

SECURED AND INTELLIGENT PARKING SYSTEM

Submitted in partial fulfillment of the requirements for the award of

**Bachelor of Engineering degree in
Electronics and Communication Engineering**

By

**ATMAKURI JYOTHI SWAROOP – 38130021
ATLA GANESH REDDY -38130019**



**SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING
DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

SATHYABAMA

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

Accredited with Grade “A” by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI – 600119**



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **ATMAKURI JYOTHI SWAROOP(38130021) & ATLA GANESH REDDY(38130019)** who carried out the Project entitled "SECURED AND INTELLIGENT PARKING SYSTEM". Under the guidance of **E.ANNA DEVI,M.E.,Ph.D.**, during the period DEC 2021 to MAY 2022

INTERNAL GUIDE

E.ANNA DEVI,M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

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

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, ATMAKURI JYOTHI SWAROOP(38130021) & ATLA GANESH REDDY(38130019) Here by declare that the Project report entitled “SECURED AND INTELLIGENT PARKING SYSTEM ” is done under the guidance of E.ANNA DEVI,M.E.,Ph.D., is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in ELECTRONICS AND COMMUNICATION ENGINEERING.

DATE:

PLACE:

SIGNATURE OF CANDIDATE

**ATMAKURI JYOTHI SWAROOP 38130021
ATLA GANESH REDDY -38130019**

ACKNOWLEDGEMENT

I pleased to acknowledge my sincere thanks to board management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY for their kind encouragement in doing this internship and for completing it successfully. I am grateful to them.

I convey my thanks to Dr.N.M.NANDHITHA,M.E.,Ph.D., Dean, school of Electrical and Electronics engineering and Dr.T.RAVI,M.E.,Ph.D., Head of the Department, Electronics and Communication engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide E.ANNA DEVI,M.E.,Ph.D., for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my Project.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Electronics and Communication Engineering who were helpful in many ways for the completion of the project

ABSTRACT:

Internet of Things (IOT) plays a vital role in connecting the surrounding environmental things to the network and made easy to access those un-internet things from any remote location.

And generally people are facing problems on parking vehicles in parking slots in a city. In this study we design a Smart Parking System (SPS) which enables the user to find the nearest parking area and gives availability of parking slots in that respective parking area.

And it mainly focus on reducing the time in finding the parking lots and also it avoids the unnecessary travelling through filled parking lots in a parking area. Thus it reduces the fuel consumption which in turn reduces carbon footprints in an atmosphere.

TABLE OF CONTENT

TITLE	PAGE.NO
ABSTRACT	5
INTRODUCTION	7
LITERATURE SURVEY	7-8
EXISTING SYSTEM	8
BLOCK DIAGRAM	10
SOFTWARE USED	10
EMBEDDED SYSTEMS	
OVERVIEW OF EMBEDDED SYSTEMS	11-12
BLOCK DIAGRAM OF EMBEDDED SYSTEMS	12-13
CHARACTERISTICS OF EMBEDDED SYSTEMS	13-16
POWER SUPPLY	16-17
TRANSFORMER	17
INTRODUCTION TO IOT	20-21
FEATURES OF IOT	22-23
EMBEDDED SYSTEM IN IOT	23-25
IOT ECOSYSTEM	25-28
IOT ARCHITECTURE	29-35
IOT PLATFROM	39-41
NODE MCU	42-46
IR(INFRARED) SENSOR	46-49
LDR(LIGHT DEPENDENT RESISTOR)	49-52
RFID(RADIO FREQUENCY IDENTIFICATION)	52-64
ARDUINO IDE	64-78
OUR PROJECT OUTPUTS	79- 82
CONCLUSION	83
REFERANCES	83

INTRODUCTION:

Road congestion caused by vehicles is alarming problem on a global scale and is growing exponentially. The problem of parking is an important factor and it was still a big problem with confined car parks urban cities. Searching for a parking space is a routine often frustrating) activities for many people in the cities around the world. This search burns about a million barrels the oil of the world every day.

These problems will get worse with the world's population continues to urbanize without removed from the car well planned, based on convenience. According to a report, Smart Parking could involve 2, 20,000 gallons of fuel until 2030 and approx. 3, 00,000 gallons fuels by 2050 if implemented properly as stated in Figure. Smart Parking systems get information about availability parking in a particular geographical area. This process is in real time to position the vehicles in the available positions.

Involved Real-time data collection with low-cost sensors automatic payment systems activated on the mobile phone allows people to book early parking. The importance of smart parking is: Detects and accurately preaches the occupation of premises / vehicles by real time. Guide residents and visitors to available parking. Optimize the use of the parking space. Simplify the parking experience and add value parking parties, such as dealers and drivers. Help the free flow of traffic in the city by exploiting the IOT technology. Enables smart decisions using data, including real data temporary status applications and historical analysis reports.

Smart parking plays an important role in the creation of urban environment by reducing CO2 emissions and other pollutants. Smart Parking allows better tracking in real time and manage the resulting free parking space significant revenue generation. Provides tools to optimize the management of the workforce.

LITERATURE SURVEY:

Smart Car Parking Management System Using Iot Aniket Gupta*, Sujata Kulkarni* , Vaibhavi Jathar, Ved Sharma, Naman Jain

Proliferation in the number of vehicles is leading to problems of vehicles parking at an appropriate place especially the car parking. This indirectly leads to traffic congestion. This is because of the fact

that current transportation infrastructure and car park facility are unable to cope with the arrival of large number of vehicles on the road. To alleviate the aforementioned problem, authors proposed a Smart Parking Management System that helps users to automatically find a free parking space with a smaller amount. Smart Parking involves use of Ultrasonic sensor, Arduino Uno, ESP8266-01 Wi-Fi Module, Cloud server. IOT based new parking platform enable to connect, analyze and automate data gathered from devices and execute smart parking possible. Smart parking would enable vehicle occupancy, monitoring and managing of available parking space in real-time that reducing the environmental pollution. Proposed system provides optimize usage of parking space and get considerable revenue generation.

Smart Car Parking System Based on lot Concept Vrushali D. Ichake.^{1, 2, 3}, Priya D. Shitole. ^{1, 2, 3} and Mohsina Momin.^{1, 2, 3} Kanchan S. Thakare, Assistant Professor ^{2, 3}

A major problem in day to day life is parking of vehicles especially the car parking at an appropriate place. And this issue indirectly leads to traffic congestion. This paper presents the basic concept of using server or cloud based smart parking services in smart cities as an important application of the Internet of Things (Iot) paradigm. This system will be accessible through a mobile app or through the webpage provided and can be used to monitor or find the empty slots in that area.

Survey paper on Smart Parking System based on Internet of Things Vishwanath Y1 , Aishwarya D Kuchalli² , Debarupa Rakshit³

This paper proposes a Smart Parking system which provides an optimal solution for parking problem in metropolitan cities. Due to rapid increase in vehicle density especially during the peak hours of the day, it is a difficult task for the drivers to find a parking space to park their vehicles. The aim of the paper is to resolve the above mentioned issue which provides the Smart Parking system. This system uses cloud computing and Internet of Things (IOT) technology.

A suitable shortest path algorithm is used to find the minimum distance between the user and each car park in the system. Thus, the waiting time of the user is minimized .The paper also introduces the usage of android application using smart phone for the interaction between the Smart Parking system and the user. RFID technology is used in this system to avoid the human intervention which minimizes

the cost

EXISTING SYSTEM:

Public lighting is has the greatest impact on energy consumption of a municipality, and may account for up to 54% of total energy consumption and 61% of electricity consumption of municipal facilities.

Multiple sections of lights in the same electric cabinet, coupled with the significant loss of tension on power lines, causes serious interference to PLC based systems.

The technology used here is face recognition and face detection in real time. As the photos are taken in real time, several problems like unequal illumination and changes in the background may affect the system.

To overcome this problem DCT normalization and background cancellation algorithms are incorporated along with basic face detection and face recognition algorithm. In this paper software and hardware details of the system are discussed. Technologically system is simple, accurate and maintainable.

DRAWBACKS:

The vehicles may theft by others

Man power is needed.

It is not suitable for large no of vehicle car parking

Maintenance cost is high

Not automated

PROPOSED SYSTEM:

The Proposed system consist of the car parking system using the Nodemcu.

The car parking system having the IR sensor to detect the vehicles count and send it to the mail in the entrance of the car parking lot.

Here we can also added rotary type. It is used to choose the slots automatically, like which slots are available.

Whenever the vehicle is detected, the empty space will be displayed in the mail and The Car parking availability will be shown using LED's intimation.

When the vehicle is moving, we can able to switch on the light automatically then after leaving the vehicle we can switch of the

vehicle. At the day time, LDR sensor detect automatically, it will switch of the light, then at the night time, we can able to switch on the light. When light are having any trouble, we can sent the message to the management of the respective places and also we can able to monitor the distance between backside of the vehicle and parking place.

ADVANTAGES:

NO man power

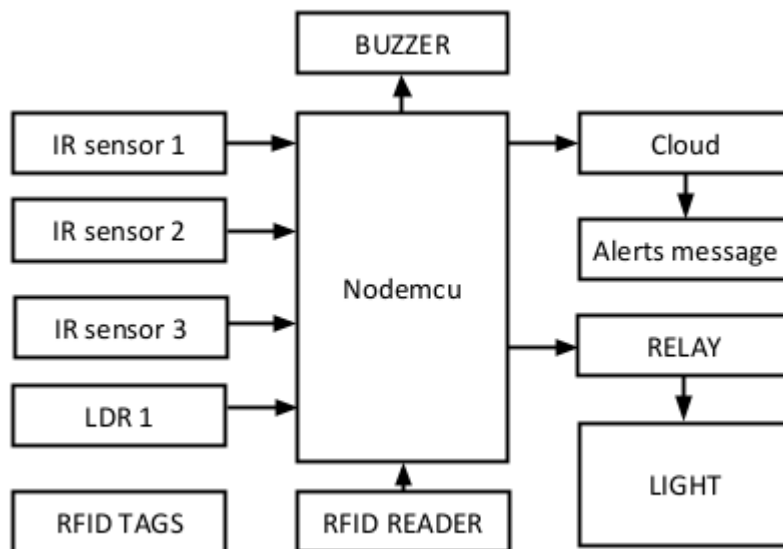
Automated

More security than existing system

Less maintenance

More comfortable.

BLOCK DIAGRAM:



HARDWARE USED:

Nodemcu

IR sensor

LDR

Servo motor

SOFTWARE USED:

Embedded C

Arduino ide

3. EMBEDDED SYSTEMS

3.1 Overview of embedded systems

An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions, often with real-time computing constraints. It is usually embedded as part of a complete device including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, can do many different tasks depending on programming. Embedded systems have become very important today as they control many of the common devices we use.

Since the embedded system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product, or increasing the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically, embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

In general, "embedded system" is not an exactly defined term, as many systems have some element of programmability. For example, Handheld computers share some elements with embedded systems — such as the operating systems and microprocessors which power them — but are not truly embedded systems, because they allow different applications to be loaded and peripherals to be connected.

Embedded systems provide several functions

- Monitor the environment; embedded systems read data from input sensors. This data is then processed and the results displayed in some format to a user or users
- Control the environment; embedded systems generate and transmit commands for actuators.

- Transform the information; embedded systems transform the data collected in some meaningful way, such as data compression/decompression

Although interaction with the external world via sensors and actuators is an important aspect of embedded systems, these systems also provide functionality specific to their applications. Embedded systems typically execute applications such as control laws, finite state machines, and signal processing algorithms. These systems must also detect and react to faults in both the internal computing environment as well as the surrounding electromechanical systems.

There are many categories of embedded systems, from communication devices to home appliances to control systems. Examples include;

- Communication devices
 - e. g.: modems, cellular phones
- Home Appliances
 - e. g.: CD player, VCR, microwave oven
- Control Systems
 - e. g.: Automobile anti-lock braking systems, robotics, satellite control

3.2 Block diagram of an embedded system:

An embedded system usually contains an embedded processor. Many appliances that have a digital interface -- microwaves, VCRs, cars -- utilize embedded systems. Some embedded systems include an operating system. Others are very specialized resulting in the entire logic being implemented as a single program. These systems are embedded into some device for some specific purpose other than to provide general purpose computing . A typical embedded system is shown in Fig 1.1

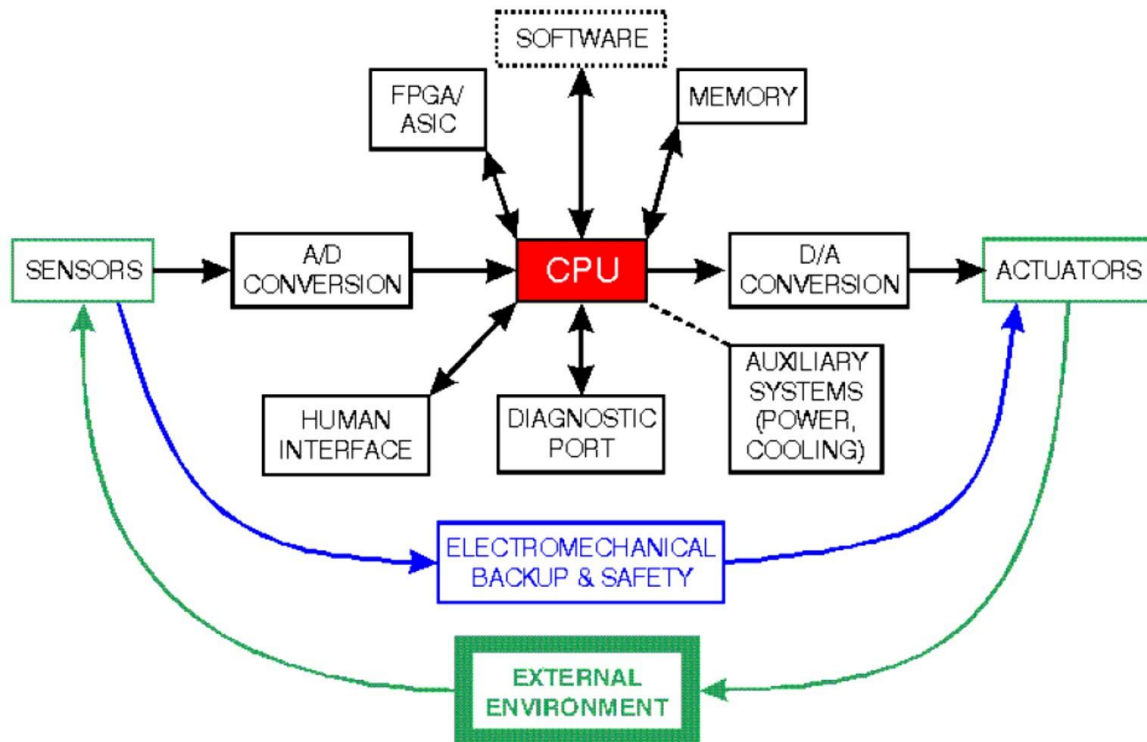


Fig 1.1 Block diagram of a typical embedded system

3.2.1 Characteristics of embedded systems

Embedded systems are characterized by a unique set of characteristics. Each of these characteristics imposed a specific set of design constraints on embedded systems designers. The challenge to designing embedded systems is to conform to the specific set of constraints for the application.

Application Specific Systems:

Embedded systems are not general-purpose computers. Embedded system designs are optimized for a specific application. Many of the job characteristics are known before the hardware is designed. This allows the designer to focus on the specific design constraints of a well-defined application. As such, there is limited user reprogram ability. Some embedded systems, however, require the flexibility of reprogram ability. Programmable DSPs are common for such applications.

Reactive Systems

As mentioned earlier, a typical embedded systems model responds to the environment via sensors and control the environment using

actuators. This requires embedded systems to run at the speed of the environment. This characteristic of embedded system is called “reactive”. Reactive computation means that the system (primarily the software component) executes in response to external events. External events can be either periodic or aperiodic. Periodic events make it easier to schedule processing to guarantee performance. Aperiodic events are harder to schedule. The maximum event arrival rate must be estimated in order to accommodate worst case situations. Most embedded systems have a significant reactive component. One of the biggest challenges for embedded system designers is performing an accurate worst case design analysis on systems with statistical performance characteristics (e.g., cache memory on a DSP or other embedded processor). Real time system operation means that the correctness of a computation depends, in part, on the time at which it is delivered. Systems with this requirement must often design to worst case performance. But accurately predicting the worst case may be difficult on complicated architectures. This often leads to overly pessimistic estimates erring on the side of caution. Many embedded systems have a significant requirement for real time operation in order to meet external I/O and control stability requirements. Many real-time systems are also reactive systems.

Distributed Systems

A common characteristic of an embedded system is one that consists of communicating processes executing on several CPUs or ASICs which are connected by communication links. The reason for this is economy. Economical 4 8-bit microcontrollers may be cheaper than a 32-bit processors. Even after adding the cost of the communication links, this approach may be preferable. In this approach, multiple processors are usually required to handle multiple time-critical tasks. Devices under control of embedded systems may also be physically distributed.

Heterogeneous Architectures

Embedded systems often are composed of heterogeneous architectures (Fig 1.2). They may contain different processors in the same system solution. They may also be mixed signal systems. The combination of I/O interfaces, local and remote memories, and sensors and actuators makes embedded system design truly unique. Embedded

systems also have tight design constraints, and heterogeneity provides better design flexibility.



Fig 1.2 Embedded Systems having Heterogeneous Architectures

Harsh environment

Many embedded systems do not operate in a controlled environment. Excessive heat is often a problem, especially in applications involving combustion (e.g., many transportation applications). Additional problems can be caused for embedded computing by a need for protection from vibration, shock, lightning, power supply fluctuations, water, corrosion, fire, and general physical abuse.

System safety and reliability

As embedded system complexity and computing power continue to grow, they are starting to control more and more of the safety aspects of the overall system. These safety measures may be in the form of software as well as hardware control. Mechanical safety backups are normally activated when the computer system loses control in order to safely shut down system operation. Software safety and reliability is a bigger issue. Software doesn't normally "break" in the sense of hardware. However software may be so complex that a set of unexpected circumstances can cause software failures leading to unsafe situations. Discussion of this topic is outside the scope of this book, but the challenges for embedded designers include designing reliable software and building cheap, available systems using unreliable components. The main challenge for embedded system designers is to obtain low-cost reliability with minimal redundancy.

