our system and existing systems is that we intend to make our system as less human independent by automating whole parking area.

THE STUDY ON INTELLIGENT ROADSIDE PARK CHARGING SYSTEMS

Currently, roadside parking charging is mainly operated manually. However, this makes it difficult to achieve effective management. Aimed at removing many inconveniences brought by this current approach, this study discusses how to apply Radio Frequency Identification (RFID) in this issue. The RFID reader can be installed on a track. Through the track, the RFID reader can detect a whole row of parking spaces along the sides of the road and therefore, can replace the current approach with which people hold machines to scan cars one by one. Both parkers and managers of the parking system can use an online platform which offers an open individual inquiry system and impartial and accurate reflection of the real-time status, and ensures a fair process of issuing parking tickets.

PARKNET: A MOBILE SENSOR NETWORK FOR HARVESTING REAL TIME VEHICULAR PARKING INFORMATION

This paper describes the architecture and design of ParkNet, a mobile sensor network consisting of vehicles, which collects and disseminates

real-time information about the availability of parking spaces in urban areas. We outline the broad challenges in real-time data collection and consumption by a mobile sensor network, as well as the design issues specific to ParkNet. Sensor nodes in ParkNet are a combination of privately owned and city owned vehicles and employ low-cost ultrasonic sensors to detect the presence of vacant parking spots as they drive by. We present early results on the performance of our sensor platform.

PARKING MONITOR SYSTEM BASED ON MAGNETIC FIELD SENSORS

To easily find an unoccupied parking space in a large car park is a problem for many drivers. Thus, it is useful to have technical solutions which can provide information on parking space occupancy. A new monitor system is described in the following. It is based on passive magnetic field sensors. It provides occupancy information for car park users and helps them to place the car in a most efficient way.

IRISNET: AN ARCHITECTURE FOR INTERNET-SCALE SENSING SERVICES

We demonstrate the design and an early proto - type of IrisNet (Internet - scale Resource - Intensive Sensor Network services), a common, scalable networked infrastructure for deploying wide area sensing services IrisNet is a potentially global network of smart sensing nodes, with webcams or other monitoring devices, and organizing nodes that provide the means to query recent and historical sensor - based data IrisNet exploits the fact that high - volume sensor feeds are typically attached to devices with

significant computing power and storage, and running a standard operating system. It uses aggressive filtering, smart query routing, and semantic caching to dramatically reduce network bandwidth utilization and improve query response times, as we demonstrate. Our demo will present two services built on Iris - Net, from two very different application domains. The first one, a parking space finder, utilizes webcams that monitor parking spaces to answer queries such as the availability of parking spaces near a user's destination. The second one, a distributed infrastructure monitor, uses measurement tools installed in individual nodes of a large distributed infrastructure to answer queries such as average network bandwidth usage of a set of nodes.

CHAPTER-3

PROJECT DESCRIPTIONS

3.1 EXISTING SYSTEM:

- In existing system, they used car system parking done manually.

DRAWBACKS

- Parking slot availability is not known to the user before.
- Parking slot mismatch for the user in manual method.

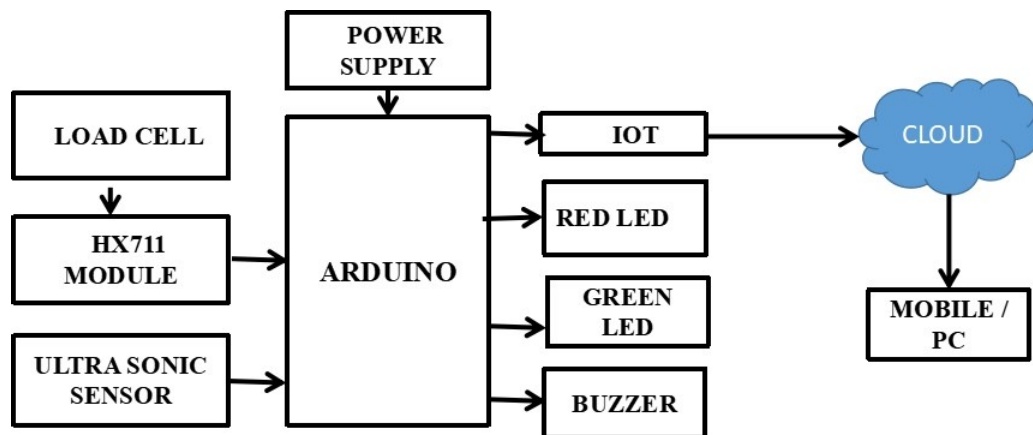
3.2 PROPOSED SYSTEM:

- In proposed system, the car parking system is done automatically with the help of internet of things (IOT).

ADVANTAGES

- User will know the parking slot availability using IOT.
- User friendly and cannot mismatch.

3.3 BLOCK DIAGRAM:



3.4 HARDWARE REQUIREMENTS

- Arduino uno
- Load Cell
- UltraSonic Sensor
- LED
- Buzzer
- ESP8266

3.5 SOFTWARE REQUIREMENTS

- ARDUINO IDE
- EMBEDDED C

CHAPTER-4

HARDWARE DESCRIPTIONS

4.1 ARDUINO UNO:

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno 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. Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode. Revision 3 of the board has the following new features:

- 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2. "Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

POWER:

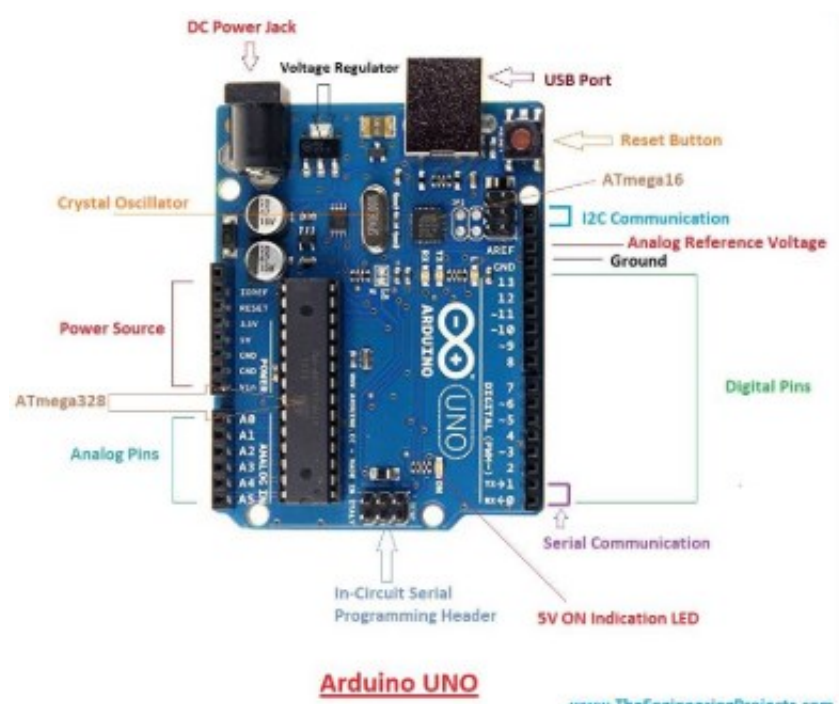
VIN. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

- 5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.

- 3V 3. A 3.3-volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

- GND. Ground pins.

ARDUINO UNO:



Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and

written with the EEPROM library). Input and Output Each of the 14 digital pins on the Uno can be used as an input or output, using pin Mode (), digital Write (), and digital Read () functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kilo Ohms.

