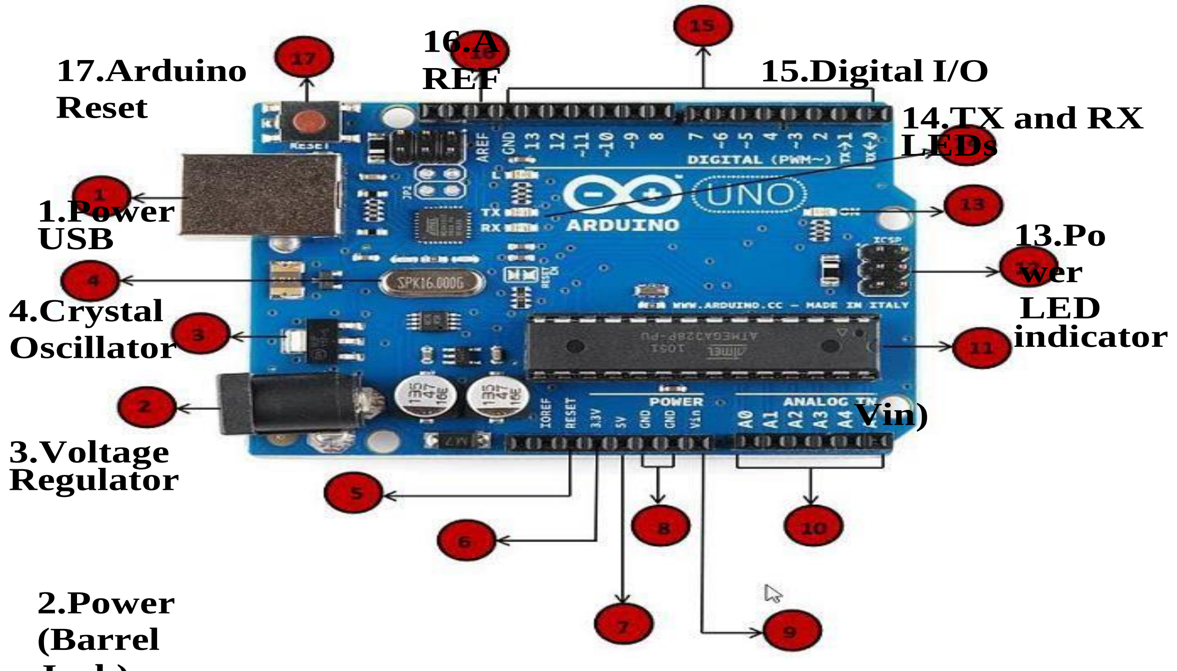
V: BUILDING IoT WITH ARDUINO &RASPBERRY PI

Building IOT with Arduino- Building IOT with RASPERRY PI- IoT Systems - Logical Design using Python – IoT Physical Devices & Endpoints - IoT Device -Building blocks - Pi - Raspberry Pi Interfaces - Pi Interfaces -Programming Raspberry Pi with Python - Other IoT Platforms - Arduino.

Building IOT with Arduino

Arduino Board:

- An Arduino is actually a microcontroller based kit.
- It is basically used in communications and in controlling or operating many devices.
- Arduino UNO board is the most popular board in the Arduino board family.
- In addition, it is the best board to get started with electronics and coding.
- Some boards look a bit different from the one given below, but most Arduino’s have majority of these components in common.
- It consists of two memories- Program memory and the data memory.
- The code is stored in the flash program memory, whereas the data is stored in the data memory.
- Arduino Uno consists of 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button



12.ICSP pin

11.Main microcontroller

10.Analog pins

Power USB

Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection (1).

Power (Barrel Jack)

Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack (2).

Voltage Regulator

The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.

Arduino Reset

You can reset your Arduino board, i.e., start your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET (5).

6,7,8,9.Pins (3.3, 5, GND, Vin)

- 3.3V (6) − Supply 3.3 output volt
- 5V (7) − Supply 5 output volt
- Most of the components used with Arduino board works fine with 3.3 volt and 5 volt.
- GND (8)(Ground) − There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- Vin (9) − This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.

Analog pins

The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.

Power LED indicator

This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.

TX and RX LEDs

On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.

Digital I/O

- The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM.

AREF

- AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Program an Arduino

- Arduino programs are written in the Arduino Integrated Development Environment (IDE).
- Arduino IDE is a special software running on your system that allows you to write sketches for different Arduino boards.
- The Arduino programming language is based on a very simple hardware programming language called processing, which is similar to the C language.
- After the sketch is written in the Arduino IDE, it should be uploaded on the Arduino board for execution.

Steps to program an Arduino

- A basic sketch consists of 3 parts

    1. Declaration of Variables
    2. Initialization: It is written in the setup () function.
    3. Control code: It is written in the loop () function.

