

# UNIT 2

## Sensors, Actuators and Interfacing

(18 Hours) — Internet of Things (CACS460)

### ⚠ Please Note

These notes were prepared quickly and **may contain errors or omissions**. Please cross-check important definitions, formulas, code and diagrams with your textbook, class notes and the official TU syllabus before relying on them in exams.

## 2.1 Roles of Sensors and Actuators

A **sensor** detects a physical quantity (temperature, light, motion...) and converts it into an electrical signal. An **actuator** does the opposite: it receives an electrical command from a controller and converts it into physical action (rotation, heat, light, sound, valve opening). Sensors are the “senses” of an IoT system and actuators are its “muscles”.

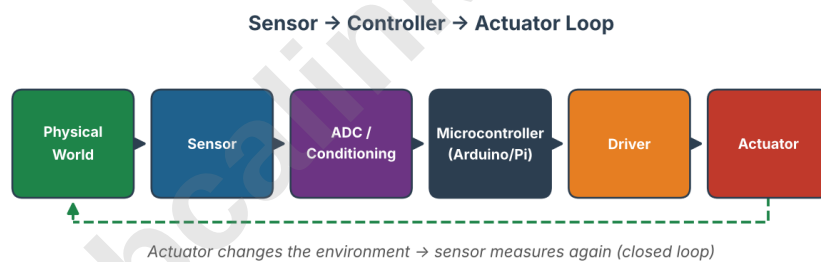


Figure 2.1 — Sensor → controller → actuator closed loop

|                  | Sensor                                  | Actuator                                                         |
|------------------|-----------------------------------------|------------------------------------------------------------------|
| Function         | Measures a physical quantity            | Produces a physical effect                                       |
| Signal direction | Environment → electrical signal (input) | Electrical signal → environment (output)                         |
| Examples         | DHT22, LDR, PIR, ultrasonic HC-SR04     | Relay, DC/servo/stepper motor, buzzer, LED, solenoid valve, pump |
| Typical role     | Monitoring and data collection          | Control and response                                             |

## 2.2 Types of Sensors

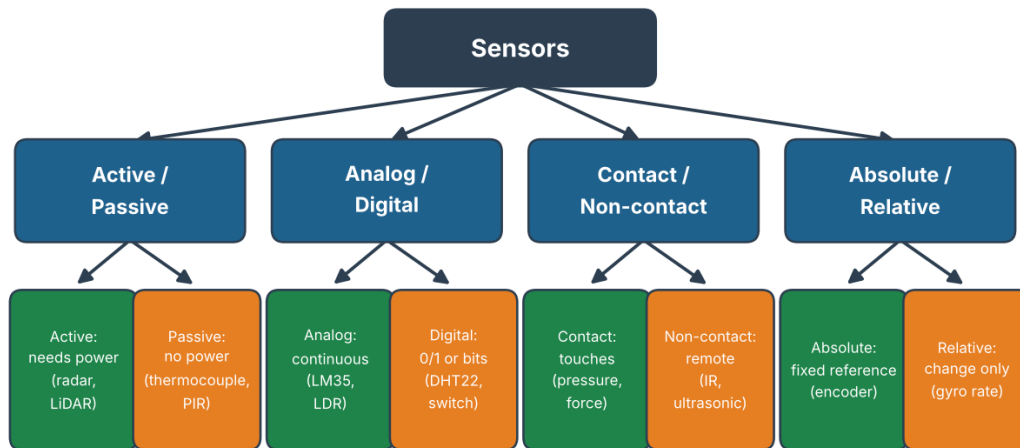


Figure 2.2 — Classification of sensors

## Active and Passive Sensors

- **Active sensors** need an external power/excitation signal to work and often emit energy into the environment. Examples: ultrasonic, radar, LiDAR, active IR proximity, strain gauge (needs excitation voltage).
- **Passive sensors** only detect energy that already exists, or generate their own signal. Examples: thermocouple, photodiode, PIR motion sensor, microphone.

