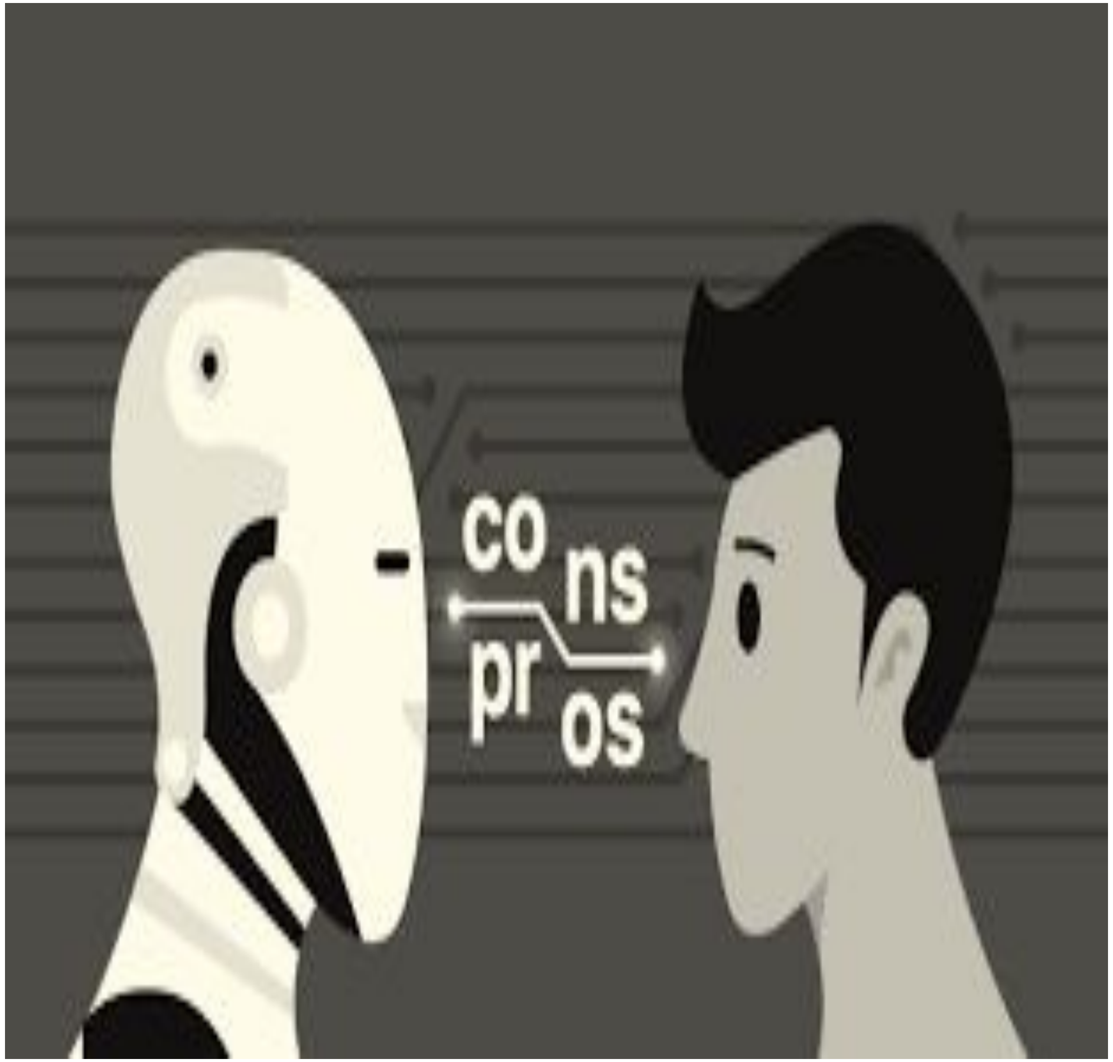
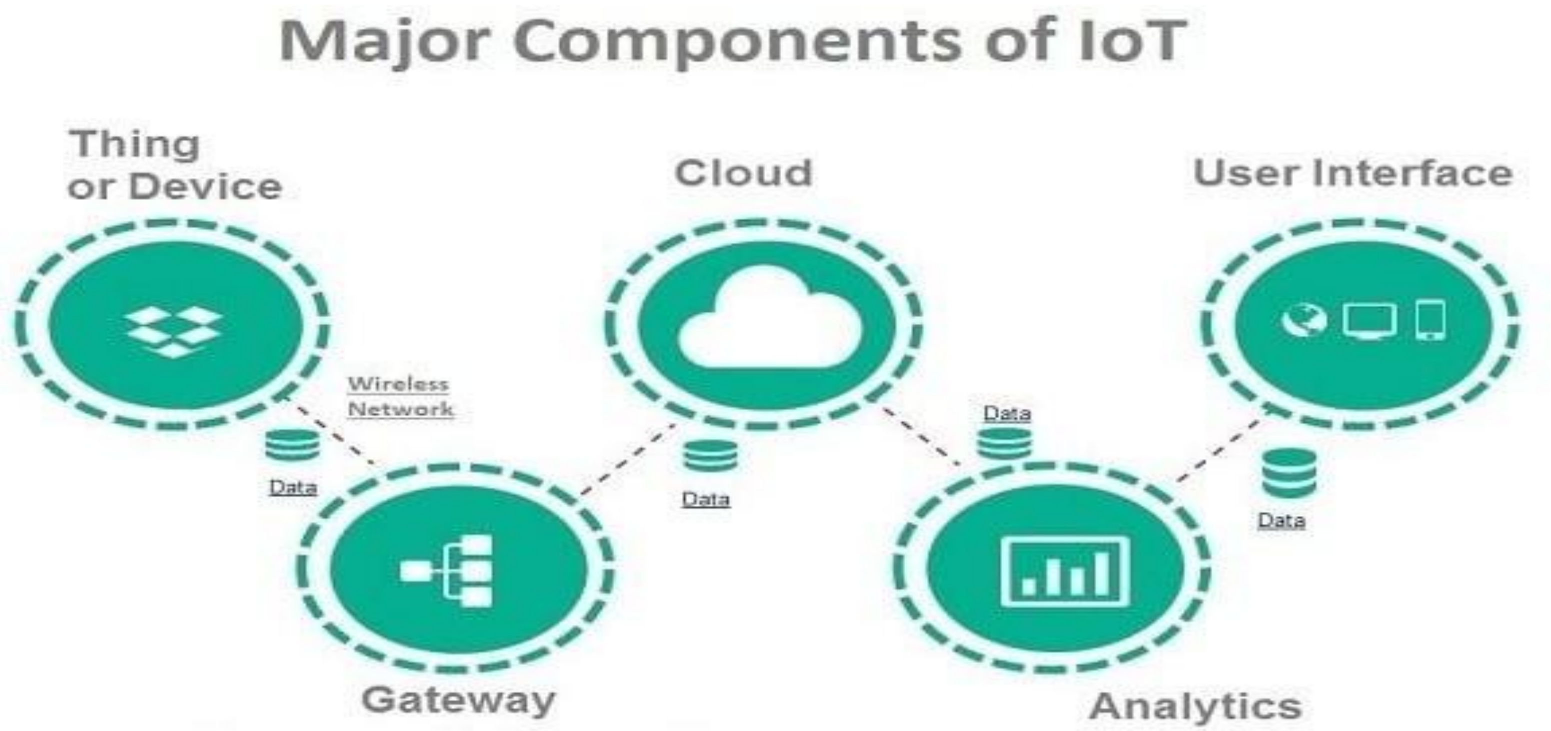
IoT Interfacing



IoT System

-



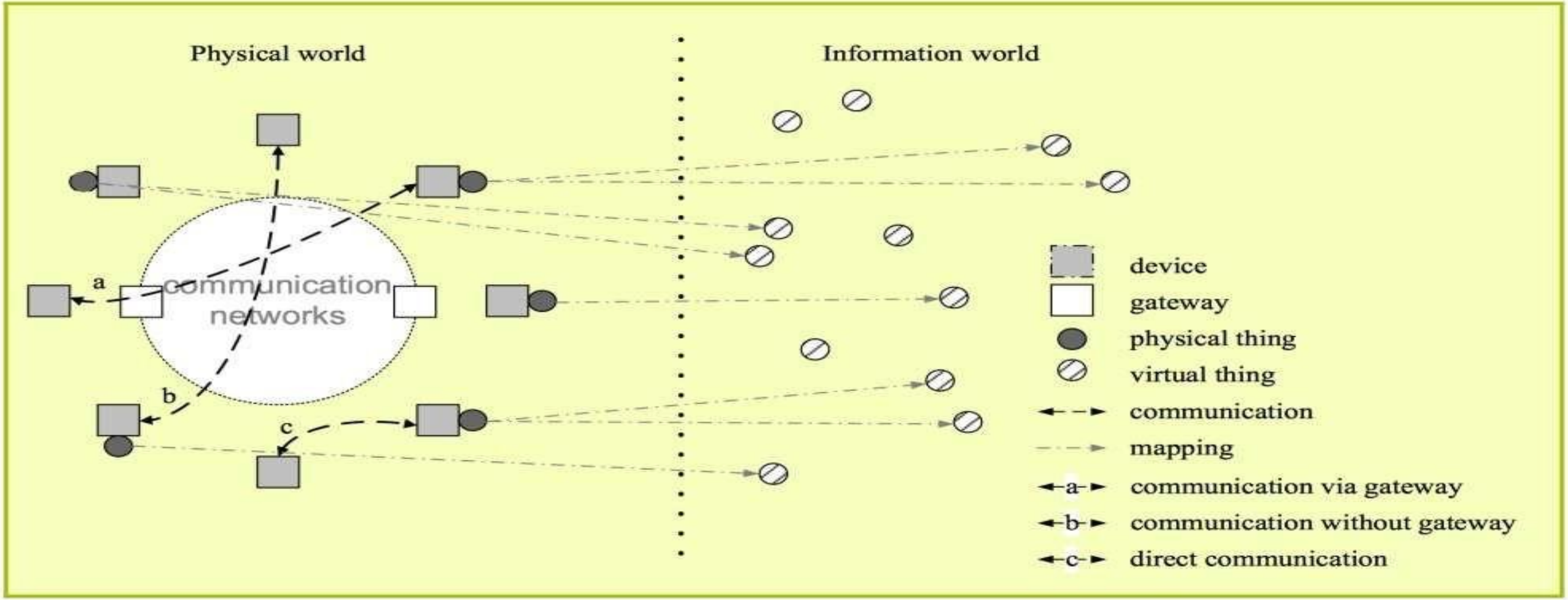
Source: Internet

Things

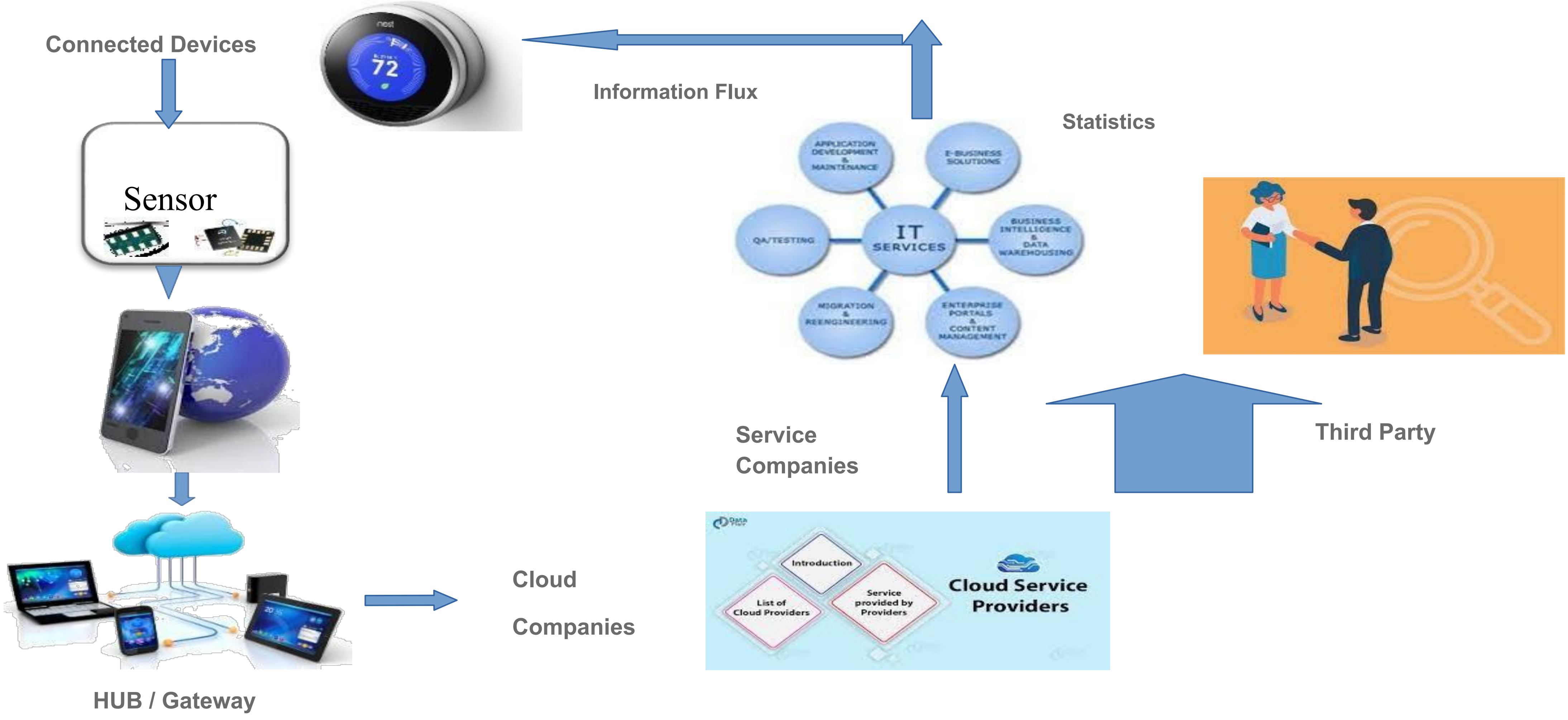
Physical things exist in the physical world and are capable of being sensed, actuated and connected. Examples of physical things include the surrounding environment, industrial robots, goods and electrical equipment.

Virtual things exist in the information world and are capable of being stored, processed and accessed. Examples of virtual things include multimedia content and application software.