Control of physical systems

One of the main reasons for embedding a computer is to interact with the environment. This is often done by monitoring and controlling external machinery. Embedded computers transform the analog signals from sensors into digital form for processing. Outputs must be transformed back to analog signal levels. When controlling physical equipment, large current loads may need to be switched in order to operate motors and other actuators. To meet these needs, embedded systems may need large computer circuit boards with many non-digital components. Embedded system designers must carefully balance system tradeoffs among analog components, power, mechanical, network, and digital hardware with corresponding software.

Small and low weight

Many embedded computers are physically located within some larger system. The form factor for the embedded system may be dictated by aesthetics. For example, the form factor for a missile may have to fit inside the nose of the missile. One of the challenges for embedded systems designers is to develop non-rectangular geometries for certain solutions. Weight can also be a critical constraint. Embedded automobile control systems, for example, must be light weight for fuel economy. Portable CD players must be light weight for portability purposes.

Cost sensitivity

Cost is an issue in most systems, but the sensitivity to cost changes can vary dramatically in embedded systems. This is mainly due to the effect of computer costs have on profitability and is more a function of the proportion of cost changes compared to the total system cost.

Power management

Embedded systems have strict constraints on power. Given the portability requirements of many embedded systems, the need to conserve power is important to maintain battery life as long as possible. Minimization of heat production is another obvious concern for embedded systems.

3.3 POWER SUPPLY

All electronic circuits works only in low DC voltage, so we need a

power supply unit to provide the appropriate voltage supply for their proper functioning .This unit consists of transformer, rectifier, filter & regulator. AC voltage of typically 230volts rms is connected to a transformer voltage down to the level to the desired ac voltage. A diode rectifier that provides the full wave rectified voltage that is initially filtered by a simple capacitor filter to produce a dc voltage. This resulting dc voltage usually has some ripple or ac voltage variation . A regulator circuit can use this dc input to provide dc voltage that not only has much less ripple voltage but also remains the same dc value even the dc voltage varies some what, or the load connected to the output dc voltages changes.

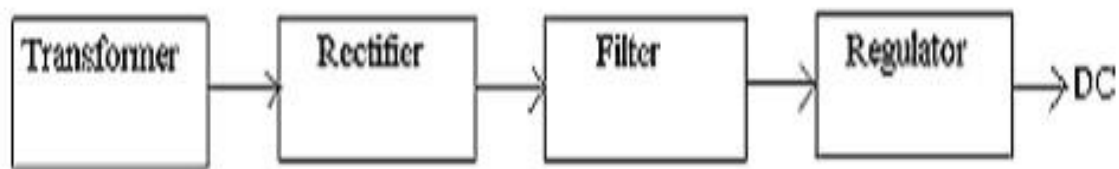


Fig 1.General Block Of Power Supply Unit

3.4 TRANSFORMER:

A transformer is a static piece of which electric power in one circuit is transformed into electric power of same frequency in another circuit. It can raise or lower the voltage in the circuit, but with a corresponding decrease or increase in current. It works with the principle of mutual induction. In our project we are using a step down transformer to providing a necessary supply for the electronic circuits. Here we step down a 230volts ac into 12volts ac.

3.5 RECTIFIER:

A dc level obtained from a sinusoidal input can be improved 100% using a process called full wave rectification. Here in our project for full wave rectification we use bridge rectifier. From the basic bridge configuration we see that two diodes(say D2 & D3) are conducting while the other two diodes (D1 & D4) are in off state during the period $t = 0$ to $T/2$. Accordingly for the negative cycle of the input the conducting diodes are D1 & D4. Thus the polarity across the load is the same.

In the bridge rectifier the diodes may be of variable types like 1N4001, 1N4003, 1N4004, 1N4005, 1N4007 etc.... can be used. But here we use 1N4007, because it can withstand up to 1000v.

3.6 FILTERS:

In order to obtain a dc voltage of 0 Hz, we have to use a low pass filter. So that a capacitive filter circuit is used where a capacitor is connected at the rectifier output & a dc is obtained across it. The filtered waveform is essentially a dc voltage with negligible ripples & it is ultimately fed to the load.

3.7 REGISTERS:

The controller IC has two 8 bit registers, an instruction register (IR) and a data register (DR). The IR stores the instruction codes and address information for display data RAM (DD RAM) and character generator RAM (CG RAM). The IR can be written, but not read by the MPU.

The DR temporally stores data to be written to /read from the DD RAM or CG RAM. The data written to DR by the MPU, is automatically written to the DD RAM or CG RAM as an internal operation.

When an address code is written to IR, the data is automatically

transferred from the DD RAM or CG RAM to the DR. data transfer between the MPU is then completed when the MPU reads the DR. likewise, for the next MPU read of the DR, data in DD RAM or CG RAM at the address is sent to the DR automatically. Similarly, for the MPU write of the DR, the next DD RAM or CG RAM address is selected for the write operation.

The dot-matrix liquid crystal display controller and driver LSI displays alphanumeric, Japanese kana characters, and symbols. It can be configured to drive a dot-matrix liquid crystal display under the control of a 4- or 8-bit microprocessor. Since all the functions such as display RAM, character generator, and liquid crystal driver, required for driving a dot-matrix liquid crystal display are internally provided on one chip, a minimal system can be interfaced with this controller/driver

3.8 REGULATORS:

The output voltage from the capacitor is more filtered & finally regulated. The voltage regulator is a device, which maintains the output voltage constant irrespective of the change in supply variations, load variations & temperature changes. Here we use fixed voltage regulator namely LM7805. The IC LM7805 is a +5v regulator which is used for microcontroller.

3.9 FEATURES AND DESCRIPTION OF REGULATORS

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

CHAPTER 4

IoT stands for Internet of Things, which means accessing and controlling daily usable equipments and devices using Internet.

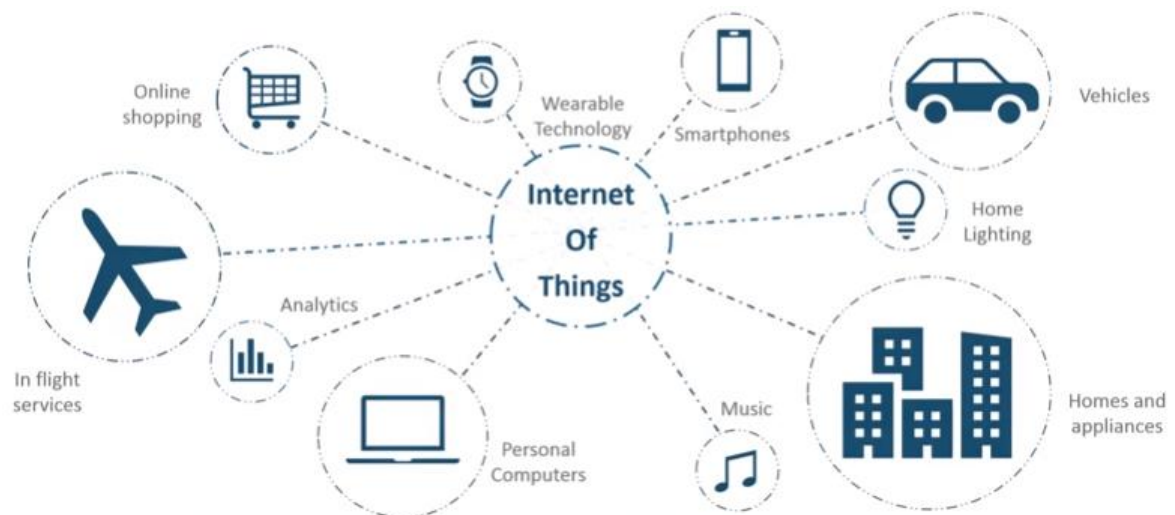
Our IoT tutorial includes all topics of IoT such as introduction, features, advantage and disadvantage, ecosystem, decision framework, architecture and domains, biometric, security camera and door unlock system, devices, etc.

What is an Internet of Things (IoT)

Let's us look closely at our mobile device which contains GPS Tracking, Mobile Gyroscope, Adaptive brightness, Voice detection, Face detection etc. These components have their own individual features, but what about if these all communicate with each other to provide a better environment? For example, the phone brightness is adjusted based on my GPS location or my direction.

Connecting everyday things embedded with electronics, software, and sensors to internet enabling to collect and exchange data without human interaction called as the Internet of Things (IoT).

The term "Things" in the Internet of Things refers to anything and everything in day to day life which is accessed or connected through the internet.



IoT is an advanced automation and analytics system which deals with artificial intelligence, sensor, networking, electronic, cloud messaging etc. to deliver complete systems for the product or services. The system created by IoT has greater transparency, control, and performance.

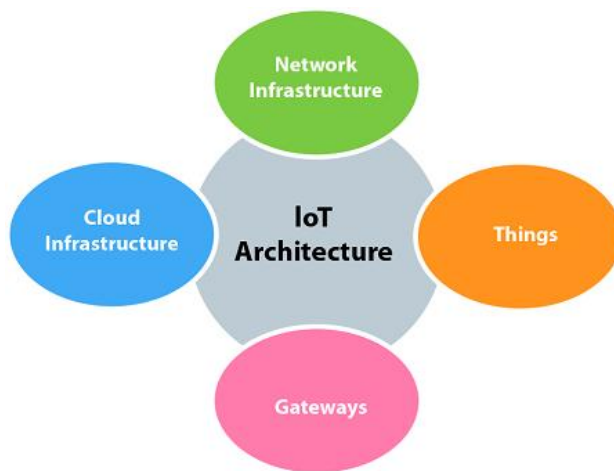
As we have a platform such as a cloud that contains all the data through which we connect all the things around us. For example, a house, where we can connect our home appliances such as air conditioner, light, etc. through each other and all these things are managed at the same platform. Since we have a platform, we can connect our car, track its fuel meter, speed level, and also track the location of the car.



If there is a common platform where all these things can connect to each other would be great because based on my preference, I can set the room temperature. For example, if I love the room temperature to be set at 25 or 26-degree Celsius when I reach back home from my office, then according to my car location, my AC would start before 10 minutes I arrive at home. This can be done through the Internet of Things (IoT).

How does Internet of Thing (IoT) Work?

The working of IoT is different for different IoT echo system (architecture). However, the key concept of there working are similar. The entire working process of IoT starts with the device themselves, such as smartphones, digital watches, electronic appliances, which securely communicate with the IoT platform. The platforms collect and analyze the data from all multiple devices and platforms and transfer the most valuable data with applications to devices.



Features of IOT

The most important features of IoT on which it works are connectivity, analyzing, integrating, active engagement, and many more. Some of them are listed below:

Connectivity: Connectivity refers to establish a proper connection between all the things of IoT to IoT platform it may be server or cloud. After connecting the IoT devices, it needs a high speed messaging between the devices and cloud to enable reliable, secure and bi-directional communication.

Analyzing: After connecting all the relevant things, it comes to real-time analyzing the data collected and use them to build effective business intelligence. If we have a good insight into data gathered from all these things, then we call our system has a smart system.

Integrating: IoT integrating the various models to improve the user experience as well.

Artificial Intelligence: IoT makes things smart and enhances life through the use of data. For example, if we have a coffee machine whose beans have going to end, then the coffee machine itself order the coffee beans of your choice from the retailer.

Sensing: The sensor devices used in IoT technologies detect and measure any change in the environment and report on their status. IoT technology brings passive networks to active networks. Without sensors, there could not hold an effective or true IoT environment.

Active Engagement: IoT makes the connected technology, product, or services to active engagement between each other.

Endpoint Management: It is important to be the endpoint management of all the IoT system otherwise, it makes the complete failure of the system. For example, if a coffee machine itself order the coffee beans when it goes to end but what happens when it orders the beans from a retailer and we are not present at home for a few days, it leads to the failure of the IoT system. So, there must be a need for endpoint management.

Advantages and Disadvantages of (IoT)

Any technology available today has not reached to its 100 % capability. It always has a gap to go. So, we can say that **Internet of Things** has a significant technology in a world that can help other technologies to reach its accurate and complete 100 % capability as well.

Let's take a look over the major, advantages, and disadvantages of the Internet of Things.

Advantages of IoT

Internet of things facilitates the several advantages in day-to-day life in the business sector. Some of its benefits are given below:

- **Efficient resource utilization:** If we know the functionality and the way that how each device work we definitely increase the efficient resource utilization as well as monitor natural resources.

- **Minimize human effort:** As the devices of IoT interact and communicate with each other and do lot of task for us, then they minimize the human effort.
- **Save time:** As it reduces the human effort then it definitely saves out time. Time is the primary factor which can save through IoT platform.
- **Enhance Data Collection:**
- **Improve security:** Now, if we have a system that all these things are interconnected then we can make the system more secure and efficient.

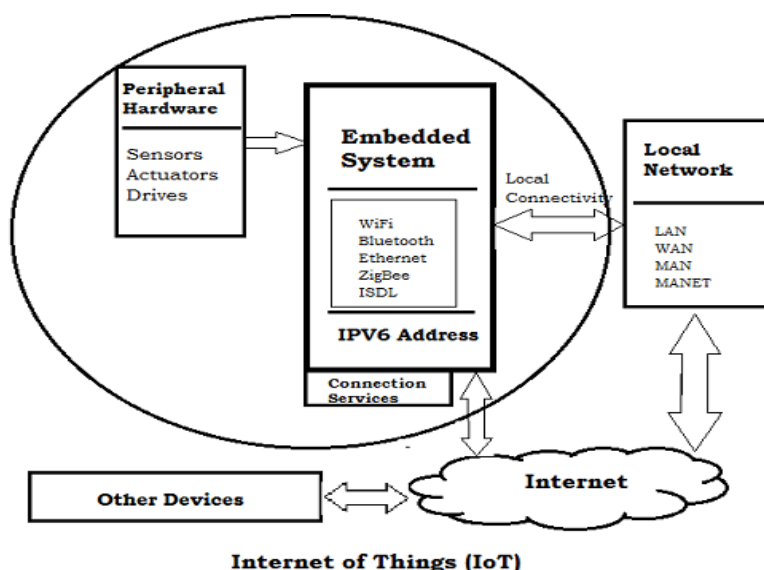
Disadvantages of IoT

As the Internet of things facilitates a set of benefits, it also creates a significant set of challenges. Some of the IoT challenges are given below:

Embedded Devices (System) in (IoT)

It is essential to know about the embedded devices while learning the IoT or building the projects on IoT. The embedded devices are the objects that build the unique computing system. These systems may or may not connect to the Internet.

An embedded device system generally runs as a single application. However, these devices can connect through the internet connection, and able communicate through other network devices.



Embedded System Hardware

The embedded system can be of type microcontroller or type microprocessor. Both of these types contain an integrated circuit (IC).

The essential component of the embedded system is a RISC family microcontroller like Motorola 68HC11, PIC 16F84, Atmel 8051 and many more. The most important factor that differentiates these microcontrollers with the microprocessor like 8085 is their internal read and writable memory. The essential embedded device components and system architecture are specified below.

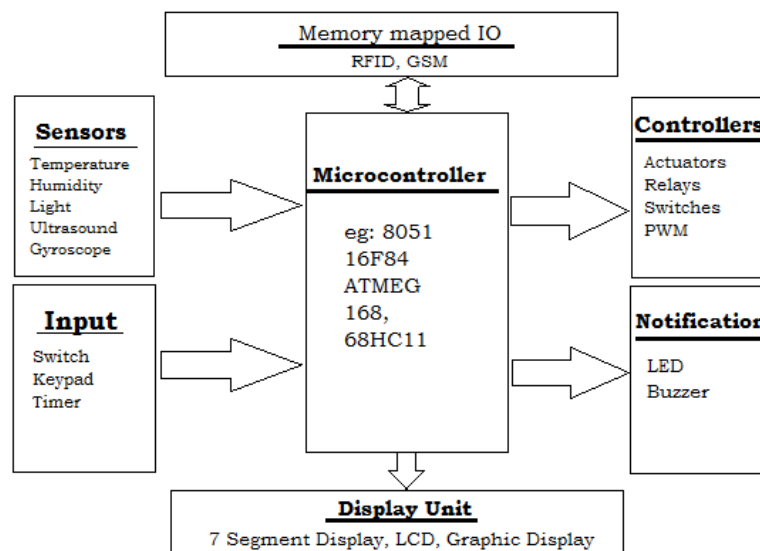


Fig: Basic Embedded System

Embedded System Software

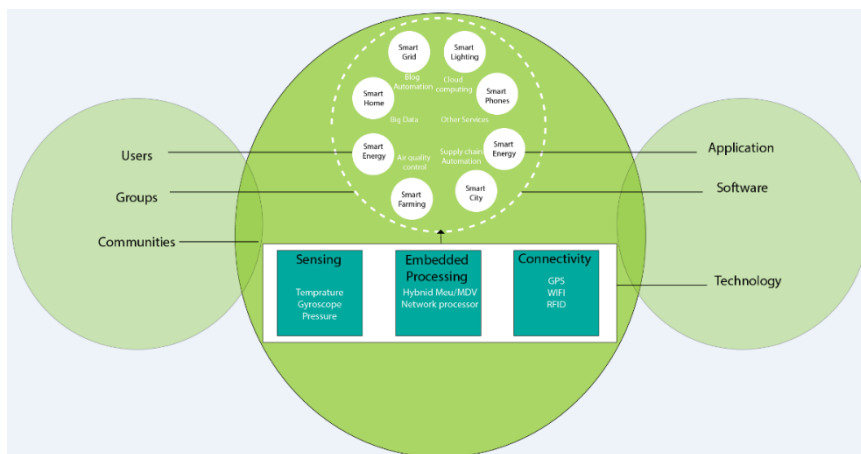
The embedded system that uses the devices for the operating system is based on the language platform, mainly where the real-time operation would be performed. Manufacturers build embedded software in electronics, e.g., cars, telephones, modems, appliances, etc. The embedded system software can be as simple as lighting controls running using an 8-bit microcontroller. It can also be complicated software for missiles, process control systems, airplanes etc.

IoT Ecosystem

The IoT ecosystem is not easy to define. It is also difficult to capture its proper image due to the vastness and emerging possibility and the rapidity with which it is expanding in the entire sector. However, the IoT ecosystem is a connection of various kind of devices that sense and analyze the data and communicates with each other over the networks.