- The sketch is saved with .ino extension. Any operations like verifying, opening a sketch, saving a sketch can be done using the buttons on the toolbar or using the tool menu.
- The sketch should be stored in the sketchbook directory.
- Chose the proper board from the tools menu and the serial port numbers.
- Click on the upload button or chose upload from the tools menu. Thus the code is uploaded onto the microcontroller.

Basic Adruino functions are:

- **digitalRead**(pin): Reads the digital value at the given pin.
- **digitalWrite**(pin, value): Writes the digital value to the given pin.
- **pinMode**(pin, mode): Sets the pin to input or output mode.
- **analogRead**(pin): Reads and returns the value.
- **analogWrite**(pin, value): Writes the value to that pin.

- **serial.begin**(baud rate): Sets the beginning of serial communication by setting the bit rate.

<u>Design your own Arduino to blink the LED</u>

Components required:

Arduino UNO R3 -1

Breadboard -1

Breadboard connectors -3

LED -1

1K resistor -1

Steps in building a breadboard connection:

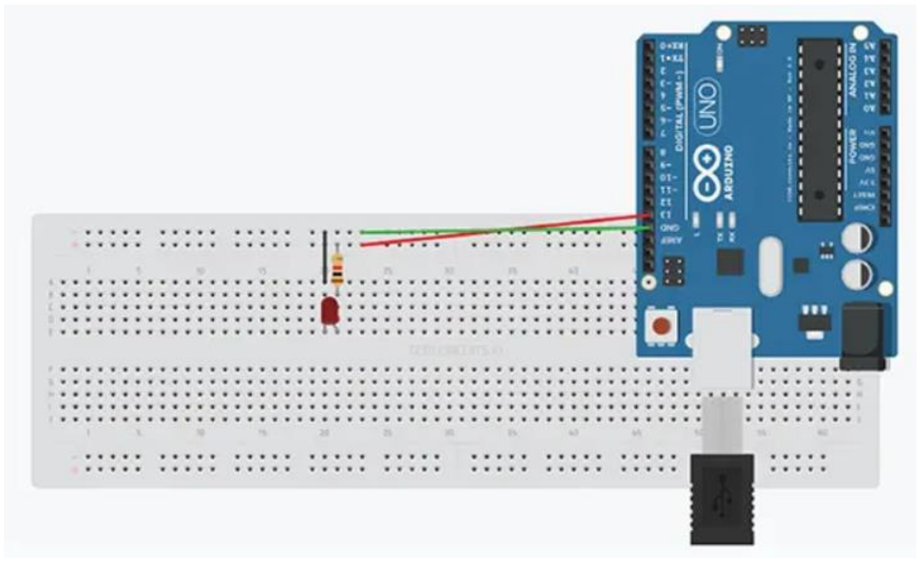
Step-1: Connect the Arduino to the Windows / Mac / Linux system via a USB cable

Step-2: Connect the 13th digital pin of Arduino to the positive power rail of the breadboard and GND to the negative

Step-3: Connect the positive power rail to the terminal strip via a 1K ohm resistor

Step-4: Fix the LED to the ports below the resistor connection in the terminal strip

Step-5: Close the circuit by connecting the cathode (the short chord) of the LED to the negative power strip of the breadboard



Arduino program for LED blink:

```
void setup ( )

{

pinMode (13, OUTPUT); //pin 13 is set as output pin

}

void loop( ) // The loop function runs again and again

{

digitalWrite (13,HIGH); // Turn ON the LED on pin 13

delay (1000); //Wait for 1sec

digitalWrite (13, LOW); //Turn OFF the LED on pin 13

}
```

<u>Advantages of Arduino Board</u>

1. It is inexpensive
2. It comes with an open source hardware feature which enables users to develop their own kit using already available one as a reference source.
3. The Arduino software is compatible with all types of operating systems like Windows, Linux, and Macintosh etc.
4. It also comes with open source software feature which enables experienced software developers to use the Arduino code to merge with the existing programming language

libraries and can be extended and modified.

