Internet of Things

Short notes for B.Tech CSE / IT / ECE · BCA · MCA · M.Tech

Uncodemy Handwritten Notes · free for students

I. Introduction

- **Internet of Things (IoT)** = a network of physical objects with sensors, software and connectivity that collect and share data over the internet.
- The term is credited to Kevin Ashton (1999).
- "Thing" = any device with a unique identity that can sense, act or communicate: smart bulb, fitness band, soil sensor.

➤ Characteristics

1. Connectivity
2. Sensing and actuation
3. Unique identity (IP address, RFID tag)
4. Dynamic and self-adapting
5. Huge scale (very many devices)
6. Heterogeneity (many hardware and network types)
7. Low power and limited resources

➤ IoT vs M2M

Point	M2M	IoT
Communication	Device to device, point to point	Devices to cloud / internet
Protocols	Often proprietary, non-IP	Mostly IP-based
Scale	Limited	Very large
Data use	Local	Cloud analytics

➤ Applications

- Smart home, smart city (traffic, parking, lighting), healthcare (remote monitoring), agriculture (soil moisture, irrigation), industry (IIoT, predictive maintenance), logistics (asset tracking), wearables.

2. IoT Architecture

➤ Three-layer architecture

Layer	Job	Examples
Perception	Sense and act on the physical world	Sensors, actuators, RFID
Network	Move data	Wi-Fi, Zigbee, LTE, gateway
Application	Give services to users	Smart home app, dashboard

➤ Five-layer architecture

- Perception → Transport → Processing (middleware) → Application → Business.

```
[Sensors/Actuators] -> [Gateway] -> [Cloud / Server] -> [App]
     perception          network      processing        application
```

➤ Edge, fog and cloud

- **Cloud computing**: central, huge storage and processing, higher latency.
- **Fog computing**: processing in local nodes (gateways, routers) between devices and cloud.
- **Edge computing**: processing on or very near the device itself.
- Benefit of fog / edge: lower latency, less bandwidth, works when internet is weak.

> **EXAM POINT** Fog = a layer between edge devices and the cloud. Edge = computing on the device or right next to it.

3. Sensors and Actuators

- **Sensor**: converts a physical quantity into an electrical signal.
- **Actuator**: converts an electrical signal into physical action.
- **Transducer**: general term for any device converting one form of energy to another.

Sensor	Measures	Example part
Temperature / humidity	Heat, moisture in air	DHT11, DHT22, LM35
Ultrasonic	Distance (echo)	HC-SR04
PIR	Motion of warm bodies	HC-SR501
LDR	Light level	Photoresistor
Gas	Smoke, LPG	MQ-2
Accelerometer / gyro	Motion, tilt	MPU6050
Soil moisture	Water in soil	Capacitive probe

- Actuators: relay, DC motor, servo motor, stepper motor, buzzer, solenoid valve, LED.

FORMULA Ultrasonic distance = (speed of sound × echo time) / 2. Speed of sound ≈ 343 m/s in air.

EXAMPLE Echo time = 2 ms. Distance = (343 × 0.002) / 2 = 0.343 m = 34.3 cm.

➤ Analog vs digital

- Analog sensor gives a varying voltage; read with an **ADC**.

FORMULA ADC value = (Vin / Vref) × (2^n − 1), n = ADC bits.

EXAMPLE 10-bit ADC, Vref = 5 V, Vin = 2.5 V → (2.5 / 5) × 1023 = 511.5, read as about 511.

4. IoT Hardware Platforms

Board	Type	Key points
Arduino Uno	Microcontroller (ATmega328P)	16 MHz, 14 digital pins (6 PWM), 6 analog inputs, no OS
ESP8266 / ESP32	Microcontroller with Wi-Fi	ESP32 also has Bluetooth, dual core
Raspberry Pi	Single-board computer	Runs Linux, has GPIO, USB, HDMI

- **Microcontroller** = CPU + RAM + flash + I/O on one chip; runs one program. **Microprocessor** needs external memory and peripherals.