In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attach Interrupt () function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analog Write () function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off. The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e., 1024 different values). By default, they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analog Reference () function. Additionally, some pins have specialized functionality:
- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library. There are a couple of other pins on the board:
- AREF. Reference voltage for the analog inputs. Used with analog Reference ().

- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board. See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical. Communication The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers.

The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed.

However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A Software Serial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

PIN CONFIGURATION:

THE ARDUINO

MCU : Atmega 328
 Input voltage : 7V-12V
 Operating voltage : 5V
 CPU Speed : 16MHZ
 Analog In/Out : 6/0
 Digital IO/PWM : 14/6
 EEPROM : 1KB
 SRAM : 2KB
 Flash : 32KB
 UART : 1
 USB : Regular

Sensor's Power - 200 mA Max
Analog In -
 Access with `analogRead(pin#)`
 returns mapped 0-5V => int 0-1023
Digital In/Out -
 Set pin with `pinmode(pin#, INPUT/OUTPUT)`
 OUTPUT - 40 mA Max
 INPUT - return HIGH if > 3V else LOW
PWM (Pulse Width Modulated) signal
 Access with `analogWrite(pin#, Value)`
 0 < Value < 255 maps to 0 - 100% Duty cycle
Serial Communication Access Points

ARDUINO PIN	MICROCONTROLLER PIN
0	- PD0(RXD)
1	- PD1(TXD)
2	- PD2(INT0)
3	- PD3(INT1)
4	- PD4
5	- PD5
6	- PD6
7	- PD7
8	- PB0
9	- PB1
10	- PB2(SS')
11	- PB3(MOSI)
12	- PB4(MISO)
13	- PB5(SCK)
A0	- PC0
A1	- PC1
A2	- PC2
A3	- PC3
A4	- PC4(SDA)
A5	- PC5(SCL)

Pin out from: <http://arduino.cc/en/arduino-boards-pin-mapping/11>

USB Overcurrent Protection

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

Physical Characteristics

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

4.2 LOAD CELL

A **load cell** is a transducer that is used to create an electrical signal whose magnitude is directly proportional to the force being measured. The various load cell types include hydraulic, pneumatic, and strain gauge.

Common types of load cells

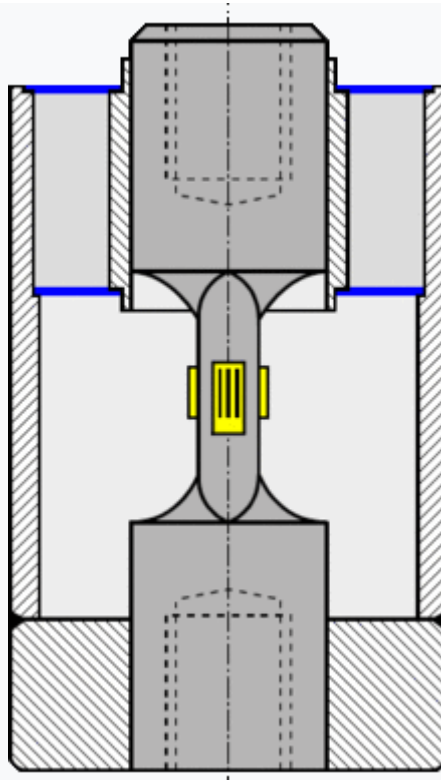
There are several types of load cells:

- Shear beam, a straight block of material fixed on one end and loaded on the other
- Bending beam, a compact, extremely versatile load cell that can easily fit into components where space is limited. These devices are commonly used in industrial and commercial measurement applications.
- Pancake, low-profile compression and wheel tension load cells that have a wide surface area for absorbing shock loads.
- Single point / platform load cells, allow the construction of a weighing scale or machine with only one load cell. Used in small / medium sized platform scales.
- Double-ended shear beam, a straight block of material fixed at both ends and loaded in the center
- Compression load cell, a block of material designed to be loaded at one point or area in compression
- S-type load cell, a S-shaped block of material that can be used in both compression and tension (load links and tension load cells are designed for tension only)
- Rope clamps, an assembly attached to a rope and measures its tension. Rope clamps are popular in hoist, crane and elevator applications due

to the ease of their installation; they have to be designed for a large range of loads, including dynamic peak loads, so their output for the rated load tends to be lower than of the other types

- Load pin, used for sensing loads on e.g., axles

Strain gauge load cell



Push-pull rod load cell spring element

Strain gauge load cells are the most common in industry. These load cells are particularly stiff, have very good resonance values, and tend to have long life cycles in application. Strain gauge load cells work on the principle that the strain gauge (a planar resistor) deforms when the material of the load cells deforms appropriately. Deformation of the strain gauge changes its electrical resistance, by an amount that is proportional to the strain. The change in resistance of the strain gauge provides an electrical value change that is calibrated to the load placed on the load cell.

A load cell usually consists of four strain gauges in a Wheatstone bridge configuration. Load cells of one strain gauge (quarter bridge) or two strain gauges (half bridge) are also available. The electrical signal output is typically in the order of a few millivolts (mV) and requires amplification by an instrumentation amplifier before it can be used. The output of the transducer can be scaled to calculate the force applied to the transducer. Sometimes a high resolution ADC, typically 24-bit, can be used directly.

The gauges themselves are bonded onto a beam or structural member that deforms when weight is applied. In most cases, four strain gauges are used to obtain maximum sensitivity and temperature compensation. Two of the gauges are usually in tension can be represented as T1 and T2, and two in compression can be represented as C1 and C2, and are wired with compensation adjustments. The strain gauge load cell is fundamentally a spring optimized for strain measurement. Gauges are mounted in areas that exhibit strain in compression or tension. When weight is applied to the load cell, gauges C1 and C2 compress decreasing their resistances. Simultaneously, gauges T1 and T2 are stretched increasing their resistances. The change in resistances causes more current to flow through C1 and C2 and less current to flow through T1 and T2. Thus, a potential difference is felt between the output or signal leads of the load cell. The gauges are mounted in a differential bridge to enhance measurement accuracy. When weight is applied, the strain changes the electrical resistance of the gauges in proportion to the load. Other load cells are fading into obscurity, as strain gauge load cells continue to increase their accuracy and lower their unit costs

Excitation and rated output

The bridge is excited with stabilized voltage (usually 10V, but can be 20V, 5V, or less for battery powered instrumentation). The difference voltage proportional to the load then appears on the signal outputs. The cell output is rated in millivolts per volt (mV/V) of the difference voltage at full rated mechanical load. So, a 2.96 mV/V load cell will provide 29.6 millivolt signal at full load when excited with 10 volts.

Typical sensitivity values are 1 to 3 mV/V. Typical maximum excitation voltage is around 15 volts.

Wiring

The full-bridge cells come typically in four-wire configuration. The wires to the top and bottom end of the bridge are the excitation (often labelled E+ and E-, or Ex+ and Ex-), the wires to its sides are the signal (labelled S+ and S-). Ideally, the voltage difference between S+ and S- is zero under zero load, and grows proportionally to the load cell's mechanical load.

Sometimes a six-wire configuration is used. The two additional wires are "sense" (Sen+ and Sen-), and are connected to the bridge with the Ex+ and Ex- wires, in a fashion similar to four-terminal sensing. With these additional signals, the controller can compensate for the change in wire resistance due to e.g., temperature fluctuations.