5. It is easy to use for beginners.

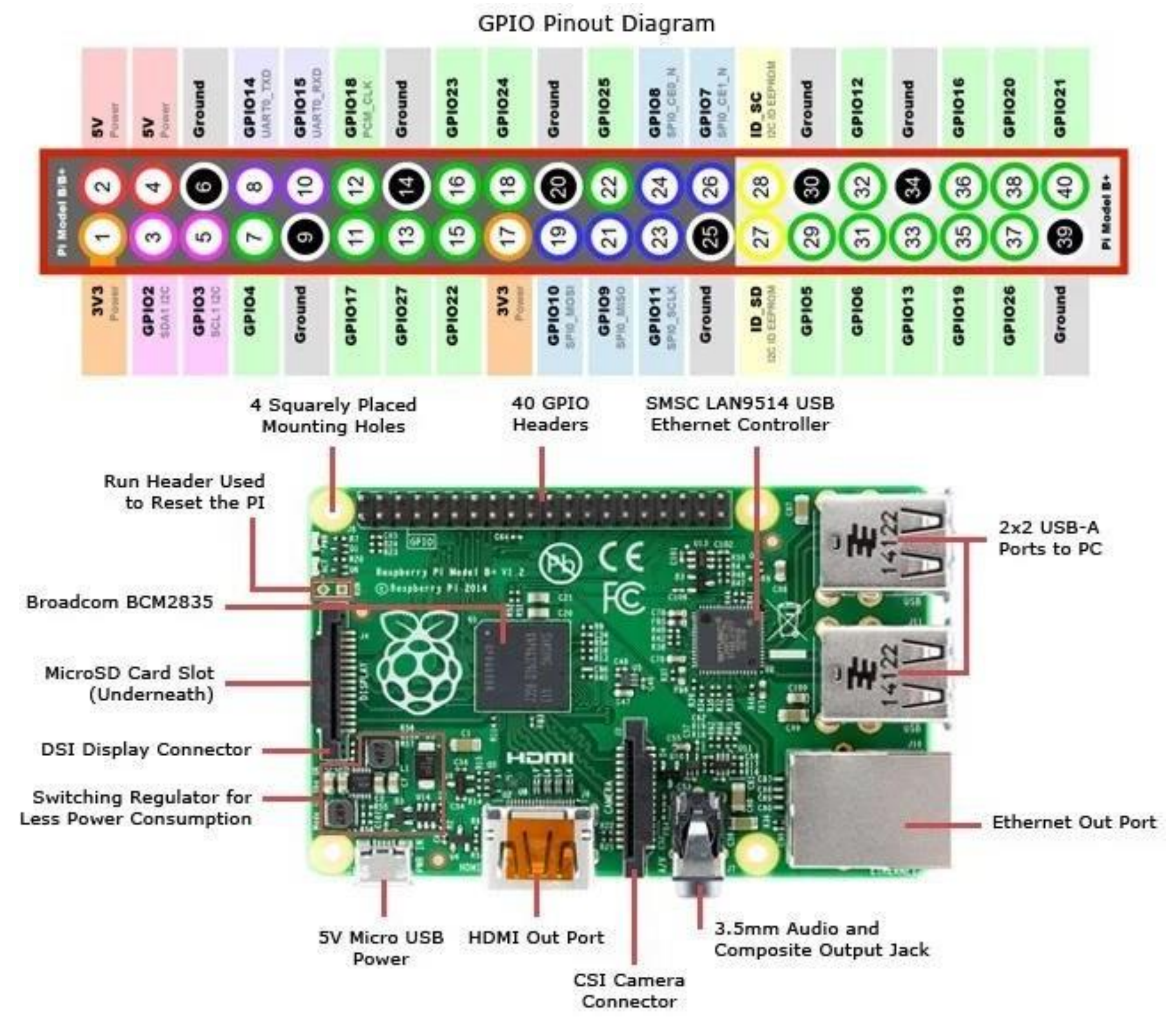
6. We can develop an Arduino based project which can be completely stand alone or projects which involve direct communication with the software loaded in the computer.

7. It comes with an easy provision of connecting with the CPU of the computer using serial communication over USB as it contains built in power and reset circuitry.

Building IOT with RASPERRY:

- The Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins that allow you to control electronic components for physical computing and explore the Internet of Things (IoT).
- Raspberry Pi was basically introduced in 2006.
- It is particularly designed for educational use and intended for Python.
- A Raspberry Pi is of small size i.e., of a credit card sized single board computer, which is developed in the United Kingdom(U.K) by a foundation called Raspberry Pi.

- There have been three generations of Raspberry Pis: Pi 1, Pi 2, and Pi 3
- The first generation of Raspberry (Pi 1) was released in the year 2012, that has two types of models namely model A and model B.
- Raspberry Pi can be plugged into a TV, computer monitor, and it uses a standard keyboard and mouse.
- It is user friendly as can be handled by all the age groups.
- It does everything you would expect a desktop computer to do like word-processing, browsing the internet spreadsheets, playing games to playing high definition videos.
- All models feature on a broadcom system on a chip (SOC), which includes chip graphics processing unit GPU(a Video Core IV), an ARM compatible and CPU.
- The CPU speed ranges from 700 MHz to 1.2 GHz for the Pi 3 and on board memory range from 256 MB to 1 GB RAM.
- An operating system is stored in the secured digital SD cards and program memory in either the MicroSDHC or SDHC sizes.
- Most boards have one to four USB slots, composite video output, HDMI and a 3.5 mm phone jack for audio. Some models have WiFi and Bluetooth.
- Several generations of Raspberry Pis have been released.
- All models feature a Broadcom system on a chip (SoC) with an integrated ARM-compatible central processing unit (CPU) and on-chip graphics processing unit (GPU).
- Processor speed ranges from 700 MHz to 1.4 GHz for the Pi 3 Model B+ or 1.5 GHz for the Pi 4; on-board memory ranges from 256 MB to 1 GB with up to 4 GB available on the Pi 4 random-access memory (RAM).
- Secure Digital (SD) cards in MicroSDHC form factor (SDHC on early models) are used to store the operating system and program memory.
- The boards have one to five USB ports. For video output, HDMI and composite video are supported, with a standard 3.5 mm tip-ring-sleeve jack for audio output.
- Lower-level output is provided by a number of GPIO pins, which support common protocols like I²C. The B-models have an 8P8C Ethernet port and the Pi 3 and Pi Zero W have on-board Wi-Fi and Bluetooth.