Arduino program structure

```
int led = 13;
void setup() {
  pinMode(led, OUTPUT);
}
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

- `setup()` runs once; `loop()` runs forever.
- Functions: `digitalRead()`, `digitalWrite()`, `analogRead()`, `analogWrite()` (PWM), `Serial.begin(9600)`.

Raspberry Pi with Python

```
import RPi.GPIO as GPIO, time
GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
GPIO.output(18, GPIO.HIGH); time.sleep(1)
GPIO.cleanup()
```

Wired interfaces

- **UART**: 2 wires (TX, RX), asynchronous.
- **I2C**: 2 wires (SDA, SCL), many devices by address.
- **SPI**: 4 wires (MOSI, MISO, SCK, SS), fast, full duplex.

5. Communication Technologies

Technology	Range	Data rate	Power	Use
RFID / NFC	cm to few m	Low	Very low (passive tags)	Tags, payments
Bluetooth Low Energy	~10–100 m	Up to 1–2 Mbps	Low	Wearables
Zigbee (IEEE 802.15.4)	~10–100 m	250 kbps	Low	Home automation, mesh
Wi-Fi (IEEE 802.11)	~50 m indoor	High	High	Cameras, home devices
LoRaWAN	Several km	Very low	Very low	Agriculture, smart city
NB-IoT / LTE-M	Cellular range	Low	Low	Meters, trackers

- **LPWAN** (Low Power Wide Area Network) = long range + low power + low data rate (LoRaWAN, Sigfox, NB-IoT).
- **6LoWPAN** = IPv6 over low-power wireless networks (802.15.4), uses header compression.
- **IPv6** is needed because IPv4 has only about 4.3 billion addresses (2^32).

TIP Range up → data rate and power usually go the other way. LPWAN trades speed for range and battery life.

6. IoT Protocols

➤ MQTT

- Lightweight publish-subscribe protocol over TCP.
- **Broker** sits in the middle. Publishers send to a **topic**, subscribers receive from it.
- QoS levels: **0** at most once, **1** at least once, **2** exactly once.
- Default port 1883 (8883 with TLS). Example topic: `home/kitchen/temp`.

➤ CoAP

- Constrained Application Protocol, REST-like over UDP. Methods GET, POST, PUT, DELETE.

- Default port 5683. Message types: confirmable, non-confirmable, acknowledgement, reset.

➤ MQTT vs CoAP vs HTTP

Point	MQTT	CoAP	HTTP
Transport	TCP	UDP	TCP
Model	Publish-subscribe	Request-response	Request-response
Overhead	Small	Very small	Large
Best for	Telemetry, many subscribers	Very constrained devices	Web, normal apps

- Others: **AMQP** (reliable message queues), **XMPP** (chat-style messaging), **DDS** (real-time data sharing), **WebSocket**.

7. IoT Data, Cloud and Analytics

- IoT data: high volume, fast stream, many types (big data V's: volume, velocity, variety, veracity, value).
- IoT cloud platforms provide device registry, data storage, rules and dashboards.
- Steps: collect → filter at edge → send to cloud → store (time-series database) → analyse → act / visualise.
- **Digital twin**: a virtual model of a physical device kept in sync with live data.
- Machine learning in IoT: anomaly detection, predictive maintenance, demand forecasting.

8. IoT Security and Challenges

- Weak points: default passwords, no updates, unencrypted traffic, physical tampering, insecure cloud APIs.
- Attacks: botnets made of IoT devices (DDoS), eavesdropping, spoofing, replay, firmware tampering.
- Protection:
  - Unique strong credentials per device
  - Encryption (TLS / DTLS)
  - Secure boot and signed firmware updates
  - Network segmentation

  - Lightweight cryptography for small devices
- Other challenges: interoperability (no single standard), power, scalability, privacy of personal data.

9. Case Studies

- **Smart home**: PIR + LDR + relay → lights on when someone enters a dark room.
- **Smart agriculture**: soil moisture sensor → ESP32 → MQTT → cloud → pump turns on below a threshold.
- **Health monitor**: pulse and temperature sensors → phone over BLE → doctor dashboard with alerts.
- **Smart parking**: ultrasonic sensor per slot → LoRa → app shows free slots.

10. Most Asked Questions (practice)

1. Define IoT. Explain its characteristics and applications.
2. Explain the three-layer and five-layer IoT architecture.
3. Difference between IoT and M2M.
4. Sensor vs actuator. Explain any four sensors.
5. Compare Arduino and Raspberry Pi. Write an LED blink program.
6. Compare Zigbee, Bluetooth LE, Wi-Fi and LoRaWAN.
7. Explain MQTT with QoS levels. MQTT vs CoAP.
8. What is fog and edge computing? Why are they needed in IoT?
9. Explain security issues in IoT and their solutions.
10. Design an IoT system for smart agriculture with a block diagram.

> **TIP** Architecture, MQTT / CoAP, sensors and one case study with a block diagram cover most of the paper. Draw neat block diagrams.