Physical and Virtual World



IoT System Design Cycle



Connected Devices

Sensor

Information Flux

Statistics

Third Party

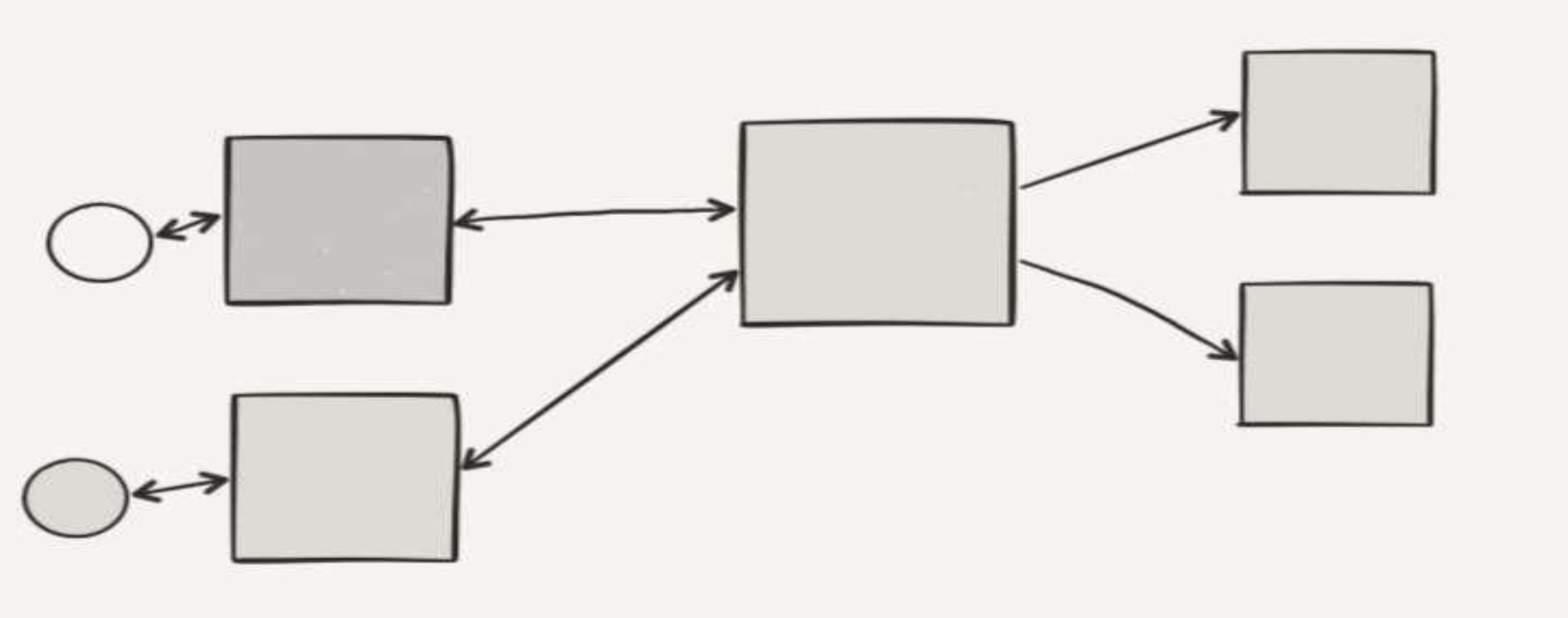
Service Companies

Cloud Companies

HUB / Gateway

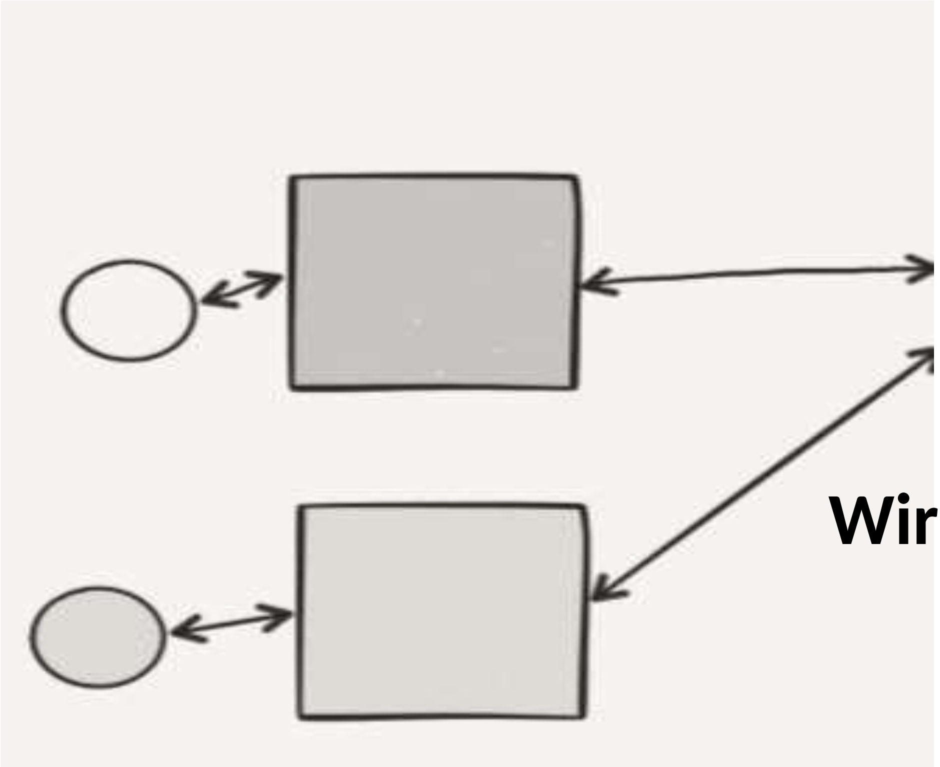
72
APPLICATION DEVELOPMENT & MAINTENANCE
E-BUSINESS SOLUTIONS
QA/TESTING
IT SERVICES
BUSINESS INTELLIGENCE & DATA WAREHOUSING
MIGRATION & REENGINEERING
ENTERPRISE PORTALS & CONTENT MANAGEMENT
Introduction
List of Cloud Providers
Service provided by Providers
Cloud Service Providers

IoT Architecture



Node Gateway Services

IoT Architecture : Node



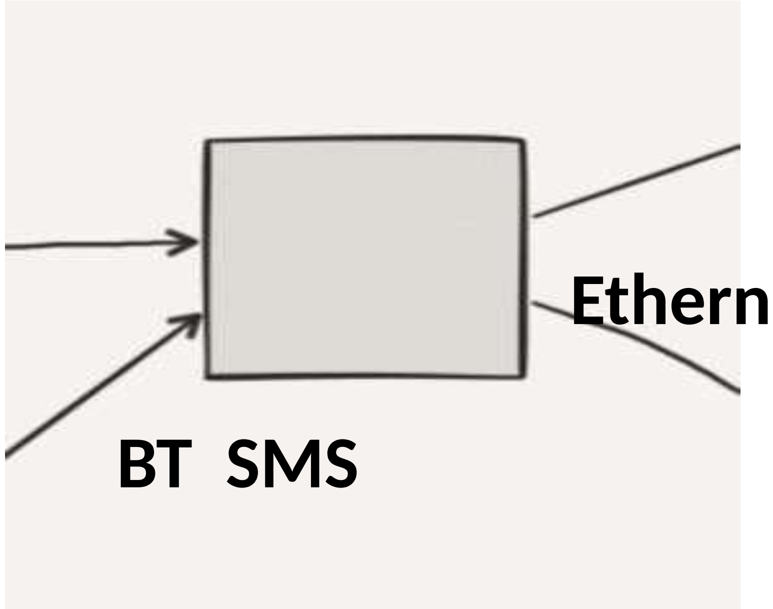
Sensor

Protocol

Wireless link+ Protocol

Controller, Memory and Power Management

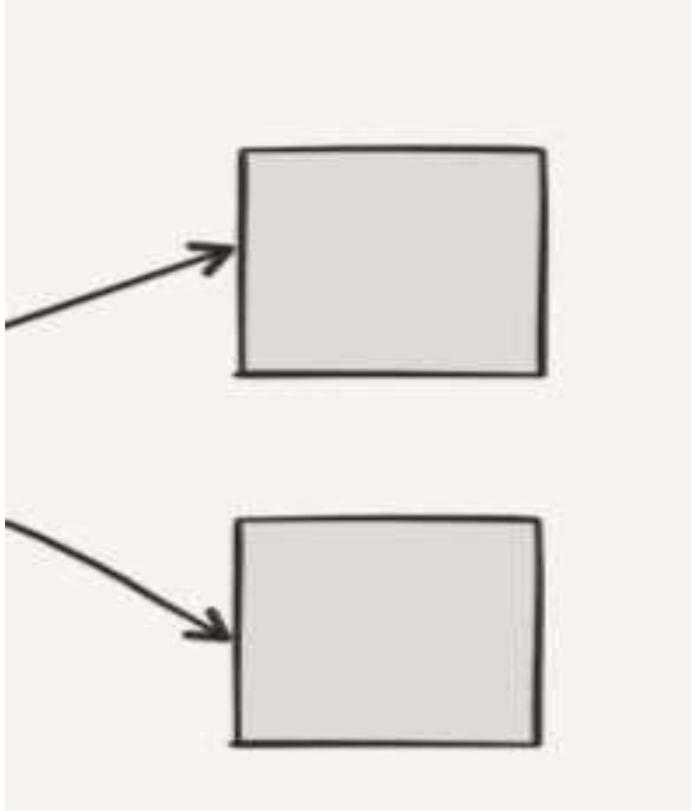
IoT Architecture : Gateway



Ethernet

BT SMS

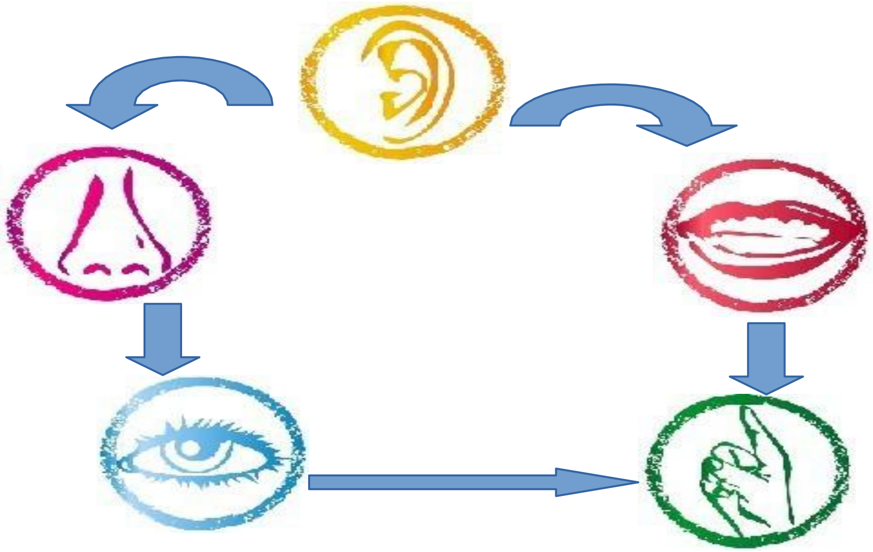
IoT Architecture : Services



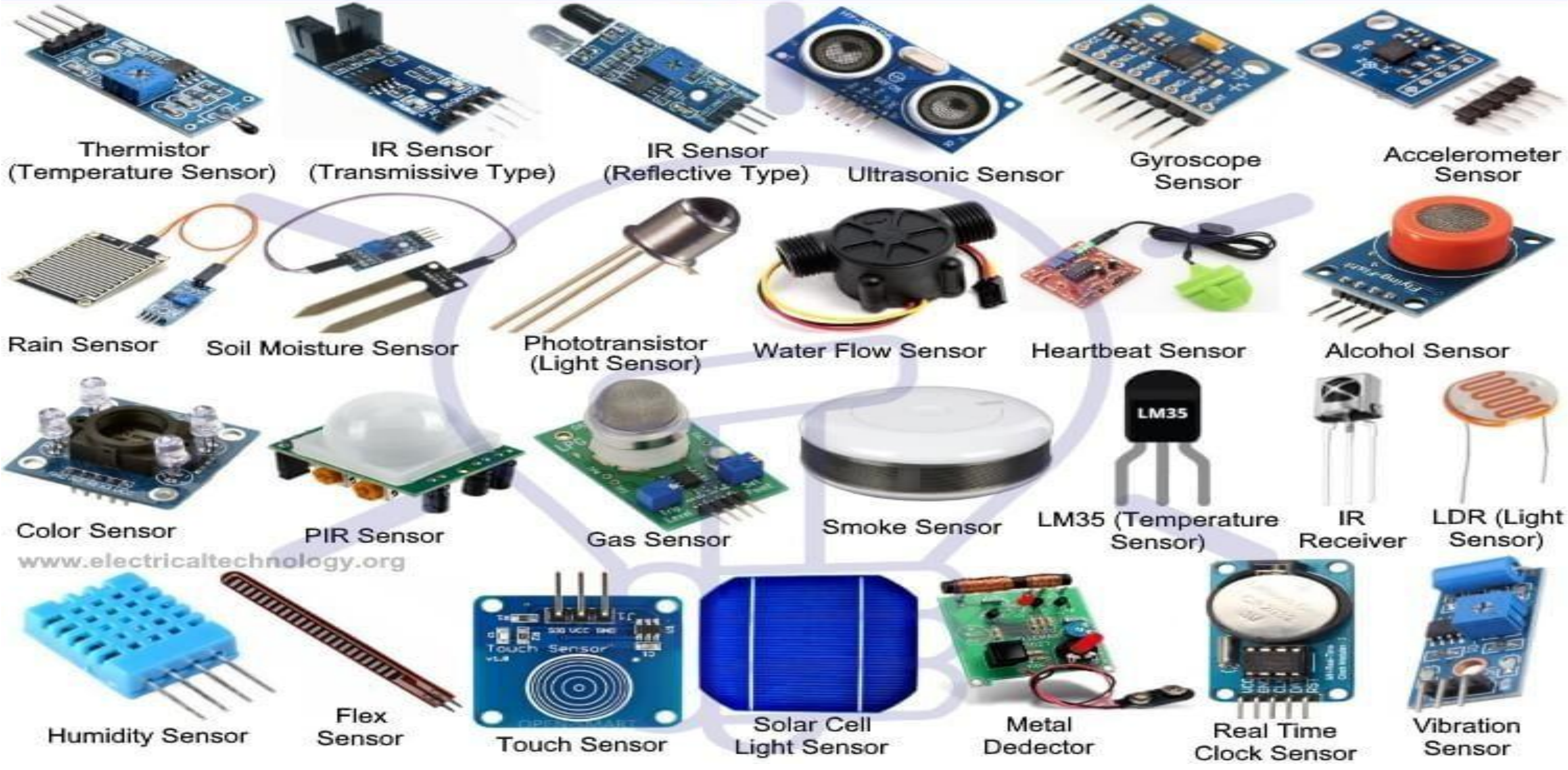
Graphing Machine Learning Alerting

Sensors

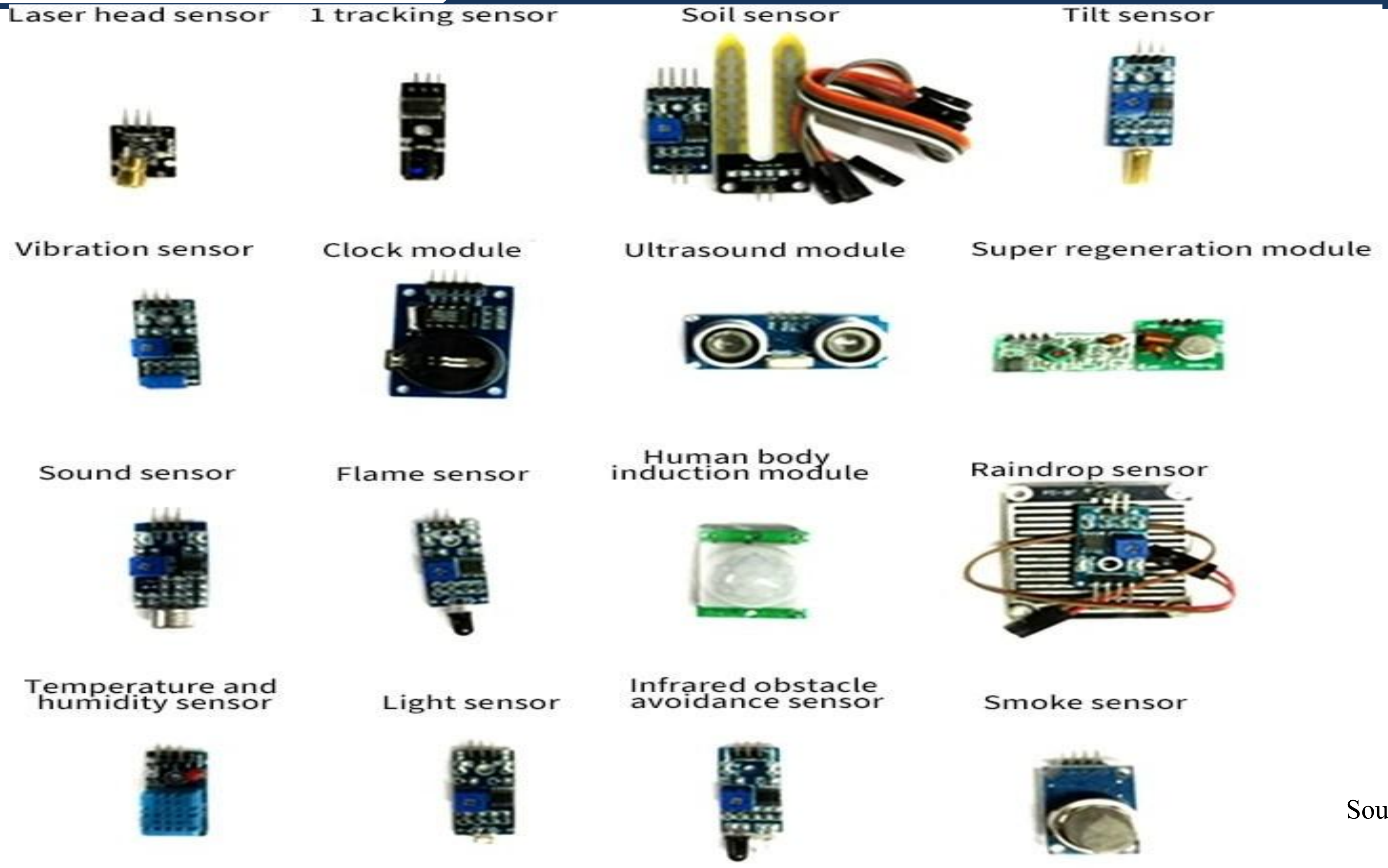
- **Sensors measure or identify a particular quantity**
- Convert physical quantities to electrical signals understood by machines



Different Types of Sensors

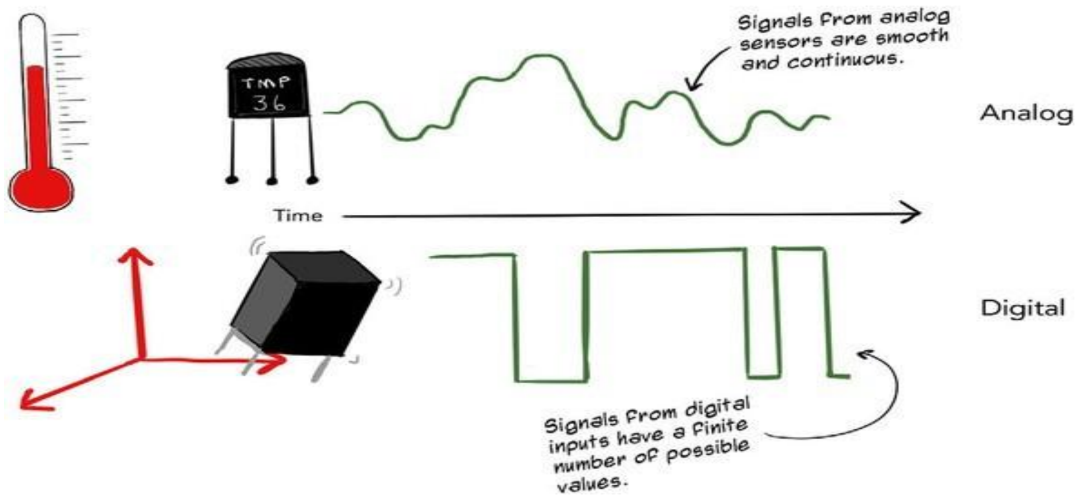


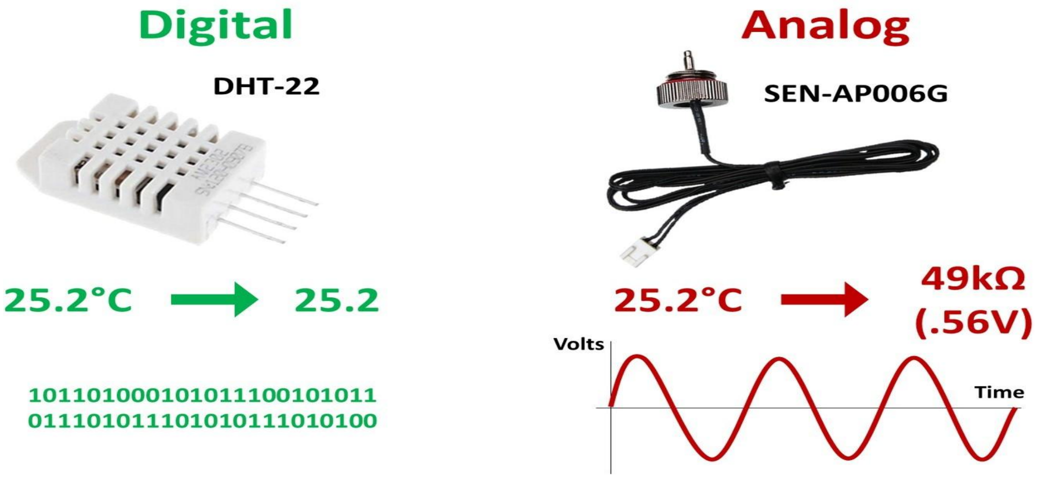
Some common Sensors



Source : Internet

Type of Sensors : Analog and Digital


TMP 36
Signals from analog sensors are smooth and continuous.
Analog
Time
Digital
Signals from digital inputs have a finite number of possible values.