## Analog and Digital Sensors

- **Analog sensors** output a continuous voltage/current proportional to the measured value (LM35, LDR, potentiometer). They need an ADC to be read by a digital controller.
- **Digital sensors** output discrete values — a binary level or a data stream over I<sup>2</sup>C/SPI/1-Wire (DHT22, DS18B20, BMP180). They have built-in ADC and are less noise-prone.

## Contact and Non-contact Sensors

- **Contact sensors** must touch the object (limit switch, pressure sensor, thermocouple probe, strain gauge).
- **Non-contact sensors** measure from a distance (IR thermometer, ultrasonic, PIR, camera, radar).

## Absolute and Relative Sensors

- **Absolute sensors** give a value against a fixed reference (absolute encoder, thermistor in °C, barometric pressure).
- **Relative sensors** give a change with respect to a previous position/value (incremental encoder, gyroscope rate, differential pressure).

## 2.3 Working of Common Sensors

### Position, Occupancy and Motion

- **Position:** potentiometer (resistance changes with shaft angle), rotary encoder, Hall-effect sensor (magnetic field), LVDT.
- **Occupancy / presence:** PIR and ultrasonic sensors, capacitive touch, IR break-beam, CO<sub>2</sub> sensors.
- **Motion — PIR:** a pyroelectric element detects changes in infrared radiation emitted by warm bodies. When a person moves across its two sensing slots, the differential signal triggers a HIGH output.

### Velocity and Acceleration

- **Velocity:** tachometer, optical/magnetic encoders (pulses per second), Doppler radar.
- **Acceleration:** accelerometer (e.g., ADXL335, MPU6050). A tiny MEMS proof mass moves under acceleration and changes capacitance, giving output proportional to acceleration ( $a = F/m$ ).
- **Gyroscope:** measures angular velocity using the Coriolis effect on a vibrating MEMS structure; IMU = accelerometer + gyro (+ magnetometer).

### Force and Pressure

- **Force:** strain gauge/load cell. Deformation changes resistance ( $\Delta R/R = GF \cdot \epsilon$ ); a Wheatstone bridge converts it to a voltage. Force-sensitive resistor (FSR) falls in resistance as force rises.
- **Pressure:** piezoresistive or capacitive diaphragm sensors (BMP180/BMP280 for atmospheric pressure, MPX series for gauge pressure). Pressure  $P = F/A$ .

## Flow

- **Hall-effect water-flow sensor (YF-S201):** a turbine rotates with flow and a Hall sensor outputs pulses; frequency  $\propto$  flow rate.
- Other types: ultrasonic, electromagnetic, thermal mass flow meters.

## Acoustic

- **Microphone** (condenser, MEMS, electret) converts sound pressure into voltage. Used for sound-level monitoring, clap switches, voice interfaces, gunshot/glass-break detection.

## Humidity

- **Capacitive humidity sensors** (DHT11/DHT22, SHT31): a polymer dielectric absorbs moisture, changing capacitance. **Resistive** types change resistance. Relative humidity (RH) is reported in %.

## Light

- **LDR (photoresistor):** resistance falls as light intensity increases; read through a voltage divider.
- **Photodiode / phototransistor:** generates current proportional to light; fast response.
- **BH1750 / TSL2561:** digital lux sensors over I<sup>2</sup>C.

## Radiation

- **Geiger-Müller tube:** ionising radiation causes a pulse of current in gas; counts per minute indicate radiation level.
- **UV sensors (e.g., GUVVA-S12SD, ML8511)** measure UV index; **IR sensors** detect infrared.

