Internet of Things (IoT)

HELLO!

I am Veneth

I am here because I love IoT.

You can contact me at sveneth23@gmail.com

1

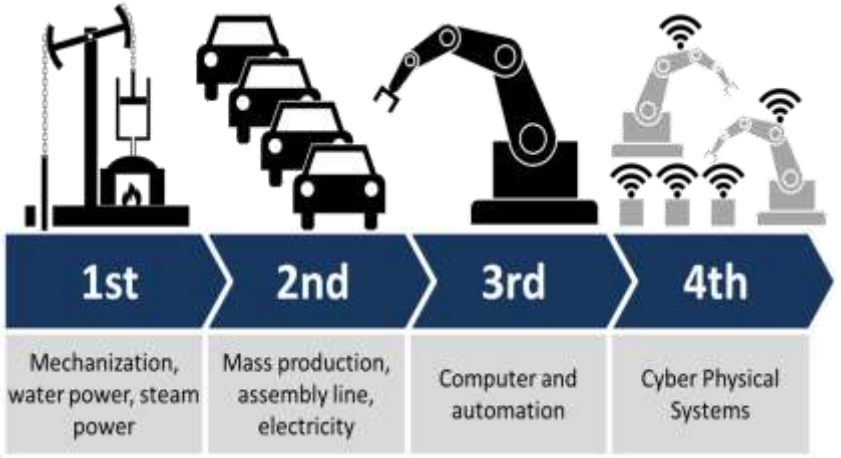
Introduction

What is IoT?

The **Internet of things (IoT)** is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and network connectivity which enables these objects to connect and exchange data.

Why IoT?

- The Internet of Things connects existing technology and existing machines to the cloud and helps to collect data that will enable cost-savings and streamline processes.



Industrial Revolution 4.0

2

Applications

Some Applications of IoT…

- Home automation
- Smart cities
- Smart energy meter
- Vehicle to Vehicle(V2V) communication
- Smart Grids
- Healthcare and Wearables

Imagine a smart traffic camera.


SMARTGRID.GOV

M2M Communication

Machine-to-machine communication, or M2M, is exactly as it sounds: two machines “communicating,” or exchanging data, without human interfacing or interaction. This includes serial connection, powerline connection (PLC)

M2M

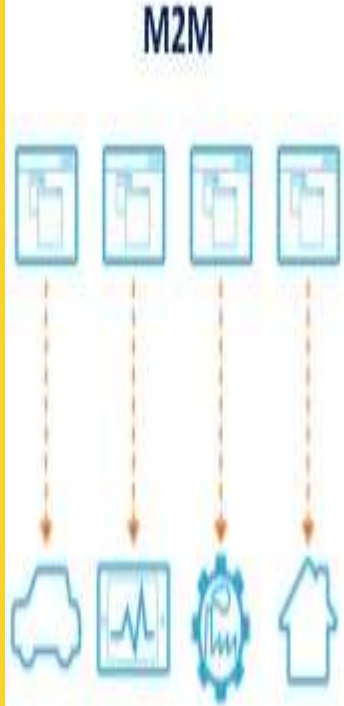
Machine to machine communication.

- Machines
- Maintenance
- Point to point
- Support and updates
- Hardware based

IOT

Internet of things

- Things (sensors)
- Integration / systems
- Cloud (HTTP)
- Big data
- Software based



3

Big Data

What is Big data?

- The **Big data** is the process of storing and analysing data to make some sense for the organization
- In simple terms, data which is very large in size and yet growing exponentially with time is called as Big data.

Why Bigdata?

- For any application that contains limited amount of data we normally use Sql/Postgresql/Oracle/MySQL, but what in case of large applications like Facebook,Google,Youtube? This data is so large and complex that none of the traditional data management system is able to store and process it.
- If we use traditional data processing applications (SQL/Oracle/MySQL) to handle it, it will lead to loss of efficiency.
- To overcome this problem, we use Big data.

Where does Big Data come from?

- Facebook generates 500+ TB data per day as people upload various images, videos, posts etc.
- Single airplane can generate 10+ TB of data in 30 minutes of a flight time.
- In CERN lab 10GB of data produced per second.
- In India the Aadhar data for each person is nearly 5MB.


Hello!, How you store such huge amount of data?

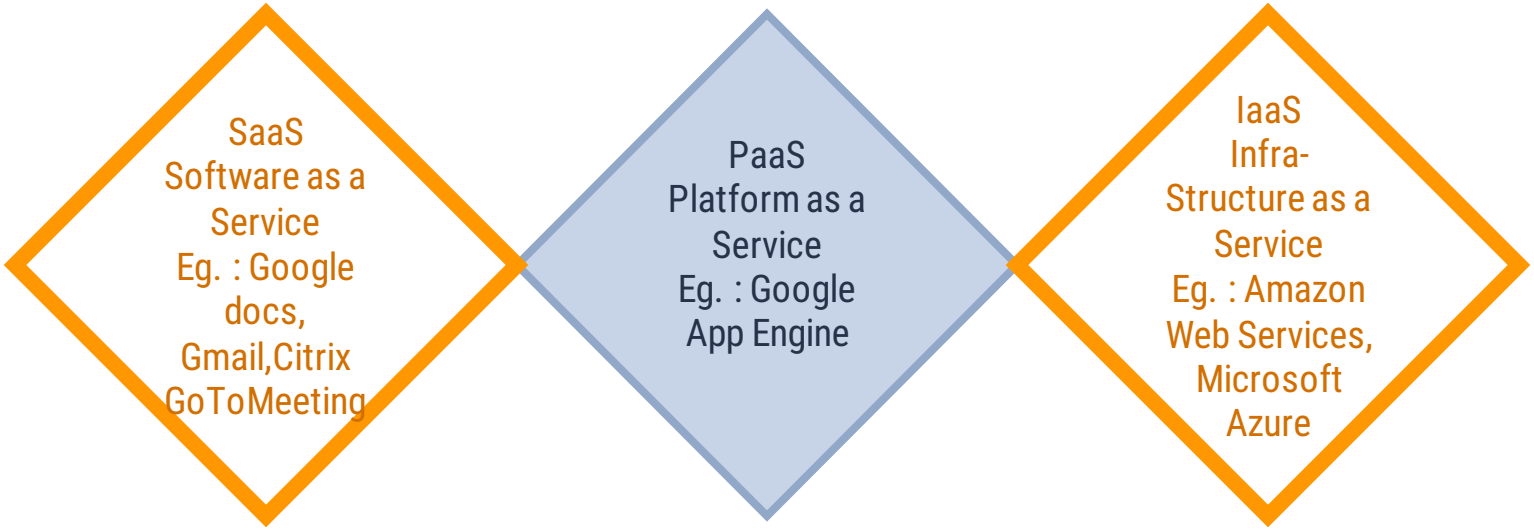
4

Cloud

Cloud Computing

- cloud computing is the delivery of computing services such as servers, storage, databases, networking, software, analytics and more over the Internet (“the cloud”).
- Companies offering these computing services are called cloud providers and typically charge for cloud computing services based on usage, similar to how you are billed for water or electricity at home.