Type of batteries

Commonly used rechargeable batteries

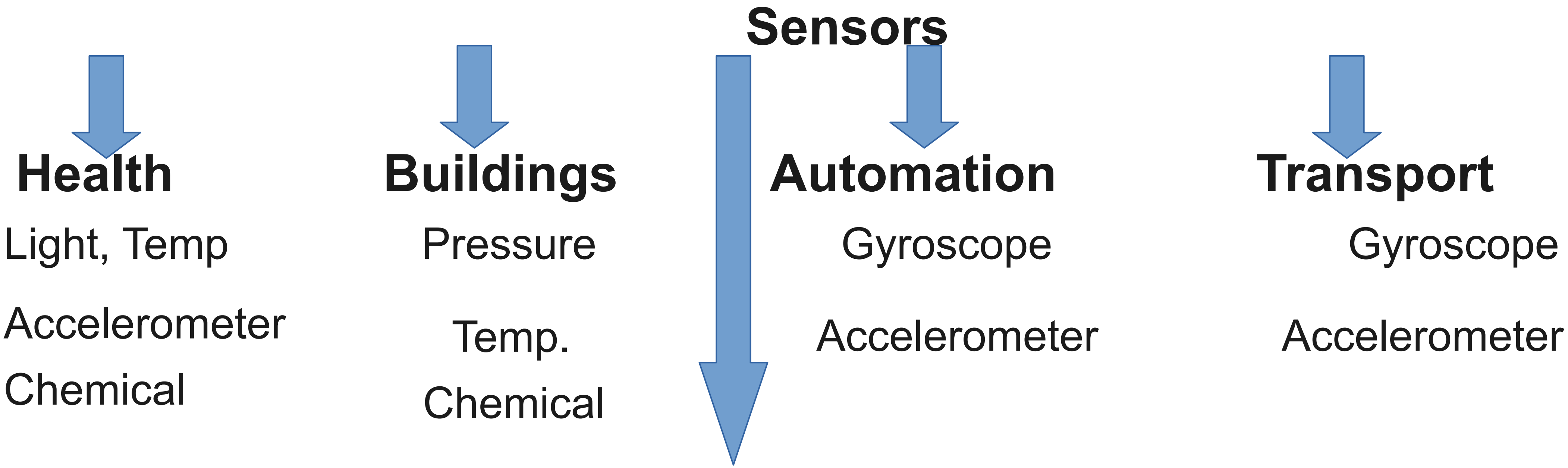
- **Li-ion / Li-Poly (Lithium ion / Lithium polymer)**
- **Pb-Acid (Lead Acid)**
- **NiCd (Nickle Cadmium)**
- **NiMH (Nickle Metal Hydride)**

- **Different chemistries, different terminal voltages**

Li-ion / Li-Poly: most popular for portable and wearable IoT

- Highest energy density
- Low maintenance
- Ease of handling

Sensors in IoT



Sensors

Health

Light, Temp

Accelerometer

Chemical

Buildings

Pressure

Temp.

Chemical

Automation

Gyroscope

Accelerometer

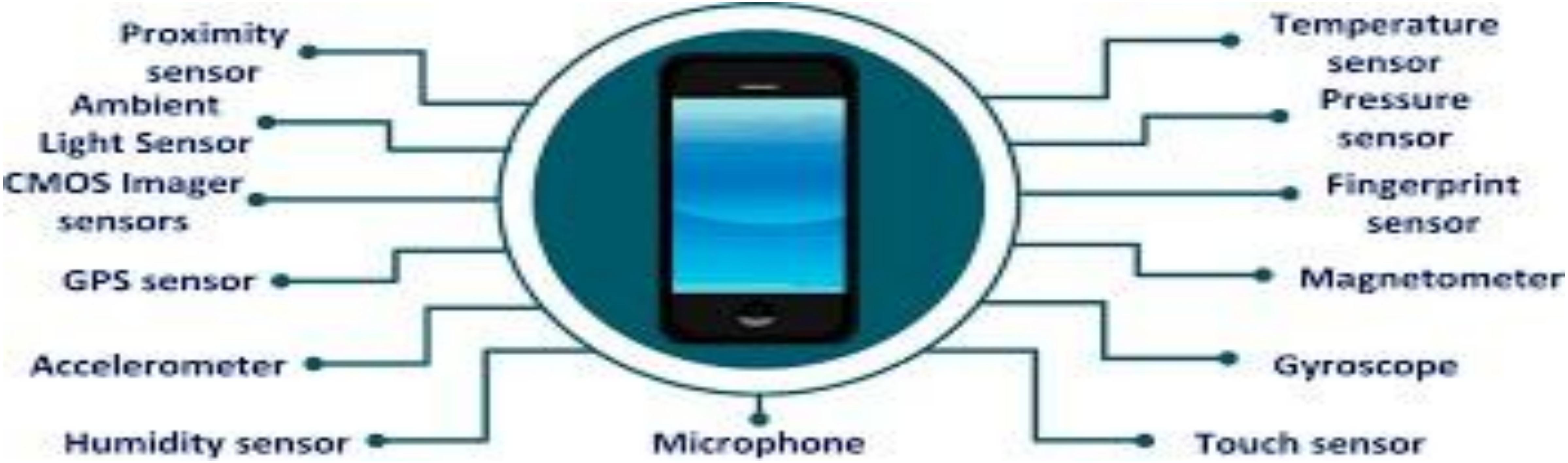
Transport

Gyroscope

Accelerometer

Industrial, Environment, Security and Public Safety, Retail and Logistics

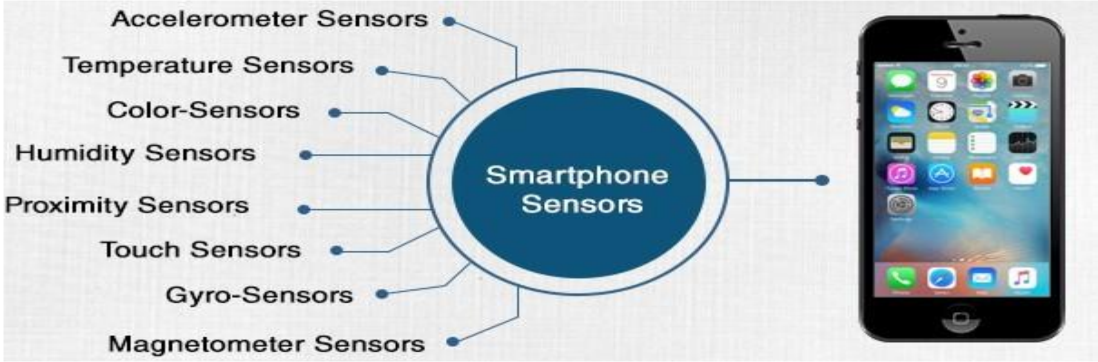
Mobile Phone : A Sensor Hub



Monitor Humans Through ?

Parental Control through mobile

Spying through mobile apps

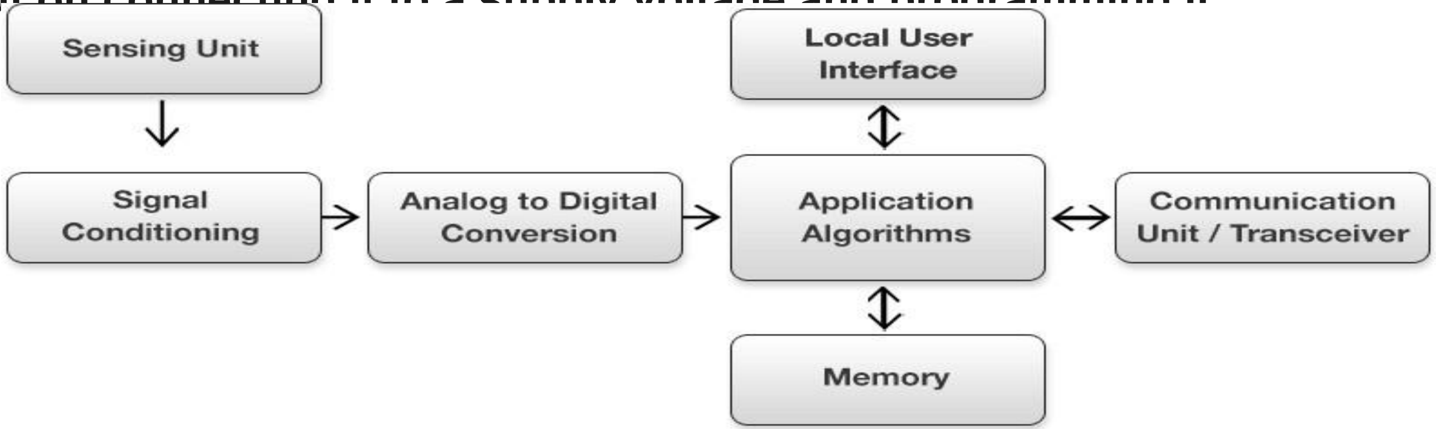


Mobile Phone : A Sensor Hub

Smart Sensor

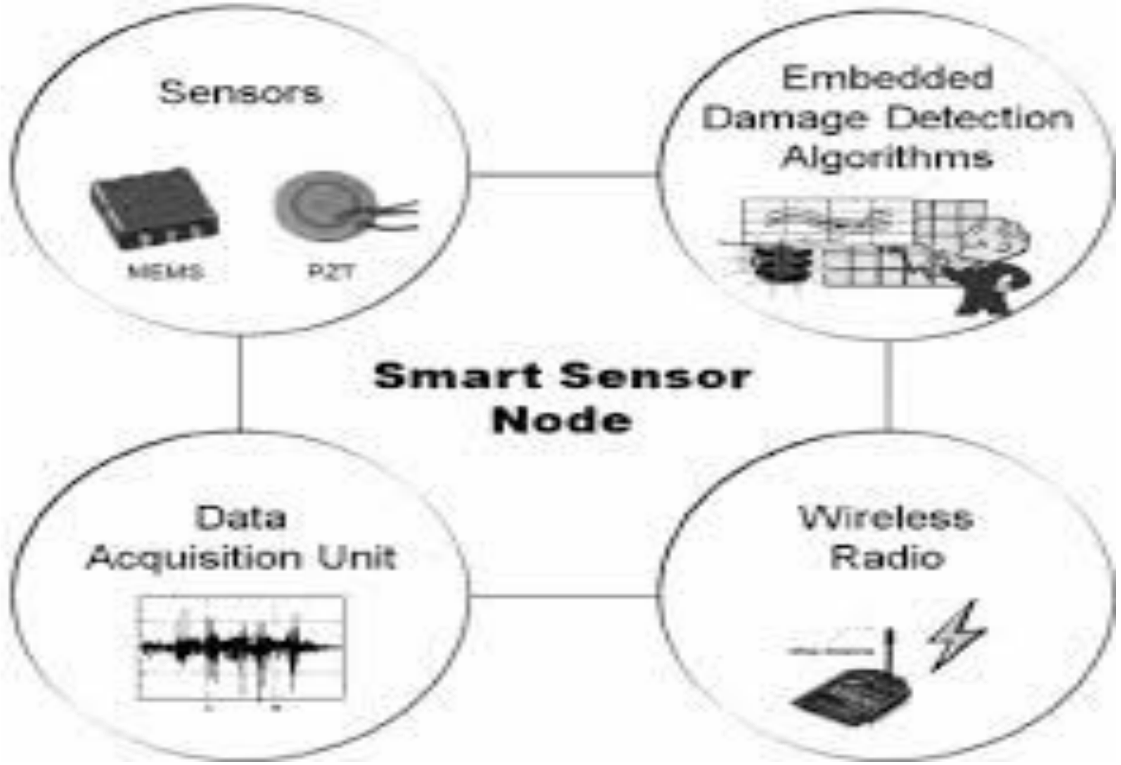
Sensors with integrated electronics that can perform Data conversion, Bidirectional communication, take decisions and perform logical operations

A sensor with built-in integrated circuit (microcontroller, and sensor) which provides the physical parameter as output on connecting it to a supply voltage and programming it

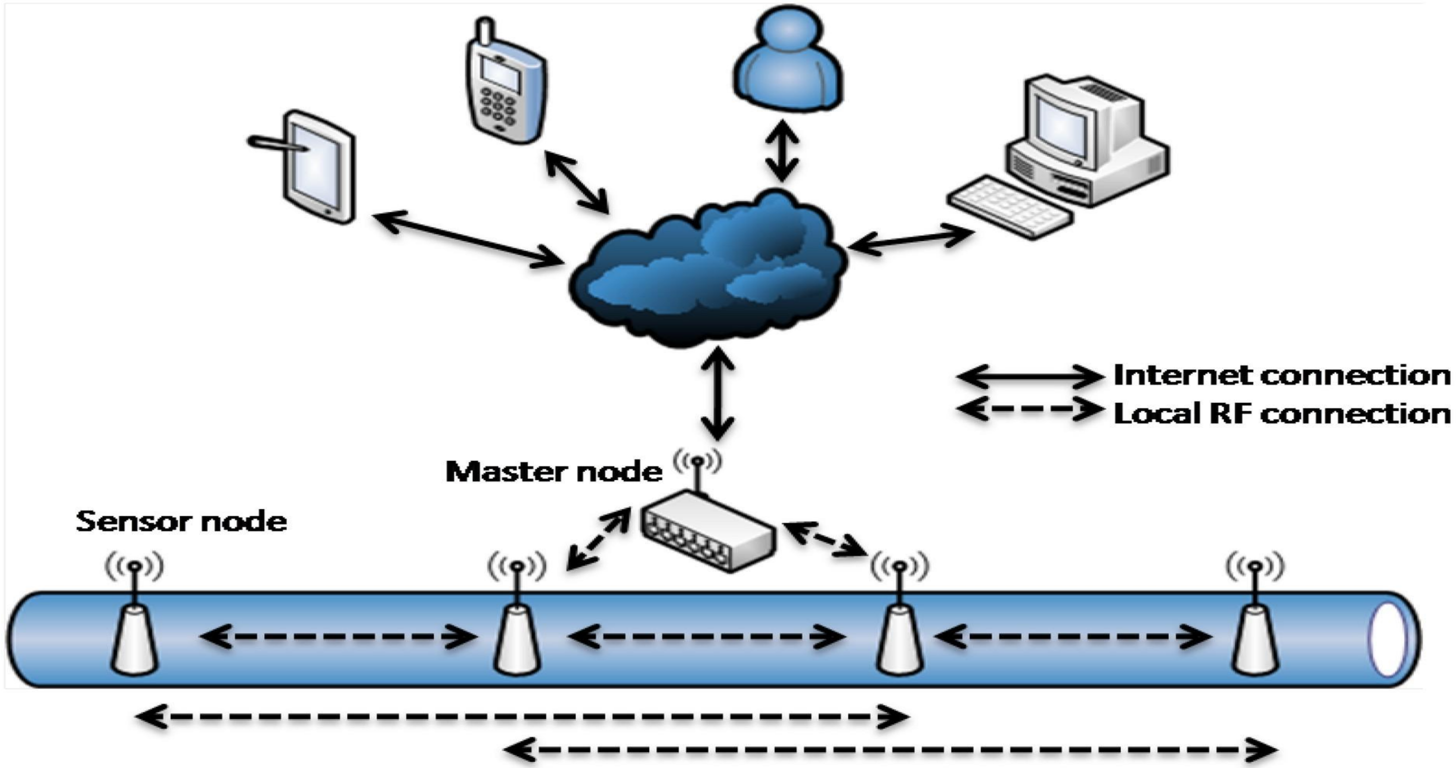


A smart sensor for temperature gives output as hex-digit - 10 UART serial bits according to the degree celsius. For ex. 01100100 is obtained for 100 degree Celsius considering the sensor has been calibrated