In the IoT ecosystem, the user uses smart devices such as smartphones, tablet, sensors, etc. to send the command or request to devices for information over the networks. The device response and performs the command to send information back to the user through networks after analyzed.

The typical IoT ecosystem is shown in below image, where the smarter devices send and receive data from the devices themselves in the environment that are integrate over network and Cloud Computing.



The IoT is itself an ecosystem of network devices that transfer the data. It is also well interconnected with Big Data and Cloud Computing.

- **Sensing, Embedded processing, Connectivity:** The IoT ecosystem senses its surrounding like temperature, gyroscope, pressure, etc. and make the embedded processing using devices. These devices are connected through any type of devices such as GPS, WiFi, RFID, etc. over the networks.
- **Smart devices and environment, Cloud Computing, Big Data:** The data transfer or receive through smart devices and

environments are communicated through Cloud Computing or others Servers and stored as Big Data.

- **Technology, Software, Application:** The IoT ecosystem uses any of different technologies, software and application to communicate and connect with smart devices and environment.
- **Users or groups of community:** The product or services generated by the IoT ecosystem are consumed by the users or the group of communities to serve the smart life.

IoT Decision Framework

The IoT decision framework provides a structured approach to create a powerful IoT product strategy. The IoT decision framework is all about the strategic decision making. The IoT Decision Framework helps us to understand the areas where we need to make decisions and ensures consistency across all of our strategic business decision, technical and more.

The IoT decision framework is much more important as the product or services communicates over networks goes through five different layers of complexity of technology.

1. Device Hardware
2. Device Software
3. Communications
4. Cloud Platform
5. Cloud Application



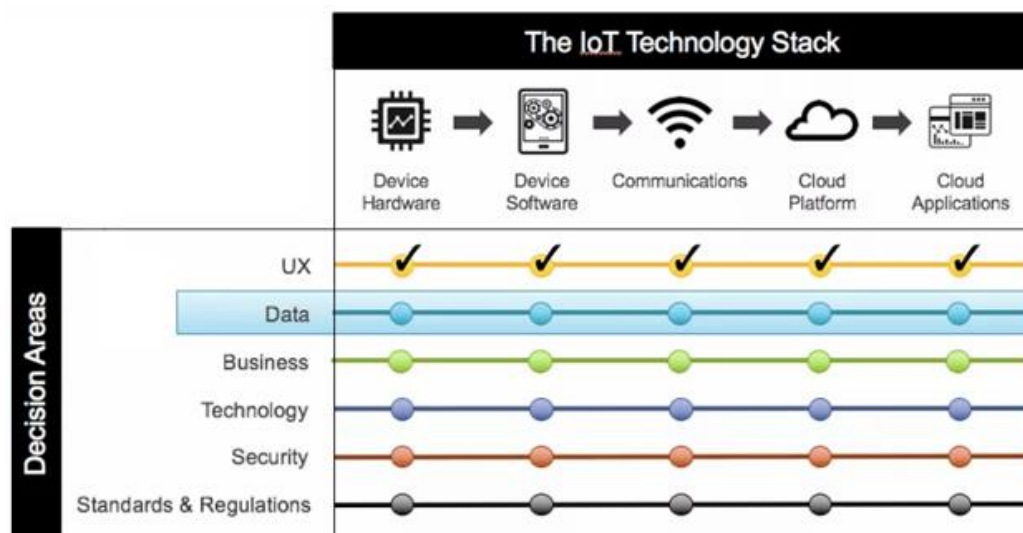
The IoT Technology Stack

Decision Area

The IoT decision framework pays attention to six key decision areas in any IoT product. These decision areas are:

1. User Experience (UX)
2. Data
3. Business
4. Technology
5. Security
6. Standards & Regulations

Each of these decision areas is evaluated at each of the IoT Technology Stack. The User Experience will be evaluated at Device Hardware, Device Software and so to provide the better user experience. Then at the next step Data Decision Area, we have to explore data considerations for all the stages of IoT Technology Stack.



Decision Area of the IoT Decision Framework

Let's see each of the Decision Area of IoT Decision Framework in detail:

1. **User Experience Decision Area:** This is the area where we concentrate about who are the users, what are their requirements and how to provide a great experience at each step of IoT stack without worrying about the technical details.
2. **Data Decision Area:** In this area, we make the overall data strategy such as the data flow over the entire IoT stack to fulfill the user's requirements.

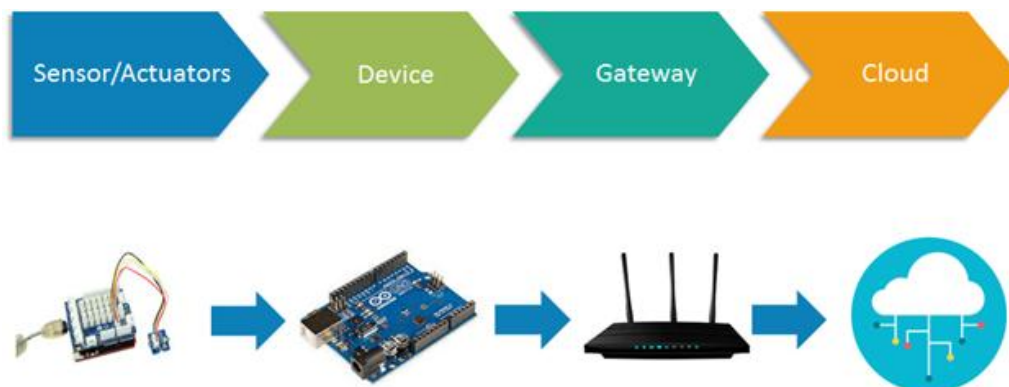
3. **Business Decision Area:** Based on the previous decisions area, we make the decision how product or services will become financial potential. At each of the IoT Stack level are monetized about the costs of providing services.
4. **Technology Decision Area:** In this area, we work with the technology for each layer to facilitate the final solution.
5. **Security Decision Area:** After going through the implementation of technology it is important to decide and provide the security at each stage of the IoT Stack.
6. **Standards & Regulations Decision Area:** At the last stage of IoT Decision Area, we identify the standards and regulations of product or services that will affect your product at each layer of the IoT Stack.

IoT Architecture

There is not such a unique or standard consensus on the Internet of Things (IoT) architecture which is universally defined. The IoT architecture differs from their functional area and their solutions. However, the IoT architecture technology mainly consists of four major components:

Components of IoT Architecture

- Sensors/Devices
- Gateways and Networks
- Cloud/Management Service Layer
- Application Layer



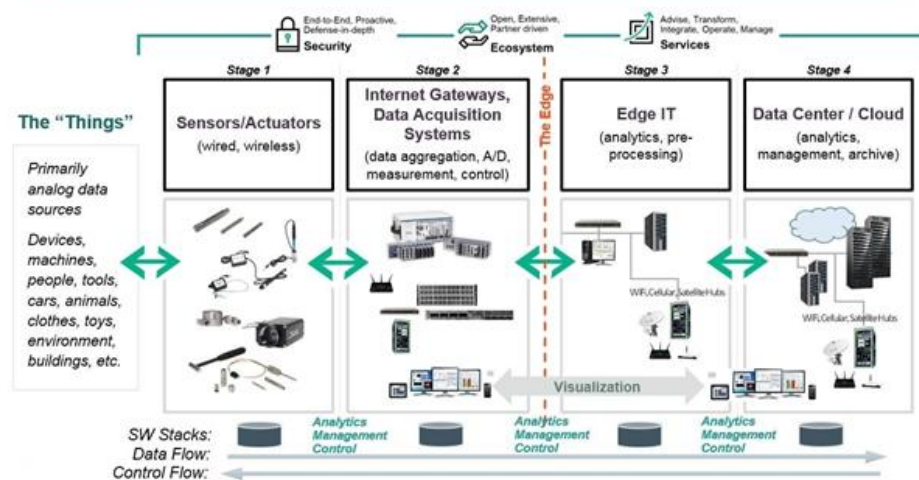
Stages of IoT Solutions Architecture

There are several layers of IoT built upon the capability and performance of IoT elements that provides the optimal solution to the business enterprises and end-users. The IoT architecture is a fundamental way to design the various elements of IoT, so that it can deliver services over the networks and serve the needs for the future.

Following are the primary stages (layers) of IoT that provides the solution for IoT architecture.

1. **Sensors/Actuators:** Sensors or Actuators are the devices that are able to emit, accept and process data over the network. These sensors or actuators may be connected either through wired or wireless. This contains GPS, Electrochemical, Gyroscope, RFID, etc. Most of the sensors need connectivity through sensors gateways. The connection of sensors or actuators can be through a Local Area Network (LAN) or Personal Area Network.
2. **Gateways and Data Acquisition:** As the large numbers of data are produced by this sensors and actuators need the high-speed Gateways and Networks to transfer the data. This network can be of type Local Area Network (LAN such as WiFi, Ethernet, etc.), Wide Area Network (WAN such as GSM, 5G, etc.).
3. **Edge IT:** Edge in the IoT Architecture is the hardware and software gateways that analyze and pre-process the data before transferring it to the cloud. If the data read from the sensors and gateways are not changed from its previous reading value then it does not transfer over the cloud, this saves the data used.
4. **Data center/ Cloud:** The Data Center or Cloud comes under the Management Services which process the information through analytics, management of device and security controls. Beside this security controls and device management the cloud transfer the data to the end users application such as Retail, Healthcare, Emergency, Environment, and Energy, etc.

The 4 Stage IoT Solutions Architecture



What are Smarts Objects in IoT

The concept of smart in IoT is used for physical objects that are active, digital, networked, can operate to some extent autonomously, reconfigurable and has local control of the resources. The smart objects need energy, data storage, etc.

A **smart object** is an object that enhances the interaction with other smart objects as well as with people also. The world of IoT is the network of interconnected heterogeneous objects (such as smart devices, smart objects, sensors, actuators, RFID, embedded computers, etc.) uniquely addressable and based on standard communication protocols.

In a day to day life, people have a lot of object with internet or wireless or wired connection. Such as:

- Smartphone
- Tablets
- TV computer

These objects can be interconnected among them and facilitate our daily life (smart home, smart cities) no matter the situation, localization, accessibility to a sensor, size, scenario or the risk of danger.



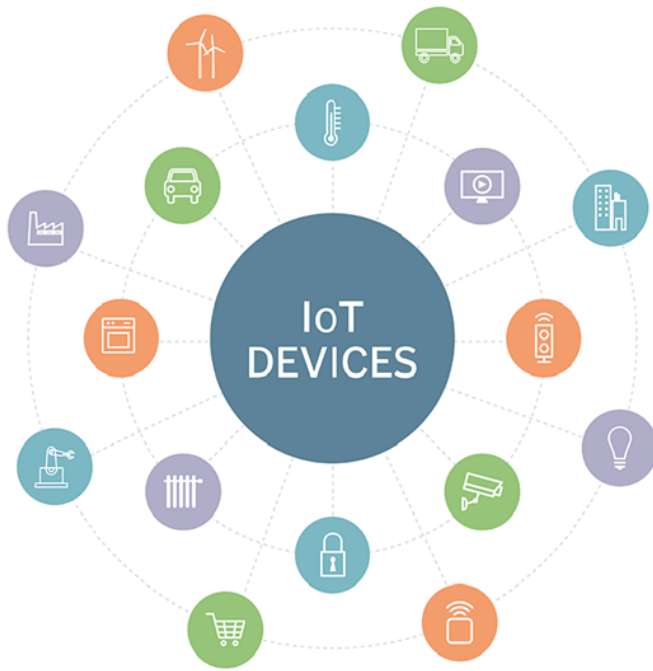
Smart objects are utilized widely to transform the physical environment around us to a digital world using the Internet of things (IoT) technologies.

A smart object carries blocks of application logic that make sense for their local situation and interact with human users. A smart object sense, log, and interpret the occurrence within themselves and the environment, and intercommunicate with each other and exchange information with people.

The work of smart object has focused on technical aspects (such as software infrastructure, hardware platforms, etc.) and application scenarios. Application areas range from supply-chain management and enterprise applications (home and hospital) to healthcare and industrial workplace support. As for human interface aspects of smart-object technologies are just beginning to receive attention from the environment.

IoT Devices

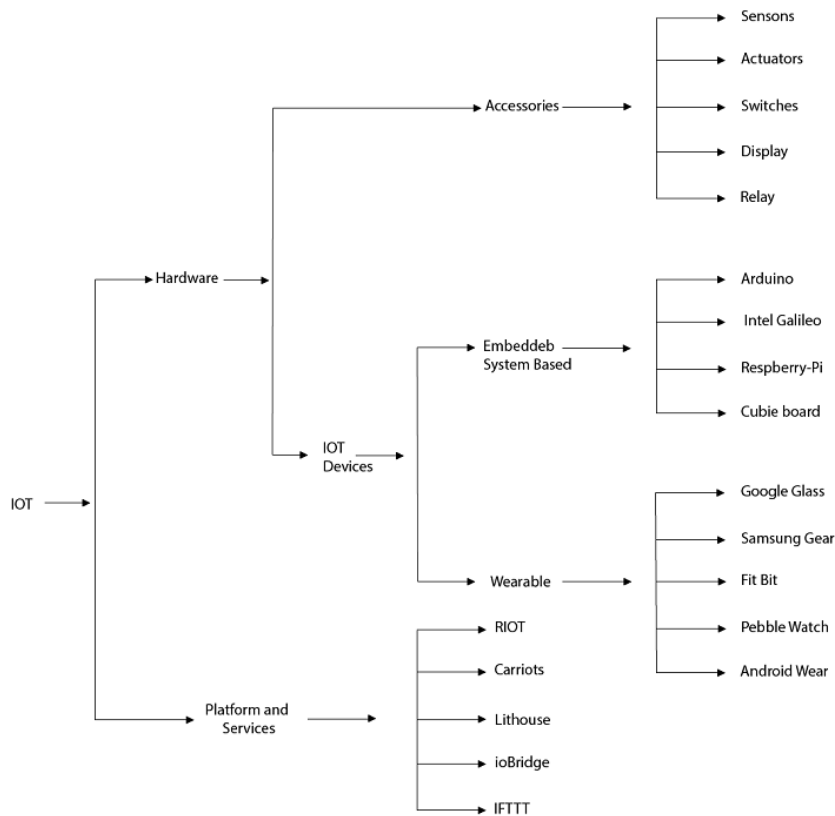
Internet of Things Devices is non-standard devices that connect wirelessly to a network with each other and able to transfer the data. IoT devices are enlarging the internet connectivity beyond standard devices such as smartphones, laptops, tablets, and desktops. Embedding these devices with technology enable us to communicate and interact over the networks and they can be remotely monitored and controlled.



There are large varieties of IoT devices available based on IEEE 802.15.4 standard. These devices range from wireless motes, attachable sensor-boards to interface-board which are useful for researchers and developers.

IoT devices include computer devices, software, wireless sensors, and actuators. These IoT devices are connected over the internet and enabling the data transfer among objects or people automatically without human intervention.

Some of the common and popular IoT devices are given below:



IoT Devices and Technologies

Arduino Device:

Arduino devices are the microcontrollers and microcontroller kit for building digital devices that can be sense and control objects in the physical and digital world. Arduino boards are furnished with a set of digital and analog input/output pins that may be interfaced to various other circuits. Some Arduino boards include USB (Universal Serial Bus) used for loading programs from the personal computer.



Intel Galileo:

The Intel Galileo Gen 2 Board includes the parts such as Intel Quark SoC processor, 256MB RAM, multiple ports and supports for Arduino device.



Samsung Gear Fit:

A Samsung Gear Fit device is a dustproof, water-resistant with fitness tracker features, a curved display, and long-lasting battery. This device receives alerts about emails and text messages, and it integrates with Samsung's S Health app.



Sensor:

A sensor is a device that reads the surrounding temperature, humidity, light, air quality control etc. There are different types of sensors available that reads different types of data. The sensors transmit these data over the networks or through which it is connected.



Bluetooth Low Energy (BLE) Intelligent Beacon:

A Bluetooth low energy beacon device is used to track the object located at a real time. Many companies use it to track the location of employees, assets, patients, and more in real time. This service primarily focuses on manufacturing, retail, and healthcare services.



Properties of IoT Devices

Some of the essential properties of IoT devices are mention below:

- **Sense:** The devices that sense its surrounding environment in the form of temperature, movement, and appearance of things, etc.
- **Send and receive data:** IoT devices are able to send and receive the data over the network connection.
- **Analyze:** The devices can able to analyze the data that received from the other device over the internet networks.
- **Controlled:** IoT devices may control from some endpoint also. Otherwise, the IoT devices are themselves communicate with each other endlessly leads to the system failure.

Major IoT Boards in Market

There are several IoT boards available in the market to build the project. Some of the major IoT Boards are described below:

Raspberry Pi:

Raspberry Pi is a much popular device used in building IoT project. The recently launched Raspberry Pi 3 includes built-in WiFi and Bluetooth making the most compact and standalone computer. It provides a powerful environment to install a variety of programming packages such as Python, Node.js, LAMP stack, Java and much more. Using 40 GPIO pins, and four USB ports you can connect many peripherals and accessories to the Pi.



Arduino:

Arduino boards are the microcontrollers and microcontroller kit for building digital devices that can be sense and control objects in the physical and digital world. Arduino boards are furnished with a set of digital and analog input/output pins that may be interfaced to various other circuits. Some Arduino boards include USB (Universal Serial Bus) to load programs from the personal computer.



ESP8266:

The ESP8266 is a low-cost Wi-Fi microchip with 32-bit microcontroller capability, standard digital peripheral interfaces. There are different types of ESP8266 boards available for different needs. The primary goal of this board is to deal with the built-in Wifi through AT commands if used as device

module, but you can 'program' using Arduino board however it also read and controls input/output, digital and analog.



Sense HAT 8x8 RGB LED matrix:

Raspberry Pi Sense HAT is an integrated sensor that can measure humidity, temperature, acceleration, and pressure. The 8x8 LED matrix display data read from Raspberry Pi Sense HAT sensors. The Sense HAT has an 8×8 RGB LED matrix includes the following sensors:

1. Gyroscope
2. Accelerometer
3. Magnetometer
4. Temperature
5. Barometric pressure
6. Humidity



Bluetooth Module HC-05:

Bluetooth Module HC-05 device is a 6 pins Bluetooth device that is used for wireless communication. Usually, this device connects small devices like mobile phones, PDAs and TVs using a short-range wireless connection to exchange data. It communicates with the microcontroller using the serial port (USART).