Types of cloud services



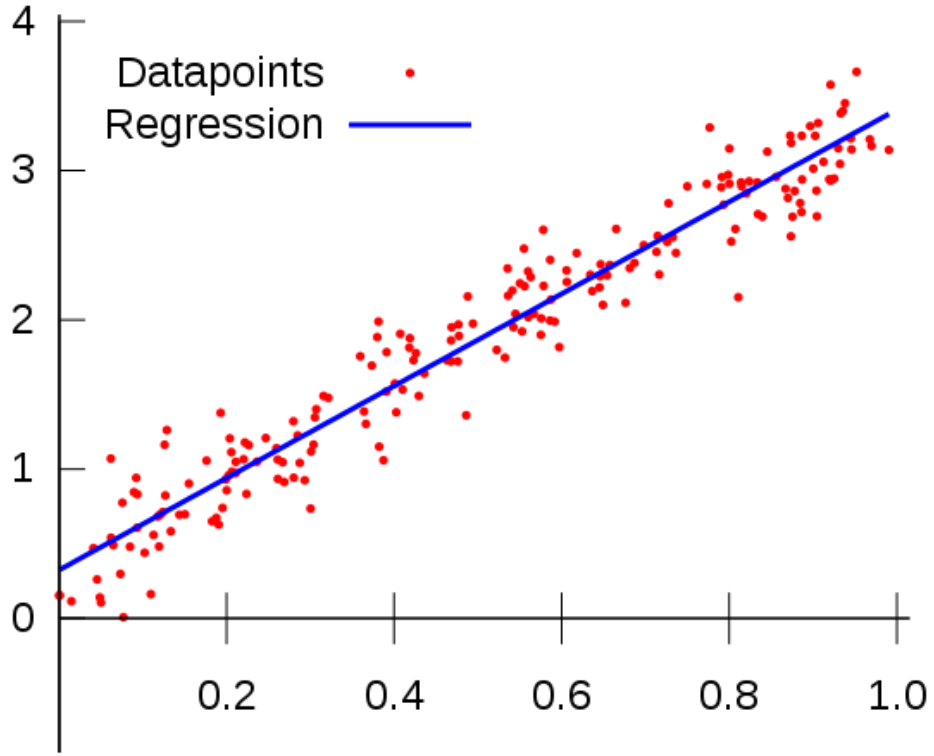

Hey! Why you store such huge amount of data?

5

Data Analytics

Data Analytics

- The main reason behind storing data is analysis.
- **Data analysis** is a process used to clean, transform and remodel data with a view to reach to a certain conclusion for a given situation.
- More accurate analyses leads to better decision making and better decision making leads to increase in efficiency and risk reduction.
- Predictive Analysis, Load Profile Analysis, etc.


Datapoints
Regression
4
3
2
1
0
0.2
0.4
0.6
0.8
1.0

How can 5 minus 2 equal 4?

Answer

IV

FIVE take away two letters F and E which leaves IV which is 4 in Roman Numerals.

6

IoT Protocols

What is protocols?

- A **protocol** is a set of rules that governs the communications between computers on a **network**. In order for two computers to talk to each other, they must be speaking the same language.
- Many different types of network protocols and standards are required to ensure that your computer can communicate with another computer located on the next desk or half-way around the world.

Contd.

- Rather than trying to fit all of the IoT Protocols on top of existing architecture models like **OSI Model**, we have broken the protocols into the following layers to provide some level of organization

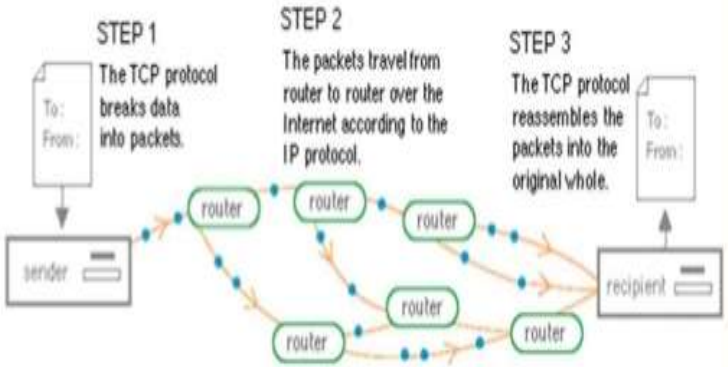
1. **Infrastructure** (ex: 6LowPAN, IPv4/IPv6, RPL)
2. **Identification** (ex: EPC, uCode, IPv6, URIs)
3. **Comms / Transport** (ex: Wifi, Bluetooth, LPWAN)
4. **Discovery** (ex: Physical Web, mDNS, DNS-SD)
5. **Data Protocols** (ex: MQTT, CoAP, AMQP, Websocket, Node)
6. **Device Management** (ex: TR-069, OMA-DM)
7. **Semantic** (ex: JSON-LD, Web Thing Model)
8. **Multi-layer Frameworks** (ex: Alljoyn, IoTivity, Weave, Homekit)

TCP

Transmission control Protocol (TCP) is a **communication protocol.**

To communicate over a network.

TCP divides message into stream of packets which are sent and then reassembled at the destination.



IP

- **Internet protocol (IP)** is **addressing protocol**.
- It is always used together with TCP.
- IP addresses of packet, routes them through different nodes and networks until it reaches its final destination.
- TCP/IP is perhaps the most used standard protocol for connecting computer networks.
- **Internet Address Protocol (IP Address)** is the address that identifies a computer on a network using TCP/IP.

	Internet Protocol version 4 (IPv4)	Internet Protocol version 6 (IPv6)
Deployed	1981	1999
Address Size	32-bit number	128-bit number
Address Format	Dotted Decimal Notation: 192.149.252.76	Hexadecimal Notation: 3FFE:F200:0234:AB00: 0123:4567:8901:ABCD
Prefix Notation	192.149.0.0/24	3FFE:F200:0234::/48
Number of Addresses	2^{32} = ~4,294,967,296	2^{128} = ~340,282,366, 920,938,463,463,374, 607,431,768,211,456

UDP

- It is simple Protocol.
- Throw some data to somebody.
- It is unsecure but there is speed.
- Mostly it is used for video conference for speed over reliability.
- Main disadvantage is duplicate is possible
- Used in Hangouts, Skype, etc.

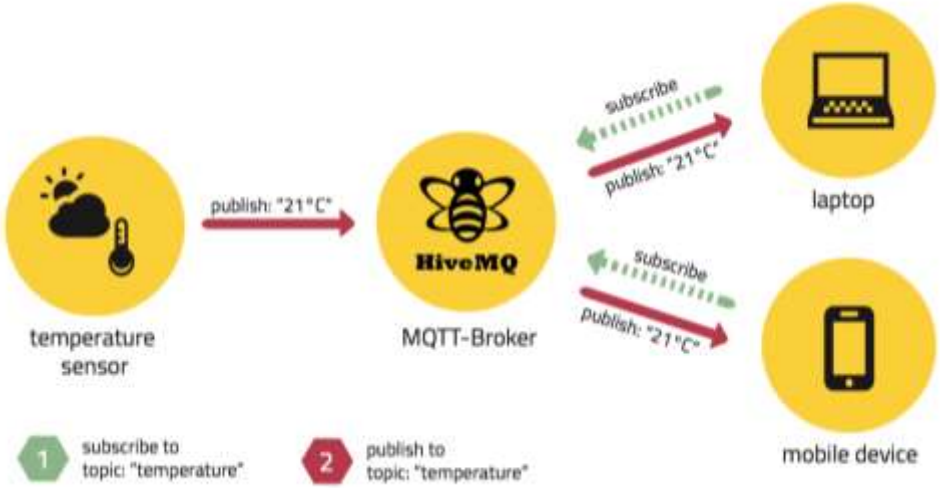
MQTT Protocol

- **Message Queue Telemetry Transport.**
- It is a publish-subscribe-based lightweight messaging protocol for use in conjunction with the TCP/IP protocol.
- MQTT was introduced by IBM in 1999 and standardized by OASIS in 2013.
- Designed to provide connectivity (mostly embedded) between applications and middle-wares on one side and networks and communications on the other side.