Smart Sensor Node



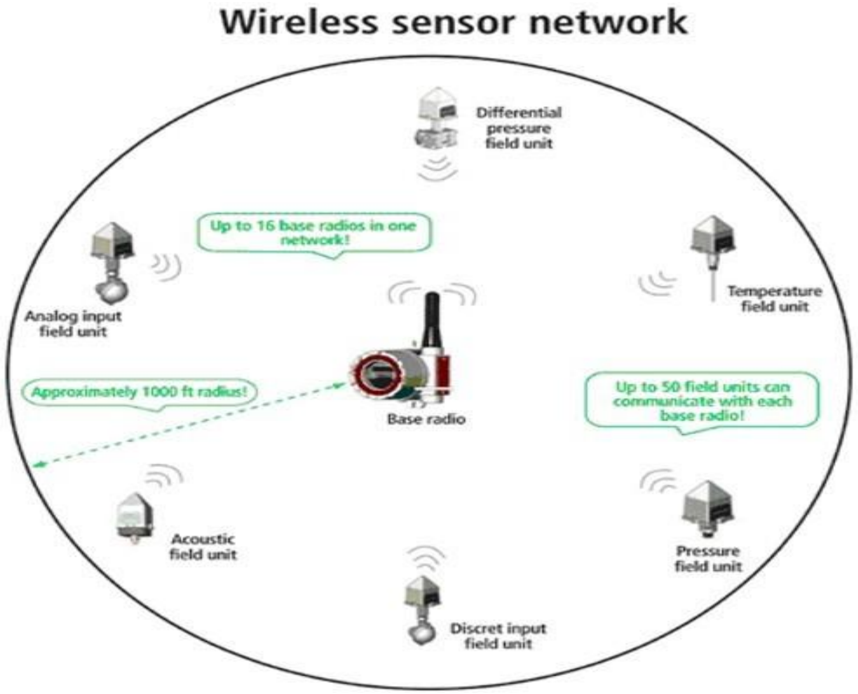
Smart Sensor Network



Wireless Sensor Network (WSN)

Network of sensor nodes which connect wirelessly

Nodes have capability of computation, data compaction, aggregation and analysis, communication and networking. Each node has independent computing power and capability to send and receive responses, data forward and routing capabilities

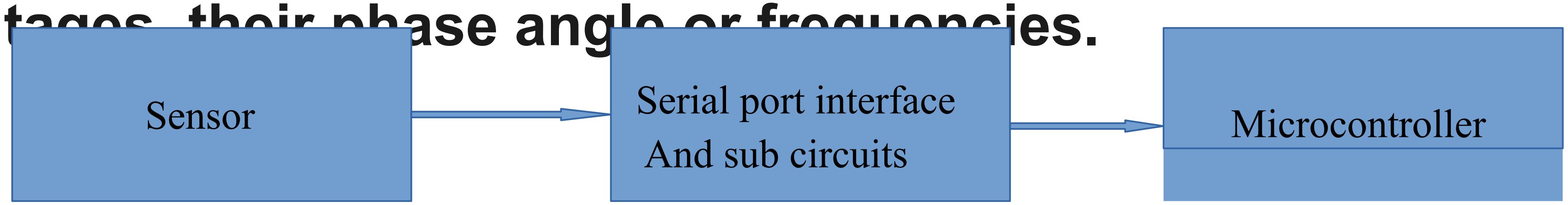

Wireless sensor network
Differential pressure field unit
Up to 16 base radios in one network!
Temperature field unit
Analog input field unit
Base radio
Approximately 1000 ft radius!
Up to 50 field units can communicate with each base radio!
Acoustic field unit
Pressure field unit
Discret input field unit

Sensing Circuit

Circuit input receives output of sensor/transducer

Circuit output variation is according to the variation in physical condition

The circuit receives energy in the form of variation in currents, voltages, their phase angle or frequencies.



Multi Sensor System

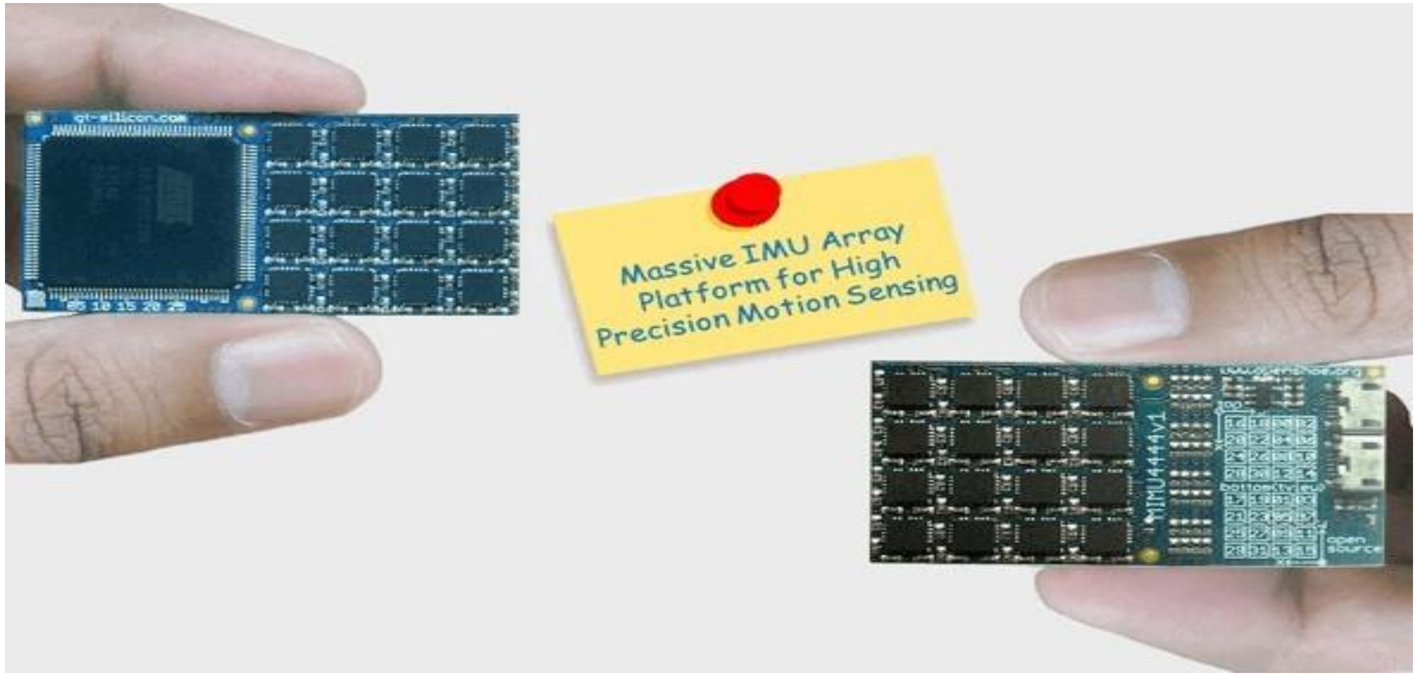

gt-silicon.com
05 10 15 20 25
Massive IMU Array Platform for High Precision Motion Sensing
MIMU4444v1
open source

Figure Source : Inertial Elements

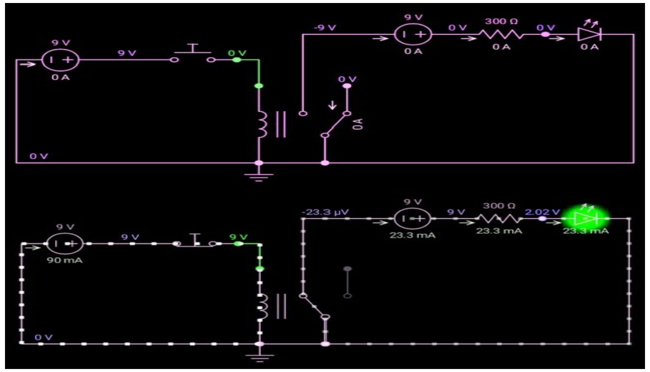
Actuator

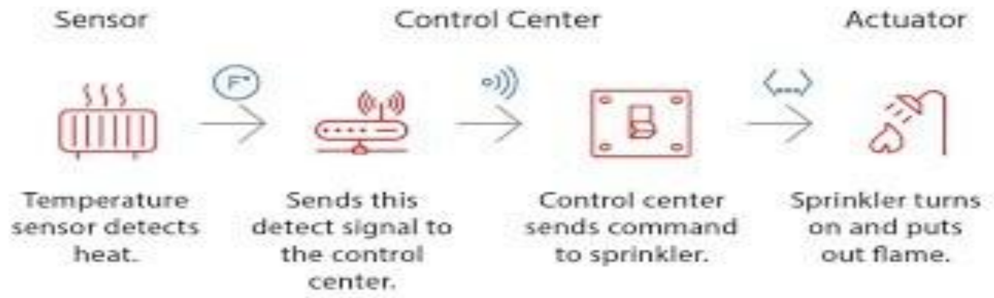
A device that takes the actions as per the input command, pulse, state (1/0), set of 1s and 0s or control signal. An attached motor, speaker, LED or an output device converts electrical energy into physical action

Piezoelectric vibrator : Piezoelectric crystals when applied varying electric voltages at input generate vibrations

Motor : can be dc/ac; I/O modules available to receive control digital inputs of 1/0 deliver high currents. A cam converts rotator motion into linear motion when motor rotates.

Relay Switch : An electronic switch can be controlled by 1/0 from the port pin of microcontroller. A relay switch makes mechanical contact when input magnetizes with a control circuit and pulls a lever to make the contact

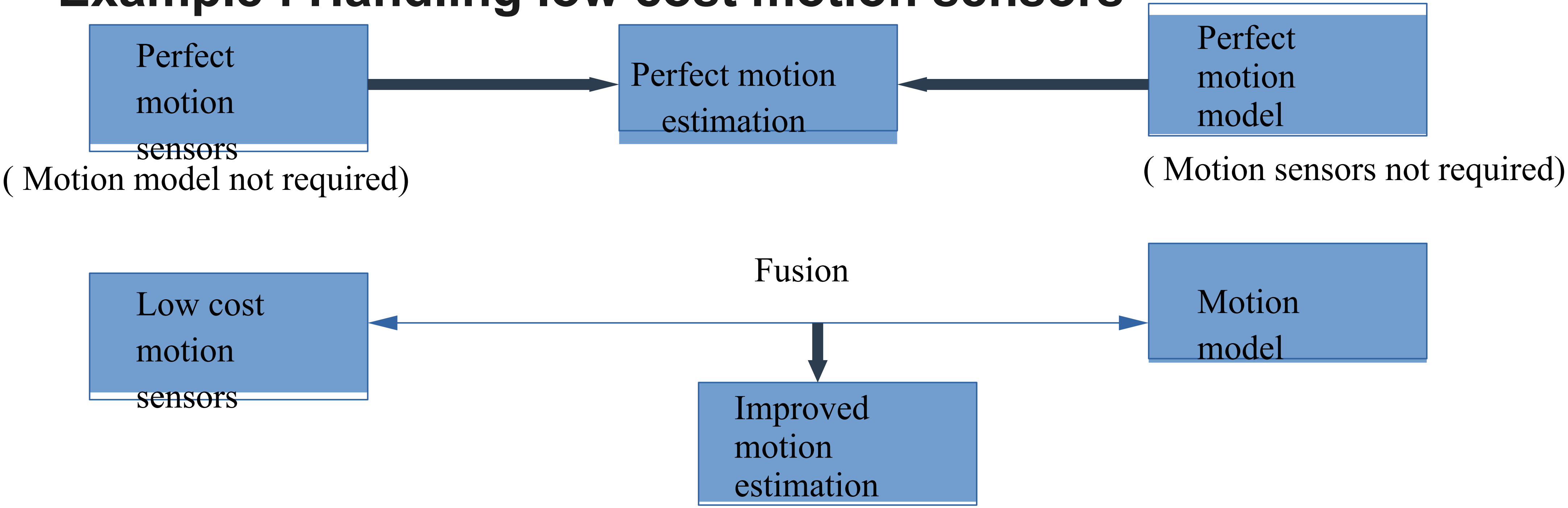



Sensor
Control Center
Actuator
Temperature sensor detects heat.
Sends this detect signal to the control center.
Control center sends command to sprinkler.
Sprinkler turns on and puts out flame.

Working with low cost Sensors

Problem : Noise and Non-aligned response Solution : Fusing sensors with computation models

Example : Handling low cost motion sensors



Perfect motion sensors

(Motion model not required)

Perfect motion estimation

Perfect motion model

(Motion sensors not required)

Low cost motion sensors

Fusion

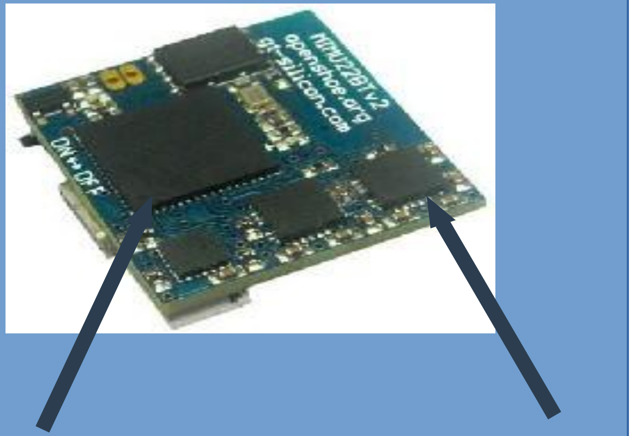
Motion model

Improved motion estimation

[redacted]

Multi Sensor System

Classical Multi Sensor System

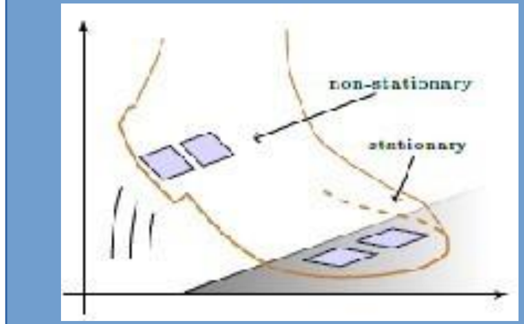


Pressure Sensor

Motion Sensor

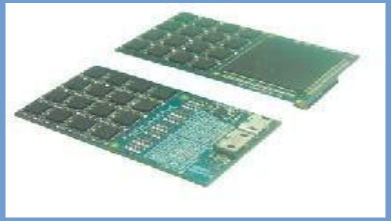
Fusing Data from different Sensors improves performance