Pin Description

- **EN:** It is the enable pin, when it is connected to 3.3V then module is enabled.
- **+5V:** This is the supply pin for connecting +5V.
- **GND:** It is the ground pin.
- **TX:** It is the transmitter pin of the UART communication.
- **RX:** It is the receiver pin of UART communication.
- **STATE:** It indicates whether the module is connected or not. It acts as a status indicator.



IoT - Platform

As in IoT, all the IoT devices are connected to other IoT devices and application to transmit and receive information using protocols. There is a gap between the IoT device and IoT application. An IoT Platform fills the gap between the devices (sensors) and application (network). Thus we can say that **an IoT platform is an integrated service that fulfills the gap between the IoT device and application and offers you to bring physical object online.**



There are several IoT Platforms available that provides facility to deploy IoT application actively. Some of them are listed below:

Amazon Web Services (AWS) IoT platform: Amazon Web Service IoT platform offers a set of services that connect to several devices and maintain the security as well. This platform collects data from connected devices and performs real-time actions.

Microsoft Azure IoT platform: Microsoft Azure IoT platform offers strong security mechanism, scalability and easy integration with systems. It uses standard protocols that support bi-directional communication between connected devices and platform. Azure IoT platform has an Azure Stream Analytics that processes a large amount of information in real-time generated by sensors. Some common features provided by this platform are:

- Information monitoring
- A rules engine
- Device shadowing
- Identity registry

Google Cloud Platform IoT: Google Cloud Platform is a global cloud platform that provides a solution for IoT devices and applications. It handles a large amount of data using Cloud IoT Core by connecting various devices. It allows

to apply BigQuery analysis or to apply Machine learning on this data. Some of the features provided by Google Cloud IoT Platform are:

- Cloud IoT Core
- Speed up IoT devices
- Cloud publisher-subscriber
- Cloud Machine Learning Engine

IBM Watson IoT platform: The IBM Watson IoT platform enables the developer to deploy the application and building IoT solutions quickly. This platform provides the following services:

- Real-time data exchange
- Device management
- Secure Communication
- Data sensor and weather data services

Artik Cloud IoT platform: Arthik cloud IoT platform is developed by Samsung to enable devices to connect to cloud services. It has a set of services that continuously connect devices to the cloud and start gathering data. It stores the incoming data from connected devices and combines this information. This platform contains a set of connectors that connect to third-party services.

Bosch IoT Suite:

Bosch cloud IoT Suit is based on Germany. It offers safe and reliable storing of data on its server in Germany. This platform supports full app development from prototype to application development.

How IoT platform help:

- IoT Platform connects sensors and devices.
- IoT platform handles different software communication protocol and hardware.
- IoT platform provides security and authentication for sensors and users.

- It collects, visualizes, and analyzes the data gathered by the sensor and device.

NODE MCU

NodeMCU is an open source [IoT](#) platform.^{[4][5]} It includes [firmware](#) which runs on the [ESP8266 Wi-Fi SoC](#) from [Espressif Systems](#), and hardware which is based on the ESP-12 module.^{[6][7]} The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the [Lua](#) scripting language. It 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^[8] and [SPIFFS](#)

NodeMCU was created shortly after the [ESP8266](#) came out. On December 30, 2013, [Espressif Systems](#)^[6] began production of the ESP8266.^[10] The ESP8266 is a Wi-Fi SoC integrated with a [Tensilica](#) Xtensa LX106 core,^[citation needed] widely used in IoT applications (see [related projects](#)). NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub.^[11] 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.^[12] Later that month, Tuan PM ported [MQTT](#) client library from [Contiki](#) to the ESP8266 SoC platform,^[13] 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^[14] to NodeMCU project,^[15] enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the NodeMCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs.

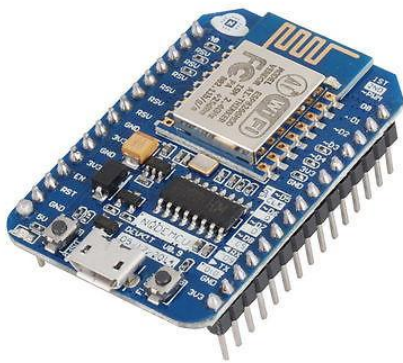
ESP8266 Arduino Core^[edit]

As [Arduino.cc](#) began developing new MCU boards based on non-[AVR](#) processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the [Arduino IDE](#) so that it would be relatively easy to change the IDE to support alternate toolchains to allow Arduino C/C++ to be compiled for these new processors. They did this with the

introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 WiFi SoC, popularly called the "ESP8266 Core for the Arduino IDE".^[16] This has become a leading software development platform for the various ESP8266-based modules and development boards, including NodeMCUs.

Introduction

NodeMCU is an open source [Lua](#) based firmware developed for ESP8266 wifi chip. By exploring functionality with ESP8266 chip, NodeMCU firmware comes with ESP8266 Development board/kit i.e. NodeMCU Development board.



NodeMCU Development Board/kit v0.9 (Version1)

Since NodeMCU is open source platform, their hardware design is open for edit/modify/build.

NodeMCU Dev Kit/board consist of ESP8266 wifi enabled chip. The **ESP8266** is a low-cost [Wi-Fi](#) chip developed by Espressif Systems with TCP/IP protocol. For more information about ESP8266, you can refer [ESP8266 WiFi Module](#).

There is Version2 (V2) available for NodeMCU Dev Kit i.e. **NodeMCU Development Board v1.0 (Version2)**, which usually comes in black colored PCB.



NodeMCU Development Board/kit v1.0 (Version2)

For more information about NodeMCU Boards available in market refer [NodeMCU Development Boards](#)

NodeMCU Dev Kit has **Arduino like** Analog (i.e. A0) and Digital (D0-D8) pins on its board.

It supports serial communication protocols i.e. UART, SPI, I2C etc.

Using such serial protocols we can connect it with serial devices like I2C enabled LCD display, Magnetometer HMC5883, MPU-6050 Gyro meter + Accelerometer, RTC chips, GPS modules, touch screen displays, SD cards etc.

How to start with NodeMCU?

NodeMCU Development board is featured with wifi capability, analog pin, digital pins and serial communication protocols.

To get start with using NodeMCU for IoT applications first we need to know about how to write/download NodeMCU firmware in NodeMCU Development Boards. And before that where this NodeMCU firmware will get as per our requirement.

There is online NodeMCU custom builds available using which we can easily get our custom NodeMCU firmware as per our requirement.

To know more about how to build custom NodeMCU firmware online and download it refer [Getting started with NodeMCU](#)

How to write codes for NodeMCU?

After setting up ESP8266 with Node-MCU firmware, let's see the IDE (Integrated Development Environment) required for development of NodeMCU.

NodeMCU with ESPlorer IDE

[Lua](#) scripts are generally used to code the NodeMCU. Lua is an open source, lightweight, embeddable scripting language built on top of C programming language.

For more information about how to write Lua script for NodeMCU refer [Getting started with NodeMCU using ESPlorerIDE](#)

NodeMCU with Arduino IDE

Here is another way of developing NodeMCU with a well-known IDE i.e. Arduino IDE. We can also develop applications on NodeMCU using Arduino development environment. This makes easy for Arduino developers than learning new language and IDE for NodeMCU.

For more information about how to write Arduino sketch for NodeMCU refer [Getting started with NodeMCU using ArduinoIDE](#)

Difference in using ESPlorer and Arduino IDE

Well, there is a programming language difference we can say while developing application for NodeMCU using ESPlorer IDE and Arduino IDE.

We need to code in C\C++ programming language if we are using Arduino IDE for developing NodeMCU applications and Lua language if we are using ESPlorer IDE.

Basically, NodeMCU is Lua Interpreter, so it can understand Lua script easily. When we write Lua scripts for NodeMCU and send/upload it to NodeMCU, then they will get executes sequentially. It will not build binary firmware file of code for NodeMCU to write. It will send Lua script as it is to NodeMCU to get execute.

In Arduino IDE when we write and compile code, ESP8266 toolchain in background creates binary firmware file of code we wrote. And when we upload it to NodeMCU then it will flash all NodeMCU firmware with newly generated binary firmware code. In fact, it writes the complete firmware.

That's the reason why NodeMCU not accept further Lua scripts/code after it is getting flashed by Arduino IDE. After getting flashed by Arduino sketch/code it will be no more Lua interpreter and we got error if we try to upload Lua scripts. To again start with Lua script, we need to flash it with NodeMCU firmware.

Since Arduino IDE compile and upload/writes complete firmware, it takes more time than ESPlorer IDE.

IR (Infrared) Sensor

An [infrared sensor](#) is an electronic device that emits in order to sense some aspects of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. These types of sensors measures only infrared radiation, rather than emitting it that is called as a [passive IR sensor](#). Usually in the infrared spectrum, all the objects radiate some form of thermal radiations. These types of radiations are invisible to our eyes that can be detected by an infrared sensor. The emitter is simply an IR LED ([Light Emitting Diode](#)) and the detector is simply an IR photodiode which is sensitive to IR light of the same wavelength as that emitted by the IR LED. When IR light falls on the photodiode, the resistances and these output voltages, change in proportion to the magnitude of the IR light received.

IR Sensor Circuit Diagram and Working Principle

An infrared sensor circuit is one of the basic and popular sensor module in an [electronic device](#). This sensor is analogous to human's visionary senses, which can be used to detect obstacles and it is one of the common applications in real time. This circuit comprises of the following components

- [LM358 IC](#) 2 IR transmitter and receiver pair
- Resistors of the range of kilo ohms.

- Variable resistors.
- LED (Light Emitting Diode).

In this project, the transmitter section includes an IR sensor, which transmits continuous IR rays to be received by an IR receiver module. An IR output terminal of the receiver varies depending upon its receiving of IR rays. Since this variation cannot be analyzed as such, therefore this output can be fed to a comparator circuit. Here an [operational amplifier](#) (op-amp) of LM 339 is used as comparator circuit. When the IR receiver does not receive a signal, the potential at the inverting input goes higher than that non-inverting input of the comparator IC (LM339). Thus the output of the comparator goes low, but the LED does not glow. When the IR receiver module receives signal to the potential at the inverting input goes low. Thus the output of the comparator (LM 339) goes high and the LED starts glowing.

IR Sensor circuit

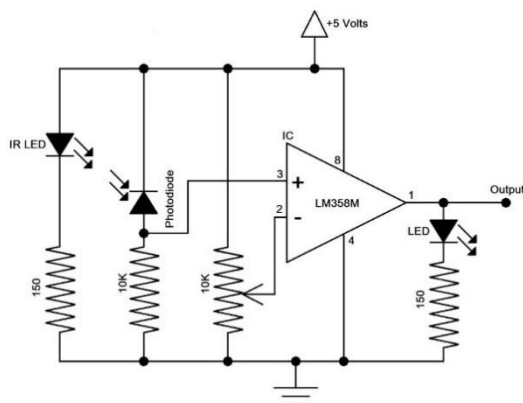


Fig. IR Sensor Circuit diagram

Resistor R1 (100 Ω), R2 (10k Ω) and R3 (330 Ω) are used to ensure that minimum 10 mA current passes through the IR LED Devices like Photodiode and normal LEDs respectively. Resistor VR2 (preset=5k) is used to adjust the output terminals. Resistor VR1 (preset=10k) is used to set the sensitivity of the circuit Diagram. Read more about IR sensors.

Different Types of IR Sensors and Their Applications

IR sensors are classified into different types depending on the

applications. The speed sensor is used for synchronizing the speed of multiple motors. The [temperature sensor](#) is used for industrial temperature control. [PIR sensor](#) is used for automatic door opening system and [Ultrasonic sensor](#) are used for distance measurement.

IR Sensor Applications

IR sensors are used in various [Sensor based projects](#) and also in various electronic devices which measures the temperature that are discussed in the below.

Radiation Thermometers

IR sensors are used in radiation thermometers to measure the temperature depend upon the temperature and the material of the object and these thermometers have some of the following features

- Measurement without direct contact with the object
- Faster response
- Easy pattern measurements

Flame Monitors

These types of devices are used for detecting the light emitted from the flames and to monitor how the flames are burning. The Light emitted from flames extend from UV to IR region types. PbS, PbSe, Two-color detector, pyro electric detector are some of the commonly employed detector used in flame monitors.

Moisture Analyzers

Moisture analyzers use wavelengths which are absorbed by the moisture in the IR region. Objects are irradiated with light having these wavelengths (1.1 μm , 1.4 μm , 1.9 μm , and 2.7 μm) and also with reference wavelengths. The Lights reflected from the objects depend upon the moisture content and is detected by analyzer to measure moisture (ratio of reflected light at these wavelengths to the reflected light at reference wavelength). In GaAs PIN photodiodes, Pbs photoconductive detectors are employed in moisture analyzer circuits.

IR Imaging Devices

IR image device is one of the major applications of IR waves, primarily by virtue of its property that is not visible. It is used for thermal imagers, night vision devices, etc.



For examples Water, rocks, soil, vegetation, an atmosphere, and human tissue all features emit IR radiation. The Thermal infrared detectors measure these radiations in IR range and map the spatial temperature distributions of the object/area on an image. Thermal imagers usually composed of a Sb (indium antimonite), Gd Hg (mercury-doped germanium), Hg Cd Te (mercury-cadmium-telluride) sensors.

An electronic detector is cooled to low temperatures using liquid helium or liquid nitrogen's. Then the Cooling the detectors ensures that the radiant energy (photons) recorded by the detectors comes from the terrain and not from the ambient temperature of objects within the scanner itself an IR imaging electronic devices.

Thus, this is all about IR sensor circuit with working and applications. These sensors are used in many sensor based [electronics projects](#).

LDR- LIGHT DEPENDENT RESISTOR

What is a Light Dependent Resistor or a Photo Resistor?

A **Light Dependent Resistor** (LDR) or a photo [resistor](#) is a device whose resistivity is a function of the incident electromagnetic radiation. Hence, they are light sensitive devices. They are also called as photo conductors, photo conductive cells or simply photocells. They are made up of [semiconductor](#) materials having high resistance. There are many different symbols used to indicate a **LDR**, one of the most commonly used symbol is shown in the figure below. The arrow indicates light falling on it.

Working Principle of LDR

A **light dependent resistor** works on the principle of photo conductivity. Photo conductivity is an optical phenomenon in which the materials conductivity (Hence resistivity) reduces when light is absorbed by the material. When light falls i.e. when the photons fall on the device, the electrons in the valence band of the [semiconductor](#) material are excited to the conduction band. These photons in the incident light should have energy greater than the band gap of the [semiconductor](#) material to make the electrons jump from the valence band to the conduction band. Hence when light having enough energy is incident on the device more & more electrons are excited to the conduction band which results in large number of charge carriers. The result of this process is more and more [current](#) starts flowing and hence it is said that the [resistance](#) of the device has decreased. This is the most common **working principle of LDR**

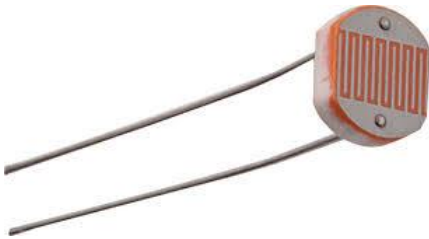
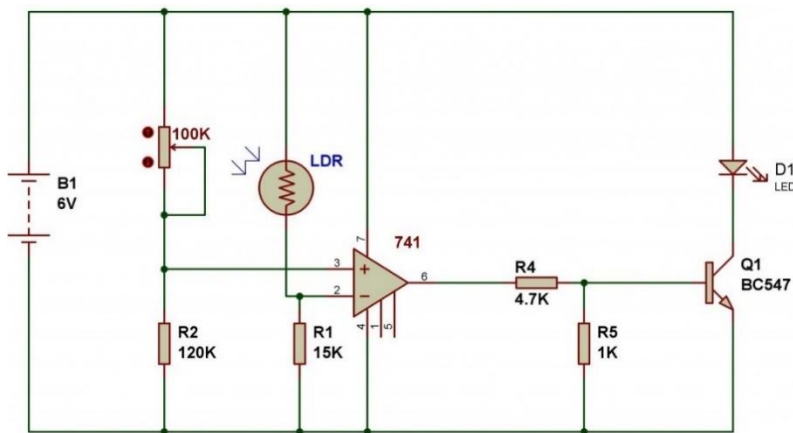


Diagram of LDR Sensor

Characteristics of LDR

LDR's are light dependent devices whose [resistance](#) decreases when light falls on them and increases in the dark. When a **light dependent resistor** is kept in dark, its resistance is very high. This resistance is called as dark [resistance](#). It can be as high as $10^{12} \Omega$. And if the device is allowed to absorb light its [resistance](#) will decrease drastically. If a constant [voltage](#) is applied to it and intensity of light is increased the [current](#) starts increasing. Figure below shows [resistance](#) vs. illumination curve for a particular **LDR**. Photocells or LDR's are non linear devices. Their sensitivity varies with the wavelength of light incident on them. Some photocells might not at all response to a certain range of wavelengths. Based on the material used different cells have different spectral response curves.



Circuit diagram of LDR Sensor

When light is incident on a photocell it usually takes about 8 to 12ms for the change in [resistance](#) to take place, while it takes seconds for the [resistance](#) to rise back again to its initial value after removal of light. This phenomenon is called as [resistance](#) recovery rate. This property is used in audio compressors. Also, **LDR's** are less sensitive than photo diodes and photo transistor. (A photo [diode](#) and a photocell (LDR) are not the same, a photo-diode is a p-n junction [semiconductor](#) device that converts light to electricity, whereas a photocell is a passive device, there is no p-n junction in this nor it “converts” light to electricity).

Types of Light Dependent Resistors:

Based on the materials used they are classified as: i) intrinsic photo [resistors](#) (Un doped semiconductor): These are pure [semiconductor](#) materials such as silicon or germanium. Electrons get excited from valance band to conduction band when photons of enough energy falls on it and number charge carriers increases.

ii) Extrinsic photo [resistor](#)s: These are [semiconductor](#) materials doped with impurities which are called as dopants. Theses dopants create new energy bands above the valence band which are filled with electrons. Hence this reduces the band gap and less energy is required in exciting them. Extrinsic photo [resistor](#)s are generally used for long wavelengths.