Components and Peripherals

- **Voltages:** Two 5V pins and two 3V3 pins are present on the board, as well as a number of ground pins (0V). The remaining pins are all general purpose 3V3 pins
- A GPIO pin designated as an output pin can be set to high (3V3) or low (0V). A GPIO pin designated as an input pin can be read as high (3V3) or low (0V).
- **Processor & RAM:** Raspberry based on ARM11 processor. Latest version supports 700MHz processor and 512MB SDRAM. The Central processing unit is the brain of the raspberry pi board and that is responsible for carrying out the instructions of the computer through logical and mathematical operations.
- **Ethernet:** The Ethernet port of the raspberry pi is the main gateway for communicating with additional devices. The raspberry pi Ethernet port is used to plug your home router to access the internet.

- **USB Ports:** It has 2 USB ports. USB port provide current upto 100mA. For connecting devices that draw current more than 100mA, an external USB powered hub is required.
- **Ethernet Port:** It has standard RJ45 Ethernet port. Connect Ethernet cable or USB wifi adapter to provide internet connectivity.
- **HDMI Output:** It supports both audio and video output. Connect raspberry Pi to monitor using HDMI cable.
- **Composite video Output:** Raspberry comes with a composite video output with an RCA jack that supports both PAL and NTSC video output.
- **Audio Output:** It has 3.5mm audio output jack. This audio jack is used for providing audio output to old television along with RCA jack for video.
- **GPIO Pins:** It has a number of general purpose input/output pins. These pins are used to connect other electronic components. For example, you can connect it to the temperature sensor to transmit digital data.
- **Display Serial Interface (DSI):** DSI interface are used to connect an LCD panel to Raspberry PI.
- **Cameral Serial Interface(CSI):** CSI interface are used to connect a camera module to Raspberry PI.
- **SD Card slot:** Raspberry does not have built in OS and storage. Plug in an SD card loaded with Linux to SD card slot.
- **Power Input:** Raspberry has a micro USP connector for power input.
- **Memory:** The raspberry pi model A board is designed with 256MB of SDRAM and model B is designed with 51MB.Raspberry pi is a small size PC compare with other PCs. The normal PCs RAM memory is available in gigabytes. But in raspberry pi board, the RAM memory is available more than 256MB or 512MB
- **Status LEDs:** Raspberry has 5 status LEDs.

Status LED	**Function**
ACT	SD card Access
PWR	3.3V power is present
FDX	Full duplex LAN Connected
LNK	Link/Network Activity
100	100 Mbit LAN connected

<u>Raspberry PI Interfaces:</u>

- It supports SPI, serial and I2C interfaces for data transfer.
- **Serial :** Serial Interface on Raspberry has receive(Rx) and Transmit(Tx) pins for communication with serial peripherals.
- **SPI:** Serial Peripheral Interface (SPI) is a synchronous serial data protocol used for communicating with one or more peripheral devices. In an SPI connection, there is one master device and one or more peripheral devices. There are 5 pins Raspberry for SPI interface.
  - **MISO(Master In Slave Out):** Master line for sending data to the peripherals.
  - **MOSI(Master Out Slave In):** Slave Line for sending data to the master.
  - **SCK(Serial Clock):** Clock generated by master to synchronize data transmission.
  - **CE0(Chip Enable 0):** To enable or disable devices.

- **CE1(Chip Enable 1):** To enable or disable devices.

- **I2C:** I2C Interface pins are used to connect hardware modules. I2C interface allows synchronous data transfer with two pins: SDA(data line) and SCL (Clock Line)

Features of Raspberry PPI

1. Where the system processing is huge. They can process high end programs for applications like Weather Station, Cloud server, gaming console etc. With **1.2GHz clock speed** and **1 GB RAM RASPBERRY PI** can perform all those advanced functions.

2. RASPBERRY PI 3 has wireless LAN and Bluetooth facility by which you can setup WIFI HOTSPOT for internet connectivity.

3. RASPBERRY PI had dedicated port for connecting touch LCD display which is a feature that completely omits the need of monitor.