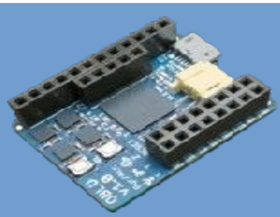
Multi Sensor Joint System



Multiple no-rigidly connect IMUs result in dynamic diversity

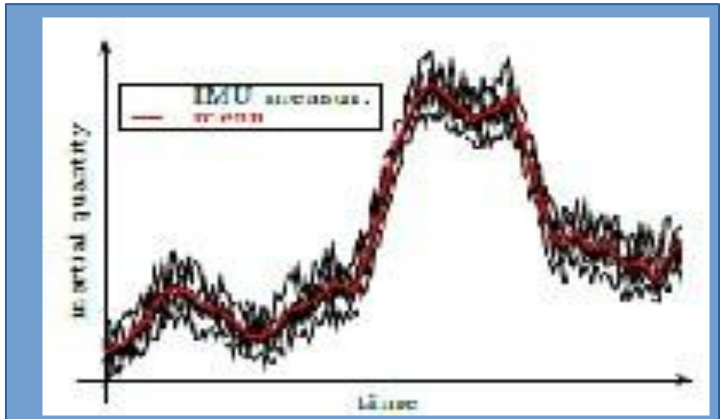
Co-located Multi Sensor System



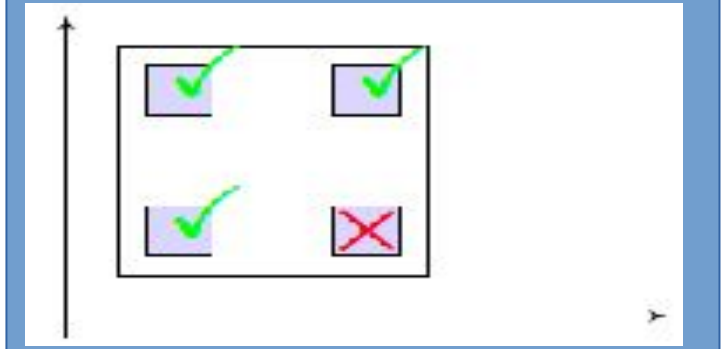


An array of well placed IMUs provides new opportunities for sensing

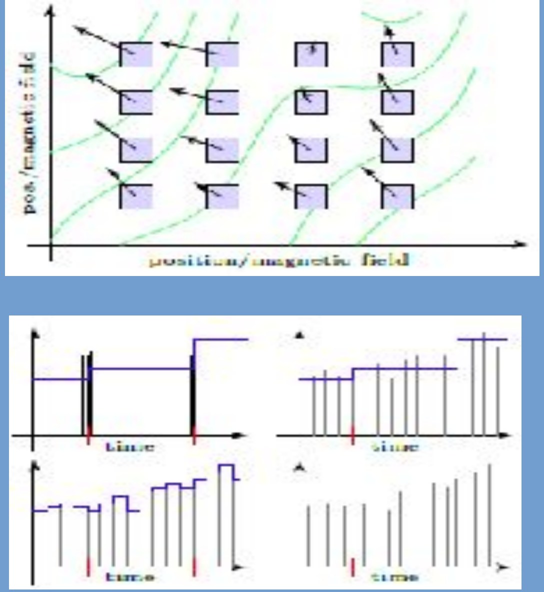
Multi Sensor System :



Noise Averaging



Fault Detection and Isolation

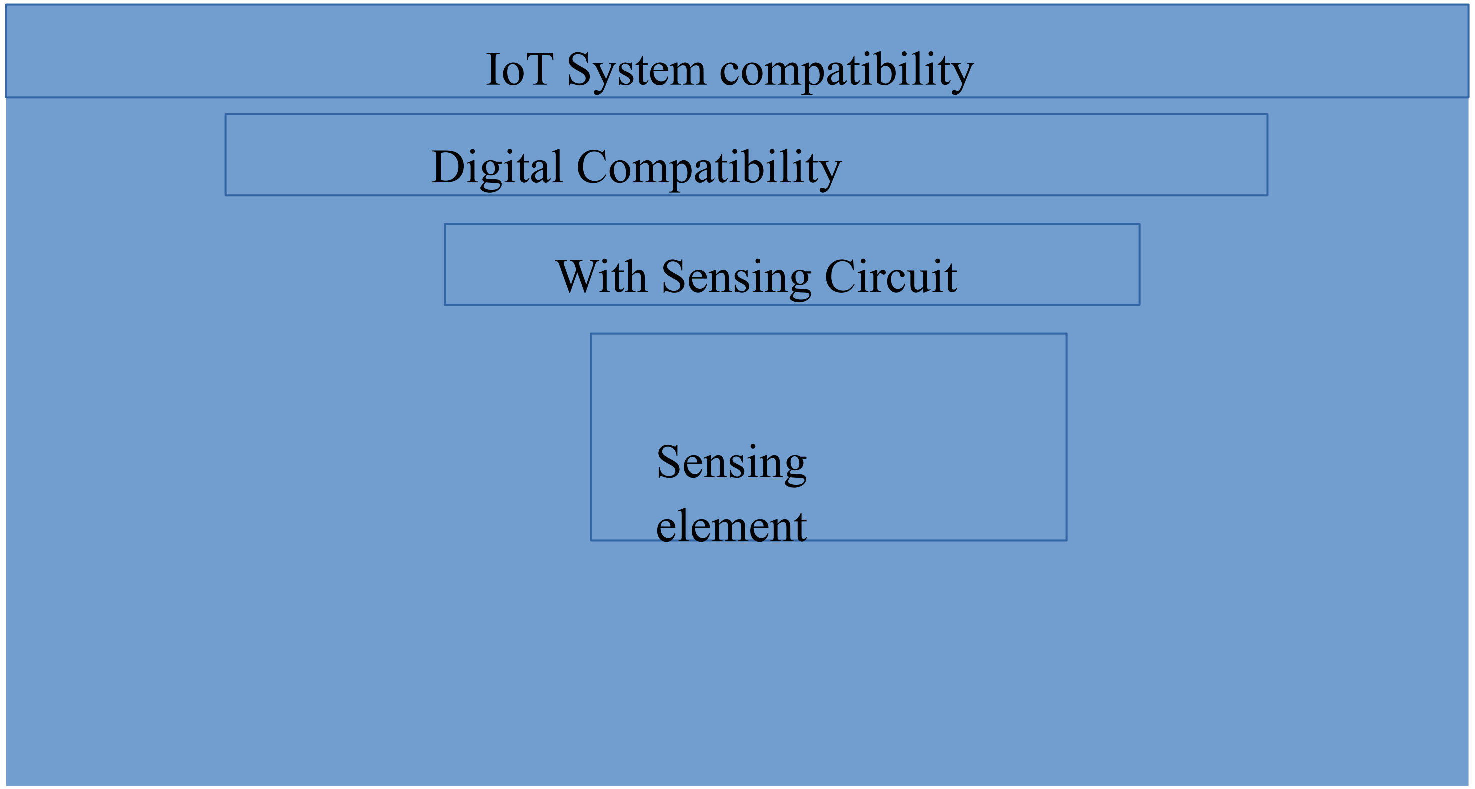


Spatial and temporal diversity

IoT System Component :

IoT System compatibility
- -Computation
- -Power mgmt. (battery)
- -Wireless
- -Data comm. Protocols

Digital Compatibility
- -Analog to Digital
- - data transfer protocols



Sensing Circuit
- -excite electrically
- -amplification

Sensing Element
- - responds to physical world

Control Unit

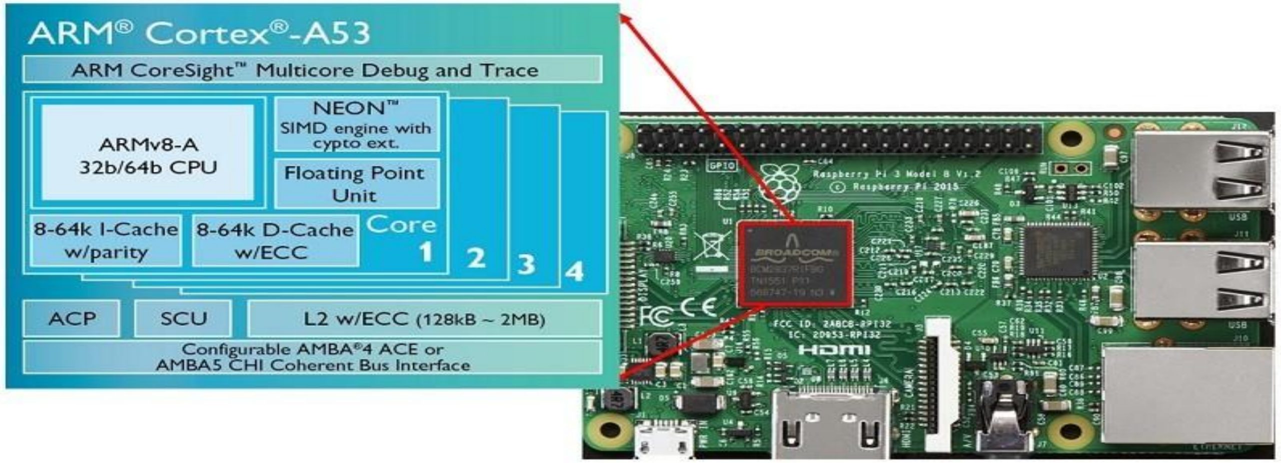
Single VLSI Chip ; A core in an application specific instruction set processor, microcontroller called

Commonly used control unit in IoT - Microcontroller unit; A core in System-on-Chip (SoC) with SD card for embedded software and OS software

Ex : ARM Cortex, ATmega328

Microcontroller components : Processor, Internal RAM, Internal Flash and

Firmware, Timers, Programmable I/O Ports, General purpose I/Os, Serial I/O Ports, PWM, ADC, Communication Netw

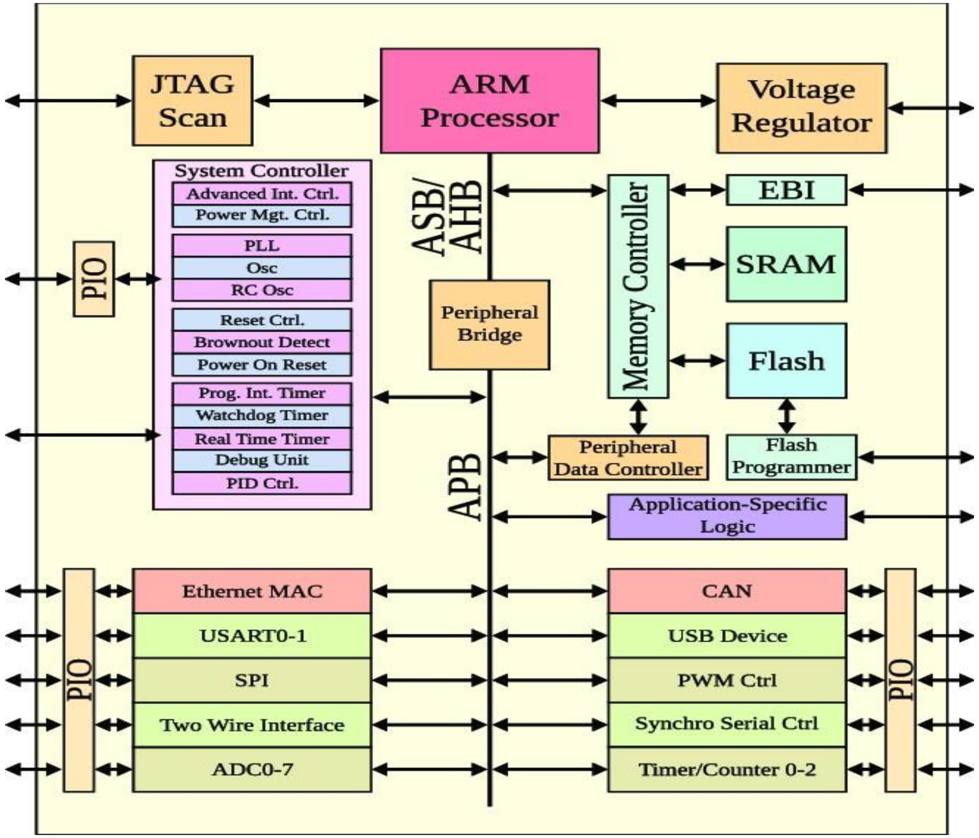


System on Chip

A circuit on a single silicon chip, consisting of multiple processors, hardware units and the embedded software

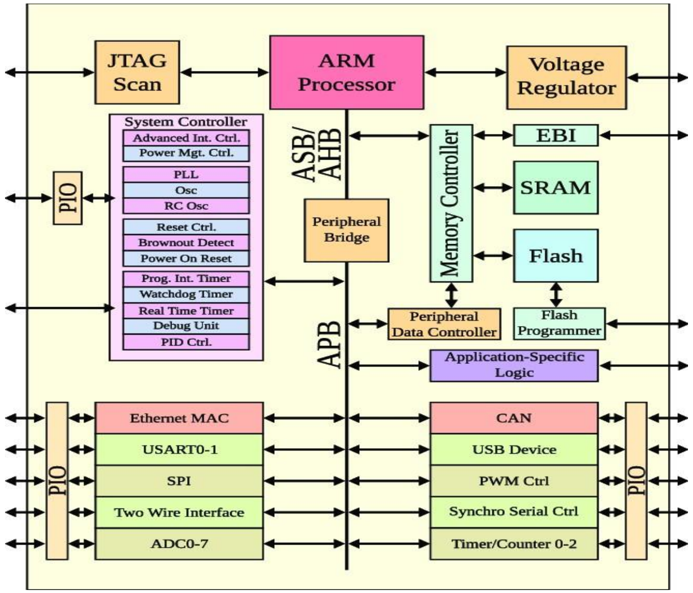
A VLSI chip that has multiple processors, software and all the needed digital as well as analog circuits' on-chip; A SD card stores external programs and OS and enables use of the chip distinctly for a particular purpose

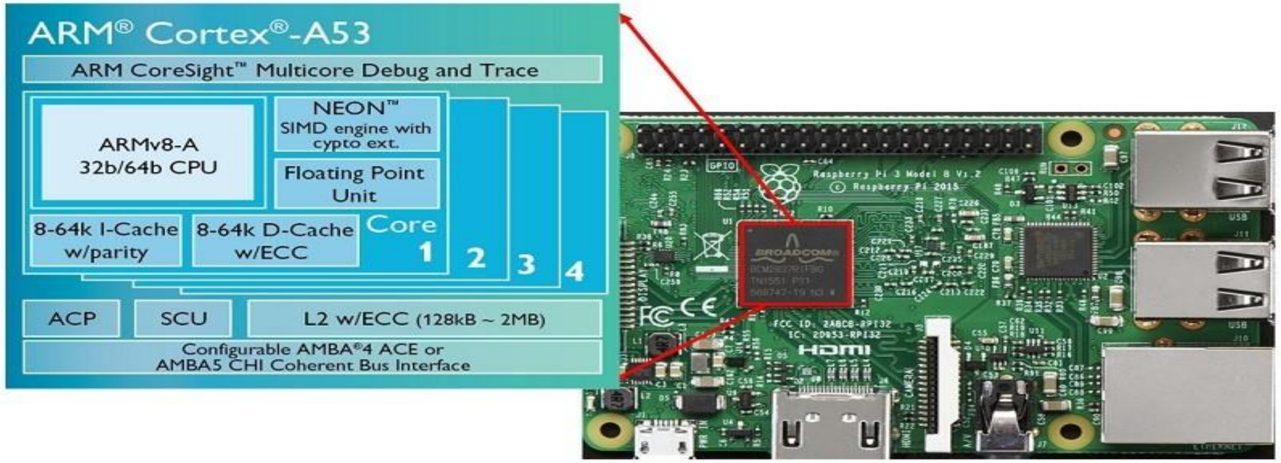
A SoC can be of different platforms : Raspberry Pi



System-on-Chip

- A system on a chip is an integrated circuit that integrates multiple processors, hardware units, analog circuits and the embedded software
- Microcontroller unit with SD card for embedded software and OS software that enables use of the chip distinctly for a particular purpose
- Example : ARM Cortex, ATmega328
- Microcontroller components : Processor, Internal RAM, Internal Flash and Firmware, Timers, Programmable I/O Ports, General purpose I/Os, Serial I/O Ports, PWM, ADC, Communication Network Interfaces

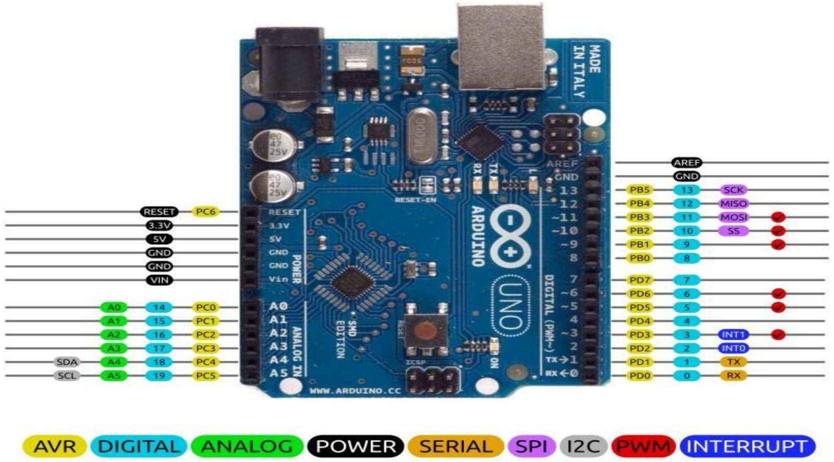



ARM® Cortex®-A53
ARM CoreSight™ Multicore Debug and Trace
ARMv8-A 32b/64b CPU
NEON™ SIMD engine with cypto ext.
Floating Point Unit
8-64k I-Cache w/parity
8-64k D-Cache w/ECC
Core 1
2
3
4
ACP
SCU
L2 w/ECC (128kB ~ 2MB)
Configurable AMBA®4 ACE or AMBA5 CHI Coherent Bus Interface

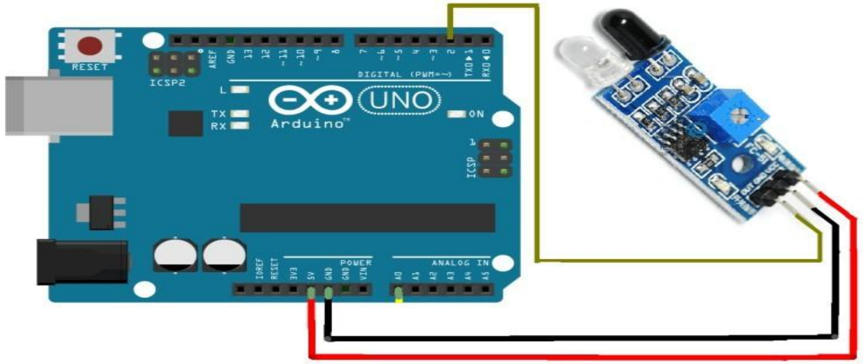
Raspberry Pi Board hosting the ARM Cortex A-53 System-on-Chip