Construction of a Photocell

The structure of a light dependent [resistor](#) consists of a light sensitive material which is deposited on an insulating substrate such as ceramic. The material is deposited in zigzag pattern in order to obtain the desired [resistance](#) & power rating. This zigzag area separates the

metal deposited areas into two regions. Then the ohmic contacts are made on the either sides of the area. The [resistance](#)s of these contacts should be as less as possible to make sure that the [resistance](#) mainly changes due to the effect of light only. Materials normally used are cadmium sulphide, cadmium selenide, indium antimonide and cadmium sulphonide. The use of lead and cadmium is avoided as they are harmful to the environment.

Applications of LDR

LDR's have low cost and simple structure. They are often used as light sensors. They are used when there is a need to detect absences or presences of light like in a camera light meter. Used in street lamps, alarm clock, burglar alarm circuits, light intensity meters, for counting the packages moving on a conveyor belt, etc.

Radio frequency identification (RFID) is a form of wireless communication that uses radio waves to identify and track objects. RFID takes the barcoding concept and digitizes it for the modern world providing the ability to:

- Uniquely identify an individual item beyond just its product type
- Identify items without direct line-of-sight

- Identify many items (up to 1,000s) simultaneously
- Identify items within a vicinity of between a few centimetres to several meters

An RFID system has *readers* and *tags* that communicate with each other by radio. RFID tags are so small and require so little power that they don't even need a battery to store information and exchange data with readers. This makes it easy and cheap to apply tags to all kinds of things that people would like to identify or track.

Why Use RFID?

RFID technology has the capability to both greatly enhance and protect the lives of consumers, and also revolutionize the way companies do business. As the most flexible auto-identification technology, RFID can be used to track and monitor the physical world automatically and with accuracy.

RFID can tell you what an object is, where it is, and even its condition, which is why it is integral to the development of the Internet of Things—a globally interconnected web of objects allowing the physical world itself to become an information system, automatically sensing what is happening, sharing related data, and responding.

RFID use is increasing rapidly with the capability to “tag” any item with an inexpensive communications chip and then read that tag with a reader. Endless applications range from supply chain management to asset tracking to authentication of frequently counterfeited pharmaceuticals. Applications are limited, in fact, only by the imagination of the user.



RFID Applications



RFID can help:

- Automate inventory and asset-tracking in healthcare, manufacturing, retail, and business sectors
- Identify the source of products, enabling intelligent recall of defective or dangerous items, such as tainted foods, defective toys, and expired or compromised medication
- Prevent use of counterfeit products in the supply chain
- Improve shopping experience for consumers, with fewer out-of-stock items and easier returns
- Provide visibility into the supply chain, yielding a more efficient distribution channel and reduced business costs
- Decrease business revenue lost to theft or inaccurate accounting of goods
- Improve civilian security through better cargo monitoring at ports
- Wirelessly lock, unlock and configure electronic devices
- Enable access control of certain areas or devices

Whatever the application, RFID has the potential to increase efficiency of operations, improve asset visibility and traceability, decrease reliance on

manual processes, reduce operations costs, and provide useful data for business analytics.

A basic RFID solution includes:

Tag

- Tag chips or integrated circuits (ICs)
- Tag antennas

Reader

- Reader antenna
- Reader control & application software

In a basic RFID system, tags are attached to all items that are to be tracked. These tags are made from a tiny tag-chip, sometimes called an integrated circuit (IC), that is connected to an antenna that can be built into many different kinds of tags including apparel hang tags, labels, and security tags, as well as a wide variety of industrial asset tags. The tag chip contains memory which stores the product's electronic product code (EPC) and other variable information so that it can be read and tracked by RFID readers anywhere.

An RFID reader is a network connected device (fixed or mobile) with an antenna that sends power as well as data and commands to the tags. The RFID reader acts like an access point for RFID tagged items so that the tags' data can be made available to business applications.

RFID Tags



An RFID tag is comprised of an integrated circuit (called an IC or chip) attached to an antenna that has been printed, etched, stamped or vapor-deposited onto a mount which is often a paper substrate or PolyEthylene Therephtalate (PET). The chip and antenna combo, called an inlay, is then

converted or sandwiched between a printed label and its adhesive backing or inserted into a more durable structure.



Tag Chip

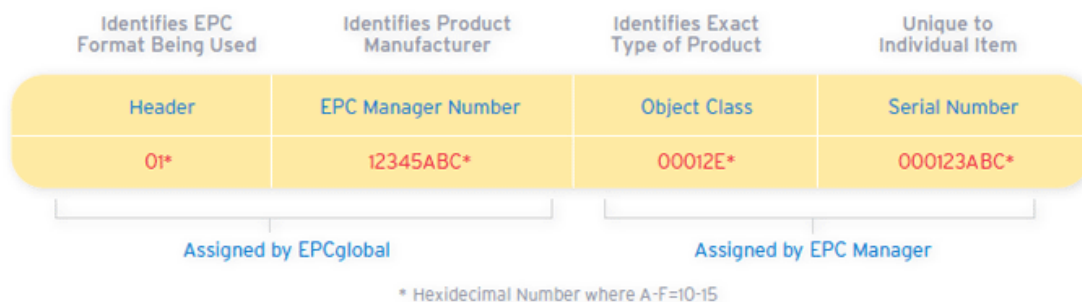
The tag's chip or integrated circuit (IC) delivers performance, memory and extended features to the tag. The chip is pre-programmed with a tag identifier (TID), a unique serial number assigned by the chip manufacturer, and includes a memory bank to store the items' unique tracking identifier (called an electronic product code or EPC).

Electronic Product Code (EPC)



950110126000000629

The electronic product code (EPC) stored in the tag chip's memory is written to the tag by an RFID printer and takes the form of a 96-bit string of data. The first eight bits are a header which identifies the version of the protocol. The next 28 bits identify the organization that manages the data for this tag; the organization number is assigned by the EPCglobal consortium. The next 24 bits are an object class, identifying the kind of product; the last 36 bits are a unique serial number for a particular tag. These last two fields are set by the organization that issued the tag. The total electronic product code number can be used as a key into a global database to uniquely identify that particular product.





Tag Antennas

Tag antennas collect energy and channel it to the chip to turn it on. Generally, the larger the tag antenna's area, the more energy it will be able to collect and channel toward the tag chip, and the further read range the tag will have.

There is no perfect antenna for all applications. It is the application that defines the antenna specifications. Some tags might be optimized for a particular [frequency band](#), while others might be tuned for good performance when attached to materials that may not normally work well for wireless communication (certain liquids and metals, for example). Antennas can be made from a variety of materials; they can be printed, etched, or stamped with conductive ink, or even vapor deposited onto labels.

Tags that have only a single antenna are not as reliable as tags with multiple antennas. With a single antenna, a tag's orientation can result in “dead zones”, or areas on the tag where incoming signals cannot be easily harvested to provide sufficient energy to power on the chip and communicate with the reader. A tag with dual antennas is able to eliminate these dead zones and increase its readability but requires a specialized chip.

RFID Readers



An RFID reader, also known as an interrogator, is a device that provides the connection between the tag data and the enterprise system software that needs the information. The reader communicates with tags that are within its field of operation, performing any number of tasks including simple continuous inventorying, filtering (searching for tags that meet certain criteria), writing (or encoding) to selected tags, etc.

The reader uses an attached antenna to capture data from tags. It then passes the data to a computer for processing. Just like RFID tags, there are many different sizes and types of RFID readers. Readers can be affixed in a stationary position in a store or factory, or integrated into a mobile device such as a portable, handheld scanner. Readers can also be embedded in electronic equipment or devices, and in vehicles.



Reader Antennas

RFID readers and reader antennas work together to read tags. Reader antennas convert electrical current into electromagnetic waves that are then radiated into space where they can be received by a tag antenna and converted back to electrical current. Just like tag antennas, there is a large variety of reader antennas and optimal antenna selection varies according to the solution's specific application and environment.

The two most common antenna types are linear- and circular-polarized antennas. Antennas that radiate linear electric fields have long ranges, and high levels of power that enable their signals to penetrate through different materials to read tags. Linear antennas are sensitive to tag orientation; depending on the tag angle or placement, linear antennas can have a difficult time reading tags. Conversely, antennas that radiate circular fields are less sensitive to orientation, but are not able to deliver as much power as linear antennas.

Choice of antenna is also determined by the distance between the RFID reader and the tags that it needs to read. This distance is called read range. Reader antennas operate in either a "near-field" (short range) or "far-field" (long range). In near-field applications, the read range is less than 30 cm and the antenna uses magnetic coupling so the reader and tag can transfer power. In near-field systems, the readability of the tags is not affected by the presence of dielectrics such as water and metal in the field.

In far-field applications, the range between the tag and reader is greater than 30 cm and can be up to several tens of meters. Far-field antennas

utilize electromagnetic coupling and dielectrics can weaken communication between the reader and tags.



Reader Control and Application Software

Reader control and application software, also known as middleware, helps connect RFID readers with the applications they support. The middleware sends control commands to the reader and receives tag data from the reader.

Creating an RFID Solution

Deploying an RFID system necessitates multiple actors and many different components. Typically, installing a system requires basic hardware-including tag chips, tag antennas, readers, and reader antennas-as well as reader control and application software, and solution providers to put it all together. When all of these components come together, an infinite number of creative applications are possible. RFID system applications can help improve the quality of business operations, inventory and customer experience in a variety of industries.

Frequency refers to the size of the radio waves used to communicate between RFID system components. RFID systems throughout the world operate in **low frequency (LF)**, **high frequency (HF)** and **ultra-high frequency (UHF) bands**. Radio waves behave differently at each of these frequencies with advantages and disadvantages associated with using each frequency band.

If an RFID system operates at a lower frequency, it has a shorter read range and slower data read rate, but increased capabilities for reading near or on metal or liquid surfaces. If a system operates at a higher frequency, it generally has faster data transfer rates and longer read ranges than lower

frequency systems, but more sensitivity to radio wave interference caused by liquids and metals in the environment.

LF RFID

The LF band covers frequencies from 30 KHz to 300 KHz. Typically LF RFID systems operate at 125 KHz, although there are some that operate at 134 KHz. This frequency band provides a short read range of 10 cm, and has slower read speed than the higher frequencies, but is not very sensitive to radio wave interference.

LF RFID applications include access control and livestock tracking.

Standards for LF animal-tracking systems are defined in ISO 14223, and ISO/IEC 18000-2. The LF spectrum is not considered a truly global application because of slight differences in frequency and power levels throughout the world.

HF RFID

The HF band ranges from 3 to 30 MHz. Most HF RFID systems operate at 13.56 MHz with read ranges between 10 cm and 1 m. HF systems experience moderate sensitivity to interference.

HF RFID is commonly used for ticketing, payment, and data transfer applications.

There are several HF RFID standards in place, such as the ISO 15693 standard for tracking items, and the ECMA-340 and ISO/IEC 18092 standards for Near Field Communication (NFC), a shortrange technology that is commonly used for data exchange between devices. Other HF standards include the ISO/IEC 14443 A and ISO/IEC 14443 standards for MIFARE technology, which used in smart cards and proximity cards, and the JIS X 6319-4 for FeliCa, which is a smart card system commonly used in electronic money cards.

UHF RFID

The UHF frequency band covers the range from 300 MHz to 3 GHz. Systems complying with the UHF Gen2 standard for RFID use the 860 to 960 MHz

band. While there is some variance in frequency from region to region, UHF Gen2 RFID systems in most countries operate between 900 and 915 MHz.

The read range of passive UHF systems can be as long as 12 m, and UHF RFID has a faster data transfer rate than LF or HF. UHF RFID is the most sensitive to interference, but many UHF product manufacturers have found ways of designing tags, antennas, and readers to keep performance high even in difficult environments. Passive UHF tags are easier and cheaper to manufacture than LF and HF tags.

UHF RFID is used in a wide variety of applications, ranging from retail inventory management, to pharmaceutical anti-counterfeiting, to wireless device configuration. The bulk of new RFID projects are using UHF opposed to LF or HF, making UHF the fastest growing segment of the RFID market.

The UHF frequency band is regulated by a single global standard called the ECPglobal Gen2 (ISO 18000-6C) UHF standard.

UHF • Single worldwide Gen2 standard • 20x the range and speed of HF • Labels cost 5¢–15¢ in 2012 • The technology for item tagging	HF and LF • Multiple competing standards • HF-based NFC for secure payment • Labels, cards, inlays cost 50¢–\$2 • Used in immobilizers, ticketing, payment
--	---

Passive, Active, and BAP RFID Systems

Active RFID Systems

In active RFID systems, tags have their own transmitter and power source. Usually, the power source is a battery. Active tags broadcast their own signal to transmit the information stored on their microchips.

Active RFID systems typically operate in the **ultra-high frequency (UHF) band** and offer a range of up to 100 m. In general, active tags are used on

large objects, such as rail cars, big reusable containers, and other assets that need to be tracked over long distances.

There are two main types of active tags: transponders and beacons. **Transponders** are “woken up” when they receive a radio signal from a reader, and then power on and respond by transmitting a signal back. Because transponders do not actively radiate radio waves until they receive a reader signal, they conserve battery life.

Beacons are used in most **real-time locating systems (RTLS)**, in order to track the precise location of an asset continuously. Unlike transponders, beacons are not powered on by the reader’s signal. Instead, they emit signals at pre-set intervals. Depending on the level of locating accuracy required, beacons can be set to emit signals every few seconds, or once a day. Each beacon’s signal is received by reader antennas that are positioned around the perimeter of the area being monitored, and communicates the tag’s ID information and position.

Passive RFID Systems

In passive RFID systems, the reader and reader antenna send a radio signal to the tag. The RFID tag then uses the transmitted signal to power on, and reflect energy back to the reader.

Passive RFID systems can operate in the **low frequency (LF)**, **high frequency (HF)** or **ultra-high frequency (UHF)** radio bands. As passive system ranges are limited by the power of the tag’s **backscatter** (the radio signal reflected from the tag back to the reader), they are typically less than 10 m. Because passive tags do not require a power source or transmitter, and only require a tag chip and antenna, they are cheaper, smaller, and easier to manufacture than active tags.

Passive tags can be packaged in many different ways, depending on the specific RFID application requirements. For instance, they may be mounted on a substrate, or sandwiched between an adhesive layer and a paper label to create smart RFID labels. Passive tags may also be embedded in a variety of devices or packages to make the tag resistant to extreme temperatures or harsh chemicals.

Passive RFID solutions are useful for many applications, and are commonly deployed to track goods in the supply chain, to inventory assets in the retail

industry, to authenticate products such as pharmaceuticals, and to embed RFID capability in a variety of devices. Passive RFID can even be used in warehouses and distribution centers, in spite of its shorter range, by setting up readers at choke points to monitor asset movement.

Battery-Assisted Passive (BAP) Systems

A Battery-Assisted Passive RFID tag is a type of passive tag which incorporates a crucial active tag feature. While most passive RFID tags use the energy from the RFID reader's signal to power on the tag's chip and backscatter to the reader, BAP tags use an integrated power source (usually a battery) to power on the chip, so all of the captured energy from the reader can be used for backscatter. Unlike transponders, BAP tags do not have their own transmitters.

	Active RFID	Passive RFID	Battery-Assisted Passive (BAP)
Tag Power Source	Internal to tag	Energy transfer from the reader via RF	Tag uses internal power source to power on, and energy transferred from the reader via RF to backscatter
Tag Battery	Yes	No	Yes
Availability of Tag Power	Continuous	Only within field of reader	Only within field of reader
Required Signal Strength from Reader to Tag	Very Low	Very high (must power the tag)	Moderate (does not need to power tag, but must power backscatter)
Available Signal Strength from Tag to Reader	High	Very Low	Moderate
Communication Range	Long Range (100m or more)	Short range (up to 10m)	Moderate range (up to 100m)

Sensor Capability	Ability to continuously monitor and record sensor input	Ability to read and transfer sensor values only when tag is powered by reader	Ability to read and transfer sensor values only when tag receives RF signal from reader
----------------------	---	--	---

Arduino ide:

Introduction

This tutorial will walk you through downloading, installing, and testing the [Arduino software](#) (also known as the Arduino IDE - short for Integrated Development Environment). Before you jump to the page for your operating system, make sure you've got all the right

equipment.



Required Materials

To follow along with this tutorial, you will need the following materials. You may not need everything though depending on what you have. Add it to your cart, read through the guide, and adjust the cart as necessary.

- A computer (Windows, Mac, or Linux)
- An Arduino-compatible microcontroller (anything from [this guide](#) should work)
- A USB A-to-B cable, or another appropriate way to connect your Arduino-compatible microcontroller to your computer (check out this [USB buying guide](#) if you're not sure which cable to get).



Ard
uin
o
Un
o -
R3

[US](#)
[B](#)
[Ca](#)
[ble](#)
[A](#)
[to](#)
[B](#)

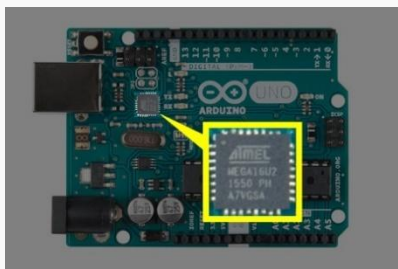


- 6 Foot

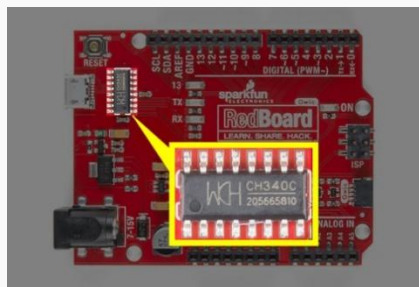
[What is an Arduino?](#)

What is this 'Arduino' thing anyway? This tutorial dives into what an Arduino is and along with Arduino projects and widgets.