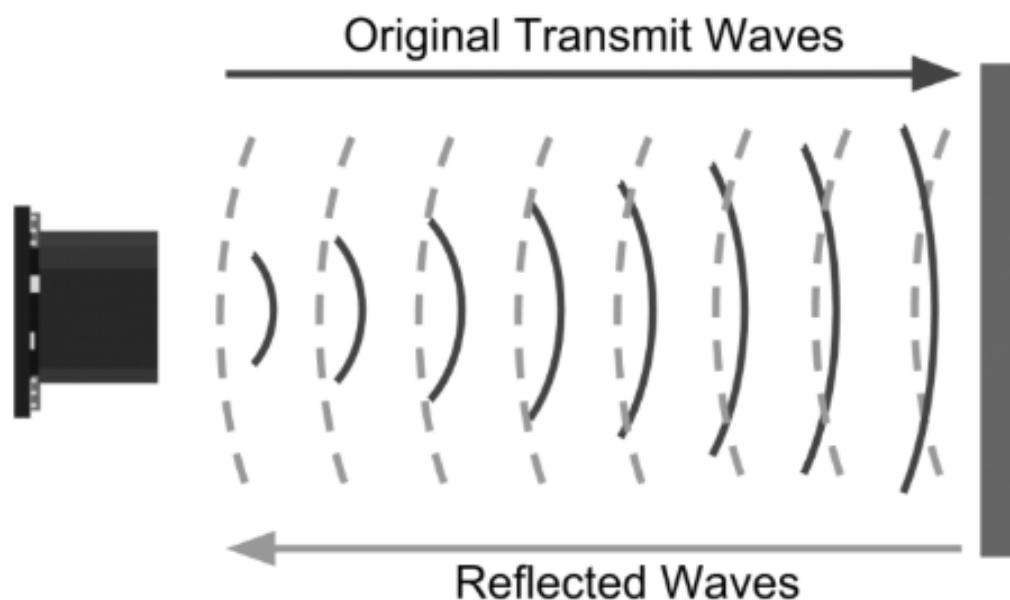
The individual resistors on the bridge usually have resistance of 350 Ω . Sometimes other values (typically 120 Ω , 1,000 Ω) can be encountered.

The bridge is typically electrically insulated from the substrate. The sensing elements are in close proximity and in good mutual thermal contact, to avoid differential signals caused by temperature differences.

4.3 ULTRASONIC SENSORS:

An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity. High-frequency sound waves reflect from boundaries to produce distinct echo patterns.

Ultrasonic sound vibrates at a frequency above the range of human hearing. Transducers are the microphones used to receive and send the ultrasonic sound. Our ultrasonic sensors, like many others, use a single transducer to send a pulse and to receive the echo. The sensor determines the distance to a tsonic pulse.



Ultrasound is reliable in any lighting environment and can be used inside or outside. Ultrasonic sensors can handle collision avoidance for a robot, and being moved often, as long as it isn't too fast.

ULTRASONIC SENSOR MODULE



Ultrasonics are so widely used, they can be reliably implemented in grain bin sensing applications, water level sensing, drone applications and sensing cars at your local drive-thru restaurant or bank.

Ultrasonic rangefinders are commonly used as devices to detect a collision.

Ultrasonic Sensors are best used in the non-contact detection of:

- Presence
- Level

Defining sound wave

Sound is a mechanical wave travelling through the mediums, which may be a solid, or liquid or gas. Sound waves can travel through the mediums with specific velocity depends on the medium of propagation. The sound waves which are having high frequency reflect from boundaries and produces distinctive echo patterns.

Laws of physics for sound waves

Sound waves are having specific frequencies or number of oscillations per second. Humans can detect sounds in a frequency range from about 20Hz to 20 KHz. However, the frequency range normally employed in ultrasonic detection is 100 KHz to 50MHz. The velocity of ultrasound at a particular time and temperature is constant in a medium.

$$W = C/F \text{ (or) } W = CT$$

Where W = Wave length

C = Velocity of sound in a medium

F = Frequency of wave

T=Time Period

The most common methods of ultrasonic examination utilize either longitudinal waves or shear waves. The longitudinal wave is a compression wave in which the particle motion is in the same direction of the propagation wave. The shear wave is a wave motion in which the particle motion is perpendicular to the direction of propagation. Ultrasonic detection introduces high frequency sound waves into a test

object to obtain information about the object without altering or damaging it in any way. Two values are measured in ultrasonic detection.

The amount of time, taking for the sound to travel through the medium and amplitude of the received signal. Based on velocity and time thickness can be calculated.

Thickness of material = Material sound velocity X Time of Flight

Transducers for Wave Propagation and particle detection

For sending sound waves and receiving echo, ultrasonic sensors, normally called transceivers or transducers will be used. They work on a principle similar to radar that will convert electrical energy into mechanical energy in the form of sound, and vice versa.

The commonly used transducers are contact transducers, angle beam transducers, delay line transducers, immersion transducers, and dual element transducers. Contact transducers are typically used for locating voids and cracks to the outside surface of a part as well as measuring thickness. Angle beam transducers use the principle of reflection and mode conversion to produce refracted shear or longitudinal waves in the test material.

Delay line transducers are single element longitudinal wave transducers used in conjunction with a replaceable delay line. One of the reasons for choosing delay line transducer is that near surface resolution can be improved. The delay allows the element to stop vibrating before a return signal from the reflector can be received.

The major advantages offered by immersion transducers over contact transducers are Uniform coupling reduces sensitivity variations, Reduction in scan time, and increases sensitivity to small reflectors.

Operation of ultrasonic sensors:

When an electrical pulse of high voltage is applied to the ultrasonic transducer it vibrates across a specific spectrum of frequencies and generates a burst of sound waves. Whenever any obstacle comes ahead of the ultrasonic sensor the sound waves will reflect back in the form of echo and generates an electric pulse. It calculates the time taken between sending sound waves and receiving echo. The echo patterns will be compared with the patterns of sound waves to determine detected signal's condition.

Applications involving Ultrasonic detection:

The distance of obstacle or discontinuities in metals is related to velocity of sound waves in a medium through which waves are passed and the time taken for echo reception. Hence the ultrasonic detection can be used for finding the distances between particles, for detecting the discontinuities in metals and for indicating the liquid level.

Ultrasonic Distance Measurement

Ultrasonic sensors are used for distance measuring applications. These gadgets regularly transmit a short burst of ultrasonic sound to a target, which reflects the sound back to the sensor. The system then measures the time for the echo to return to the sensor and computes the distance to the target using the speed of sound within the medium.

Different sorts of transducers are utilized within industrially accessible ultrasonic cleaning devices. An ultrasonic transducer is affixed to a stainless-steel pan which is filled with a solvent and a square wave is applied to it, conferring vibration energy on the liquid.



The ultrasonic distance sensors measure distance using sonar; an ultrasonic (well above human hearing) beat is transmitted from the unit and distance-to-target is determined by measuring the time required for the echo return. Output from the ultrasonic sensor is a variable-width beat that compares to the distance to the target.

8 Features of Ultrasonic Distance Sensor:

1. Supply voltage: 5V (DC).
2. Supply current: 15mA.
3. Modulation frequency: 40Hz.
4. Output: 0 – 5V (Output high when obstacle detected in range).
5. Beam Angle: Max 15 degree.
6. Distance: 2cm – 400cm.
7. Accuracy: 0.3cm.

8. Communication: Positive TTL pulse.

4.4 LED

A **light-emitting diode (LED)** is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons. This effect is called electroluminescence. The color of the light (corresponding to the energy of the photons) is determined by the energy required for electrons to cross the band gap of the semiconductor. White light is obtained by using multiple semiconductors or a layer of light-emitting phosphor on the semiconductor device.