4. RASPBERRY PI also has dedicated camera port so one can connect camera without any hassle to the PI board.

5. RASPBERRY PI also has PWM outputs for application use.

6. It supports HD steaming

Applications

- ✓ Hobby projects.
- ✓ Low cost PC/tablet/laptop
- ✓ IoT applications
- ✓ Media center
- ✓ Robotics
- ✓ Industrial/Home automation
- ✓ Server/cloud server
- ✓ Print server
- ✓ Security monitoring
- ✓ Web camera
- ✓ Gaming
- ✓ Wireless access point

5.2 IoT Systems - Logical Design using Python

5.4 IoT physical Devices & Endpoints

5.4.1 IoT Device

- Thing in Internet of Things(IoT) can be any object that has a unique identifier and which can send/receive data over a network.

- IoT devices are connected to the internet and send information about themselves or about their surroundings over a network or allow actuation upon the physical entities or environment around them remotely.
- Some example of IoT devices are:
    - A Home Automation device that allows remotely control and monitor the appliances.
    - An Industrial Machine which sends information about its operation and health monitoring data to a server.
    - A car sends information about its location to a cloud based service.
    - A wireless enabled wearable device that measures data about a person such as number of steps walked and send the data to a cloud based service.

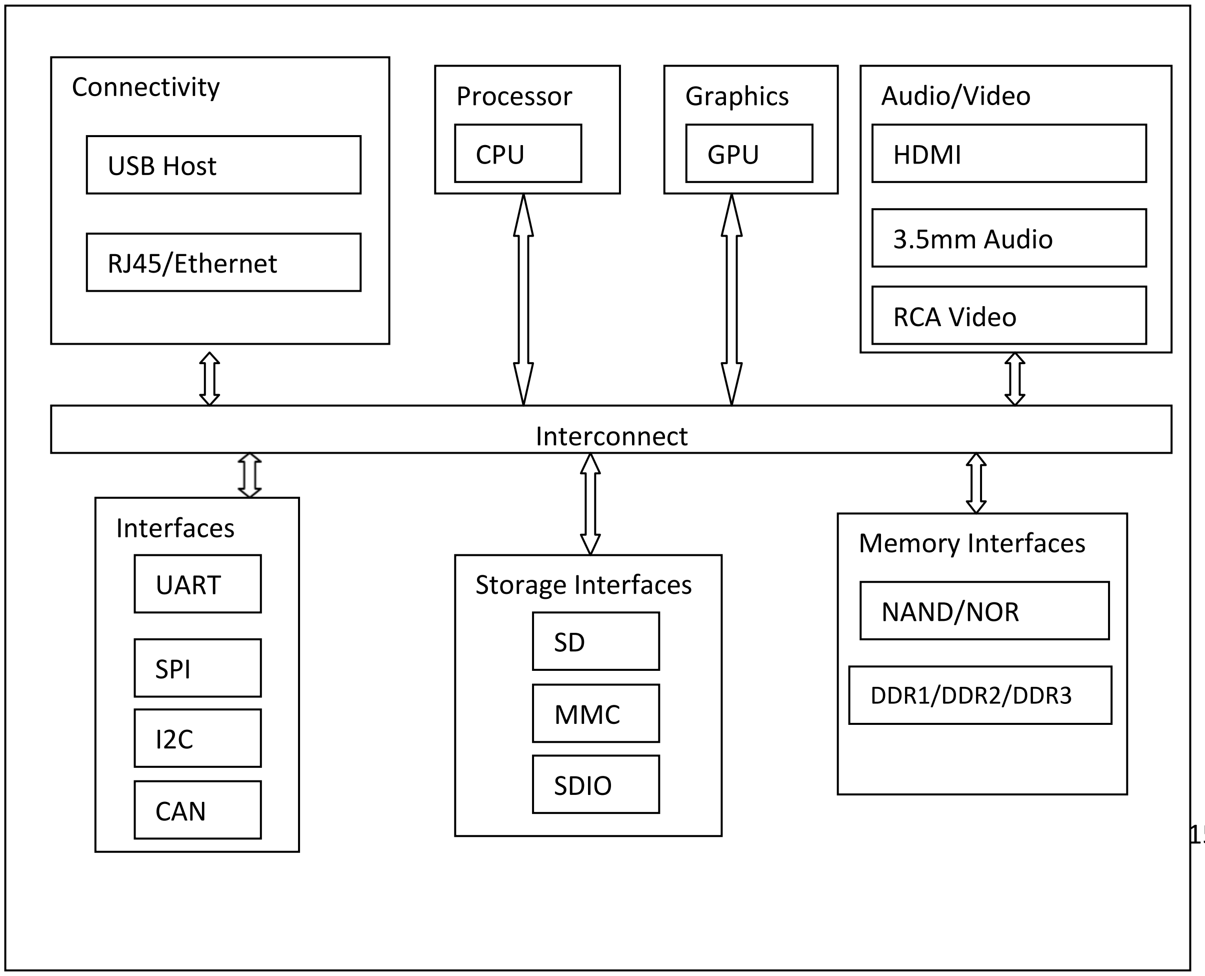
5.4.2 Basic Building block of an IoT device

- IoT stands for "Internet of Things," which means using the Internet to connect different things. IoT is an intersection between the physical and virtual worlds. It essentially maps virtual operations onto real interactions.