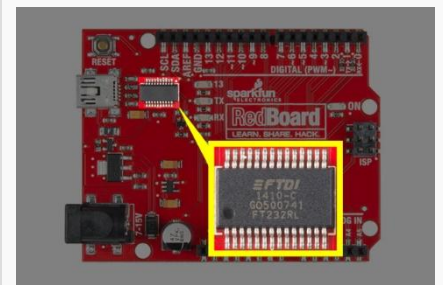
Note: There are several variants that use the Arduino Uno R3 footprint. Depending on the design, you may need to install additional drivers for your USB-to-serial converter before you are able to upload code to your microcontroller. For example, the [RedBoard uses an FTDI](#) while the [RedBoard Plus](#) uses the [CH340](#).



Atmega16U2 on the Arduino Uno R3



FTDI on the RedBoard



CH340 on the RedBoard Qwiic

Both use different drivers compared to the Arduino Uno R3. Make sure to look closely at your board and its respective hookup guide to determine USB-to-serial converter that is on board. You will probably have either an FTDI or CH340 populated on the board.

[How to Install FTDI Drivers](#)

How to install drivers for the FTDI Basic on Windows, Mac OS X, and Linux.

[How to Install CH340 Drivers](#)

How to install CH340 drivers (if you need them) on Windows, Mac OS X, and Linux.

Other boards using a different architecture like the [RedBoard Turbo \(SAMD21\)](#) have a built-in USB communication, eliminating the need to have a separate piece of hardware. For more information, check out the blog post on drivers: [What Drives your SparkFun Inventor's Kit?](#).

Downloading the Arduino IDE

You can [download the Arduino IDE](#) from their website. They have [installation](#)

[instructions](#), but we will also go over the installation process as well. Make sure you download the version that matches your operating system.

[CLICK FOR ARDUINO IDE DOWNLOAD PAGE](#)

The installation procedure is fairly straightforward, but it does vary by OS. Here are some tips to help you along.

Troubleshooting Tips

- We recommend using a computer with a full desktop operating system like Windows 7/10 (**avoid** Windows 8 if you can), Mac OSX, and certain flavors Linux (check the [Arduino FAQ page for compatibility](#)).
- If you are not a technical or computer savy individual and you have your choice of computers, I highly recommend using a **Windows 7 or 10** computer. You will usually run into the the least issues, if any, with these operating systems.
- We do **NOT** recommend using a Chromebook, Netbook, tablet, phone, or the Arduino Web IDE in general. You will be responsible for troubleshooting any driver or Arduino Web IDE issues.
- As of the writing of this tutorial (updated 12-14-2018), the most recent and stable release of the Arduino IDE is version 1.8.5. We recommend using that version of the Arduino IDE; you can download the [previous releases here](#).
- On Windows 10, we do **NOT** recommend installing the Arduino IDE from the app store. You may run into issues because the OS will automatically update to the most recent release of the Arduino IDE, which may have unknown bugs (like the compiler errors in versions 1.8.6 and 1.8.7).
- Raspberry Pi users with Raspbian installed should use the **Linux ARM** download. We do not recommend using the command line installation. It will install the oldest release of Arduino, which is useless when it comes to installing new boards definitions or libraries.

- For additional troubleshooting tips, here is a [troubleshooting guide from Arduino](#).

Windows

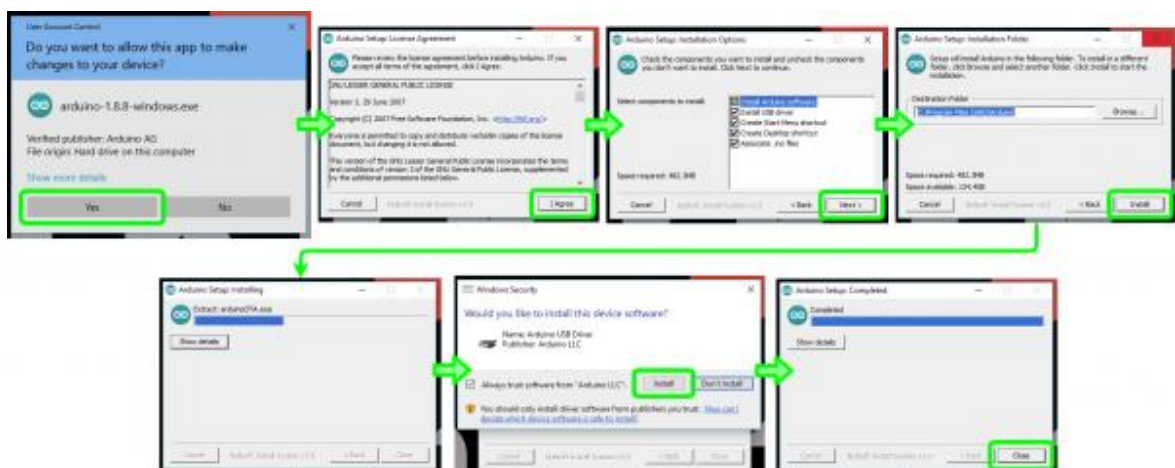
This page will show you how to install and test the Arduino software with a Windows operating system (Windows 10, Windows 7, Vista, and XP).

Go to the Arduino [download page](#) and download the latest version of the Arduino software for Windows if you have not already.

[CLICK FOR ARDUINO IDE DOWNLOAD PAGE](#)

Installer

The Windows version of Arduino is offered in two options: an installer or a zip file. The **installer** is the easier of the two options, just download that, and run the executable file to begin the installation.



Windows install steps. Click the image to get a bigger view.

When you're prompted to install a driver during installation, select "Install". This will install drivers for Arduino specific boards (like the Uno, Nano, etc.) that you may use in the future.

ZIP

If you choose to download the **zip file** version of Arduino, you'll need to extract the files yourself. Don't forget which folder you extract the files into! You will need to run the executable Arduino file in the folder to start the Arduino IDE.

When the download is finished, un-zip it and open up the Arduino

folder to confirm that yes, there are indeed some files and sub-folders inside. The file structure is important so don't be moving any files around unless you really know what you're doing.

Note: On Windows 10, there is an option to install Arduino through their app store. we do not recommend installing the Arduino IDE from the app store. You may run into issues because the OS will automatically update to the most recent release of the Arduino IDE, which may have unknown bugs.

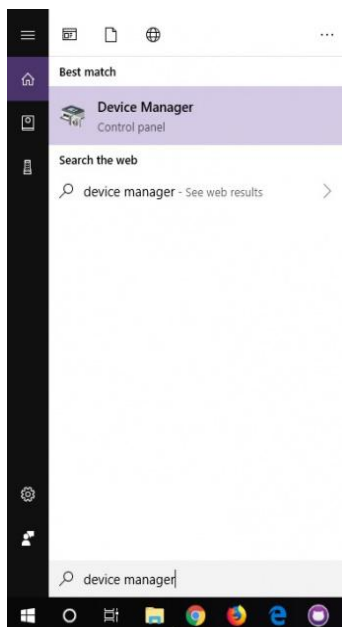
Connecting Your Arduino

Power up your Arduino by connecting your Arduino board to your computer with a USB cable (or FTDI cable if you're using an Arduino Pro). You should see the an LED labeled 'ON' light up. ([this diagram](#) shows the placement of the power LED on the UNO).

Drivers for Arduino Uno on Windows

To install the drivers for the Arduino Uno, you will need to plug in your board to your computer's USB port. Once the board is connected, you will need to wait for Windows to begin it's driver installation process. After a few moments, the process will probably fail, despite its best efforts.

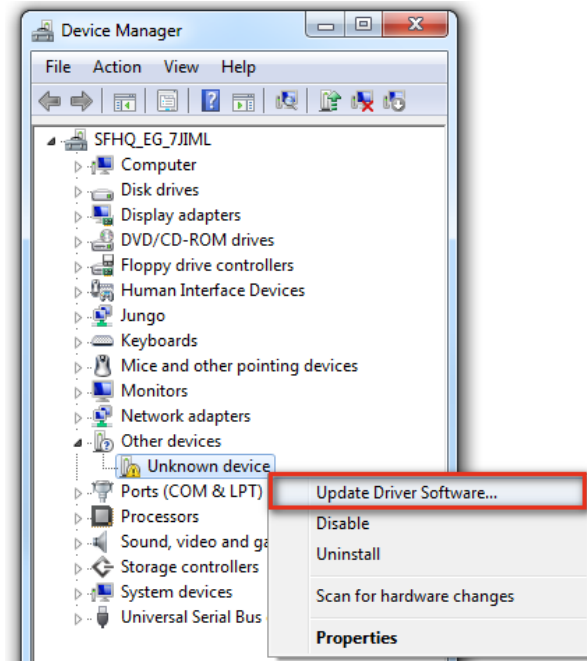
Open up a search, type in **Device Manager**, and hit **ENTER**.



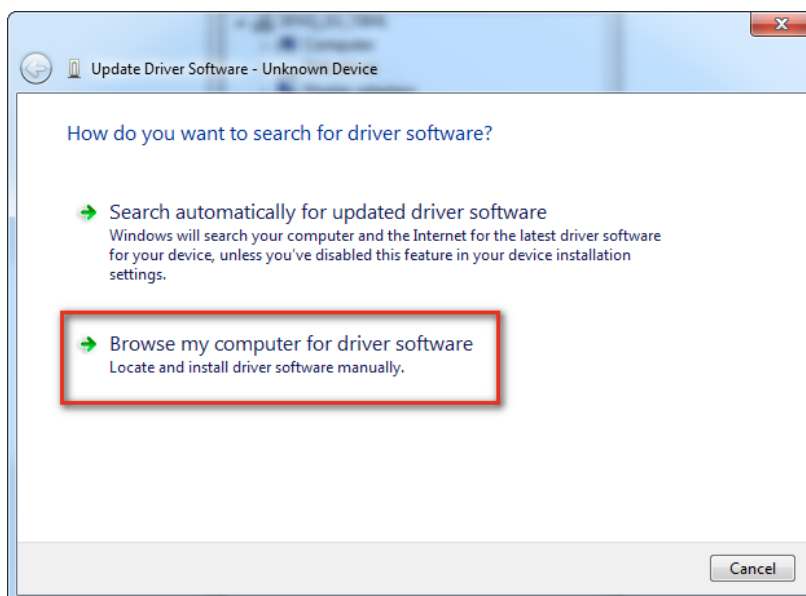
Note: Searching is the easiest method to open the Device Manager. However, there is more than one method of opening the device

manager. The longer method is to click on the **Start Menu > Windows System > System and Security > System > Device Manager**.

Look under Ports (COM & LPT) tree. You should see an open port named "**Arduino UNO (COMxx)**". If there is no COM & LPT section, look under "**Other devices**" for "**Unknown Device**". Right click on the "**Arduino UNO (COMxx)**" or "**Unknown device**" port and choose the "**Update Driver Software**" option.

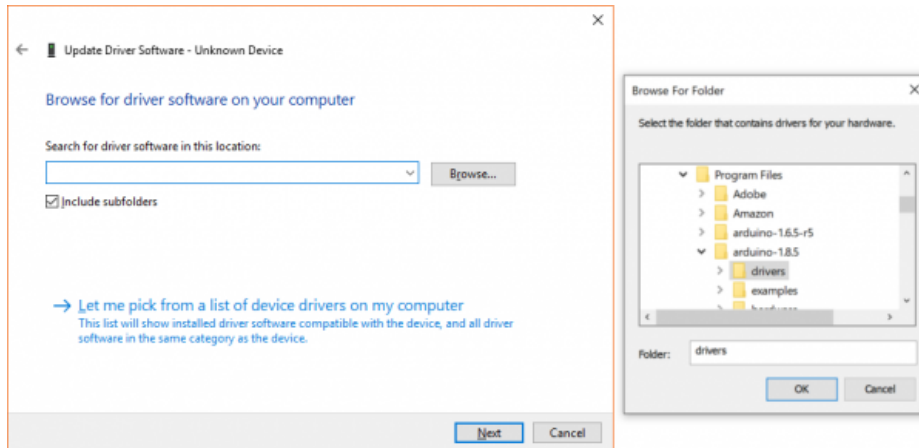


Next, choose the "**Browse my computer for Driver software**" option

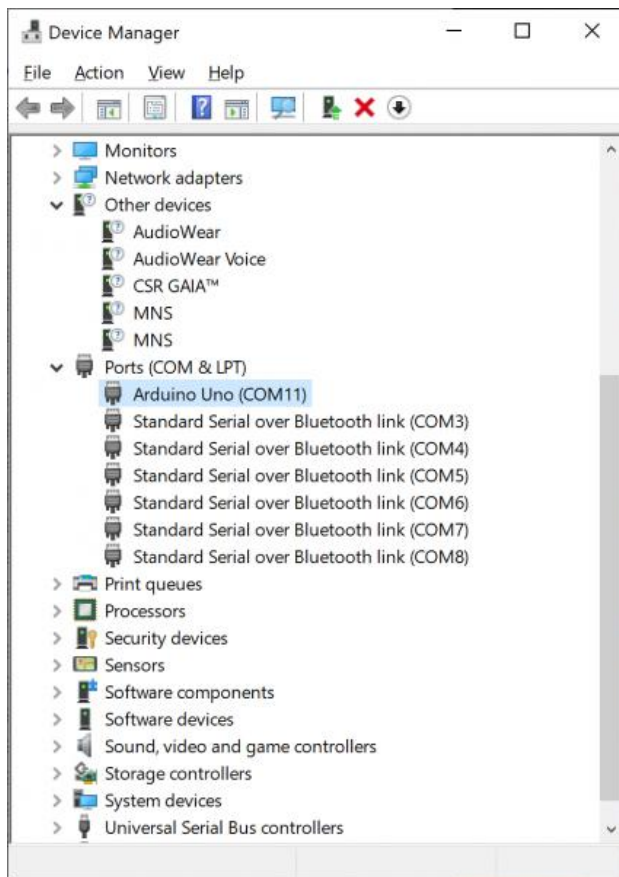


Finally, navigate to the Arduino IDE folder. This should be where you unzipped the Arduino IDE (e.g. it should be similar to the following path with a different version number: *C:\Program Files\arduino-*

1.8.5\drivers). Depending on what version of Windows you have, you may be able to select the Uno's driver file, named "**Arduino.inf**", located in the "**Drivers**" folder (not the "FTDI USB Drivers" sub-directory). If you cannot see the *.inf file, it is probably just hidden. You can select the "**drivers**" folder with the "**include sub-folders**" option selected instead.



Windows will finish up the driver installation from there! Your computer will enumerate with a COM port. You may see a COM port number depending on what is currently saved in your computer. Try to remember what the number is when uploading. If not, you can always navigate back to the device manager and power cycle the Arduino to determine what number your Arduino enumerated on.



For earlier versions of the Arduino boards (e.g. Arduino Duemilanove, Nano, or Diecimila) check out [this page](#) for specific directions.

ARDUINO.CC - GUIDE: WINDOWS

Drivers for RedBoard on Windows

Depending on your board, there may be different USB-to-serial converters on your board. If you are using a RedBoard you will need to go to the [How to Install FTDI Drivers](#) tutorial. If you are using the or RedBoard Qwiic, you will need to go to the [How to Install CH340 Drivers](#) tutorial. Make sure to look closely at your board and its respective hookup guide to determine USB-to-serial converter that is on board.

[How to Install FTDI Drivers](#)

JUNE 4, 2013

How to install drivers for the FTDI Basic on Windows, Mac OS X, and Linux.

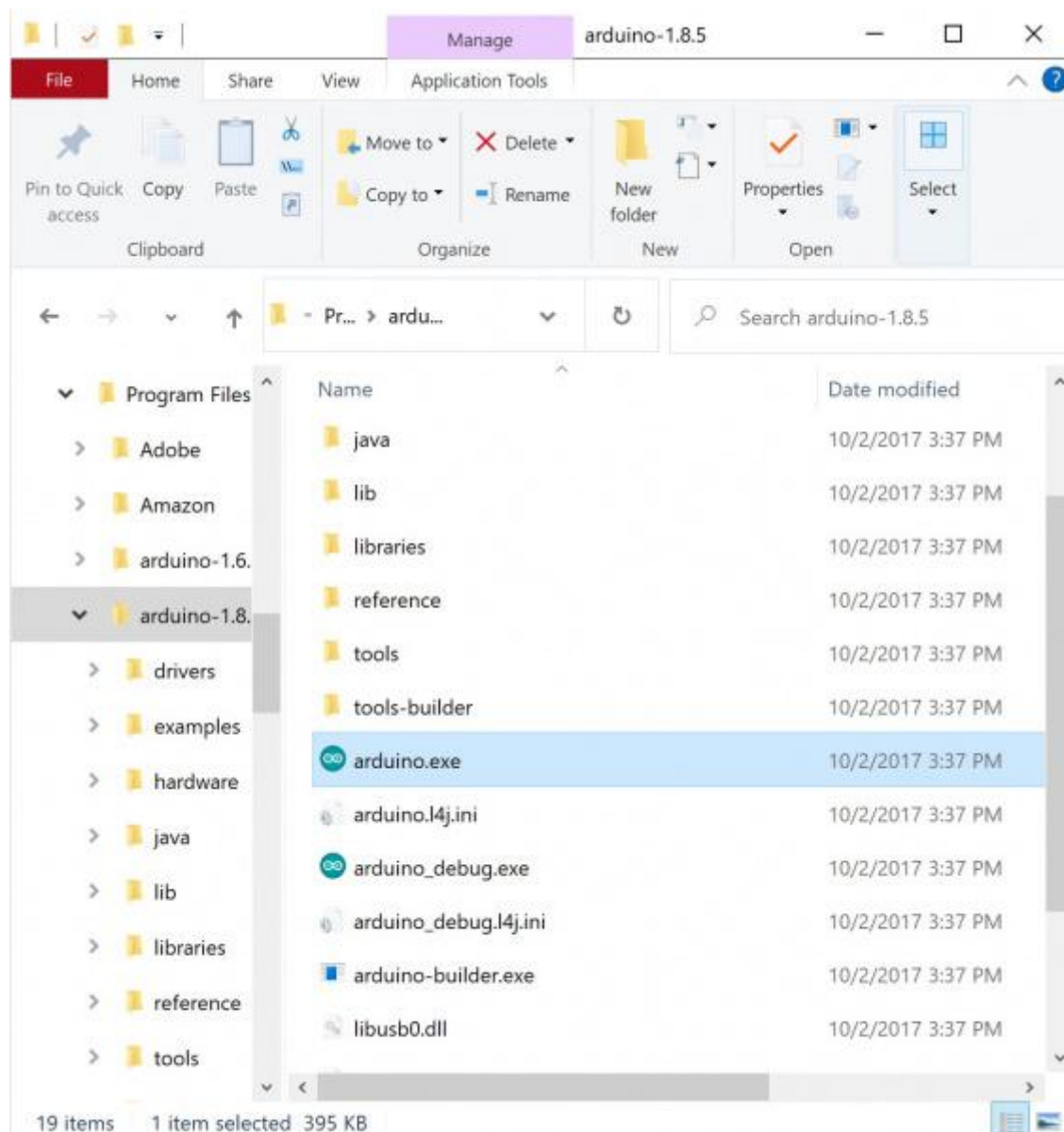
[How to Install CH340 Drivers](#)

AUGUST 6, 2019

How to install CH340 drivers (if you need them) on Windows, Mac OS X, and Linux.