Appearing as practical electronic components in 1962, the earliest LEDs emitted low-intensity infrared light. Infrared LEDs are used in remote-control circuits, such as those used with a wide variety of consumer electronics. The first visible-light LEDs were of low intensity and limited to red. Modern LEDs are available across the visible, ultraviolet, and infrared wavelengths, with high light output.

Early LEDs were often used as indicator lamps, replacing small incandescent bulbs, and in seven-segment displays. Recent developments have produced white-light LEDs suitable for room lighting. LEDs have led to new displays and sensors, while their high switching rates are useful in advanced communications technology.

LEDs have many advantages over incandescent light sources, including lower energy consumption, longer lifetime, improved physical robustness, smaller size, and faster switching. Light-emitting diodes are used in applications as diverse as aviation lighting, automotive

headlamps, advertising, general lighting, traffic signals, camera flashes, lighted wallpaper and medical devices.

Unlike a laser, the color of light emitted from an LED is neither coherent nor monochromatic, but the spectrum is narrow with respect to human vision, and functionally monochromatic

TYPES

LEDs are made in different packages for different applications. A single or a few LED junctions may be packed in one miniature device for use as an indicator or pilot lamp. An LED array may include controlling circuits within the same package, which may range from a simple resistor, blinking or color changing control, or an addressable controller for RGB devices. Higher-powered white-emitting devices will be mounted on heat sinks and will be used for illumination. Alphanumeric displays in dot matrix or bar formats are widely available. Special packages permit connection of LEDs to optical fibers for high-speed data communication links.

MINIATURE

These are mostly single-die LEDs used as indicators, and they come in various sizes from 2 mm to 8 mm, through-hole and mount packages. Typical current ratings range from around 1 mA to above 20 mA. Multiple LED dies attached to a flexible backing tape form an LED strip light.

Common package shapes include round, with a domed or flat top, rectangular with a flat top (as used in bar-graph displays), and triangular or square with a flat top. The encapsulation may also be clear or tinted

to improve contrast and viewing angle. Infrared devices may have a black tint to block visible light while passing infrared radiation.

Ultra-high-output LEDs are designed for viewing in direct sunlight

5 V and 12 V LEDs are ordinary miniature LEDs that have a series resistor for direct connection to a 5 V or 12 V supply.

HIGH POWER

High-power LEDs (HP-LEDs) or high-output LEDs (HO-LEDs) can be driven at currents from hundreds of mA to more than an ampere, compared with the tens of mA for other LEDs. Some can emit over a thousand lumens. LED power densities up to 300 W/cm^2 have been achieved. Since overheating is destructive, the HP-LEDs must be mounted on a heat sink to allow for heat dissipation. If the heat from an HP-LED is not removed, the device fails in seconds. One HP-LED can often replace an incandescent bulb in a flashlight, or be set in an array to form a powerful LED lamp

AC DRIVEN

LEDs developed by Seoul Semiconductor can operate on AC power without a DC converter. For each half-cycle, part of the LED emits light and part is dark, and this is reversed during the next half-cycle. The efficacy of this type of HP-LED is typically 40 lm/W . A large number of LED elements in series may be able to operate directly from line voltage. In 2009, Seoul Semiconductor released a high DC voltage LED, named as 'Acrich MJT', capable of being driven from AC power with a simple

controlling circuit. The low-power dissipation of these LEDs affords them more flexibility than the original AC LED design

Application-specific variations

Flashing

Flashing LEDs are used as attention seeking indicators without requiring external electronics. Flashing LEDs resemble standard LEDs but they contain an integrated multivibrator circuit that causes the LED to flash with a typical period of one second. In diffused lens LEDs, this circuit is visible as a small black dot. Most flashing LEDs emit light of one color, but more sophisticated devices can flash between multiple colors and even fade through a color sequence using RGB color mixing.

Bi-color

Bi-color LEDs contain two different LED emitters in one case. There are two types of these. One type consists of two dies connected to the same two leads antiparallel to each other. Current flow in one direction emits one color, and current in the opposite direction emits the other color. The other type consists of two dies with separate leads for both dies and another lead for common anode or cathode so that they can be controlled independently. The most common bi-color combination is red/traditional green; however, other available combinations include amber/traditional green, red/pure green, red/blue, and blue/pure green.

RGB Tri-color

Tri-color LEDs contain three different LED emitters in one case. Each emitter is connected to a separate lead so they can be controlled

independently. A four-lead arrangement is typical with one common lead (anode or cathode) and an additional lead for each color. Others, however, have only two leads (positive and negative) and have a built-in electronic controller.



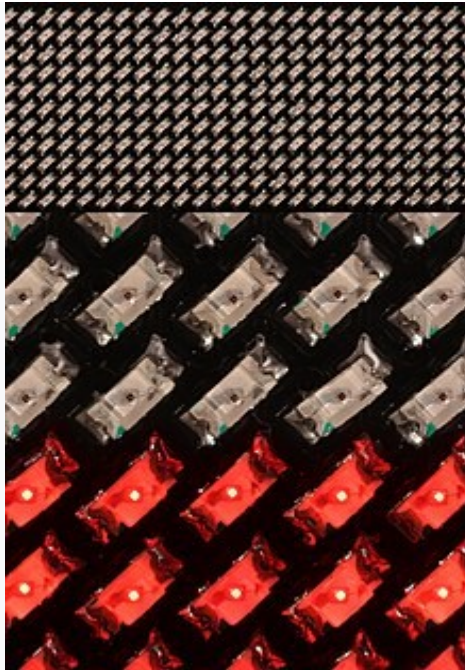
RGB-SMD-LED

RGB LEDs consist of one red, one green, and one blue LED.^[104] By independently adjusting each of the three, RGB LEDs are capable of producing a wide color gamut. Unlike dedicated-color LEDs, however, these do not produce pure wavelengths. Modules may not be optimized for smooth color mixing.

Decorative-multicolor

Decorative-multicolor LEDs incorporate several emitters of different colors supplied by only two lead-out wires. Colors are switched internally by varying the supply voltage.

Alphanumeric



Composite image of a 11x44 LED matrix lapel name tag display using 1608/0603-type SMD LEDs. Top: A little over half of the 21x86 mm display. Center: Close-up of LEDs in ambient light. Bottom: LEDs in their own red light.

Alphanumeric LEDs are available in seven-segment, starburst, and dot-matrix format. Seven-segment displays handle all numbers and a limited set of letters. Starburst displays can display all letters. Dot-matrix displays typically use 5x7 pixels per character. Seven-segment LED displays were in widespread use in the 1970s and 1980s, but rising use of liquid crystal displays, with their lower power needs and greater display flexibility, has reduced the popularity of numeric and alphanumeric LED displays.

Digital RGB

Digital RGB addressable LEDs contain their own "smart" control electronics. In addition to power and ground, these provide connections for data-in, data-out, and sometimes a clock or strobe signal. These are connected in a daisy chain. Data sent to the first LED of the chain can control the brightness and color of each LED independently of the others. They are used where a combination of maximum control and minimum visible electronics are needed such as strings for Christmas and LED matrices. Some even have refresh rates in the kHz range, allowing for basic video applications. These devices are known by their part number (WS2812 being common) or a brand name such as NeoPixel

Filament

An LED filament consists of multiple LED chips connected in series on a common longitudinal substrate that forms a thin rod reminiscent of a traditional incandescent filament. These are being used as a low-cost decorative alternative for traditional light bulbs that are being phased out in many countries. The filaments use a rather high voltage, allowing them to work efficiently with mains voltages. Often a simple rectifier and capacitive current limiting are employed to create a low-cost replacement for a traditional light bulb without the complexity of the low voltage, high current converter that single die LEDs need. Usually, they are packaged in bulb similar to the lamps they were designed to replace, and filled with inert gas to remove heat efficiently.