MQTT Methods

- Connect
- Disconnect
- Subscribe
- Unsubscribe
- Publish

- A message broker controls the publish-subscribe messaging pattern.
- A topic to which a client is subscribed is updated in the form of messages and distributed by the message broker.
- Some message brokers are Mosquitto, HiveMQ, mosca, etc.



Applications.

- ✓ **Facebook Messenger** uses MQTT for online chat.
- ✓ **Amazon Web Services** use Amazon IoT with MQTT.
- ✓ **Microsoft Azure** IoT Hub uses MQTT as its main protocol for telemetry messages.
- ✓ The **EVERYTHING IoT platform** uses MQTT as an M2M protocol for millions of connected products.
- ✓ **Adafruit** launched a free MQTT cloud service for IoT experimenters called Adafruit IO.

SMQTT

Secure MQTT is an extension of MQTT which uses encryption based on lightweight attribute based encryption

The main advantage of using such encryption is the broadcast encryption feature, in which one message is encrypted and delivered to multiple other nodes, which is quite common in IoT applications

In general, the algorithm consists of four main stages: setup, encryption, publish and decryption

- ✓ In the setup phase, the subscribers and publishers register themselves to the broker and get a master secret key according to their developer's choice of key generation algorithm.
- ✓ When the data is published, it is encrypted and published by the broker which sends it to the subscribers, which is finally decrypted at the subscriber end having the same master secret key.
- ✓ The key generation and encryption algorithms are not standardized.
- ✓ SMQTT is proposed only to enhance MQTT security features.

CoAP

- CoAP – **Constrained Application Protocol**
- **Web transfer protocol** for use with constrained nodes and networks.
- **Designed for Machine to Machine** (M2M) applications such as smart energy and building automation
- Based on **Request-Response model** between end-points

Features.

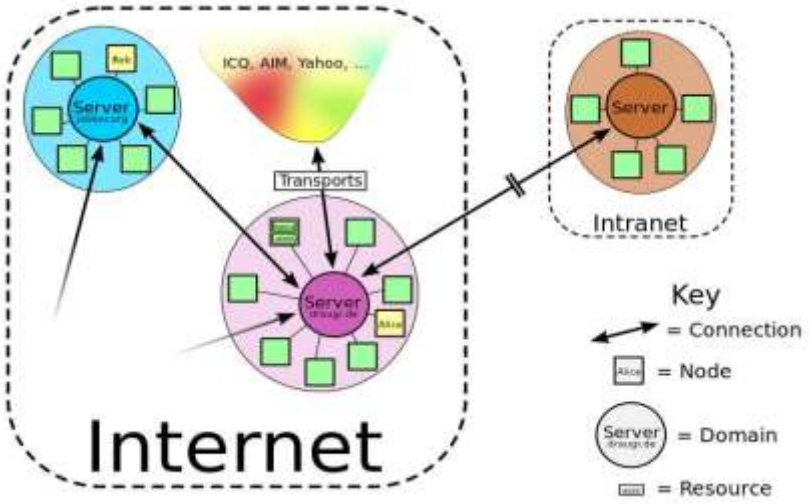
- ✓ Reduced overheads and parsing complexity.
- ✓ URL and content-type support.
- ✓ Support for the discovery of resources provided by known CoAP services.
- ✓ Simple subscription for a resource, and resulting push notifications.
- ✓ Simple caching based on maximum message age.

XMPP

- **XMPP – Extensible Messaging and Presence Protocol**
- A communication protocol for **message-oriented middleware** based on XML (Extensible Markup Language).
- Real-time exchange of structured data
- It is an open standard protocol
- XMPP uses a **client-server architecture**.

Contd..

- ✓ As the model is **decentralized**, no central server is required
- ✓ Well-suited for cloud computing where virtual machines, networks, and firewalls would otherwise present obstacles to alternative service discovery and presence-based solutions.
- ✓ Open means to support machine-to-machine or peer-to-peer communications across a diverse set of networks.