## Temperature

| Sensor               | Principle                                           | Output              | Notes                                                        |
|----------------------|-----------------------------------------------------|---------------------|--------------------------------------------------------------|
| Thermistor (NTC/PTC) | Resistance changes with temperature (non-linear)    | Analog              | Cheap, sensitive; needs divider and calibration              |
| Thermocouple         | Seebeck effect: junction of two metals generates mV | Analog (mV)         | Very wide range (up to ~1000+ °C); needs amplifier (MAX6675) |
| RTD (Pt100)          | Platinum resistance rises linearly with temperature | Analog              | Accurate, stable, costlier                                   |
| LM35                 | IC giving 10 mV per °C                              | Analog              | Simple: $T(^{\circ}\text{C}) = V \times 100$                 |
| DS18B20              | Digital IC, 9–12 bit                                | 1-Wire digital      | Waterproof probe versions; unique 64-bit ID                  |
| DHT11 / DHT22        | Thermistor + capacitive humidity                    | Single-wire digital | DHT22 more accurate than DHT11                               |

## Chemical (Gas) Sensors

- **MQ series (MQ-2, MQ-7, MQ-135):** a metal-oxide semiconductor (e.g.,  $\text{SnO}_2$ ) heated by a coil changes resistance when target gas molecules are adsorbed. Detects smoke, LPG, CO, air quality. Needs warm-up and calibration.
- **Electrochemical sensors** produce current proportional to gas concentration ( $\text{CO}$ ,  $\text{O}_2$ ); **pH sensors** measure ion activity in liquids.

## Biosensors

A **biosensor** combines a biological recognition element (enzyme, antibody, DNA) with a transducer that converts the biological response into an electrical signal. Examples: glucose meter (enzyme glucose oxidase on a test strip), pulse oximeter ( $\text{SpO}_2$  using red/IR LEDs and photodiode, e.g., MAX30100), ECG and heart-rate sensors (AD8232), body-temperature sensors.

## Camera Sensors

A camera uses a **CCD or CMOS** image sensor — a grid of photosites that convert photons to charge/voltage. In IoT, cameras (Raspberry Pi Camera, ESP32-CAM) feed computer-vision tasks such as surveillance, face/number-plate recognition, crop-health monitoring and people counting. They generate large data volumes, so edge processing is often used.

## 2.4 Actuators

| Actuator               | Working                                                                     | Driven by                 |
|------------------------|-----------------------------------------------------------------------------|---------------------------|
| Relay                  | Electromagnet switches a high-voltage/current load using a low-power signal | Transistor / relay module |
| DC motor               | Rotates when current flows; speed via PWM, direction via H-bridge           | L293D / L298N driver      |
| Servo motor            | Rotates to a commanded angle (0–180°) using PWM pulse width 1–2 ms          | PWM pin directly          |
| Stepper motor          | Moves in discrete steps; precise positioning                                | ULN2003 / A4988 driver    |
| Solenoid valve         | Coil opens/closes a fluid valve                                             | Relay or MOSFET           |
| Buzzer / LED / display | Sound, light or text output                                                 | GPIO pin (with resistor)  |

*Note: Never drive motors or relays directly from a microcontroller pin; use a driver/transistor and a flyback diode.*

## 2.5 Development Boards

### Arduino

**Arduino** is an open-source microcontroller board family (commonly Arduino Uno with ATmega328P). It has no operating system; code runs directly on the chip (*bare-metal*), making it ideal for real-time sensing and control with very low power.

| Feature               | Arduino Uno                                 |
|-----------------------|---------------------------------------------|
| Microcontroller       | ATmega328P, 8-bit, 16 MHz                   |
| Operating voltage     | 5 V (logic level)                           |
| Flash / SRAM / EEPROM | 32 KB / 2 KB / 1 KB                         |
| Digital I/O pins      | 14 (6 can give PWM)                         |
| Analog input pins     | 6 (10-bit ADC, 0–1023)                      |
| Interfaces            | UART, SPI, I <sup>2</sup> C                 |
| Programming           | Arduino IDE (C/C++ based sketches) over USB |

### Arduino Installation and Programming

- Download and install the **Arduino IDE**, connect the board by USB.
- Select **Tools → Board → Arduino Uno** and the correct **Port**.

- Write a sketch with `setup()` (runs once) and `loop()` (repeats forever), then click Upload.