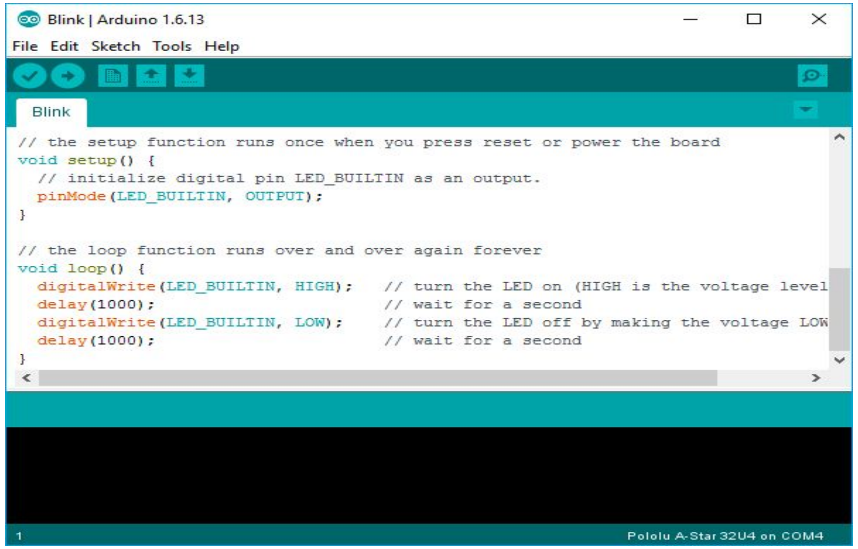
Common IoT Computing Platforms : Arduino

Arduino Uno SMD Pinout



2014 by Bouni
Photo by Arduino.cc



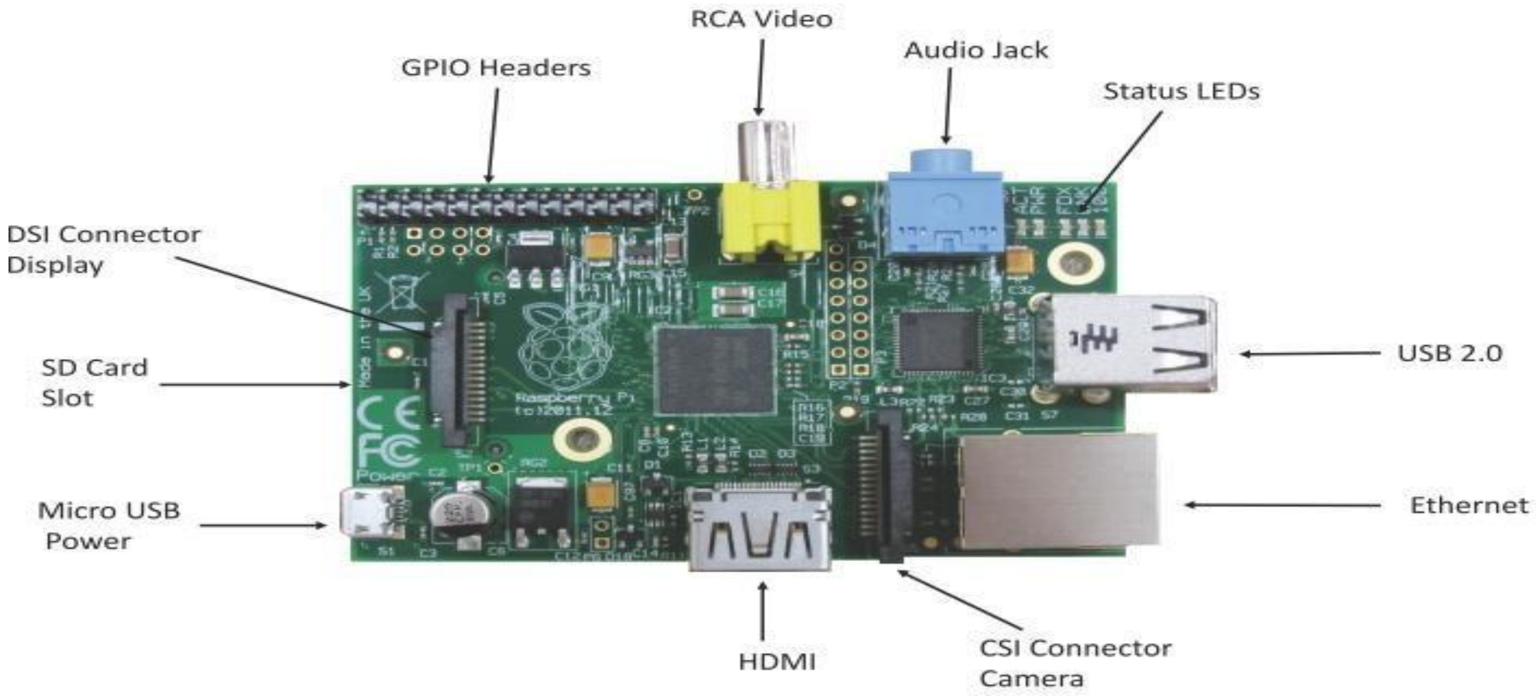


```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

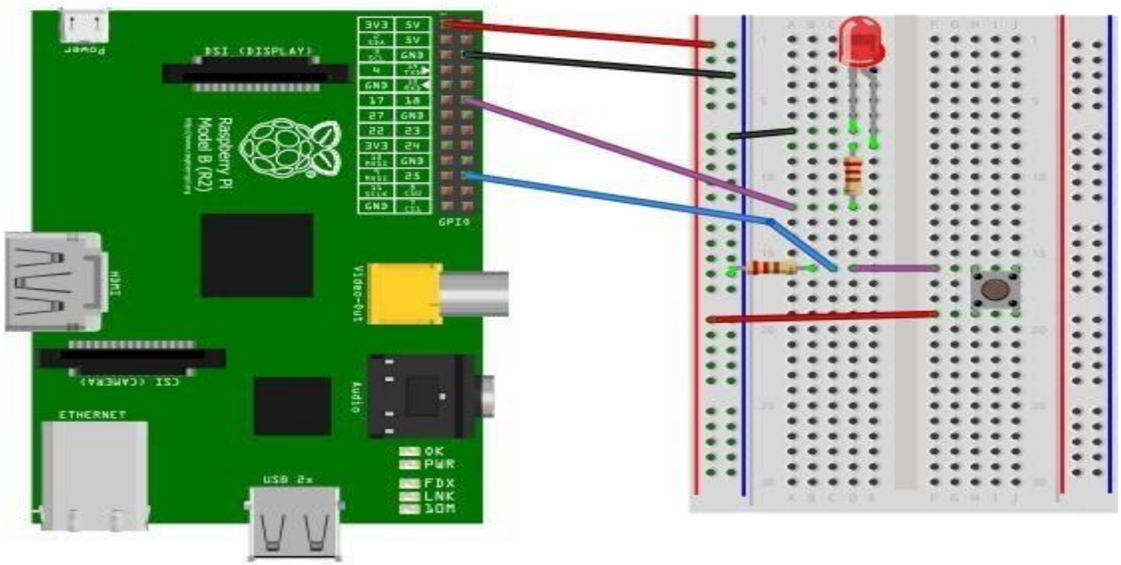
```
1                                                    Pololu A-Star 32U4 on COM4
```

Common IoT Computinng Platforms : Raspberry Pi

Low cost mini computer, allows interfacing sensors though GPIOs, runs Raspbian OS (a Liux variant), supports Python



Interfacing devices to



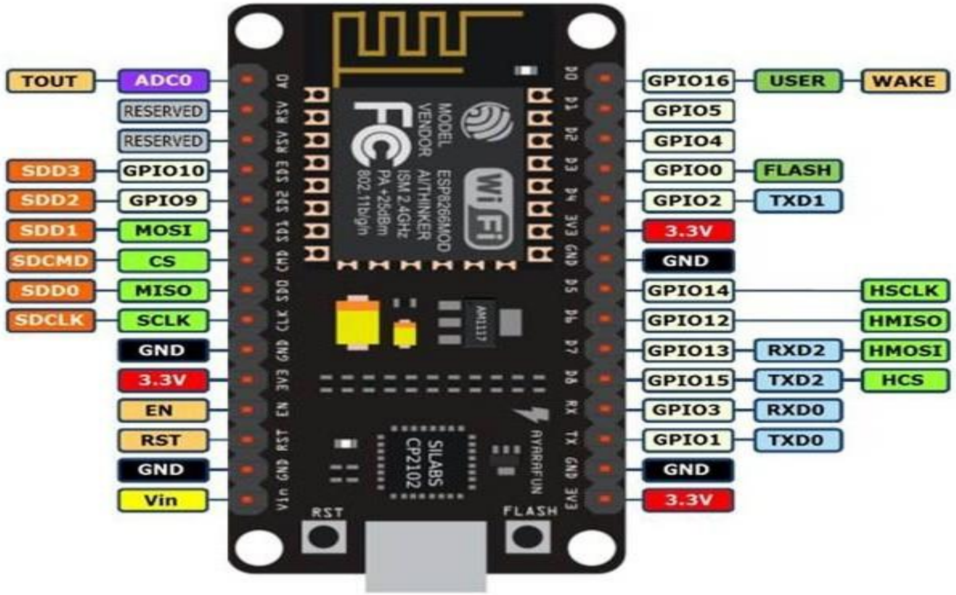
```
from time import sleep import
RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)

 #Switch Pin
 GPIO.setup(25, GPIO.IN) #LED Pin
 GPIO.setup(18, GPIO.OUT) state=false

   def toggleLED(pin): state =

          not state

          GPIO.output(pin, state)

    while True:
            try:

                if (GPIO.input(25) == True):