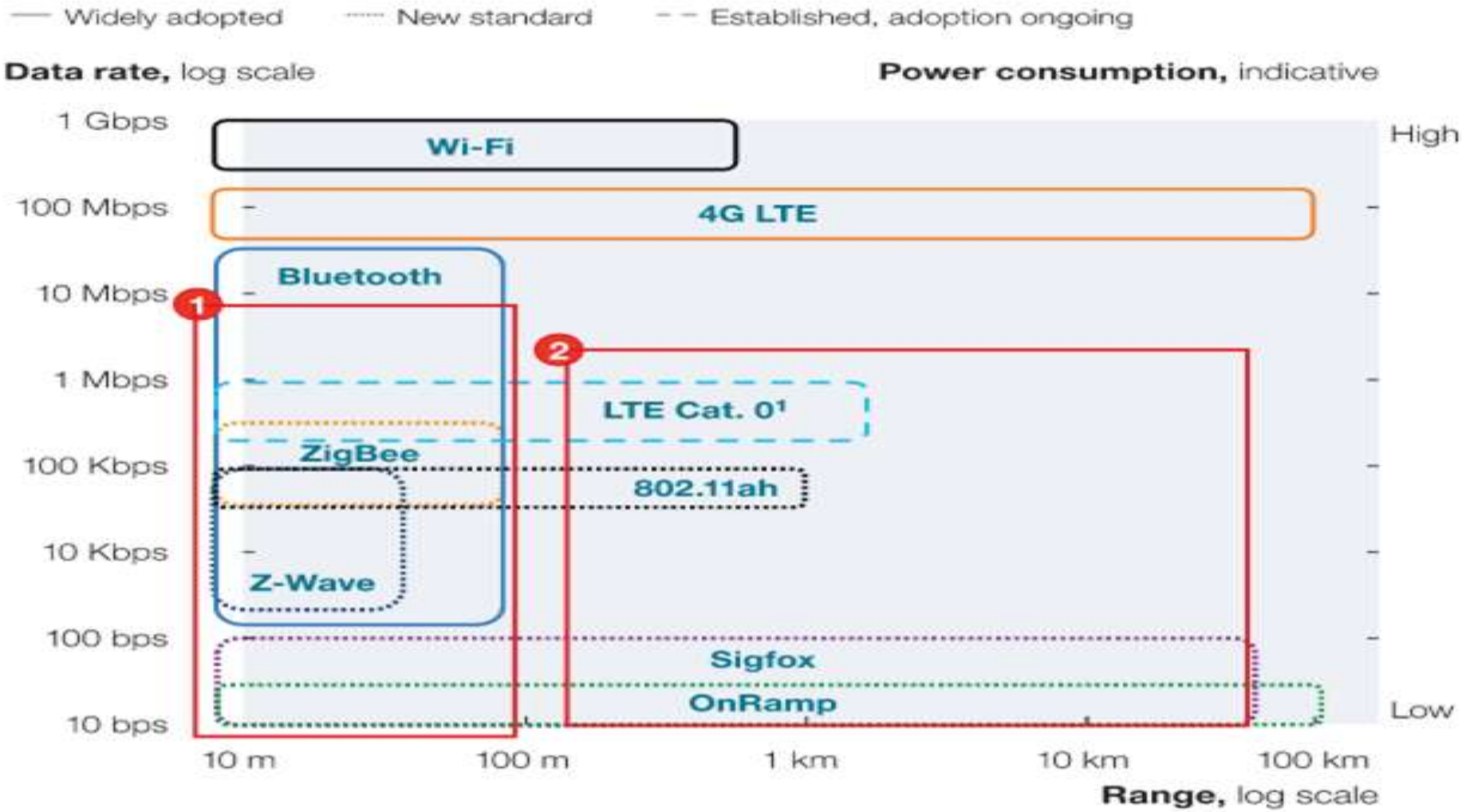


Applications

- ✓ File transfer
- ✓ Gaming
- ✓ Internet of Things applications
  - Smart grid
  - Social networking services

IoT Frinendly Protocols

- Hyper Text Transfer Protocol (HTTP)
- Secure Message Queue Telemetry Transport (SMQTT)
- Constrained Application Protocol (CoAP)
- Extensible Messaging and Presence Protocol (XMPP)
- Advanced Message Queuing Protocol (AMQP)
- Simple Text Oriented Messaging Protocol (STOMP)


Widely adopted
New standard
Established, adoption ongoing
Data rate, log scale
Power consumption, indicative
1 Gbps
100 Mbps
10 Mbps
1 Mbps
100 Kbps
10 Kbps
100 bps
10 bps
High
Low
Wi-Fi
4G LTE
Bluetooth
LTE Cat. 0¹
ZigBee
802.11ah
Z-Wave
Sigfox
OnRamp
1
2
10 m
100 m
1 km
10 km
100 km
Range, log scale

If the word MOM becomes WOW when you turn it upside down, what do you get when you spell it backwards?

Answer

ti

When you spell the word “it” backwards it becomes “ti”

How many numbers are there on a dice?

Answer

none

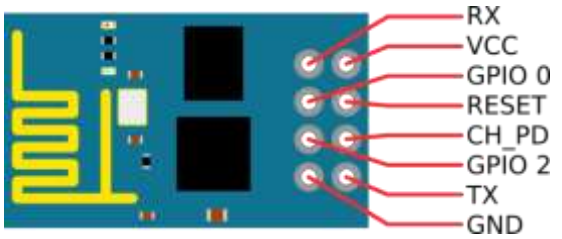
There are only dots on a dice.

7

Implementations

Arduino + ESP8266

- The **ESP8266** is a low-cost Wi-Fi chip with full TCP/IP stack
- This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using Hayes-style commands.



Pros and Cons

+ built-in Wi-Fi

+ smaller size

+ Arduino IDE supported

— Not all Arduino libraries supported and work

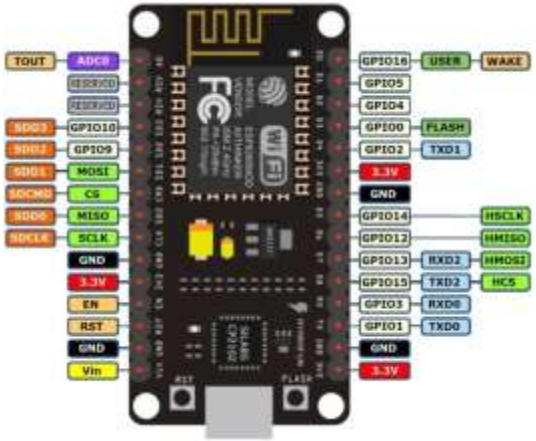
— Not as many GPIO pins

— ESP-1 does NOT have ANY analog pins!

— Only supports 3.3V power rail (5V devices would need level shifter)

*Note preferable

NodeMCU



- **NodeMCU** is an open source IoT platform.
- It includes firmware which runs on the ESP8266 Wi-FiSoC from Espressif Systems, and hardware which is based on the ESP-12 module.
- The NodeMCU board is based on an ESP8266-12 but features a built-in serial over USB interface and other amenities like 2 buttons and 2 LEDs.
- NodeMCU = Arduino + ESP8266

Features of NodeMCU

- 802.11 b/g/n
- Wi-Fi 2.4 GHz, support WPA/WPA2
- Integrated low power 32-bit MCU
- Operating Voltage: 3.0 ~ 3.6 V
- Operating Current: 80mA
- Integrated 10-bit ADC
- It can be programed by Arduino, Python & Lua script

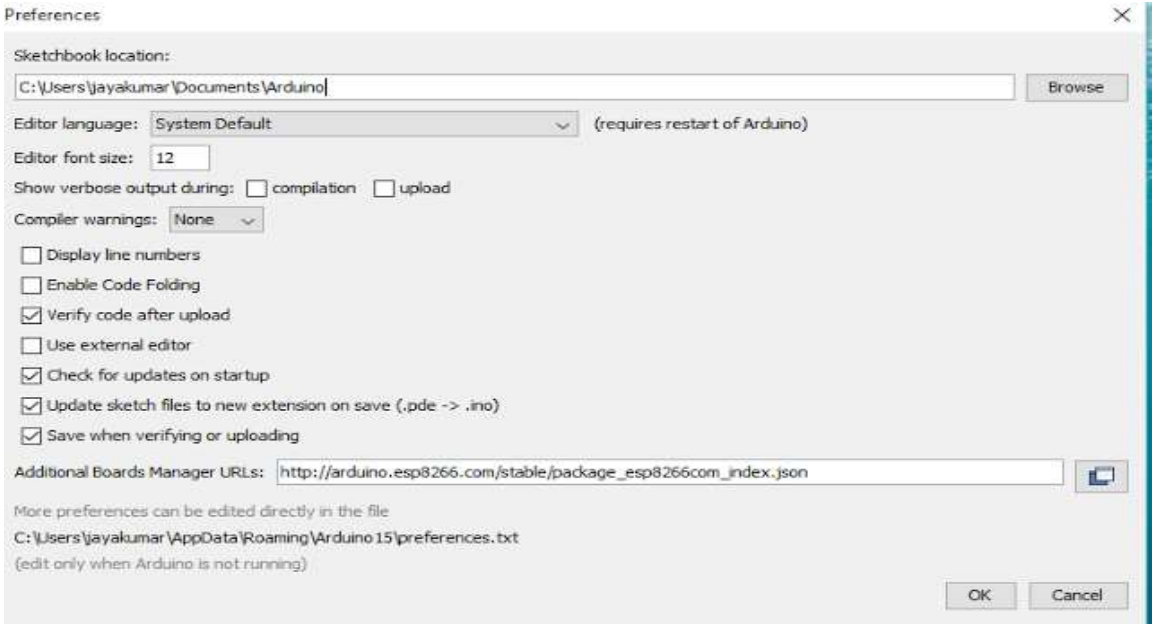
Pros and Cons

+ ESP is cheaper than Arduino or clones
+ ESP has built-in WiFi
+ ESP is smaller than most Arduinos
+ ESP can now be programmed with Arduino IDE so you can leverage your Arduino experience
+ Supports Python program and Lua script also.