### Example 2.1 — Blink LED and read LM35 temperature sensor

Task: print room temperature on Serial Monitor and blink the LED on pin 13 as a heartbeat.

```
const int LM35 = A0;
void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  int adc = analogRead(LM35);           // 0..1023
  float volts = adc * (5.0 / 1023.0);   // convert to volts
  float tempC = volts * 100.0;          // LM35: 10 mV per deg C
  Serial.print("Temp = "); Serial.println(tempC);
  digitalWrite(13, HIGH); delay(500);
  digitalWrite(13, LOW);  delay(500);
}
```

## Raspberry Pi

**Raspberry Pi** is a credit-card-sized single-board *computer* (SBC) running Linux (Raspberry Pi OS). It has a multi-core ARM CPU, RAM, USB, HDMI, Ethernet/WiFi, Bluetooth and a 40-pin GPIO header. It can run full programs, web servers, databases and AI models, so it often acts as an IoT gateway.

| Feature       | Raspberry Pi (e.g., Pi 4 Model B)                                                    |
|---------------|--------------------------------------------------------------------------------------|
| Processor     | Quad-core ARM Cortex-A72, 1.5 GHz (64-bit)                                           |
| Memory        | 1–8 GB RAM depending on model; storage via microSD                                   |
| Connectivity  | WiFi, Bluetooth 5.0, Gigabit Ethernet, USB 2.0/3.0, HDMI                             |
| GPIO          | 40 pins at 3.3 V logic (NOT 5 V tolerant), includes I <sup>2</sup> C, SPI, UART, PWM |
| Analog input  | None — external ADC (e.g., MCP3008) needed                                           |
| OS / Language | Linux; Python is the most common language                                            |

## Raspberry Pi Installation (Setup)

- Download **Raspberry Pi Imager** and write **Raspberry Pi OS** to a microSD card (enable SSH/WiFi in advanced options).
- Insert the card, connect power (and monitor/keyboard, or use SSH headless).
- Boot and update: `sudo apt update && sudo apt upgrade`.

- Enable interfaces (I<sup>2</sup>C, SPI, camera) using `sudo raspi-config`.
- Install libraries, e.g. `pip install RPi.GPIO adafruit-circuitpython-dht`.

| Basis             | Arduino                        | Raspberry Pi                         |
|-------------------|--------------------------------|--------------------------------------|
| Type              | Microcontroller board          | Single-board computer                |
| OS                | None (bare-metal firmware)     | Linux (full OS)                      |
| Speed / memory    | 16 MHz / 2 KB SRAM             | GHz-class / GBs RAM                  |
| Analog input      | Built-in ADC                   | No ADC (needs external)              |
| Power use         | Very low                       | Higher (2–7 W)                       |
| Real-time control | Excellent                      | Limited (OS scheduling)              |
| Best for          | Simple sensing/actuating nodes | Gateways, vision, analytics, servers |
| Programming       | C/C++ (Arduino IDE)            | Python, C, Java, etc.                |

## 2.6 Interfacing and Programming with Python on Raspberry Pi

Python on the Pi accesses pins using the **RPi.GPIO** or **gpiozero** libraries. Always check numbering: **BCM** (Broadcom GPIO number) vs **BOARD** (physical pin number).

### Example 2.2 — Blink an LED on GPIO 17

Wiring: GPIO17 → 330 Ω resistor → LED anode; LED cathode → GND.

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(17, GPIO.OUT)
try:
    while True:
        GPIO.output(17, GPIO.HIGH); time.sleep(1)
        GPIO.output(17, GPIO.LOW);  time.sleep(1)
except KeyboardInterrupt:
    pass
finally:
    GPIO.cleanup()
```

### Example 2.3 — PIR motion sensor controlling a buzzer

Wiring: PIR OUT → GPIO4 (input), buzzer + → GPIO18, buzzer – → GND, PIR VCC → 5 V, GND → GND.

```
import RPi.GPIO as GPIO
import time

PIR, BUZ = 4, 18
GPIO.setmode(GPIO.BCM)
GPIO.setup(PIR, GPIO.IN)
GPIO.setup(BUZ, GPIO.OUT)
try:
    while True:
        if GPIO.input(PIR):
            print("Motion detected!")
            GPIO.output(BUZ, True); time.sleep(1); GPIO.output(BUZ, False)
            time.sleep(0.2)
finally:
    GPIO.cleanup()
```

### Example 2.4 — Read DHT22 temperature and humidity

Wiring: DHT22 data → GPIO4 (with 10 kΩ pull-up to 3.3 V), VCC → 3.3 V, GND → GND.