                     toggleLED(pin)
                sleep(.01)
                except KeyboardInterrupt: exit()
```

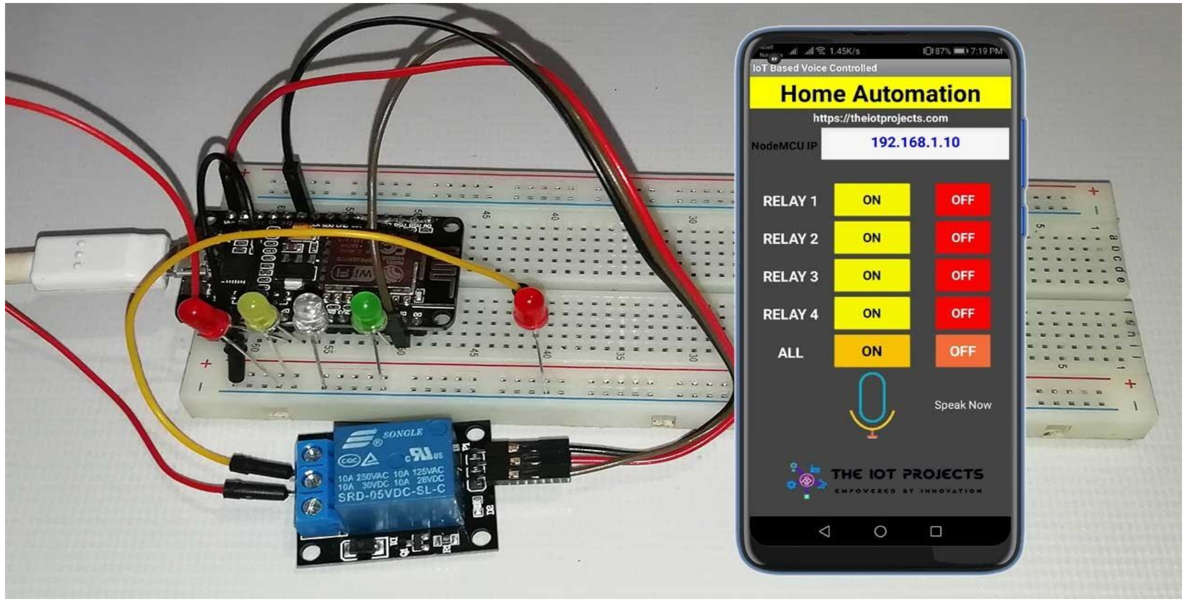
Node MCU

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

- Memory: 128kBytes

- Developer: ESP8266 Opensource Community CPU:

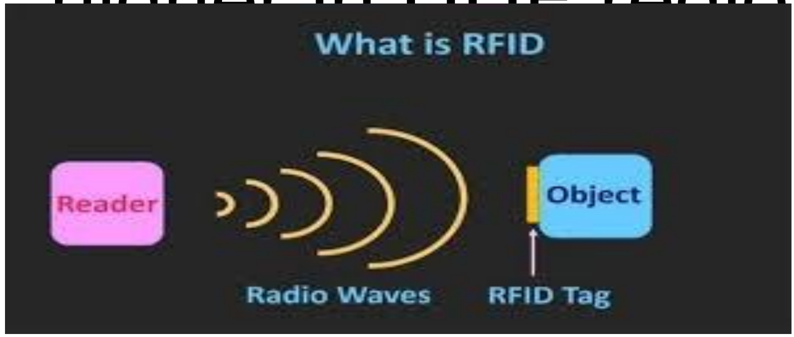


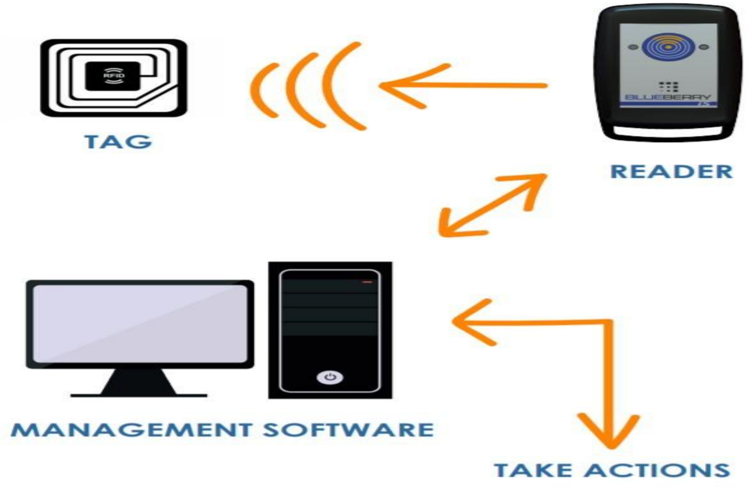


RFID (Radio Frequency ID) Technology in IoT

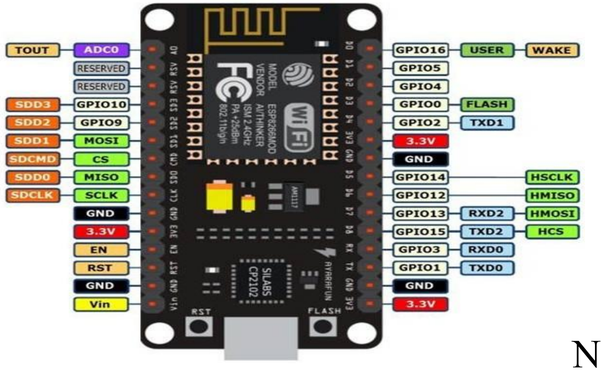
- Identification System – Tagging and Labelling
- Tiny chips : Passive/Active; battery powered when reader near wireless Communication range : 10cm to 200m
- Standard frequency : 120- MHz and higher in UHF regions

ing, in

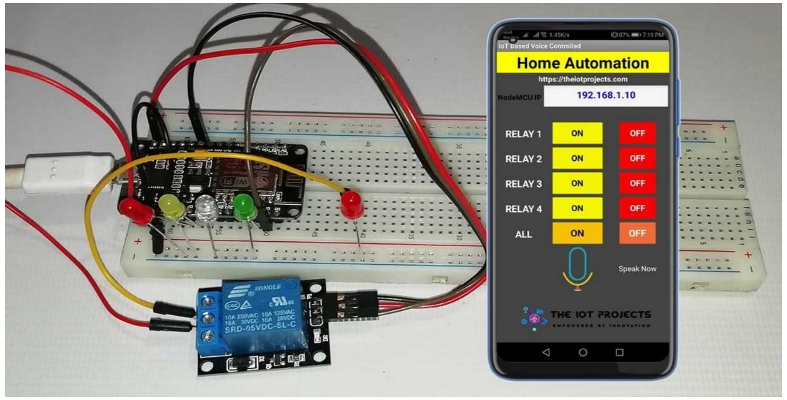

What is RFID

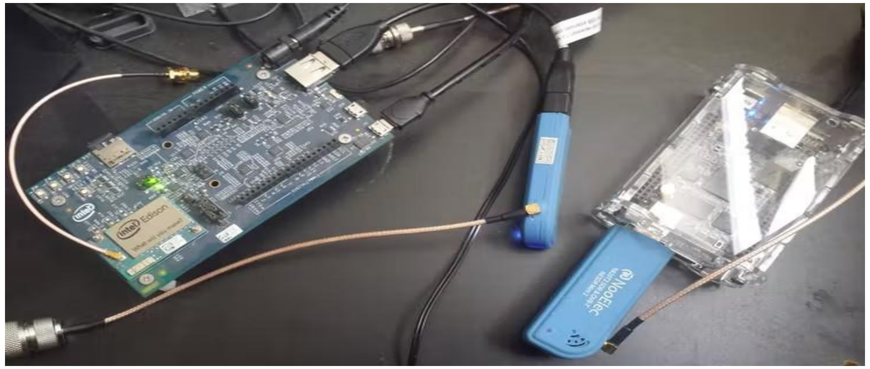


Other IoT Computing Platf

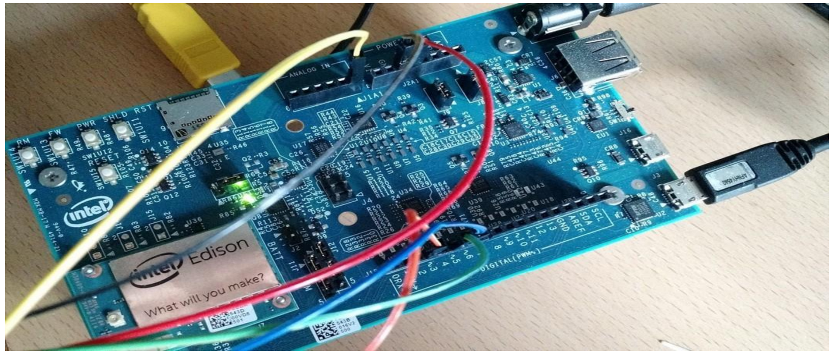


Node MCU





Beagle board



Intel Edison Board

Interfacing Sensors to Microcontrollers

The process of connecting devices together so that they can exchange is called interfacing

In order for these devices to swap their information, they must share a common communication protocol. Communication

protocols are of two types :

Parallel - multi line channel with each line capable of transmitting several bits of data simultaneously. usually require buses of data transmitting across eight, sixteen, or more wires

data is transferred as streams of 0’s and 1’s

Serial - stream their data, one single bit at a time.

operate as little as one wire, usually never more than four, Simple wiring,

serial interface cables can be longer than parallel interface cables since less crosstalk among conductors

Most hardware interfaces are serial interfaces sacrificing potential speed in parallel.

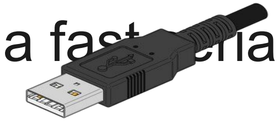
Serial interfaces generally use multiple wires to control the flow and timing of binary information along the primary data wire.

Each type of hardware interface defines a method of communicating between a peripheral and the central processor

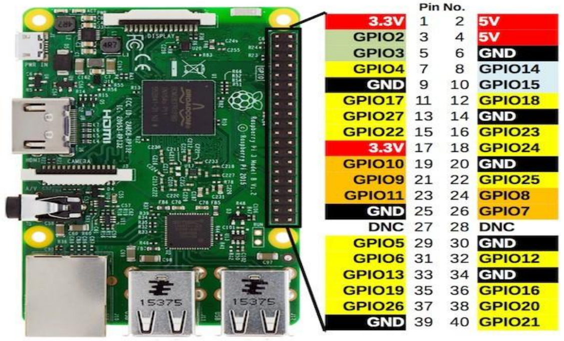
Interfacing Sensors to Microcontrollers

IoT hardware platforms use a number of common interfaces. Sensor and actuator modules can support one or more of these interfaces:

1. **Universal Serial Bus (USB)** - a technology that allows a person to connect an electronic device to a microcontroller. It is a fast serial bus.

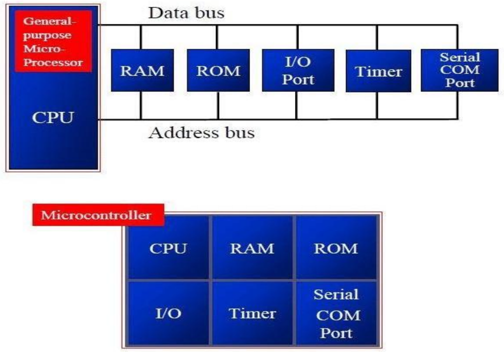


2. **General-purpose input/output pins (GPIO)** - generic pin on an integrated circuit or computer board whose behaviour (whether it is an input or output pin) is controllable by the user at run time. GPIO pins have no predefined purpose, and go unused by default. GPIO pins can be designed to carry digital or ar al pins have only two states: HIGH or LOW.

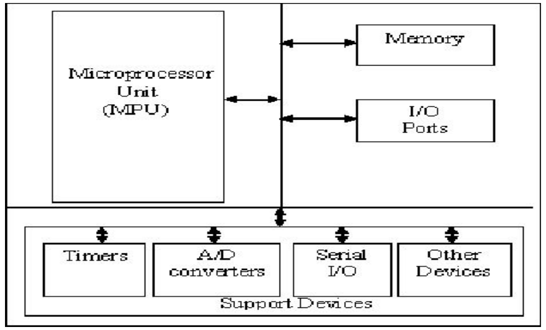


	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

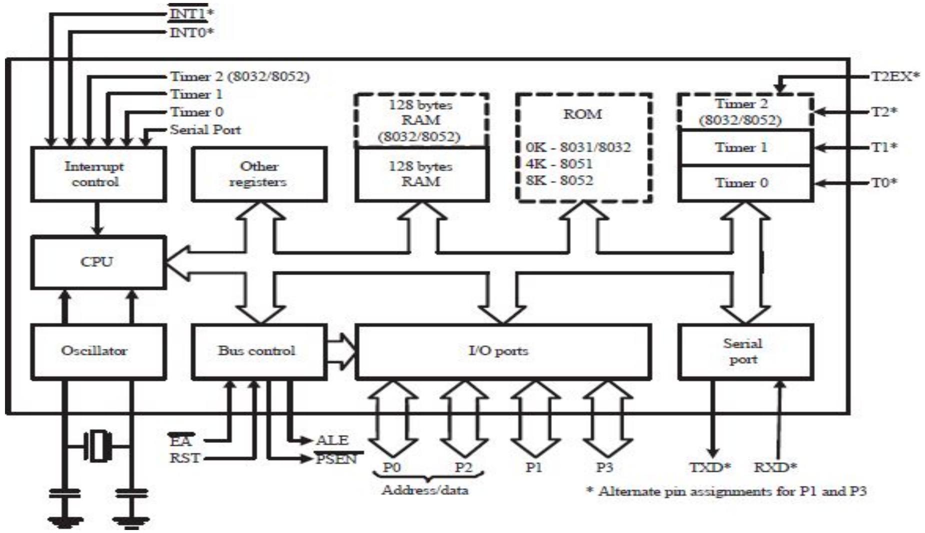
Microcontroller



Microprocessor Vs. Microcontroller



Block Diagram of Microcontroller showing its components

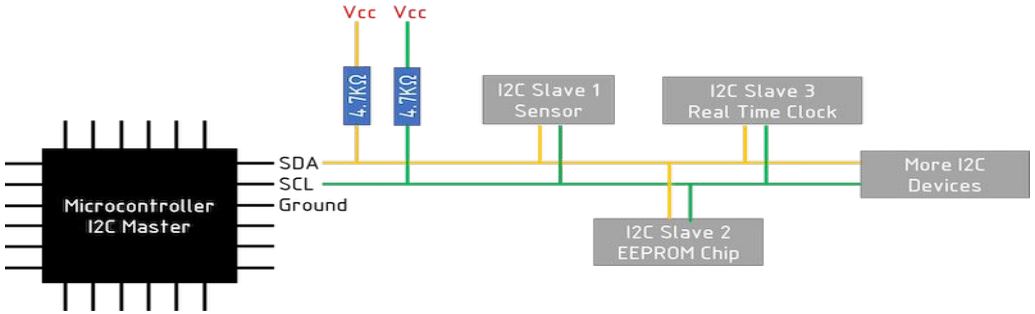


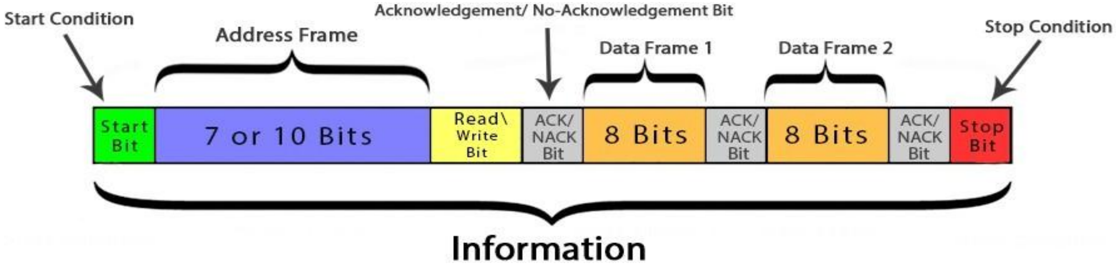
* Alternate pin assignments for P1 and P3

Block Diagram of 8051 Microcontroller

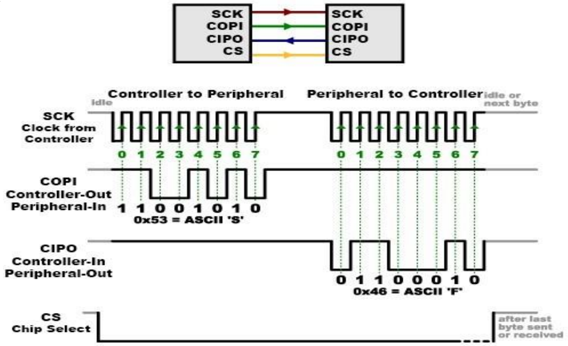
Interfacing Sensors to Microcontrollers

3. Inter-Integrated Circuit serial bus (I2C) - uses a protocol that enables multiple modules to be assigned a discrete address on the bus. I2C is sometimes pronounced “I two C”, “I-I-C”, or “I squared C”. I has two wires, a clock and data wire.



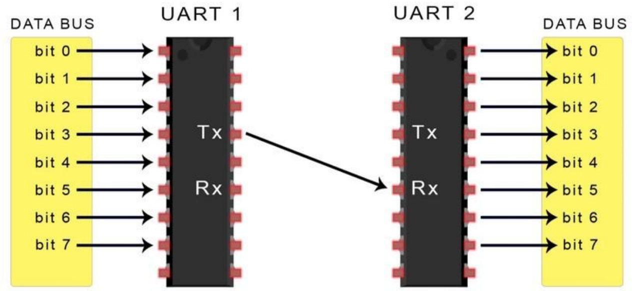


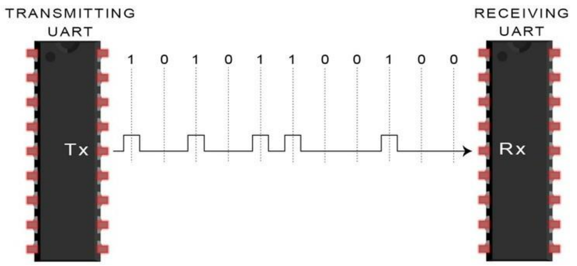
4. Serial Peripheral Interface/Interchange (SPI) - Bus devices employ a master- slave architecture, with a single master and full-duplex communication

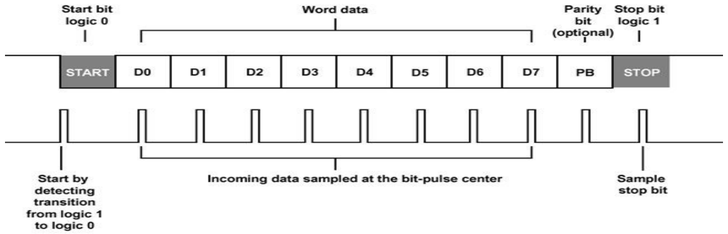


Interfacing Sensors to

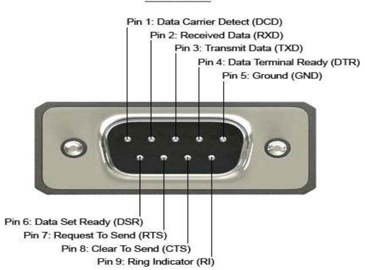
5. Universal Asynchronous Receiver/Transmitter (UART) - it is not a communication protocol like SPI and I2C, but a physical circuit in a microcontroller, or a stand-alone IC**.** devices translate data between serial and parallel forms at the point where the data is acted on by the processor. UART is required when serial data must be laid out in memory in a parallel fashion.

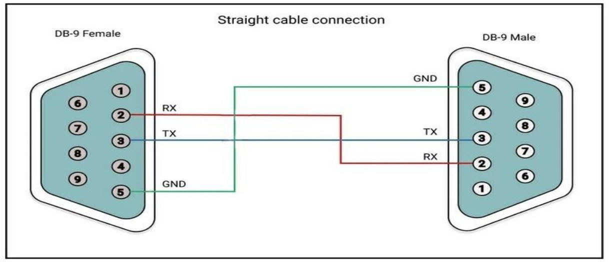






6. Recommended Standard 232(RS232) - is used for obtaining communication between the computer and circuit in order to transfer data

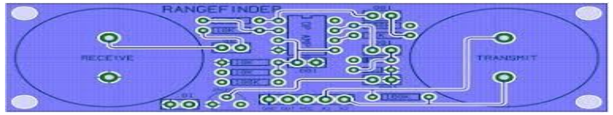

RS232 Pinout
Pin 1: Data Carrier Detect (DCD)
Pin 2: Received Data (RXD)
Pin 3: Transmit Data (TXD)
Pin 4: Data Terminal Ready (DTR)
Pin 5: Ground (GND)
Pin 6: Data Set Ready (DSR)
Pin 7: Request To Send (RTS)
Pin 8: Clear To Send (CTS)
Pin 9: Ring Indicator (RI)

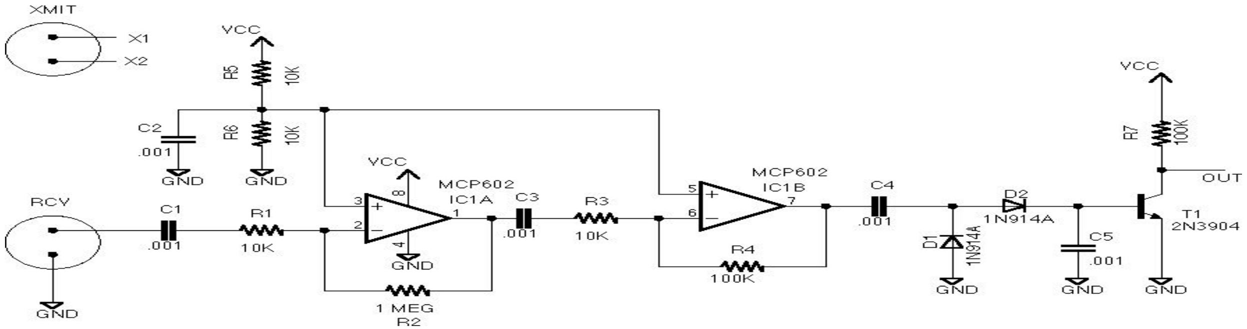


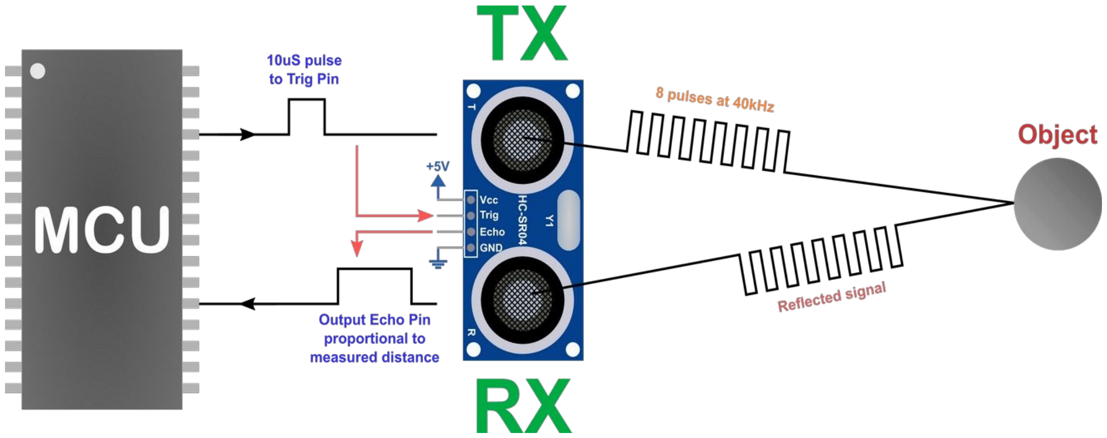
terfacing devices to

- **Interfacing Sonar Range finder to MCU**

Only two inputs are required : INIT : Start transmitting; ECHO – Return signal

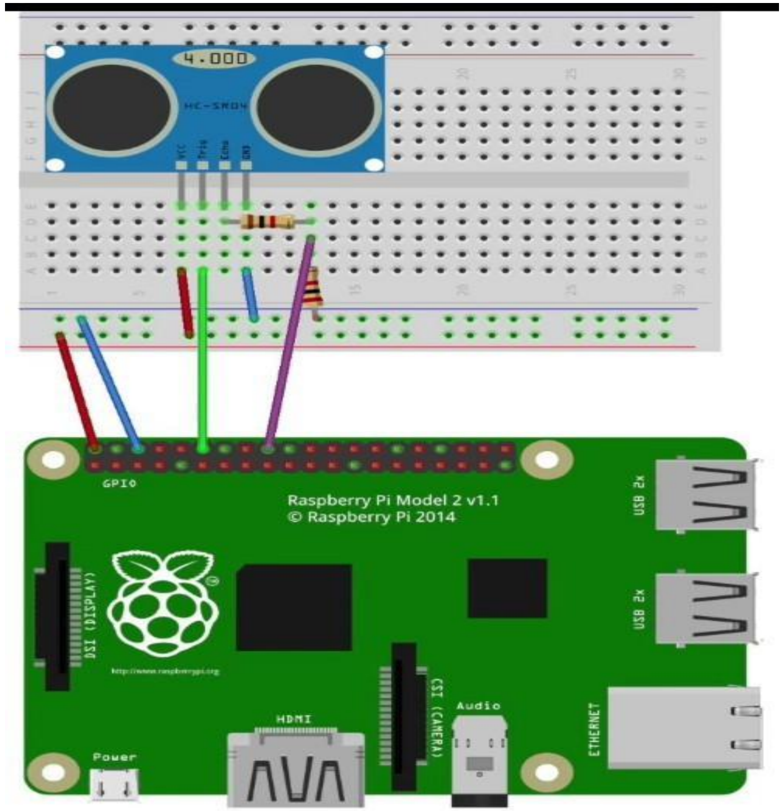


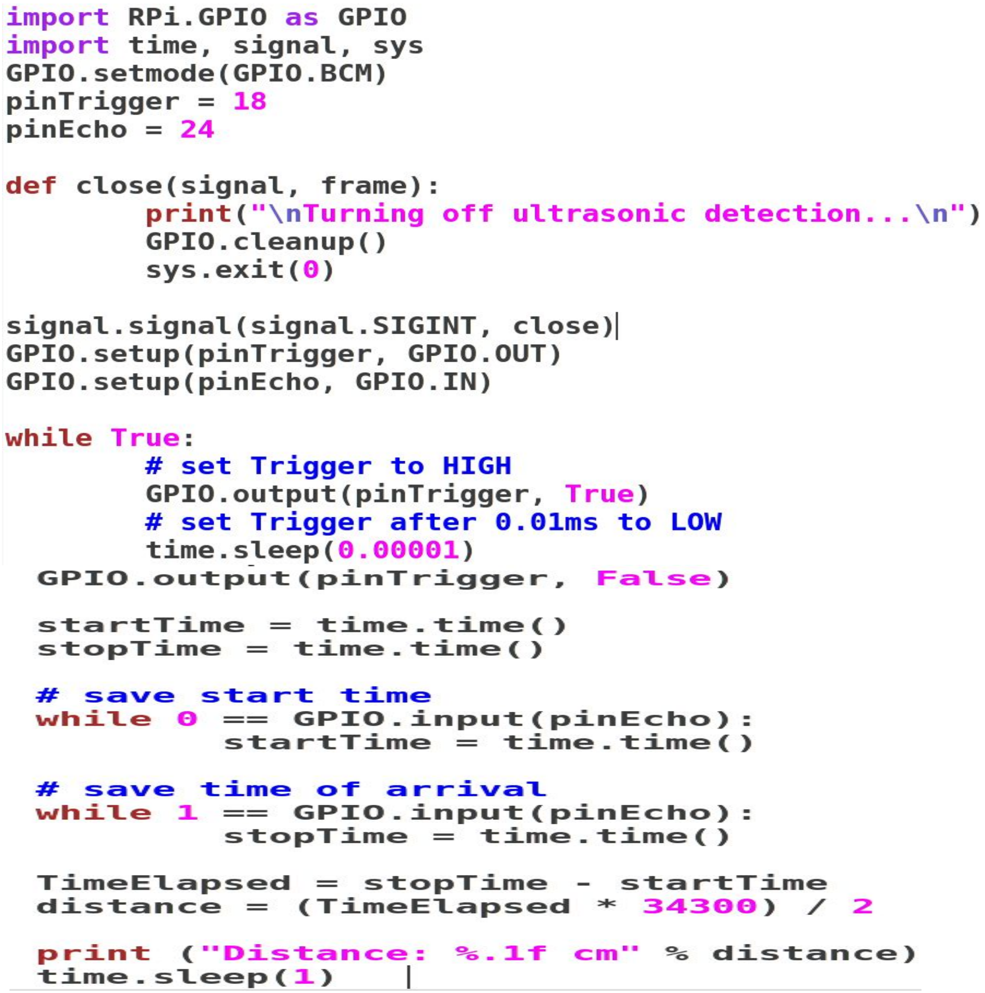