— It is a 3.3V device, so it may not be compatible with some peripherals
— Less # of analogy pin
— Lack of official documentation

Get Started with NodeMCU

- Firstly open the Arduino IDE -> Go to files and click on the preference in the Arduino IDE
- copy the below code in the Additional boards Manager http://arduino.esp8266.com/stable/package_esp8266com_index.json click OK to close the preference Tab.



After completing the above steps , go to Tools and board, and then select board Manager

Navigate to esp8266 by esp8266 community and install the software for Arduino. Once all the above process been completed we are read to program our esp8266 with Arduino IDE.

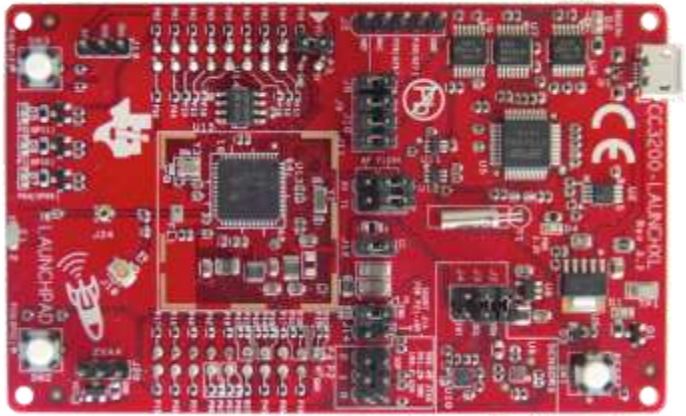
```
void setup() {
  pinMode(D4, OUTPUT);        // Initialize the LED_BUILTIN pin as an output
}
void loop() {
  digitalWrite(D4, LOW);
  delay(1000);
  digitalWrite(D4, HIGH);
  delay(1000);
}
```

Connecting to your WiFi

```
#include <ESP8266WiFi.h>
const char* ssid = " type your WiFi name";
const char* password = " type your password ";
 WiFiServer server(80);//Service Port
void WiFiEvent(WiFiEvent_t event) {
   Serial.printf("[WiFi-event] event: %d\n", event);
   switch(event) {
      case WIFI_EVENT_STAMODE_GOT_IP:
         Serial.println("WiFi connected");
         Serial.println("IP address: ");
         Serial.println(WiFi.localIP());
         break;
case WIFI_EVENT_STAMODE_DISCONNECTED:
         Serial.println("WiFi lost connection");
         break;
 }}
void setup() {
Serial.begin(115200);
WiFi.disconnect(true);
delay(1000);
WiFi.onEvent(WiFiEvent);
WiFi.begin(ssid, password);
Serial.println();
Serial.println();
Serial.println("Wait for WiFi... ");
Serial.println("");
Serial.println("WiFi connected");
}
```

CC3200

- The high performance CC3200 is the industry's first single-chip Microcontroller (MCU) with built-in Wi-Fi connectivity for the Launchpad Ecosystem!
- Created for the Internet of Things (IoT)
- The CC3200 device is a wireless MCU that integrates a high-performance ARM Cortex-M4 MCU allowing customers to develop an entire application with a single IC



Features

- ✓ 80MHz 32-bit ARM Cortex-M4 CC3200 MCU with 256 KB of RAM.
- ✓ 3 LEDs, one reset plus two buttons built-in debugger, two sensors — the TMP006 thermometer and the BMA222 accelerometer on the I²C bus and connectors for a 40-pin BoosterPack.
- ✓ Analog inputs are limited to 1.5 V. Higher voltage may *zombify* the MCU.
- ✓ Advance low power mode.
- ✓ Hibernate mode with RTC.

Pros & Cons

+ Very nice package with USB cable,

+ Large offer in IDEs: 3 professional IDEs from Texas Instruments, Arduino-like Energia, RTOS-based Energia MT, and embedXcode.

+ Very easy WiFi provisioning

— Analog inputs limited to 1.5 V

— No FPU (Floating Point Unit)

— Some pins aren't connected

— Multiple and confusing pins naming schemes

Raspberry Pi

- The Raspberry Pi is a credit-card-sized computer that plugs into your TV and a keyboard.
- It is a capable little computer which can be used in electronics projects, and for many of the things that your desktop PC does, like spreadsheets, word processing, browsing the internet, and playing games.
- We want to see it being used by adults and children all over the world to learn programming and digital making.
- It can run with Rasbian, Noobs, Windows 10, RISC OS, etc.

Features

- ✓ CPU: Quad-core 64-bit ARM Cortex A53 clocked at 1.2 GHz
- ✓ GPU: 400MHz VideoCore IV multimedia
- ✓ Memory: 1GB LPDDR2-900 SDRAM (i.e. 900MHz)
- ✓ USB ports: 4
- ✓ Video outputs: HDMI, composite video (PAL and NTSC) via 3.5 mm jack
- ✓ Network: 10/100Mbps Ethernet and 802.11n Wireless LAN
- ✓ Peripherals: 17 GPIO plus specific functions, and HAT ID bus
- ✓ Bluetooth: 4.1
- ✓ Power source: 5 V via MicroUSB or GPIO header

Pros

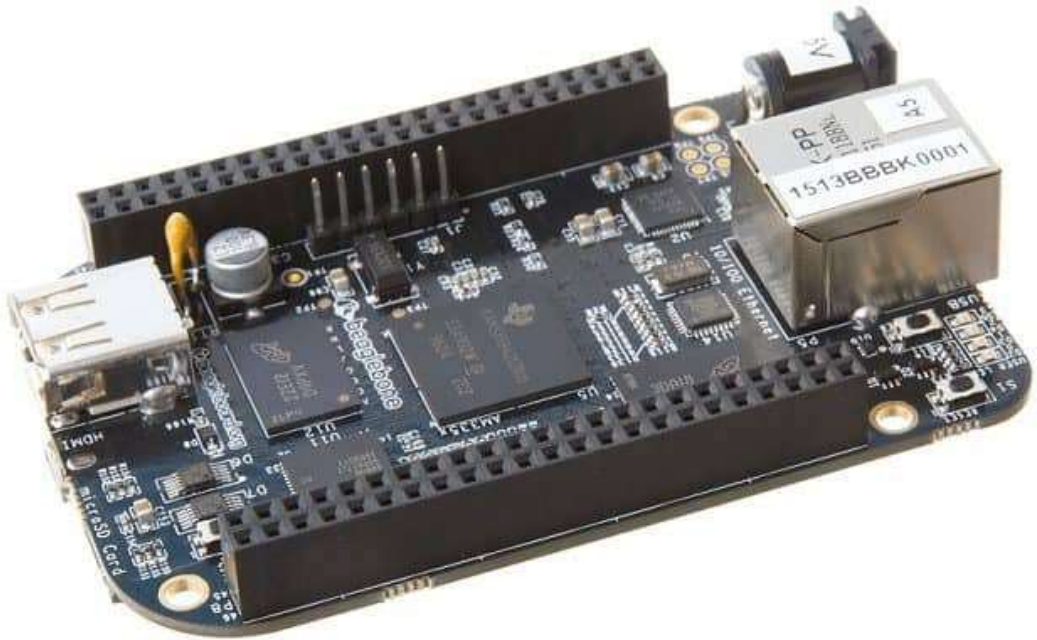
+ Super powerful with lot of memory and processing capabilities. Expandable memory.