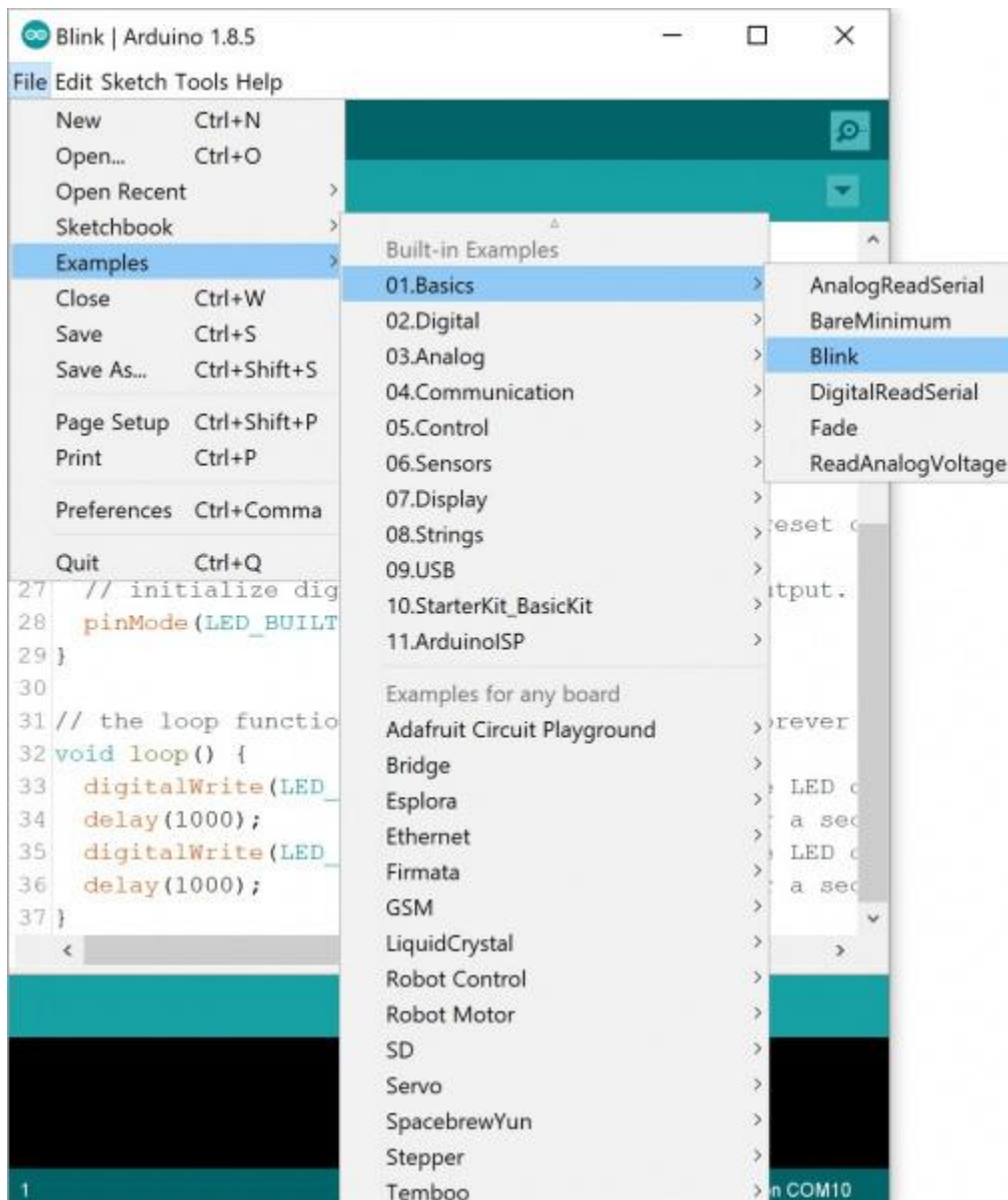
Blink!

After following the appropriate steps for your software install, we are now ready to test your first program with your Arduino board! Launch the Arduino application. If you disconnected your board, plug it back in.

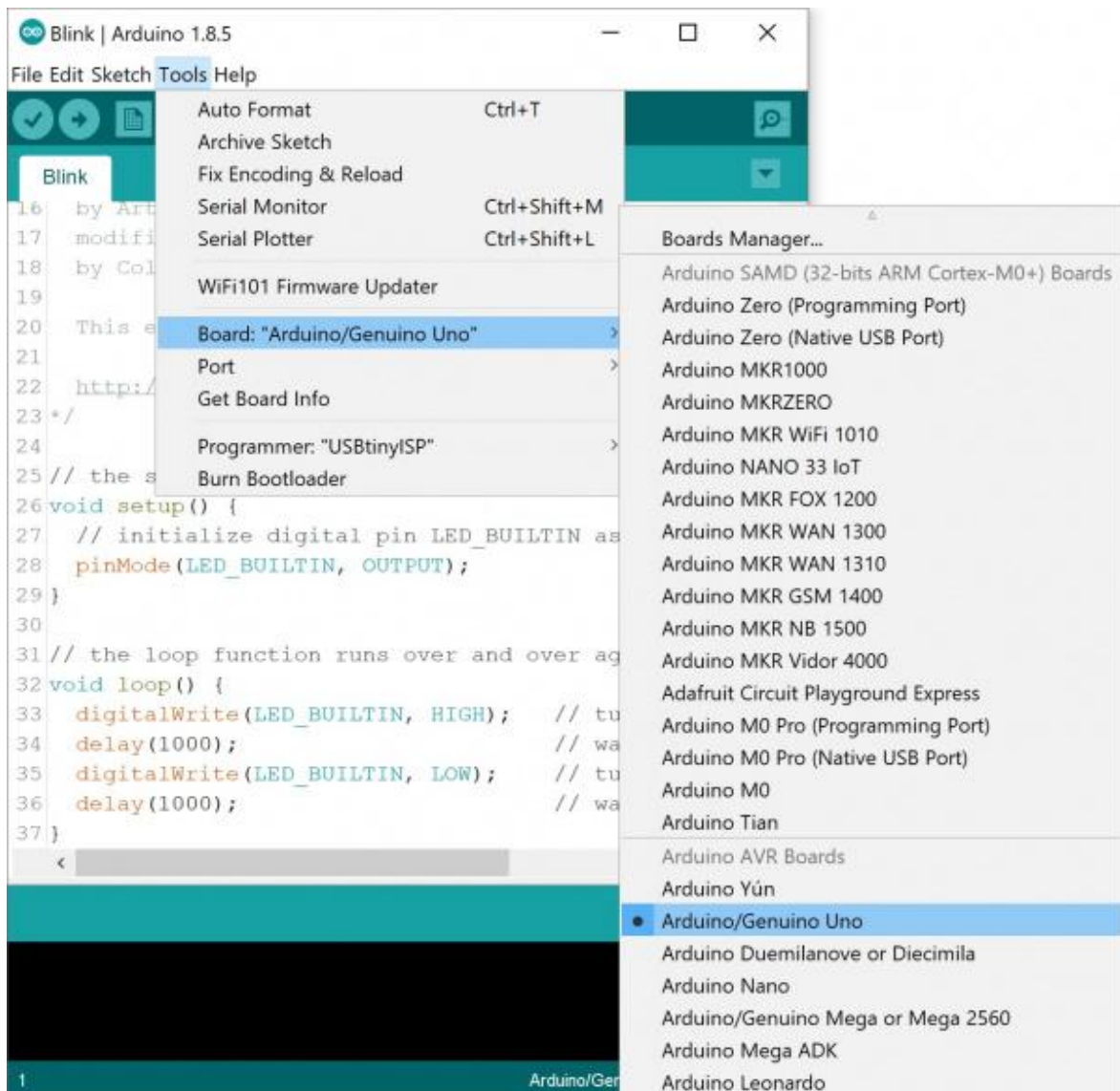


Note: Depending on your method of installing the Arduino IDE, the application may be on your desktop or the program folder.

Open the Blink example sketch by going to: **File > Examples > 01.Basics > Blink.**

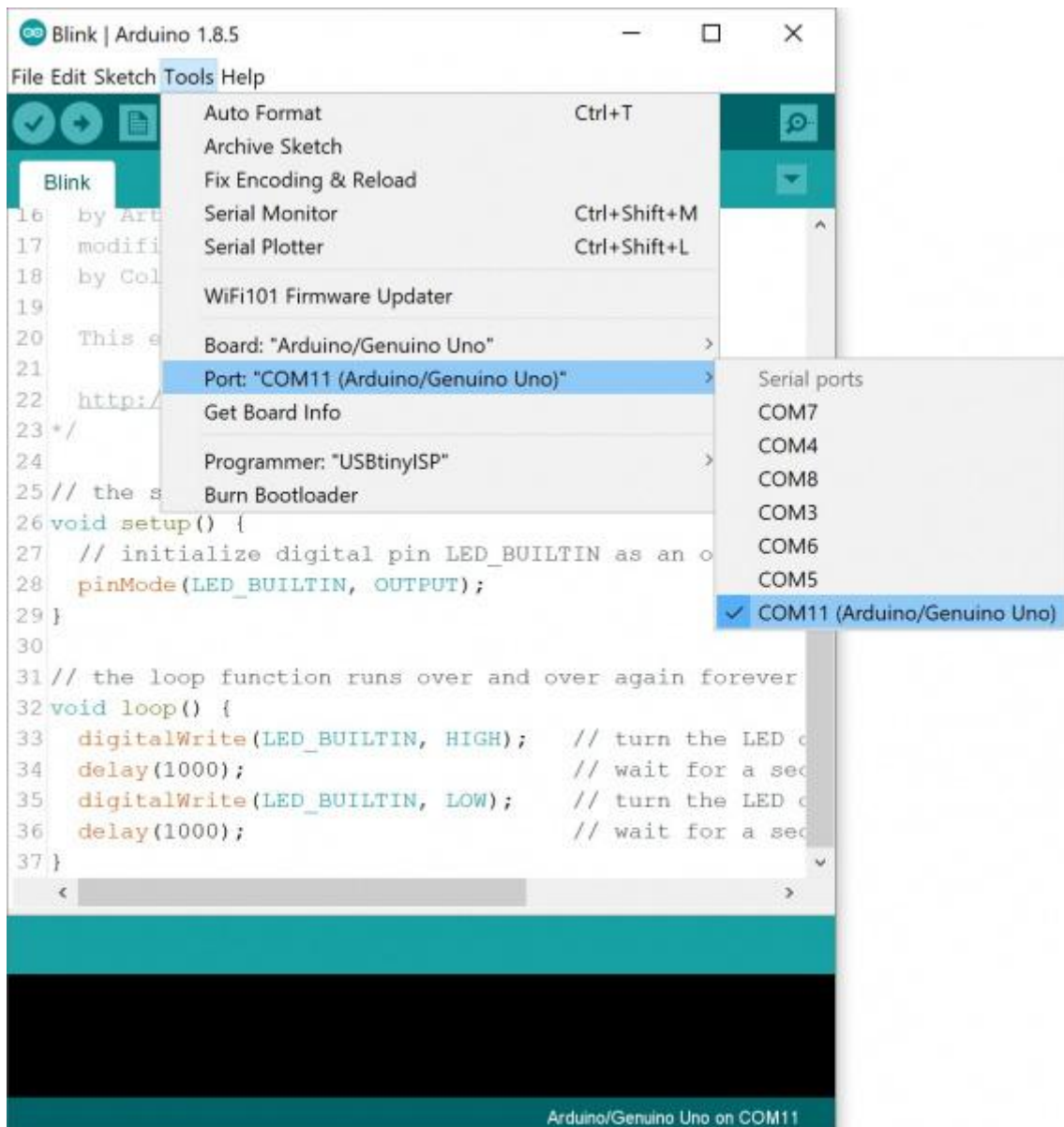


Select the type of Arduino board you're using: **Tools > Board > Arduino Uno.**



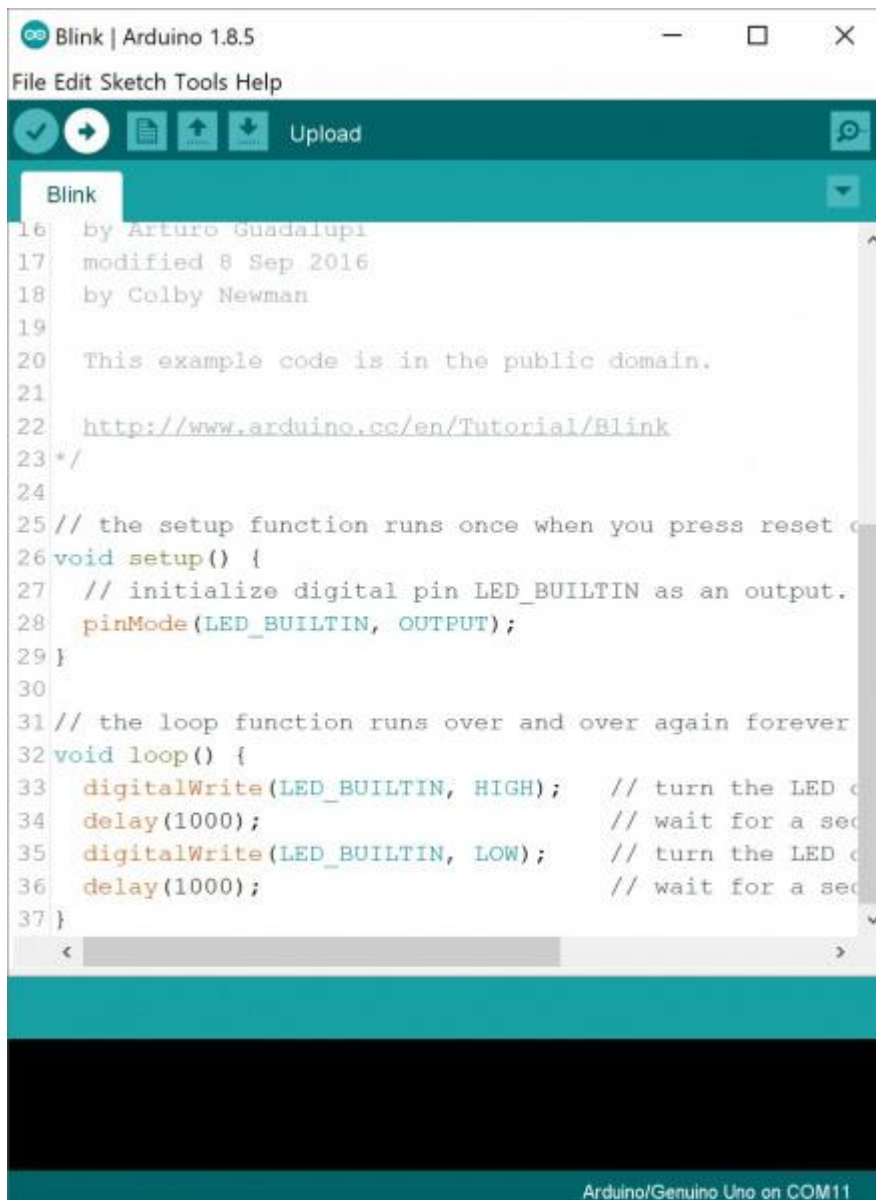
Note: As you move to other architectures, you may need to select a different board definition depending on your development board. For the Arduino Uno R3 and RedBoard development boards with ATmega328P, you can simply select **Arduino Uno**. Certain Arduino IDE versions may have you select **Arduino/ Genuino Uno**.

Select the serial/COM port that your Arduino is attached to: **Tools > Port > COMxx**. In this case it was **COM11**.

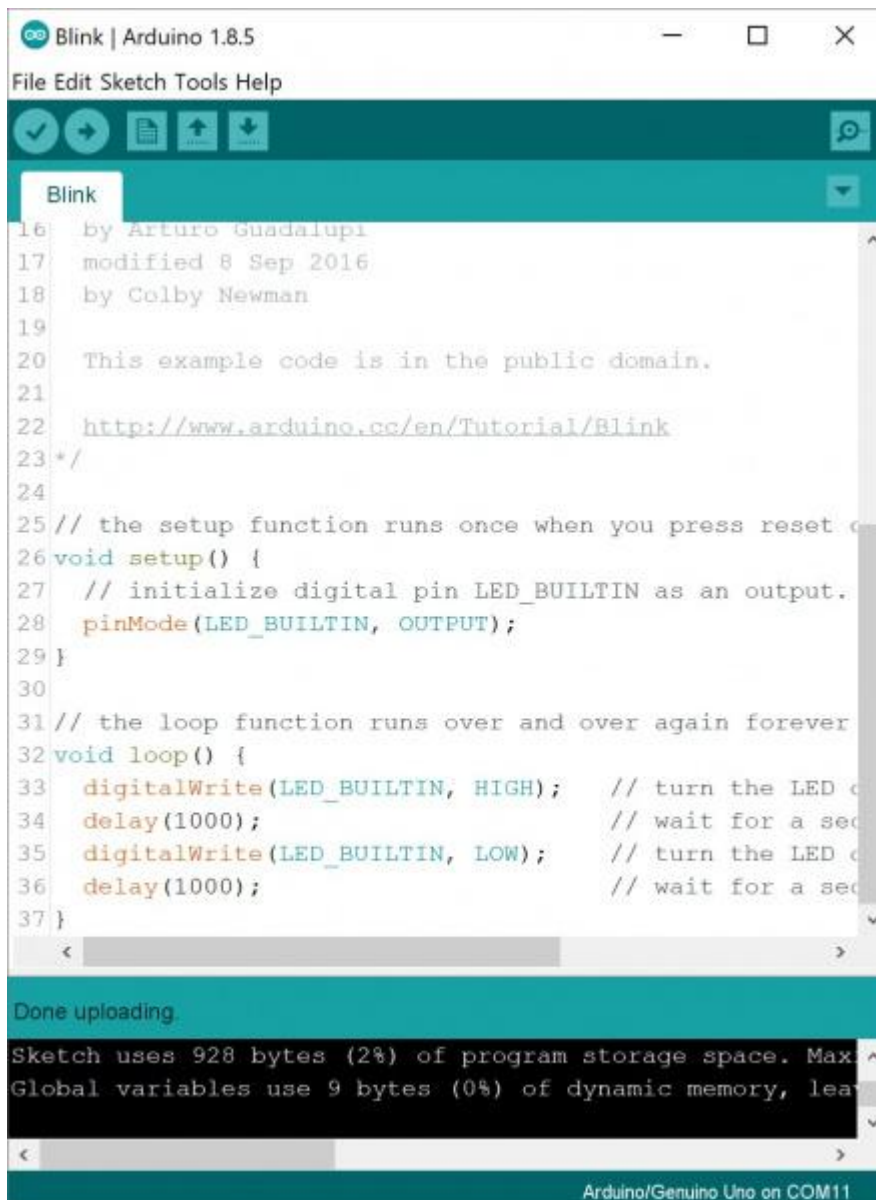


Note: If you're not sure which serial device is your Arduino, take a look at the available ports, then unplug your Arduino and look again. The one that disappeared is your Arduino.