Chip-on-board arrays

Surface-mounted LEDs are frequently produced in chip on board (COB) arrays, allowing better heat dissipation than with a single LED of comparable luminous output. The LEDs can be arranged around a cylinder, and are called "corn cob lights" because of the rows of yellow LEDs.

4.5 BUZZER

A **buzzer** or **beeper** is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.



TYPES

Electromechanical

Early devices were based on an electromechanical system identical to an electric bell without the metal gong. Similarly, a relay may be

connected to interrupt its own actuating current, causing the contacts to buzz. Often these units were anchored to a wall or ceiling to use it as a sounding board. The word "buzzer" comes from the rasping noise that electromechanical buzzers made.

Mechanical

A joy buzzer is an example of a purely mechanical buzzer and they require drivers. Other examples of them are doorbells.



Piezoelectric

A piezoelectric element may be driven by an oscillating electronic circuit or other audio signal source, driven with a piezoelectric audio amplifier. Sounds commonly used to indicate that a button has been pressed are a click, a ring or a beep.

A piezoelectric buzzer/beeper also depends on acoustic cavity resonance or Helmholtz resonance to produce an audible beep

APPLICATIONS

While technological advancements have caused buzzers to be impractical and undesirable, there are still instances in which buzzers and similar circuits may be used. Present day applications include:

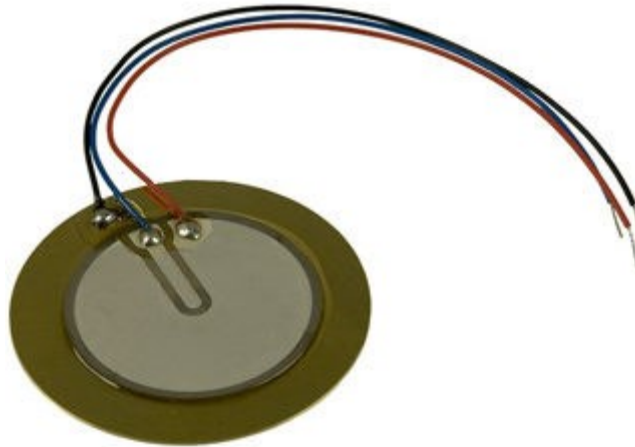
- Novelty uses
- Judging panels
- Educational purposes
- Annunciator panels
- Electronic metronomes
- Game show lock-out device
- Microwave ovens and other household appliances
- Sporting events such as basketball games
- Electrical alarms
- Joy buzzer (mechanical buzzer used for pranks)

Simple Piezo Buzzer circuit diagram

Piezo buzzers are one of the most common buzzers available around, they got their name from the piezoelectric material used as the active element.

These buzzers are usually driven at a relatively higher voltage but low current, consumes a little power, but still capable of producing very high sound.

The Piezo element must be a three terminal one, like in the picture.

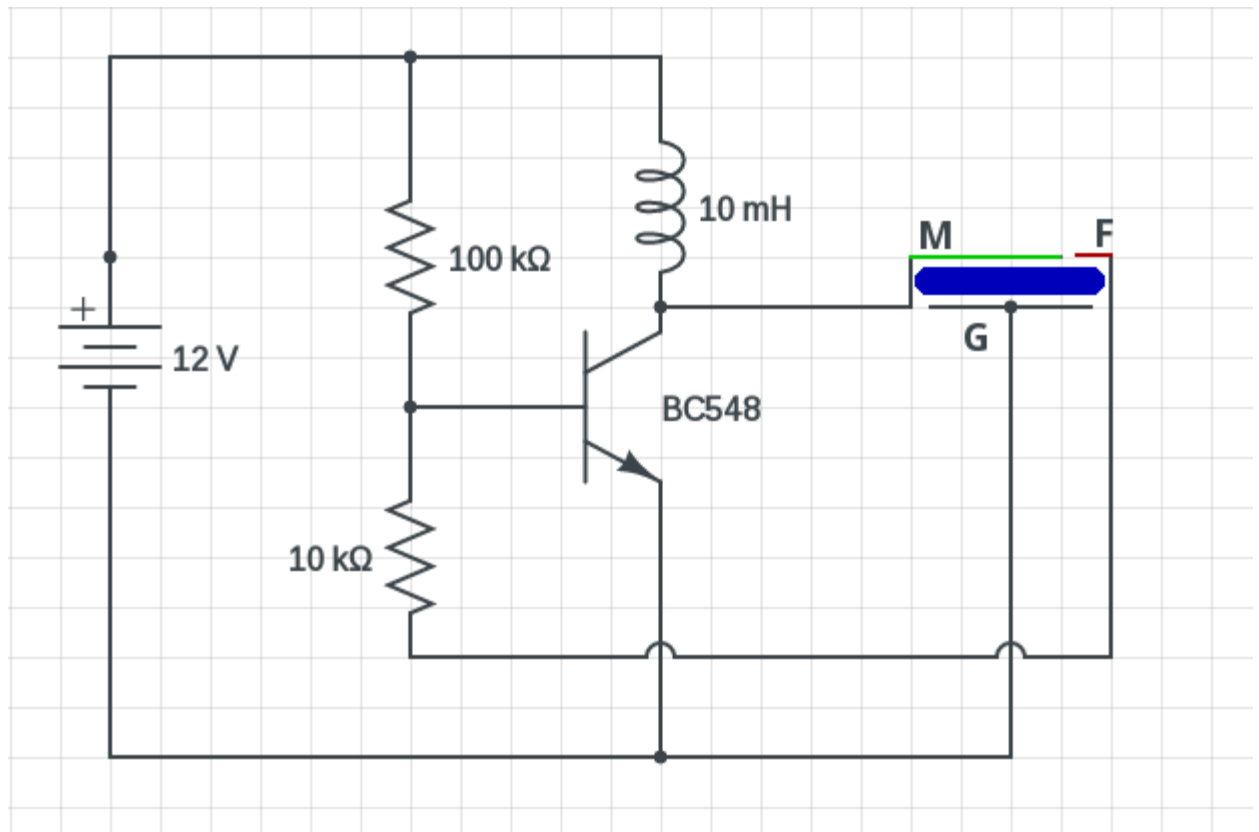


The blue wire is connected to feedback(**F**) terminal, red wire to the main (**M**) terminal and the black wire to the piezo element's ground(**G**) plate.

The inductor coil's value and shape is not crucial. You can use any coil from **1mH** to **10mH** or more, or even no measured value at all. I used a 40 turn coil on a small ferrite toroid in the final design.