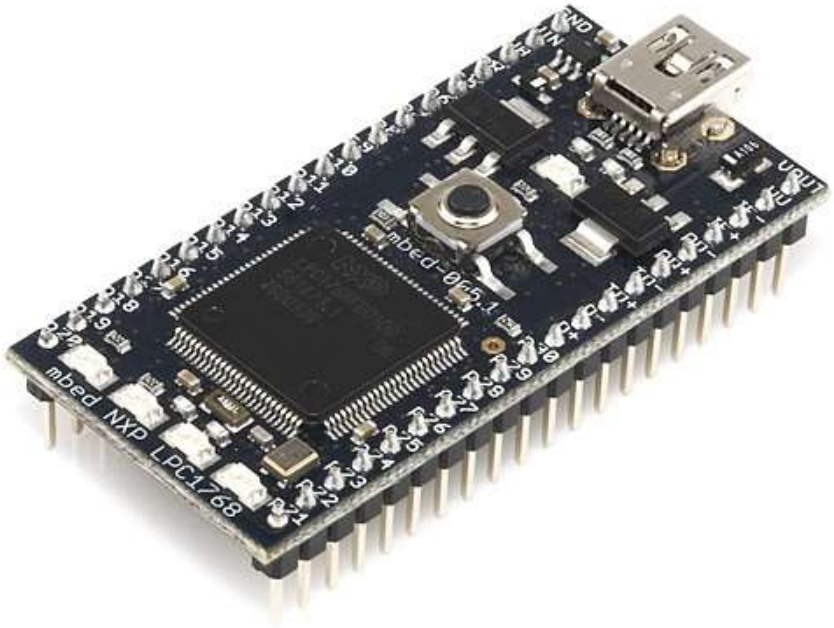
+ Linux based OS and now even Windows 10 can be run on top of it to make processing more user friendly.

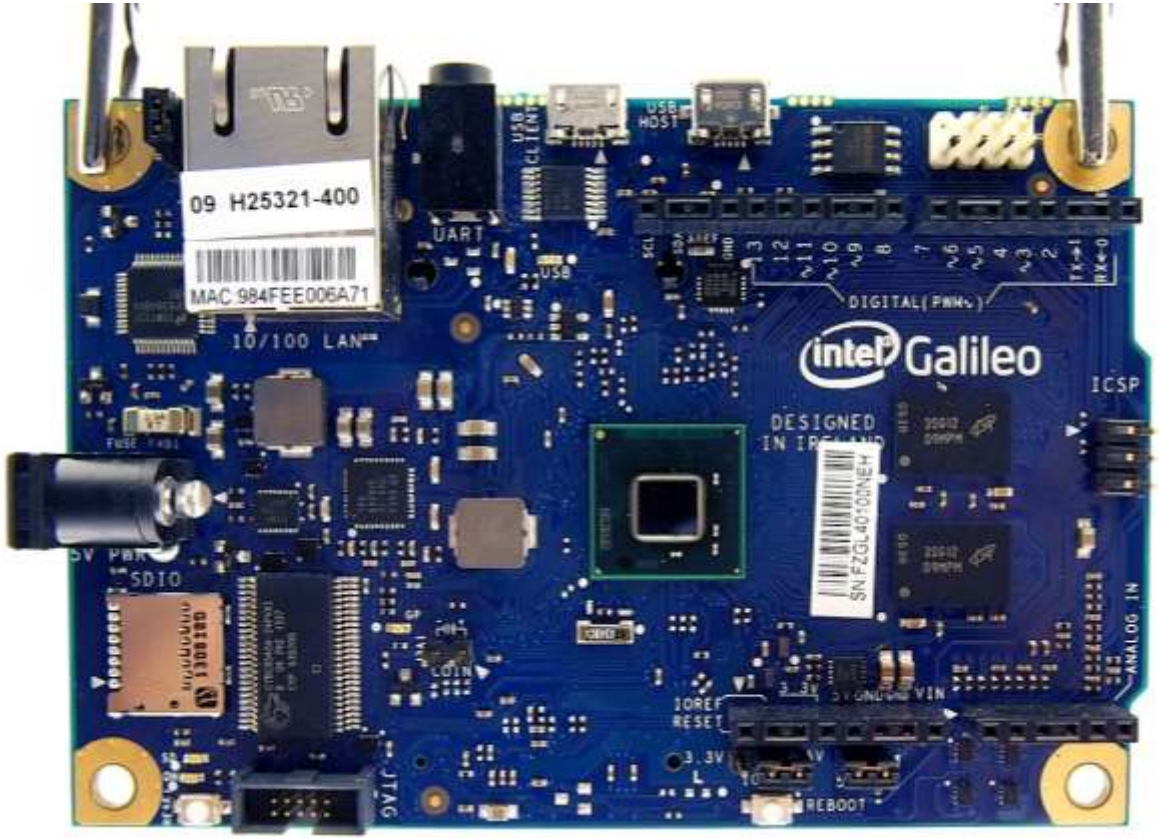
+ A lot of GPIOs available and the more the GPIOs, the more sensors you can interface.

+ Python, C, C++, Ruby, Go and many more can be used to program the Pi exactly the way you can code any computer.

+ In terms of cost, better than an Arduino with Ethernet shield.


1513BBBK0001
beagleboard


mbed-0965.1
mbed NXP LPC1768


09 H25321-400
MAC 984FEE006A71
10/100 LAN
UART
USB
USB HOST
DIGITAL(PWM~)
intel Galileo
DESIGNED
ICSP
SDIO
IOREF
RESET
3.3V
VIN
REBOOT
JTAG
ANALOG IN


intel Edison
What will you make?
ANALOG IN
POWER
DIGITAL(PWM~)

Word Facts

- Word with all Vowels without repeating any letters
- Largest Word with out Vowels
- Longest English Word that can be spelled without repeating any letters

8

Python

Python

- Intepreted general purpose high level language
- Supports multi paradigm such as OOPs,Procedural, Imperative, Functional
- Memory management is easy
- Objective : Teaching should be simple (Inspired from ABC programming language for educational purpose)
- Open source

Starting with Python

Simple printing statement at the python interpreter prompt,

```
>>>print(“Hi, Welcome to Python!”)
```

output: Hi, Welcome to Python!