IoT device consists of a number of modules based on functional attributes:

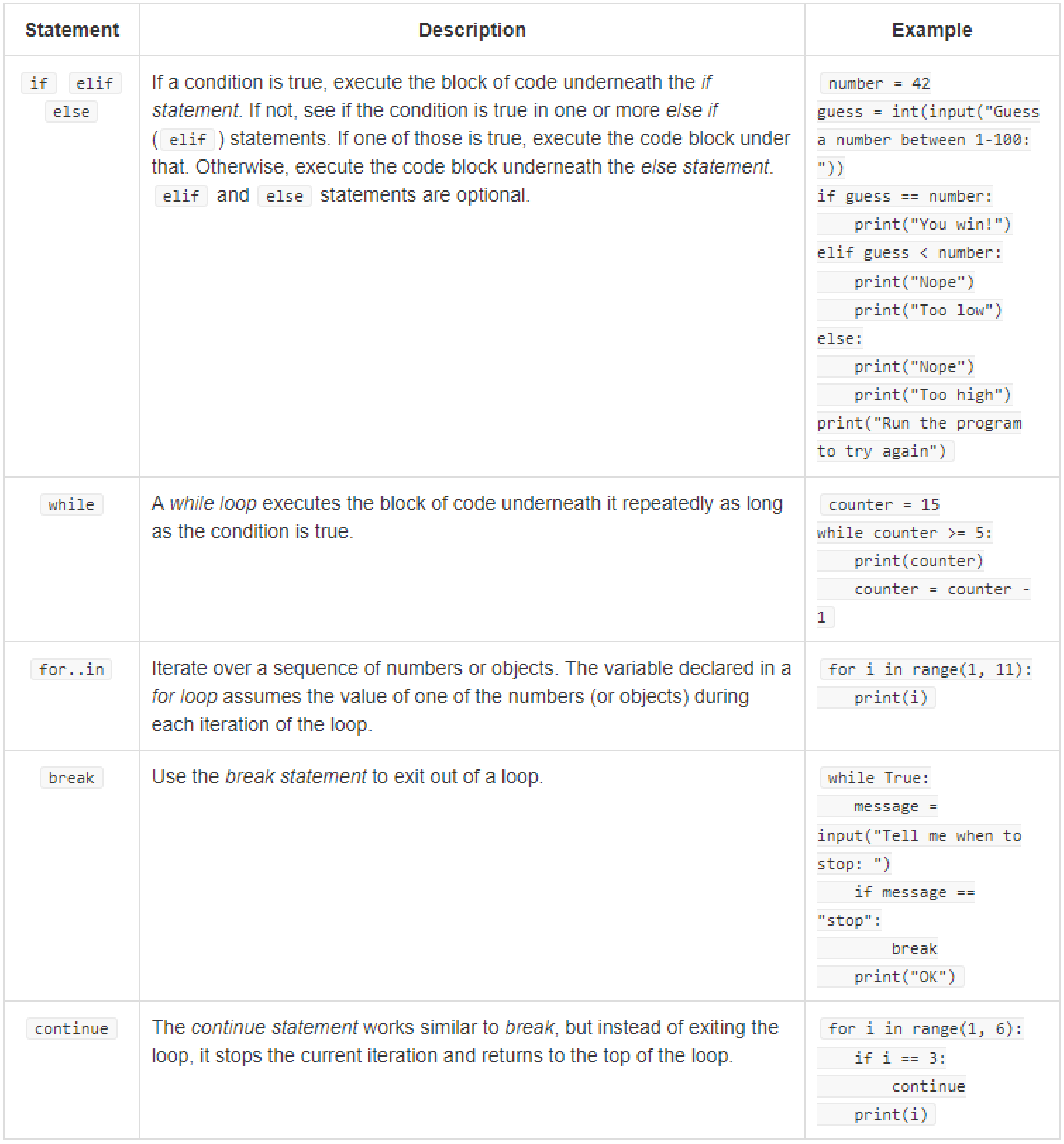
- **Sensors:** It can be either on board the IoT device or attached to the device. IoT device can collect various types of information from the onboard or attached sensors such as temperature, humidity, light intensity etc. The sensed information can be communicated either to other devices or cloud based server/storage. These form the front end of the IoT devices. These are the so called “Things” of the system. Their main purpose is to collect data from its surrounding (sensors) or give out data to its surrounding (actuators). These have to be uniquely identifiable devices with a unique IP address so that they can be easily identifiable over a large network. It should be able to collect real time data. Examples of sensors are: gas sensor, water quality sensor, moisture sensor etc.