With your Arduino board connected, and the Blink sketch open, press the **"Upload"** button.



After a second, you should see some LEDs flashing on your Arduino, followed by the message "**Done Uploading**" in the status bar of the Blink sketch.



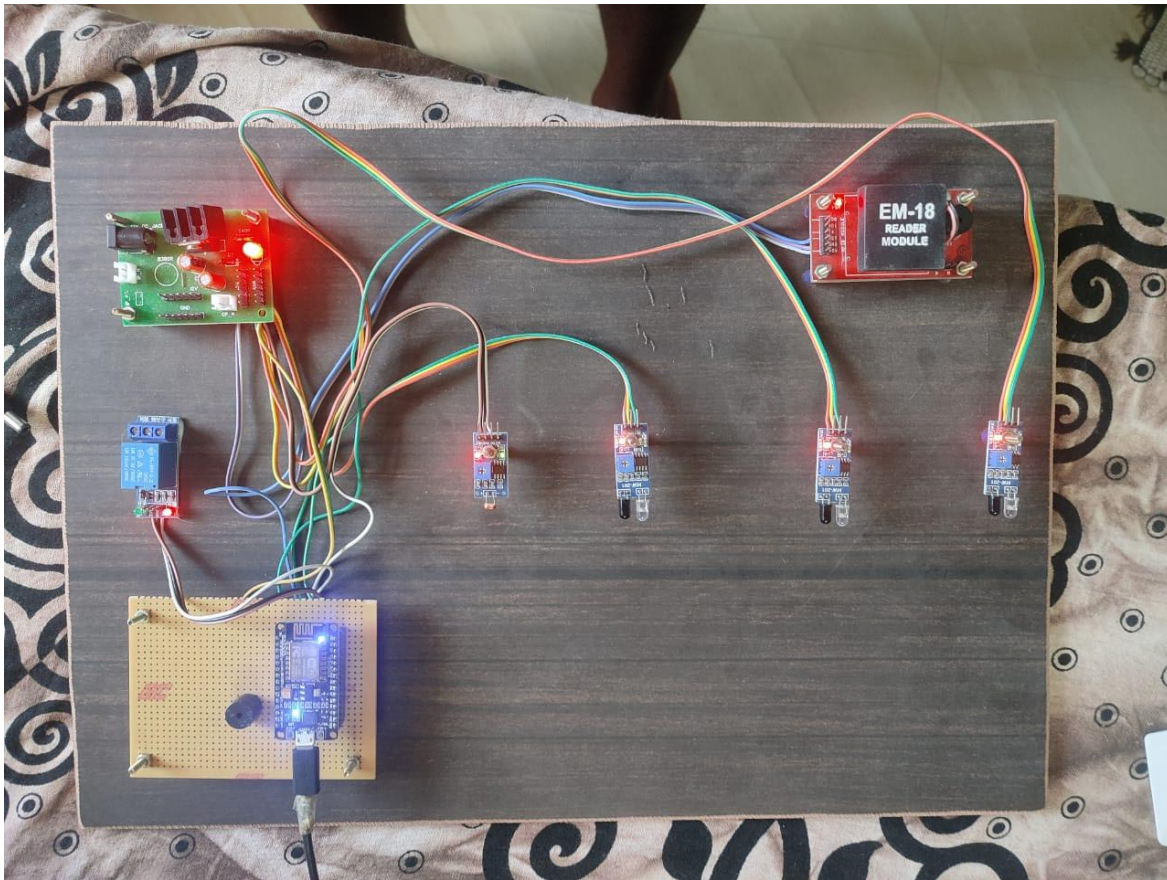
The screenshot shows the Arduino IDE interface. The title bar reads "Blink | Arduino 1.8.5". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for checking, running, saving, and uploading. The "Blink" tab is selected. The code editor displays the following text:

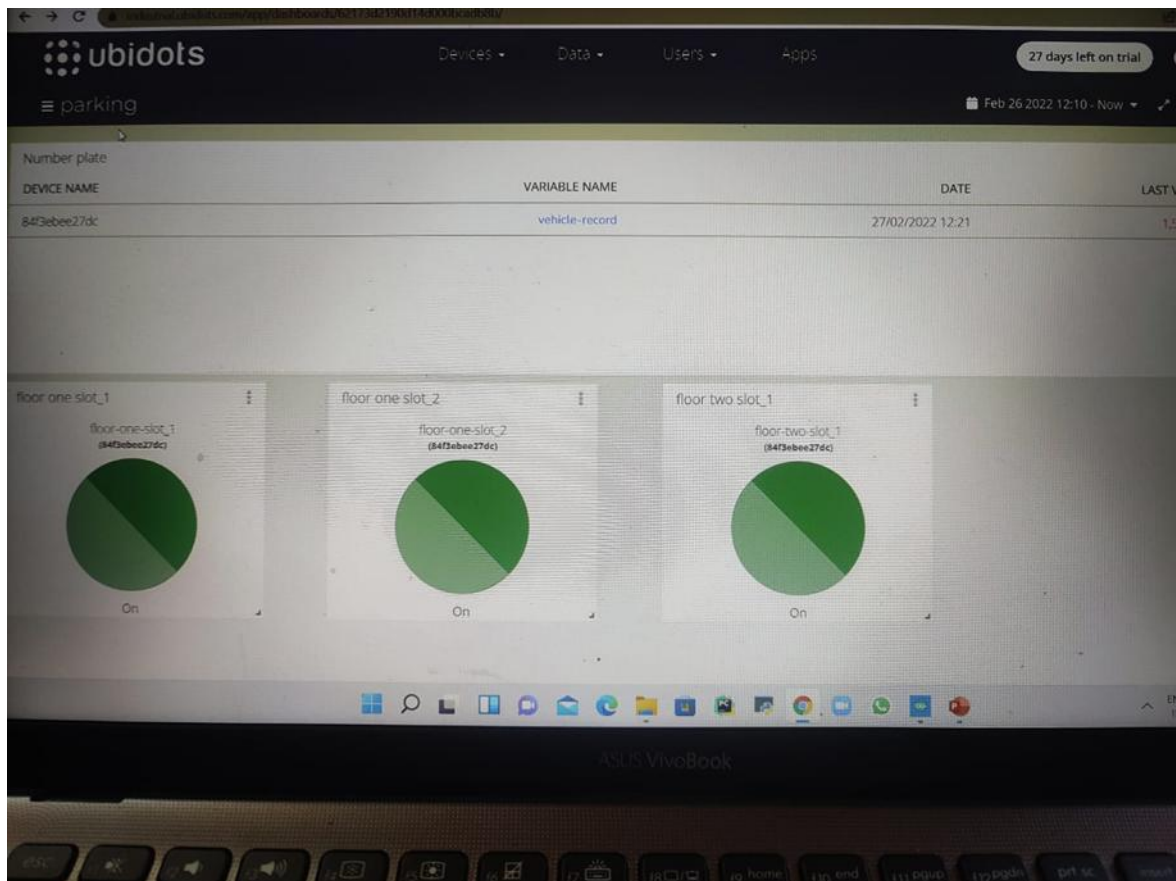
```
16 by Arturo Guadalupi
17 modified 8 Sep 2016
18 by Colby Newman
19
20 This example code is in the public domain.
21
22 http://www.arduino.cc/en/Tutorial/Blink
23 */
24
25 // the setup function runs once when you press reset
26 void setup() {
27   // initialize digital pin LED_BUILTIN as an output.
28   pinMode(LED_BUILTIN, OUTPUT);
29 }
30
31 // the loop function runs over and over again forever
32 void loop() {
33   digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
34   delay(1000);                      // wait for a second
35   digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by making the voltage LOW
36   delay(1000);                      // wait for a second
37 }
```

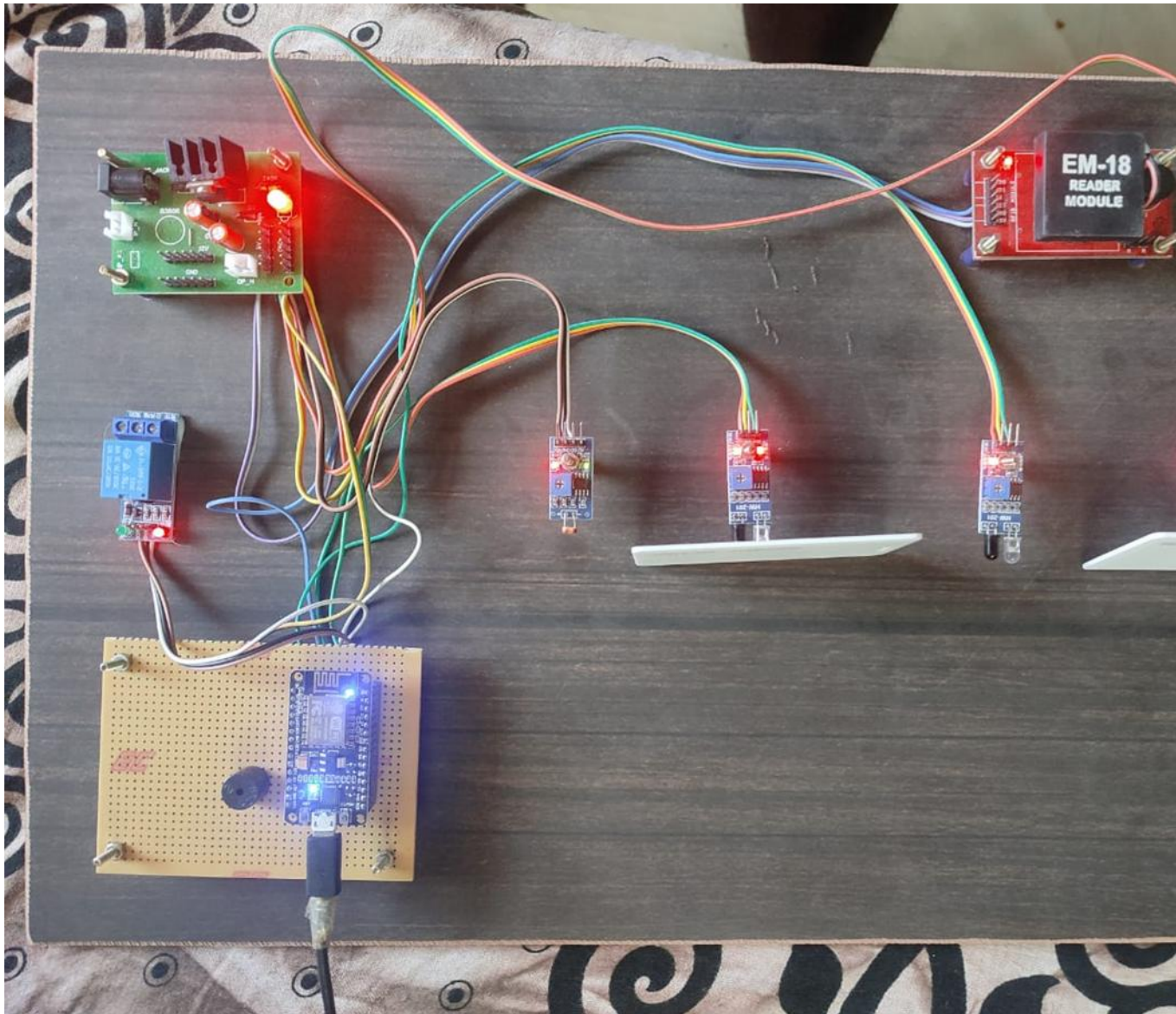
Below the code editor, a status bar indicates "Done uploading." and "Sketch uses 928 bytes (2%) of program storage space. Maximum is 32768 bytes. Global variables use 9 bytes (0%) of dynamic memory, leaving 2048 bytes free. Maximum is 2048 bytes." At the bottom right, it says "Arduino/Genuino Uno on COM11".

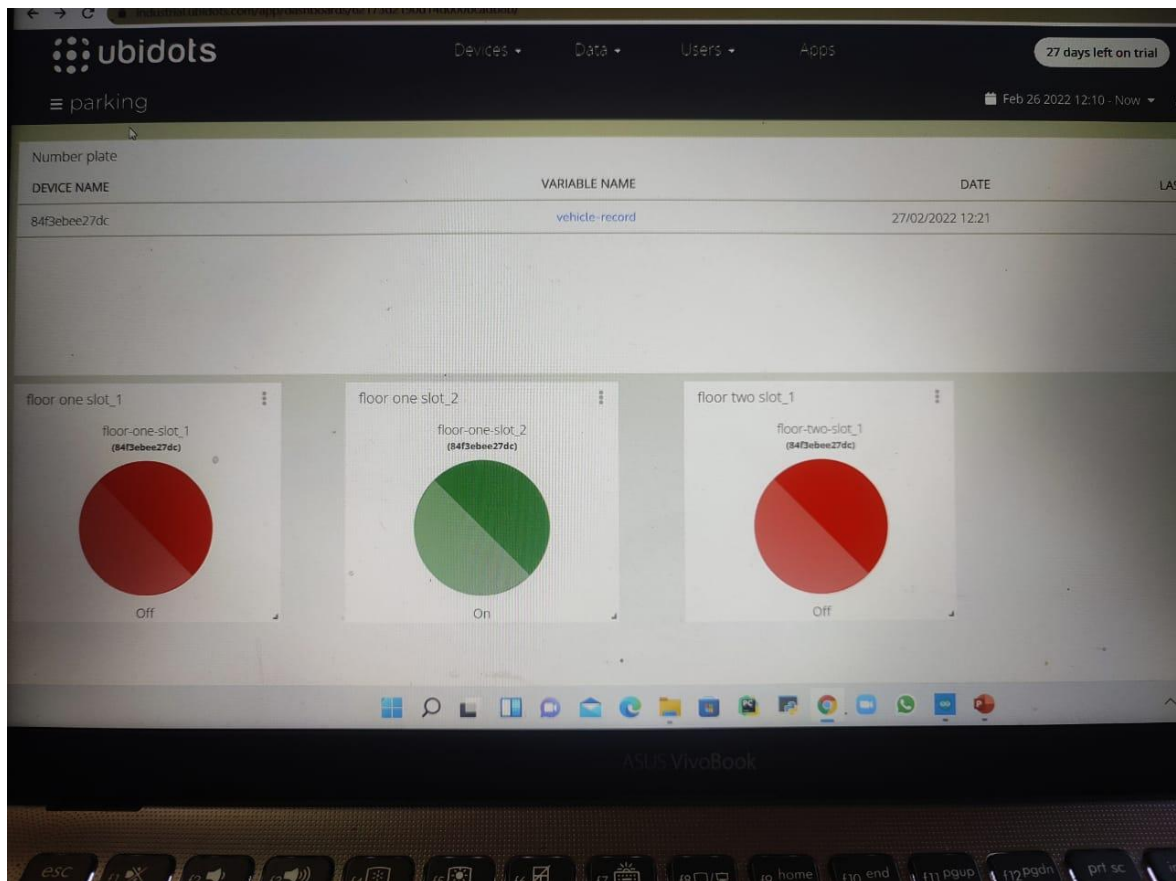
If everything worked, the onboard LED on your Arduino should now be blinking! You just programmed your first Arduino!

Note: Depending on the architecture and development board, the built-in LED may be defined on a different pin. You may need to adjust `LED_BUILTIN` or pin `13` to a different value before uploading.









CONCLUSION:

This designed automatic smart parking system which is simple, economic and provides effective solution to reduce carbon footprints in the atmosphere. It is well managed to access and map the status of parking slots from any remote location through web browser. Thus it reduces the risk of finding the parking slots in any parking area and also it eliminates unnecessary travelling of vehicles across the filled parking slots in a city. So it reduces time and it is cost effective also.

References:

- [1] <http://www.mdpi.com/journal/sensors> Sensors 2014, 14, 22372-22393; doi:10.3390/s141222372
- [2] Bilodeau, V.P. Intelligent Parking Technology Adoption. Ph.D. Thesis, University of Southern Queensland: Queensland, Australia, 2010.
- [3] Li, T.S.; Ying- Chieh, Y.; Jyun-Da, W.; Ming-Ying, H.; Chih-Yang, C. Multifunctional intelligent autonomous parking controllers for carlike mobile robots. IEEE Trans. Ind. Electron. 2010, 57, 1687–1700.
- [4] Faheem¹, S.A. Mahmud, G.M. Khan, M. Rahman and H. Zafar, ||A Survey of Intelligent Car Parking System||, October 2013
- [5] S. Alam, M. M. R. Chowdhury, and J. Noll, "Senaas: An event-driven sensor virtualization approach for internet of things cloud," in Networked Embedded Systems for Enterprise Applications (NESEA), 2010 IEEE International