To indicate different blocks of code, it follows rigid indentation.

```
if True:
    print (“Correct“)
else:
    print (“Error“)
```

- There are 5 data types in Python:
  - ✓ Numbers

    x, y, z = 10, 10.2, " Python "

  - ✓ String

```
x = 'This is Python'          >>This is Python
print (x)                     >>T
print (x[0])                  >>is
print (x[2:4])
```

- ✓ List
  x = [10, 10.2, 'python']
- ✓ Tuple
  x = (10, 10.2, ‘python’)
- ✓ Dictionary
  d = {1:‘item','k':2}

```
if (cond.):
        statement 1
        statement 2
elif (cond.):
        statement 1
        statement 2
else:
        statement 1
        statement 2
```

```
while (cond.):
        statement 1
        statement 2

x = [1,2,3,4]
for i in x:
        statement 1
        statement 2
```

- Defining a function
  - ✓ Without return value

```
def funct_name(arg1, arg2, arg3):          # Defining the function
        statement 1
        statement 2
```

  - ✓ With return value

```
def funct_name(arg1, arg2, arg3):          # Defining the function

        statement 1
        statement 2
        return x                           # Returning the value
```

- Calling a function

```
def example (str):
    print (str + “!”)

example(“Hi”)          # Calling the function
```

Output::
>>> Hi!

- Example showing function returning multiple values

```
def greater(x, y):
    if x > y:
        return x, y
        else:
        return y, x

val = greater(10, 100)
print(val)
```

Output:: (100,10)

- An error that is generated during execution of a program, is termed as exception.
- Syntax:

```
try:
    statements
except _Exception_:
    statements
else:
  statements
```

- Example:

```
while True:
        try:
                n = input ("Please enter an integer: ")
                n = int (n)
                break
        except ValueError:
                print ("No valid integer! ")
print (“It is an integer!")
```

File Read Write Operations

- Python allows you to read and write files
- No separate module or library required
- Three basic steps
  - Open a file
  - Read/Write
  - Close the file

- Mode: Four basic modes to open a file
  - r: read mode
  - w: write mode
  - a: append mode
  - r+: both read and write mode

Read from a file:

- read(): Reads from a file

```
file=open('data.txt', 'r')
file.read()
```

Write to a file:

- Write(): Writes to a file

```
file=open('data.txt', 'w')
file.write('writing to the file')
```

Modules

- Python has a way to put related code in a file and use that file in other Python files. It is called a module
- Grouping related code into module makes it easier to understand and use
- Module is nothing but a Python file with '.py' extension

Built in Module

```
import random

for i in range(1,10):
        val = random.randint(1,10)
        print (val)
```

User Defined Module

module1.py

```
def factorial(n):
    if n == 1:
        return 1
    else:
        return n * factorial(n-1)
```

module2.py

```
import module1
fact = module1.factorial(5)
print(fact)
```

Networking

- Python provides network services for client server model.
- Socket support in the operating system allows to implement clients and servers for both connection-oriented and connectionless protocols.
- Python has libraries that provide higher-level access to specific application-level network protocols

- Syntax for creating a socket:

  s = socket.socket (socket_family, socket_type, protocol=0)

socket_family – AF_UNIX or AF_INET

socket_type – SOCK_STREAM or SOCK_DGRAM

protocol – default ‘0’.

Simple Server Client Implemenation

```python
import socket
import sys

# Create a TCP/IP socket
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Bind the socket to the port
server_address = ('10.14.88.82', 2017)
print >>sys.stderr, 'starting up on %s port %s' % server_address
sock.bind(server_address)

# Listen for incoming connections
sock.listen(1)

connection, client_address = sock.accept()

#Receive command
data = connection.recv(1024)
print(data)
sock.close()
```

Server side code

Client side code

```python
import socket
import sys

# Create a TCP/IP socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

#Connect to Listener socket
client_socket.connect(("10.14.88.82", 2017))
print>>sys.stderr,'Connection Established'

#Send command
client_socket.send('Message to the server')
print('Data sent successfully')
```

Output

```
starting up on 10.14.88.82 port 2017
Message to the server
saswati@saswati-BK361AA-ACJ-CQ3236IX:~/Desktop$
```

```
Connection Established
Data sent successfully
saswati@saswati-BK361AA-ACJ-CQ3236IX:~/Desktop$
```

To Check Prime or Not

```
x = int (input("Enter a number: "))
def prime (num):
        if num > 1:
            for i in range(2,num):
                    if (num % i) == 0:
                            print (num,"is not a prime number")
                            print (i,“is a factor of”,num)
                            break
            else:
                    print(num,"is a prime number")
        else:
            print(num,"is not a prime number")
prime (x)
```

Tasks

- Odd or Even
- Generate Prime Numbers
- String Reverse
- Reverse the Number
- Factorial

If you throw a white hat with a blue ribbon into the Red Sea what does it become?

Answer

wet!

9

MicroPython

MicroPython

MicroPython is a software implementation of the Python 3 programming language, written in C, that is optimized to run on a microcontroller.

MicroPython is a full Python compiler and runtime that runs on the micro-controller hardware.

It ensures that the memory size/microcontroller performance is optimised and fit for purpose for the application it serves.

The simplicity of the Python programming language makes MicroPython an excellent choice for beginners who are new to programming and hardware.

```
import RPi.GPIO as GPIO          #GPIO library
import time
GPIO.setmode(GPIO.BOARD)         # Set the type of board for pin numbering
GPIO.setup(11, GPIO.OUT)         # Set GPIO pin 11as output pin
for i in range (0,5):
    GPIO.output(11,True)         # Turn on GPIO pin 11
    time.sleep(1)
    GPIO.output(11,False)
    time.sleep(2)
    GPIO.output(11,True)
GPIO.cleanup()
```

Turn LED on for 2 seconds and off for 1 second, loop forever

```
import RPi.GPIO as GPIO
import time
def main():
        GPIO.cleanup()
        GPIO.setmode(GPIO.BOARD) # to use Raspberry Pi board pin numbers
        GPIO.setup(11, GPIO.OUT) # set up GPIO output channel
        while True:
                GPIO.output(11, GPIO.LOW) # set RPi board pin 11 low. Turn off LED.
                time.sleep(1)
                GPIO.output(11, GPIO.HIGH) # set RPi board pin 11 high. Turn on LED.
                time.sleep(2)
main()
```

Check input using polling

```
input = True
prev_input = True
while True:
        input = GPIO.input(17)
        if (prev_input and (not input)):
                print("Button pressed")
        #update previous input
        prev_input = input
        #slight pause to debounce
        time.sleep(0.05)
```

S E Q U E N C _

What letter is needed to complete the word?