```
// Pseudo-code showing the interfacing technique
    // Setup
        Set the Trig pin as output Set the Echo pin
        as input
        Set the baud rate for serial communication
  // Execution Loop()
{
        //Sets the trigPin on HIGH state for 10 micro seconds
            Write Trig pin = ‘ 1’ ; delay 10microsec; Write Trig pin = ‘ 0’ ;
        // Reads the echoPin, returns the sound wave travel time in microseconds

            Read the echo pin and record the duration of the pulse received;
        // calculate the distance from the duration of the echo pulse and print Distance =

            (duration of the echo pulse * 34300 ) / 2;

            Print Distance on the serial monitor; Insert a suitable
            delay;
}
```

nterface Sonar Range Finder with Raspberry Pi



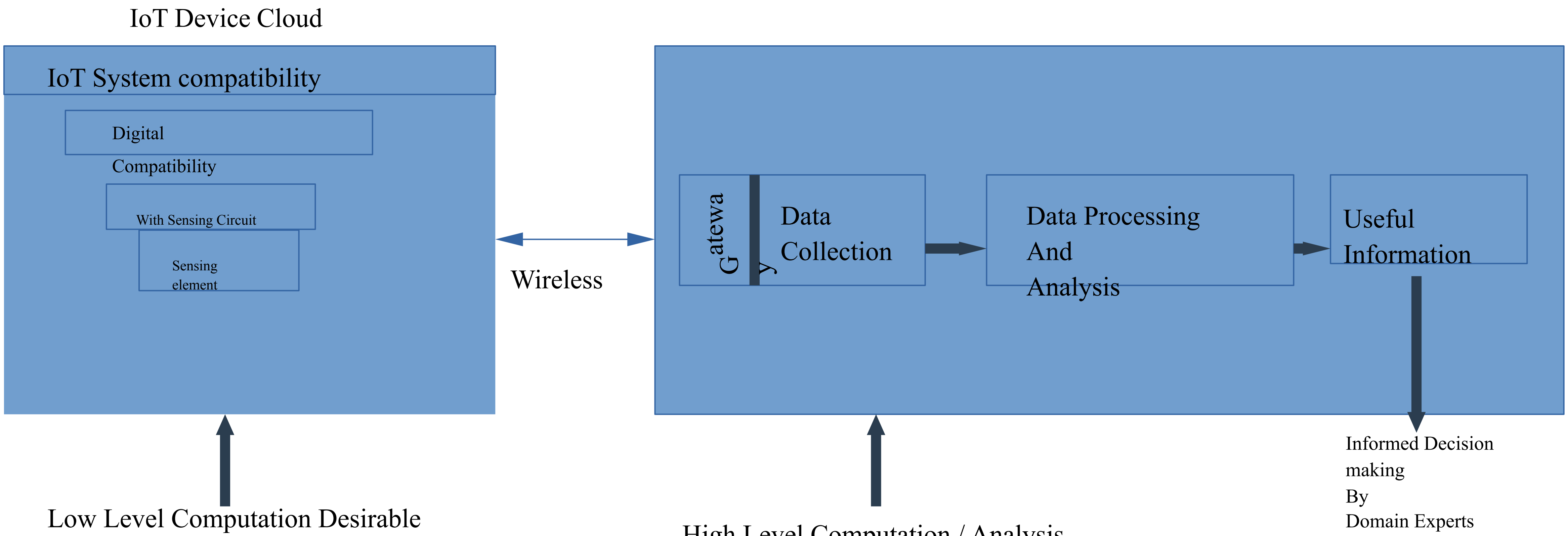


```
import RPi.GPIO as GPIO
import time, signal, sys
GPIO.setmode(GPIO.BCM)
pinTrigger = 18
pinEcho = 24

def close(signal, frame):
        print("\nTurning off ultrasonic detection...\n")
        GPIO.cleanup()
        sys.exit(0)

signal.signal(signal.SIGINT, close)
GPIO.setup(pinTrigger, GPIO.OUT)
GPIO.setup(pinEcho, GPIO.IN)

while True:
        # set Trigger to HIGH
        GPIO.output(pinTrigger, True)
        # set Trigger after 0.01ms to LOW
        time.sleep(0.00001)
 GPIO.output(pinTrigger, False)

 startTime = time.time()
 stopTime = time.time()

 # save start time
 while 0 == GPIO.input(pinEcho):
         startTime = time.time()

 # save time of arrival
 while 1 == GPIO.input(pinEcho):
         stopTime = time.time()

 TimeElapsed = stopTime - startTime
 distance = (TimeElapsed * 34300) / 2

 print ("Distance: %.1f cm" % distance)
 time.sleep(1)
```

Need for Computing

1. Sensors convert real world entities to electrical signals
2. Real world generates raw data that has to be interpreted to extract meaningful information
3. There is a strong need to store, transport and sort data
4. There is a strong need to process data and extract information
5. Right information means informed decision making
6. Informed decision means better life !!!

IoT System: Distribution of Computation



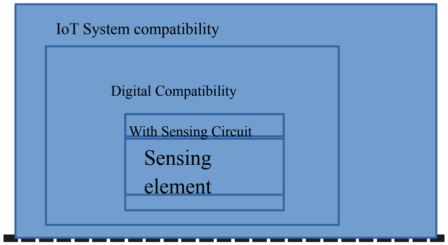
Low Level Computation Desirable

- Increased capabilities at local node
- Reduced requirements of connectivity
- Providing backend with high level information
- Simple data interface

High Level Computation / Analysis

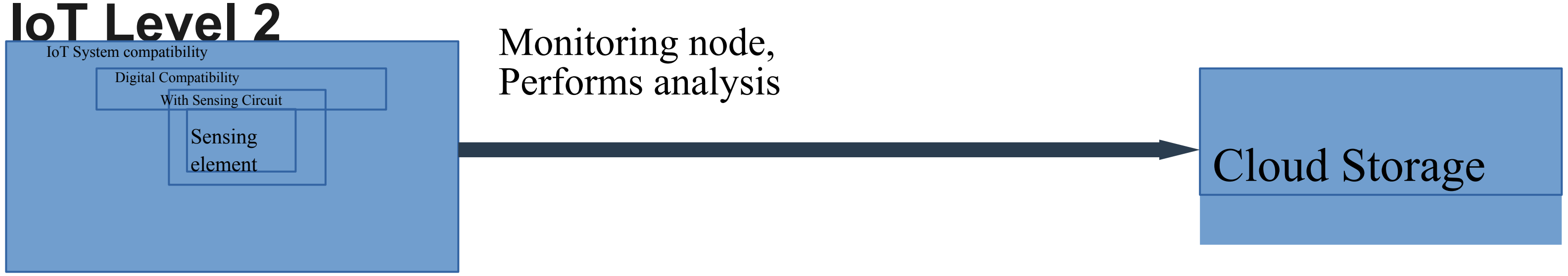
Types of IoT System: Distribution of Computation

IoT Level 1



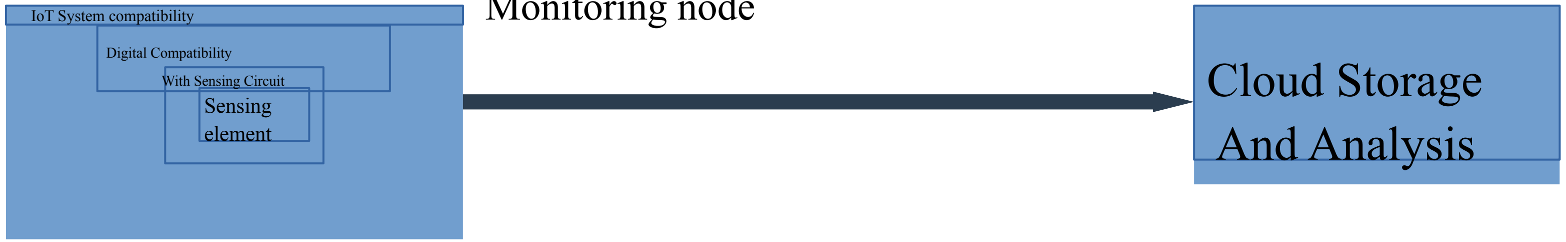
Monitoring node, Performs storage and analysis

IoT Level 2



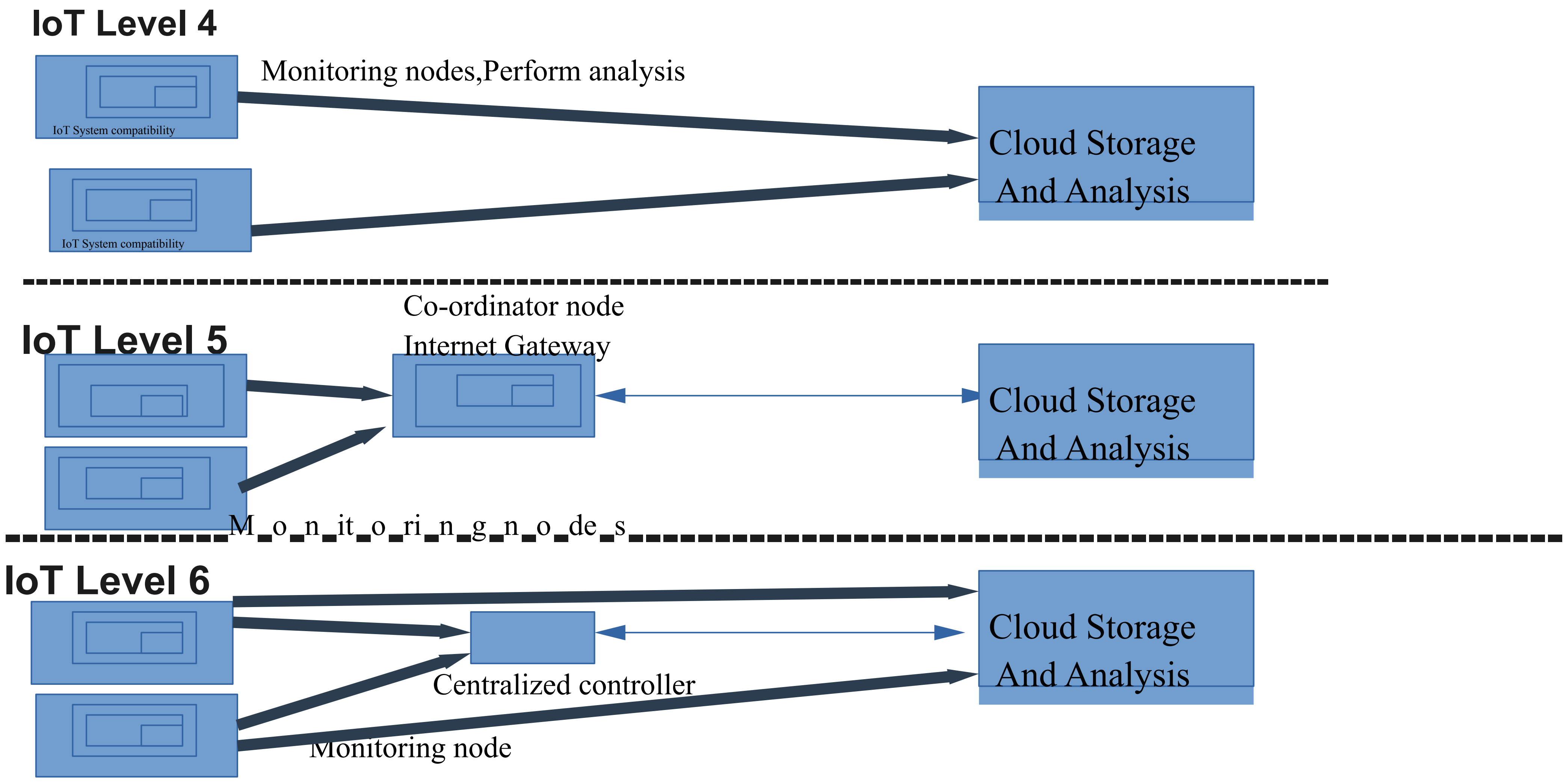
Monitoring node,
Performs analysis

IoT Level 3



Monitoring node

Types of IoT System: Distribution of Computation



IoT Level 4

Monitoring nodes,Perform analysis

IoT System compatibility

IoT System compatibility

Cloud Storage
And Analysis

IoT Level 5

Co-ordinator node
Internet Gateway

Cloud Storage
And Analysis

Monitoring nodes

IoT Level 6

Cloud Storage
And Analysis

Centralized controller

Monitoring node