- **Actuation:** IoT devices can have various types of actuators attached that allow taking actions upon the physical entities in the vicinity of the device. Example relay switch connected to an IoT device can turn appliances on/off based on the commands send to the device.
- **Processors:** Processors are the brain of the IoT system. Their main function is to process the data captured by the sensors and process them so as to extract the valuable data from the enormous amount of raw data collected. It gives intelligence to the data. Processors mostly work on real-time basis and can be easily controlled by applications. These are also responsible for securing the data – that is performing encryption and decryption of data.
- **Communication:** It is responsible for sending collected data to other devices or cloud based servers/storage and receiving data from other devices and commands from remote applications. Gateways are responsible for routing the processed data and send it to proper locations for its (data) proper utilization. Gateway helps in to and fro communication of the data. It provides network connectivity to the data. Network connectivity is essential for any IoT system to communicate. LAN, WAN, PAN etc are examples of network gateways.
- **Analysis & Processing:** These are responsible for taking decision based upon the collected data.
- Given diagram shows the Single Board Computer(SBC) based IoT device that includes CPU, GPU,RAM, storage and various types of interfaces and peripheral



Logical Design using Python:

Control Flow:



Statement	Description	Example

`if` `elif` `else`	If a condition is true, execute the block of code underneath the *if statement*. If not, see if the condition is true in one or more *else if* (`elif`) statements. If one of those is true, execute the code block under that. Otherwise, execute the code block underneath the *else statement*. `elif` and `else` statements are optional.	`number = 42`
		`guess = int(input("Guess a number between 1-100: "))`
		`if guess == number:`
		`    print("You win!")`
		`elif guess < number:`
		`    print("Nope")`
		`    print("Too low")`
		`else:`
		`    print("Nope")`
		`    print("Too high")`
		`print("Run the program to try again")`

`while`	A *while loop* executes the block of code underneath it repeatedly as long as the condition is true.	`counter = 15`
		`while counter >= 5:`
		`    print(counter)`
		`    counter = counter - 1`

`for..in`	Iterate over a sequence of numbers or objects. The variable declared in a *for loop* assumes the value of one of the numbers (or objects) during each iteration of the loop.	`for i in range(1, 11):`
		`    print(i)`

`break`	Use the *break statement* to exit out of a loop.	`while True:`
		`    message = input("Tell me when to stop: ")`
		`    if message == "stop":`
		`        break`
		`    print("OK")`

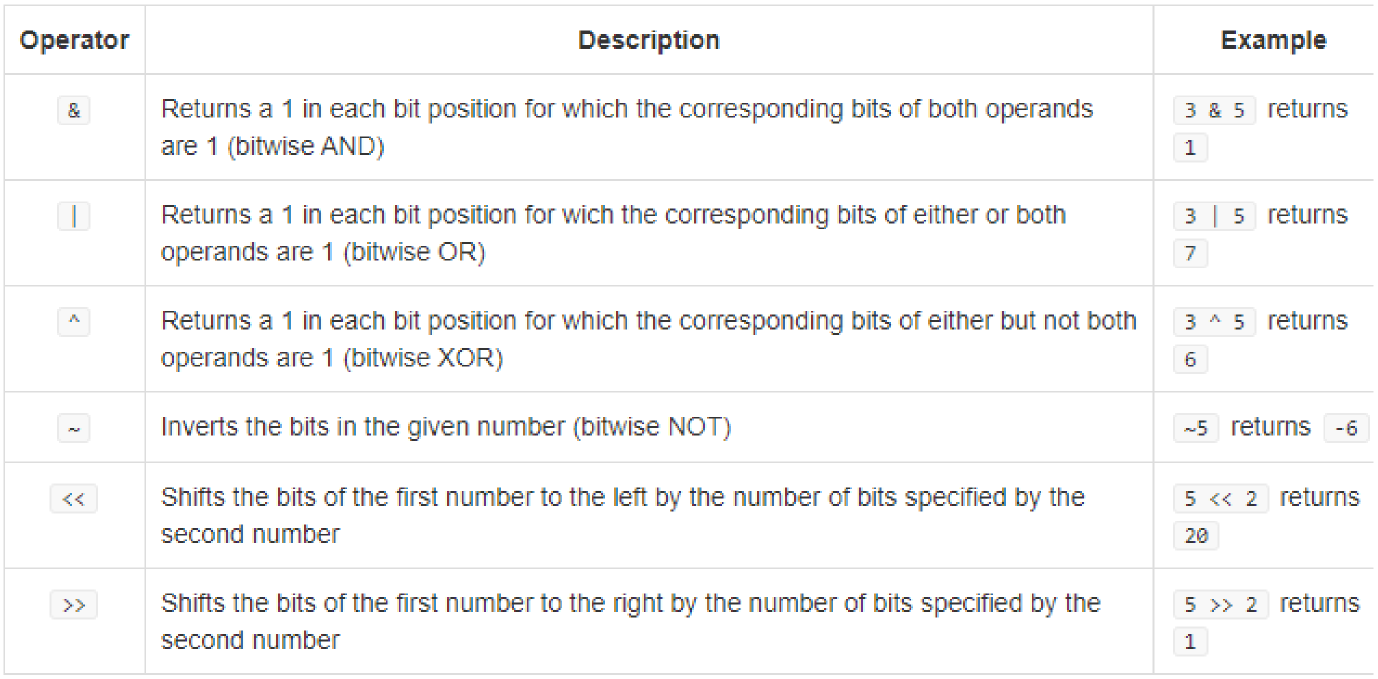
`continue`	The *continue statement* works similar to *break*, but instead of exiting the loop, it stops the current iteration and returns to the top of the loop.	`for i in range(1, 6):`
		`    if i == 3:`
		`        continue`
		`    print(i)`