Circuit diagram and construction

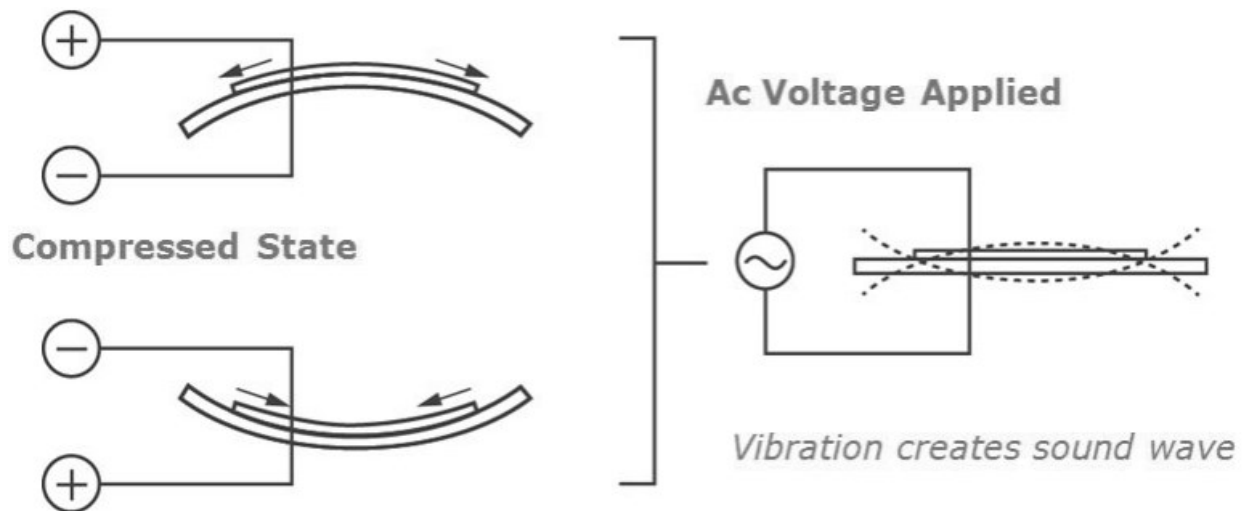
Lets have a look at the circuit diagram,



Note the piezoelectric element's pinout, **M** is the main terminal, **F** is the feedback terminal and **G** is the ground plate.

The circuit is fairly simple, you can use a little piece of strip board to make it. As this piezo buzzer circuit uses very few components, so it also could be constructed by soldering the components to each other.

When A voltage is applied to the electrodes of the piezo element, they produces flex in either way. This flex force bends the ground plate up and down.



The exact opposite thing happens too, when a piezoelectric element is subjected to varying pressure, it produces voltage.

As you've seen before, self drive piezo buzzers are constructed with an extra electrically isolated feedback electrode. The voltage created by the flex force is available in the feedback terminal.

The piezo buzzer is placed in a resonant cavity, there is a hole in the opposite side of the resonant cavity from where the buzzing sound comes out.

The driver circuit and piezo buzzer co-operates soon between themselves and they start oscillating on the resonant frequency of the piezo buzzer.

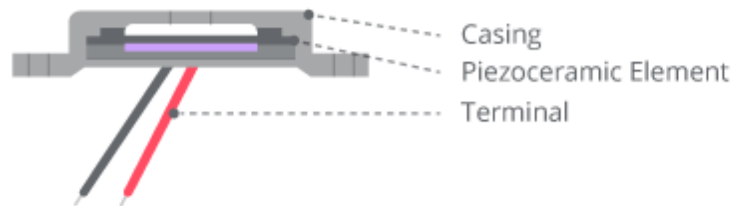
Piezo Buzzer Characteristics

- Wide operating voltage: 3~250 V
- Lower current consumption: less than 30 mA higher rated frequency

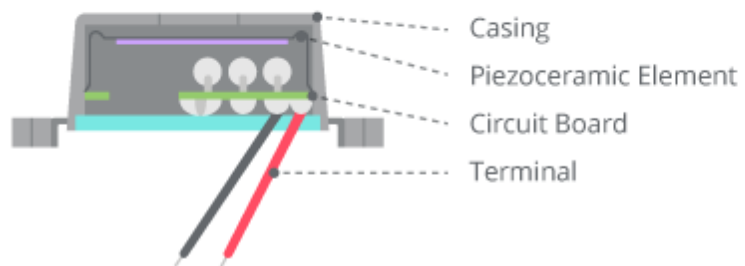
- Larger footprint
- Higher sound pressure level

STRUCTURE OF PIEZO BUZZER

Transducer *(without driving circuitry)*



Indicator *(with driving circuitry)*

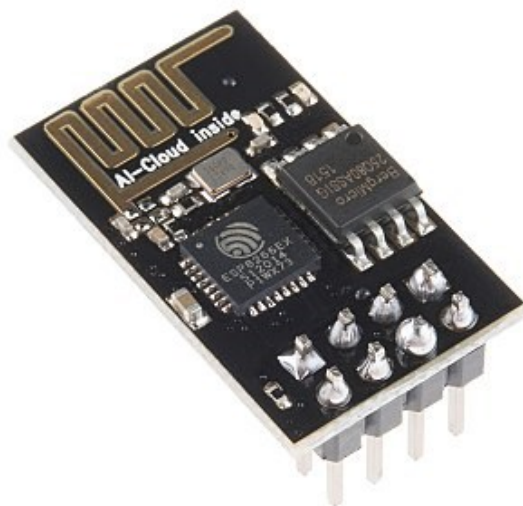


There are two types of piezo buzzers - transducers and indicators. Transducers consist of a casing, a piezoceramic element and a terminal. In order to operate a transducer, the user must send a square wave signal to the buzzer. Indicators consist of a casing, a piezoceramic element, a circuit board and a terminal. In order to operate an indicator, the user must send the buzzer a specified dc voltage.

4.6 ESP 8266 MODULE

The ESP8266 is a low-cost Wi-Fi microchip, with a full TCP/IP stack and microcontroller capability, produced by Espressif Systems^[1] in Shanghai, China. 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-Fi network and make simple TCP/IP connections using Hayes-style commands. However, at first 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, the chip, and the software on it, as well as to translate the Chinese documentation. The ESP8285 is an ESP8266 with 1 MiB of built-in flash, allowing the building of single-chip devices capable of connecting to Wi-Fi. These microcontroller chips have been succeeded by the ESP32 family of devices, including the pin-compatible ESP32-C3.



ESP8266

FEATURE:

- Processor: L106 32-bit RISC microprocessor core based on the Tensilica Xtensa Diamond Standard 106Micro running at 80 MHz^[5]
- Memory:
 - 32 KiB instruction RAM
 - 32 KiB instruction cache RAM
 - 80 KiB user-data RAM
 - 16 KiB ETS system-data RAM
- External QSPI flash: up to 16 MiB is supported (512 KiB to 4 MiB typically included)
- IEEE 802.11 b/g/n Wi-Fi
 - Integrated TR switch, balun, LNA, Power amplifier and matching network.
 - WEP or WPA/WPA2 authentication, or open networks
- 16 GPIO pins
- SPI
- I²C (software implementation)^[6]
- I²S interfaces with DMA (sharing pins with GPIO)
- UART on dedicated pins, plus a transmit-only UART can be enabled on GPIO2
- 10-bit ADC (successive approximation ADC)

Pinout Of ESP

The pinout is as follows for the common ESP-01 module:

- GND, Ground (0 V)

- GPIO 2, General-purpose input/output No. 2
- GPIO 0, General-purpose input/output No. 0
- RX, Receive data bit X, also GPIO3
- VCC, Voltage (+3.3 V; can handle up to 3.6 V)
- RST, Reset
- CH_PD, Chip power-down
- TX, Transmit data bit X, also GPIO1

Espressif modules

This is the series of ESP8266-based modules made by Espressif:

Name	Active pins	Pitch	Form factor	LEDs	Antenna	Shielded	Dimensions (mm)	Notes
ESP-WROOM-02	18	1.5 mm	2×9 castellated	No	PCB trace	Yes	18 × 20	FCC ID 2AC7Z-ESPWROOM02.
ESP-WROOM-02D	18	1.5 mm	2×9 castellated	No	PCB trace	Yes	18 × 20	FCC ID 2AC7Z-ESPWROOM02D. Revision of ESP-WROOM-02 compatible with both 150-mil and 208-mil flash memory chips.
ESP-WROOM-02U	18	1.5 mm	2×9 castellated	No	U.FL socket	Yes	18 × 20	Differs from ESP-WROOM-02D in that includes an U.FL compatible antenna socket connector.