Answer

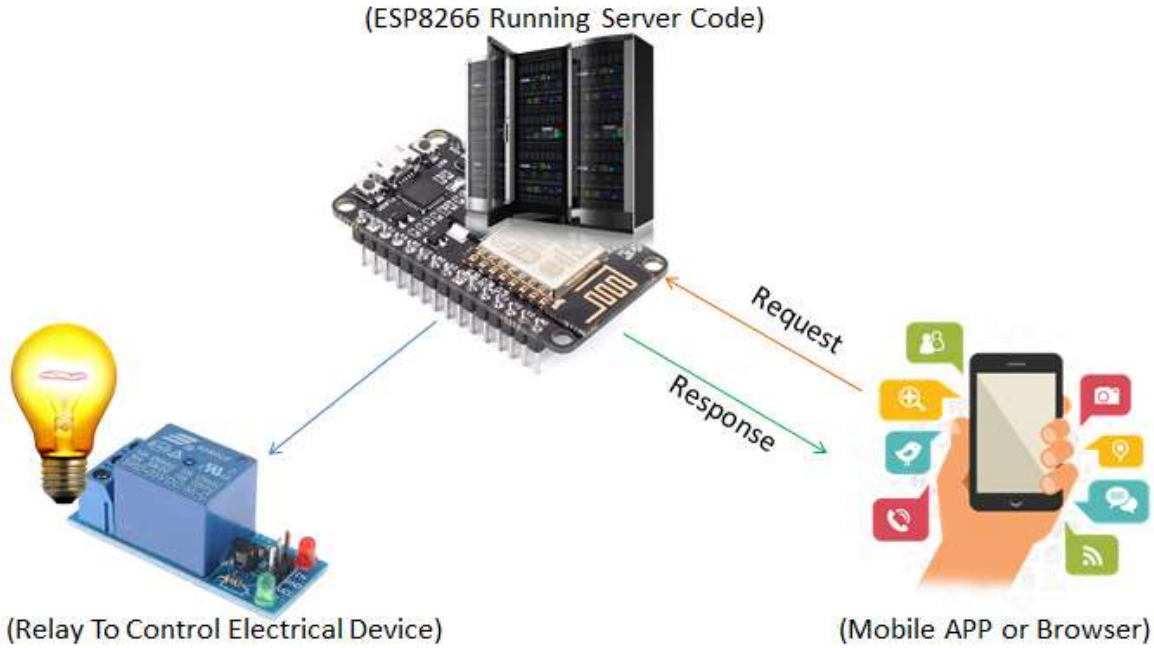
F

Place an F on the _ to form an E.

10

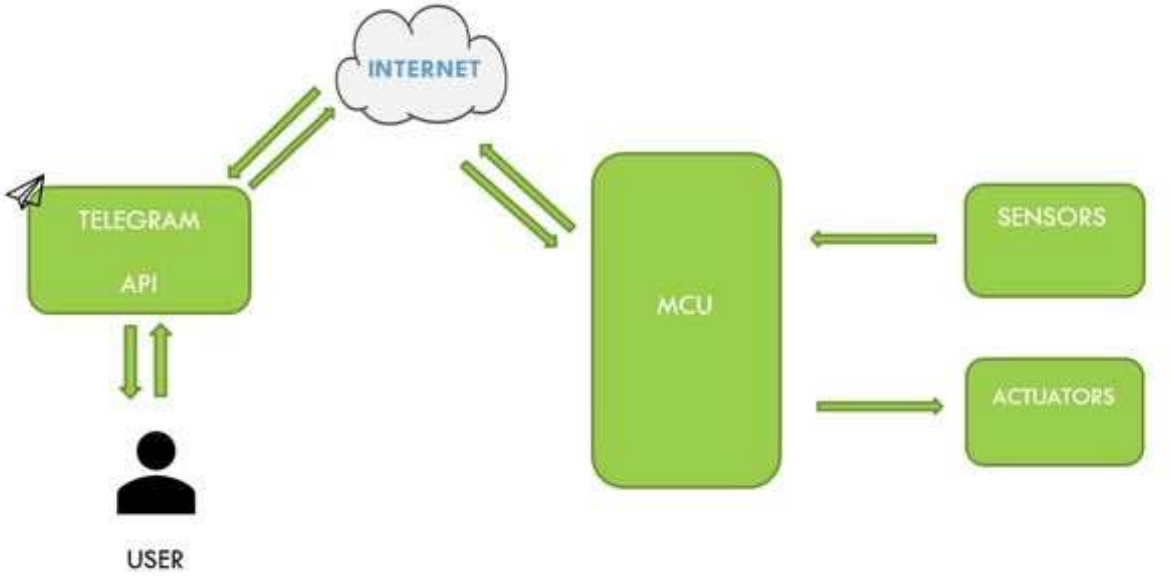
Implemention of Home Automation

By NodeMCU


(ESP8266 Running Server Code)
Request
Response
(Relay To Control Electrical Device)
(Mobile APP or Browser)

Code In Arduino IDE

- We can easily create the app by using MIT App Inventor **http://ai2.appinventor.mit.edu/**
- The app front end is created by just drag the required components.
- In text box the IP Address of NodeMCU is entered with the help of WebViewer (Globe Logo) we can control it like web page.


INTERNET
TELEGRAM
API
USER
MCU
SENSORS
ACTUATORS

(contd..)

This API allows you to connect bots to our system. **Telegram Bots** are special accounts that do not require an additional phone number to set up. These accounts serve as an interface for code running somewhere on your server.

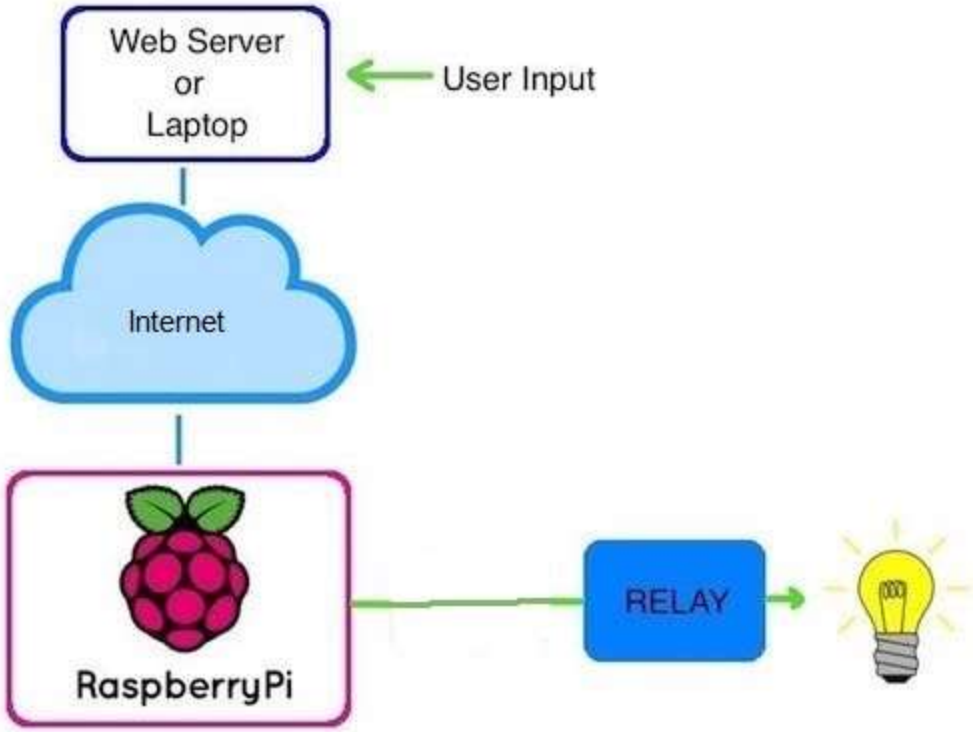
To use this, you don't need to know anything about how our MTProto encryption protocol works — our intermediary server will handle all encryption and communication with the Telegram API for you. You communicate with this server via a simple HTTPS-interface that offers a simplified version of the Telegram API.

Library file https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot

1.Register your Telegram bot and make a Telegram channel

Code In Arduino IDE

By RPi


Web Server
or
Laptop
User Input
Internet
RaspberryPi
RELAY

THANKS!

Any questions?

You can find me at

Veneth Sampath Kumar & sveneth23@gmail.com