Operators:

Operator	Description	Example
`+`	Adds two numbers	`2 + 3` returns `5`
`-`	Subtracts one number from another	`8 - 5` returns `3`
`*`	Multiplies two numbers together	`4 * 6` returns `24`
`**`	Raises the first number to the power of the second number	`2 ** 4` returns `16`
`/`	Divides the first number by the second number	`5 / 4` returns `1.25`
`//`	Divides the two numbers and rounds down to the nearest integer (divide and floor)	`5 / 4` returns `1`
`%`	Divides the first number by the second number and gives the remainder (modulo)	`19 % 8` returns `3`

Operator	Description	Example
`<`	`True` if the first number is less than the second, `False` otherwise	`5 < 3` returns `False`
`>`	`True` if the first number is greater than the second, `False` otherwise	`5 > 3` returns `True`
`<=`	`True` if the first number is equal to or less than the second, `False` otherwise	`2 <= 8` returns `True`
`>=`	`True` if the first number is equal to or greater than the second, `False` otherwise	`2 >=` returns `False`
`==`	`True` if the first number is equal to the second, `False` otherwise	`6 == 6` returns `True`
`!=`	`True` if the first number is not equal to the second, `False` otherwise (not equal)	`6 != 6` returns `False`

Operator	Description	Example
`not`	Gives the opposite (`True` becomes `False` and vice versa)	`x = False; not x` returns `True`
`and`	Returns `True` if both operands are `True` , `False` otherwise	`x = True; y = False; x and y` returns `False`
`or`	Returns `True` if either of the operands are `True` , `False` otherwise	`x = True; y = False; c or y` returns `True`



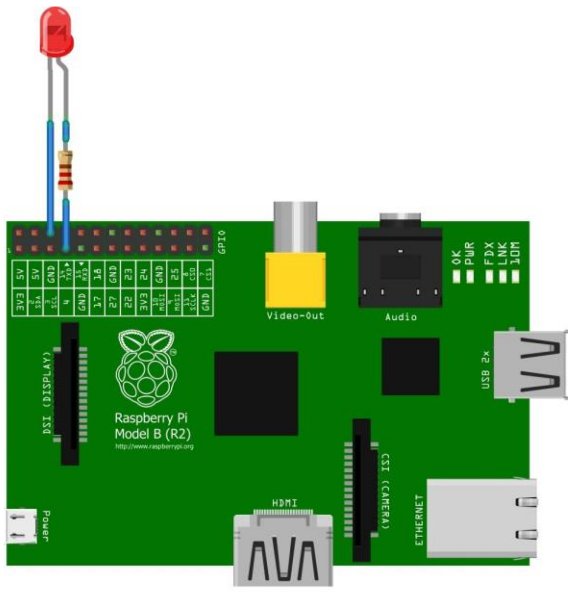
Operator	Description	Example
`&`	Returns a 1 in each bit position for which the corresponding bits of both operands are 1 (bitwise AND)	`3 & 5` returns `1`
`|`	Returns a 1 in each bit position for wich the corresponding bits of either or both operands are 1 (bitwise OR)	`3 | 5` returns `7`
`^`	Returns a 1 in each bit position for which the corresponding bits of either but not both operands are 1 (bitwise XOR)	`3 ^ 5` returns `6`
`~`	Inverts the bits in the given number (bitwise NOT)	`~5` returns `-6`
`<<`	Shifts the bits of the first number to the left by the number of bits specified by the second number	`5 << 2` returns `20`
`>>`	Shifts the bits of the first number to the right by the number of bits specified by the second number	`5 >> 2` returns `1`

Python program to glow LED using Raspberry pi:

Components required:

- One led
- 100 ohm resistor
- Jumper cables
- Raspberry pi 3

Diagram:



Python code:

```
import time
import RPi.GPIO as GPIO       ## Import GPIO library
GPIO.setmode(GPIO.BOARD)      ## Use board pin numbering
GPIO.setup(11, GPIO.OUT)      ## Setup GPIO Pin 11 to OUT
while True:
            GPIO.output(11,True)  ## Turn on Led
            time.sleep(1)         ## Wait for one second
            GPIO.output(11,False) ## Turn off Led
            time.sleep(1)         ## Wait for one second
```