ESP-WROOM-S2	20	1.5 mm	2×10 castellated	No	PCB trace	Yes	16 × 23	FCC ID 2AC7Z-ESPWROOMS2.
--------------	----	--------	------------------	----	-----------	-----	---------	--------------------------

In the table above (and the two tables which follow), "Active pins" include the GPIO and ADC pins with which external devices can be attached to the ESP8266 MCU. The "Pitch" is the space between pins on the ESP8266 module, which is important to know if the device will be used on a breadboard. The "Form factor" also describes the module packaging as "2 × 9 DIL", meaning two rows of 9 pins arranged "Dual In Line", like the pins of DIP ICs. Many ESP-xx modules include a small on-board LED which can be programmed to blink and thereby indicate activity. There are several antenna options for ESP-xx boards including a trace antenna, an on-board ceramic antenna, and an external connector which allows an external Wi-Fi antenna to be attached. Since Wi-Fi communications generates a lot of RFI (Radio Frequency Interference), governmental bodies like the FCC like shielded electronics to minimize interference with other devices. Some of the ESP-xx modules come housed within a metal box with an FCC seal of approval stamped on it. First and second world markets will likely demand FCC approval and shielded Wi-Fi devices.

CHAPTER-5

SOFTWARE DESCRIPTIONS

5.1 ARDUINO IDE:

The Arduino IDE is incredibly minimalistic, yet it provides a near-complete environment for most Arduino-based projects. The top menu bar has the standard options, including “File” (new, load save, etc.), “Edit” (font, copy, paste, etc.), “Sketch” (for compiling and programming), “Tools” (useful options for testing projects), and “Help”. The middle section of the IDE is a simple text editor that where you can enter the program code. The bottom section of the IDE is dedicated to an output window that is used to see the status of the compilation, how much memory has been used, any errors that were found in the program, and various other useful messages.



Projects made using the Arduino are called sketches, and such sketches are usually written in a cut-down version of C++ (a number of C++ features are

not included). Because programming a microcontroller is somewhat different from programming a computer, there are a number of device-specific libraries (e.g., changing pin modes, output data on pins, reading analog values, and timers). This sometimes confuses users who think Arduino is programmed in an “Arduino language.” However, the Arduino is, in fact, programmed in C++. It just uses unique libraries for the device.

The 6 Buttons

While more advanced projects will take advantage of the built-in tools in the IDE, most projects will rely on the six buttons found below the menu bar.



The button bar

1. The **check mark** is used to verify your code. Click this once you have written your code.
2. The **arrow** uploads your code to the Arduino to run.
3. The **dotted paper** will create a new file.
4. The **upward arrow** is used to open an existing Arduino project.
5. The **downward arrow** is used to save the current file.
6. The far-right button is a **serial monitor**, which is useful for sending data from the Arduino to the PC for debugging purposes.

The source code for the IDE is released under the GNU General Public License, version 2. 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

Projects made using the Arduino are called sketches, and such sketches are usually written in a cut-down version of C++ (a number of C++ features are not included). Because programming a microcontroller is somewhat different from programming a computer, there are a number of device-specific libraries (e.g., changing pin modes, output data on pins, reading analog values, and timers). This sometimes confuses users who think Arduino is programmed in an “Arduino language.” However, the Arduino is, in fact, programmed in C++. It just uses unique libraries for the device.

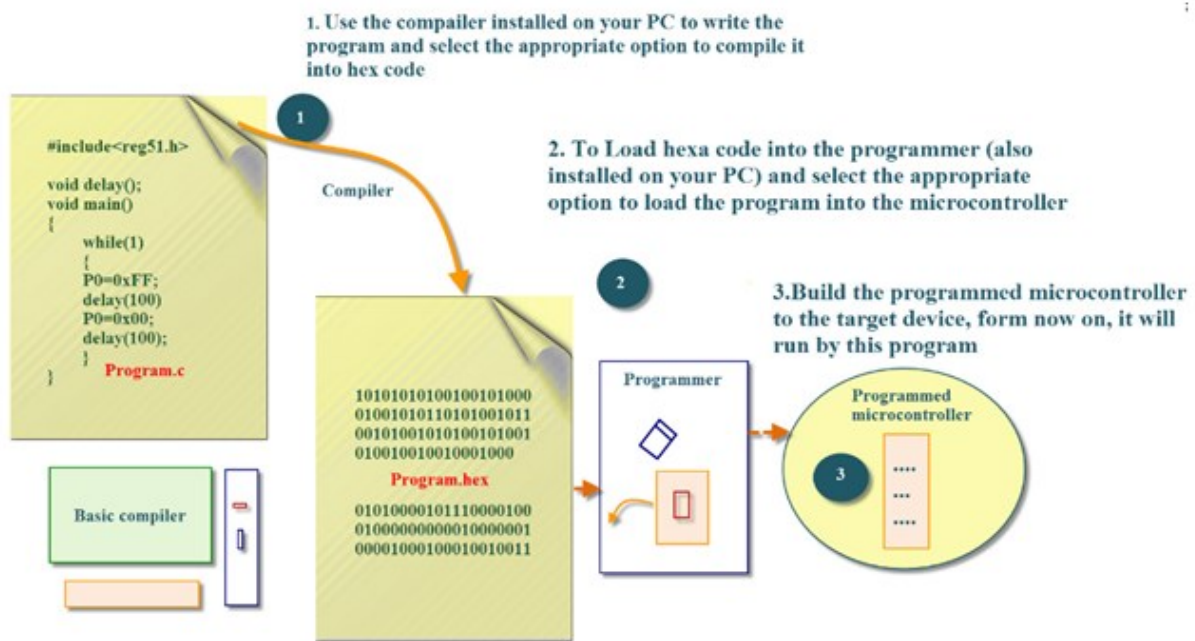
The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main 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.

5.2 EMBEDDED C:

Embedded C is most popular programming language in software field for developing electronic gadgets. Each processor used in electronic system is associated with embedded software.

Embedded C programming plays a key role in performing specific function by the processor. In day-to-day life we used many electronic devices such as mobile phone, washing machine, digital camera, etc. These all-device working is based on microcontroller that are programmed by embedded C.



The Embedded C code written in above block diagram is used for blinking the LED connected with Port0 of microcontroller.

In embedded system programming C code is preferred over other language. Due to the following reasons:

- Easy to understand
- High Reliability
- Portability
- Scalability

Function is a collection of statements that is used for performing a specific task and a collection of one or more functions is called a programming language.

Most consumers are familiar with application software that provide functionality on a computer. Embedded software however is often less visible, but no less complicated. Unlike application software, embedded software has fixed hardware requirements and capabilities, and addition of third-party hardware or software is strictly controlled.

Embedded software needs to include all needed device drivers at manufacturing time, and the device drivers are written for the specific hardware. The software is highly dependent on the CPU and specific chips chosen. Most embedded software engineers have at least a passing knowledge of reading schematics, and reading data sheets for components to determine usage of registers and communication system. Conversion between decimal, hexadecimal and binary is useful as well as using bit manipulation.

Web applications are rarely used, although XML files and other output may be passed to a computer for display. File systems with folders are typically absent as are SQL databases.