```
import time, board, adafruit_dht

dht = adafruit_dht.DHT22(board.D4)
while True:
    try:
        t = dht.temperature
        h = dht.humidity
        print(f"Temp: {t:.1f} C Humidity: {h:.1f} %")
    except RuntimeError as e:
        print("Retrying:", e)
        time.sleep(2)
```

### Example 2.5 — Servo and ultrasonic distance sensor (HC-SR04)

Distance = (echo time × speed of sound) / 2, speed of sound ≈ 34300 cm/s.

```
import RPi.GPIO as GPIO, time
TRIG, ECHO = 23, 24
GPIO.setmode(GPIO.BCM)
GPIO.setup(TRIG, GPIO.OUT); GPIO.setup(ECHO, GPIO.IN) # ECHO needs a voltage divider (5V -

def distance_cm():
    GPIO.output(TRIG, True); time.sleep(0.00001); GPIO.output(TRIG, False)
    while GPIO.input(ECHO) == 0: start = time.time()
    while GPIO.input(ECHO) == 1: stop = time.time()
    return (stop - start) * 34300 / 2

print(round(distance_cm(), 1), "cm")
GPIO.cleanup()
```

*Note: Sensors that output 5 V to a 3.3 V Raspberry Pi pin need a level shifter or voltage divider, otherwise the Pi can be damaged.*

## 2.7 Common Communication Interfaces

| Interface        | Wires               | Features                                    | Typical use                          |
|------------------|---------------------|---------------------------------------------|--------------------------------------|
| GPIO / Digital   | 1 per signal        | HIGH/LOW logic                              | Buttons, LEDs, relays, PIR           |
| ADC (analog)     | 1 per sensor        | Converts voltage to number (Arduino 10-bit) | LM35, LDR, MQ sensors                |
| PWM              | 1                   | Duty cycle controls average power           | Motor speed, LED dimming, servo      |
| UART (serial)    | TX, RX              | Asynchronous, point-to-point, baud rate     | GPS, Bluetooth HC-05, serial monitor |
| I <sup>2</sup> C | SDA, SCL            | 2 wires, many devices with unique addresses | BMP180, OLED, RTC                    |
| SPI              | MOSI, MISO, SCK, CS | Fast, full-duplex, one CS per device        | SD card, TFT, MCP3008                |
| 1-Wire           | 1 data              | Many devices on single wire                 | DS18B20                              |

### Example 2.6 — ADC resolution calculation

An Arduino Uno has a 10-bit ADC with 5 V reference. (a) What is its resolution? (b) What temperature is shown if an LM35 gives ADC reading 82?

### Solution

(a) Levels =  $2^{10} = 1024$ . Resolution =  $5 \text{ V} / 1024 \approx 4.88 \text{ mV}$  per step.

(b) Voltage =  $82 \times 4.88 \text{ mV} \approx 0.400 \text{ V}$ . LM35 gives  $10 \text{ mV}/^{\circ}\text{C}$ , so  $T = 0.400 / 0.010 = 40^{\circ}\text{C}$ .

### Key Takeaways

- ✓ Sensors convert physical quantities to electrical signals; actuators convert commands into physical action.
- ✓ Classify sensors as active/passive, analog/digital, contact/non-contact, absolute/relative.
- ✓ Know the working principle of PIR, accelerometer, strain gauge, DHT22, LDR, MQ gas, biosensor and camera.
- ✓ Arduino = microcontroller, real-time, has ADC; Raspberry Pi = Linux SBC, 3.3 V GPIO, no ADC.
- ✓ Interfaces to remember: GPIO, ADC, PWM, UART, I<sup>2</sup>C, SPI, 1-Wire.
- ✓ In Python on the Pi: setmode → setup → input/output → cleanup.

— End of Unit 2 —