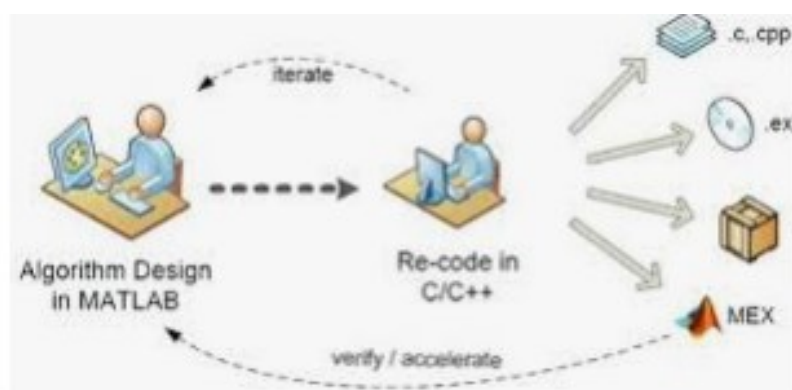
Software development requires use of a cross compiler, which runs on a computer but produces executable code for the target device. Debugging requires use of an in-circuit emulator, JTAG or SWD. Software developers often have access to the complete kernel (OS) source code.

Size of the storage memory and RAM can vary significantly. Some systems run in 16 KB of Flash and 4 KB of RAM with a CPU operating at 8 MHz, other systems can rival contemporary computers.[8] These space requirements lead to more work being done in C or embedded C++, instead of C++. Interpreted languages like BASIC (while e.g. Parallax Propeller can use compiled BASIC) and Java (Java ME Embedded 8.3[9] is available for e.g. ARM Cortex-M4, Cortex-M7 microcontrollers and older ARM11 used in Raspberry Pi and Intel Galileo Gen. 2) are not commonly used; while an implementation of the interpreted Python 3 language – MicroPython – is however available expressly for microcontroller use, e.g. 32-bit ARM-based (such as BBC micro:bit) and 16-bit PIC microcontrollers.

Communications between processors and between one processor and other components are essential. Besides direct memory addressing, common protocols include I²C, SPI, serial ports, and USB.

Communications protocols designed for use in embedded systems are available as closed source from companies including InterNiche Technologies and CMX Systems. Open-source protocols stem from uIP, lwip, and others

- This program explains how to use structure within structure in C using normal variable. “student_college_detail” structure is declared inside “student_detail” structure in this program. Both structure variables Please note that members of “student_college_detail” structure are accessed by 2 dot(.) operator and members of “student are normal structure variables.
- _detail” structure are accessed by single dot(.) operator.



Embedded C Program with Examples ...

A Keyword is a special word with a special meaning to the compiler (a C Compiler for example, is a software that is used to convert program written in C to Machine Code). For example, if we take the Keil’s Cx51 Compiler (a popular

C Compiler for 8051 based Microcontrollers) the following are some of the keywords:

- bit
- sbit
- sfr
- small
- large

These are few of the many keywords associated with the Cx51 C Compiler along with the standard C Keywords.

CHAPTER-6

CONCLUSION

We addressed some of the challenges surrounding parking in this article and demonstrated an IoT-based smart outdoor parking solution. Upto this we completed the test of load cell and ultrasonic sensor interfacing with arduino and it also provides the expected output.

REFERENCE

- ❖ F. Losilla, A.J Garcia-Sanchez, F. Garcia-Sanchez, J. GarciaHaro, “On the Role of Wireless Sensor Networks in intelligent transportation Systems, ICTON, 2-5 July, 2012, pp. 2161-2056 .
- ❖ J. Chinrungrueng, S.Dumnin, R. Pongthornseri, “ iParking: A Parking Management Framework”, 11th International Conference on ITS Telecommunications, 23-25 August, 2011, pp. 63-68 .
- ❖ Y. Hirakata, A. Nakamura, K. Ohno, M. Itami, “Navigations System using Zigbee Wireless Sensor Network for Parking”, 12th International Conference on ITS Telecommunications, 5-8 Nov, 2012, pp. 605-609.
- ❖ W. Lumpkins, “The internet of things meets cloud computing,” IEEE Consum. Electron. Mag., vol. 2, no. 2, pp. 47–51, Apr. 2013.
- ❖ Aijaz, A., Aghvami, A.H. “Cognitive Machine-to-Machine Communications for Internet-of-Things: A Protocol Stack Perspective” Internet of Things Journal, IEEE, JAN 2015.
- ❖ Chinrungrueng, S. Dumnin, and R. Pongthornseri. iParking: a Parking Management Framework, Proceedings of the 11th International IEEE Conference on ITS Telecommunications (ITST), St. Petersburg, FL, (2011) 63-68.
- ❖ S. Mathur, S. Kaul, M. Gruteser, and W. Trappe. ParkNet: a Mobile Sensor Network for Harvesting Real Time Vehicular Parking Information,

Proceedings of the MobiHoc S3 Workshop on MobiHoc S3 (MobiHoc S3 '09), New York, (2009) 25-28.

- ❖ J. Wolff, T. Heuer, H. Gao, M. Weinmann, S. Voit, and U. Hartmann. Parking Monitor System Based on Magnetic Field Sensor, Proceedings of the 2006 IEEE Intelligent Transportation Systems Conference (ITSC '06), Toronto, Canada, (2006) 1275-1279.
- ❖ S. Nath, A. Deshpande, Y. Ke, P. Gibbons, B. Karp, and S. Seshan. IrisNet: An Architecture for Internet-Scale Sensing Services, Proceedings of the 29th International Conference on Very Large Data Bases (VLDB '2003), Berlin, Germany, Vol. 29, (2003) 1137-1140.
- ❖ J. Propst, K. Poole, and J. Hallstrom. An Embedded Sensing Approach to Monitoring Parking Lot Occupancy, Proceedings of the 50th Annual Southeast Regional Conference (ACM-SE '12), Tuscaloosa, AL, (2012) 309-314.
- ❖ U. Hartmann. Parking Monitor System Based on Magnetic Field Sensor, Proceedings of the 2006 IEEE Intelligent Transportation Systems Conference (ITSC '06), Toronto, Canada, (2006) 1275-1279.
- ❖ S. Nath, A. Deshpande, Y. Ke, P. Gibbons, B. Karp, and S. Seshan. IrisNet: An Architecture for Internet-Scale Sensing Services, Proceedings of the 29th International Conference on Very Large Data Bases (VLDB '2003), Berlin, Germany, Vol. 29, (2003) 1137-1140.
- ❖ J. Propst, K. Poole, and J. Hallstrom. An Embedded Sensing Approach to Monitoring Parking Lot Occupancy, Proceedings of the 50th Annual

Southeast Regional Conference (ACM-SE '12), Tuscaloosa, AL, (2012) 309-314.

- ❖ S. Banerjee, P. Choudekar, M. Muju. Real Time Car Parking System Using Image Processing, Proceedings of the 3rd International IEEE Conference on Electronics Computer Technology (ICECT), Kanyakumari, India, Vol. 2, (2011) 99- 103.
- ❖ D.J. Bonde, R. Shende, K. Gaikwad, A. Kedari and A. Bhokre, "Automated car parking system commanded by android application", International Conference on Computer Communication and Informatics, (2014) 4.
- ❖ E. Soni, K. Kaur and A. Kumar, "Design And Development Of RFID Based Automated Car Parking System", The International Journal of Mathematics, Science, Technology and Management, Vol. 2. (2016).
- ❖ L. Chen, J. Hsieh, W. Lai, C. Wu, and S. Chen. VisionBased Vehicle Surveillance and Parking Lot Management Using Multiple Cameras, Proceedings of the 6th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Darmstadt, Germany, (2010) 631-634.