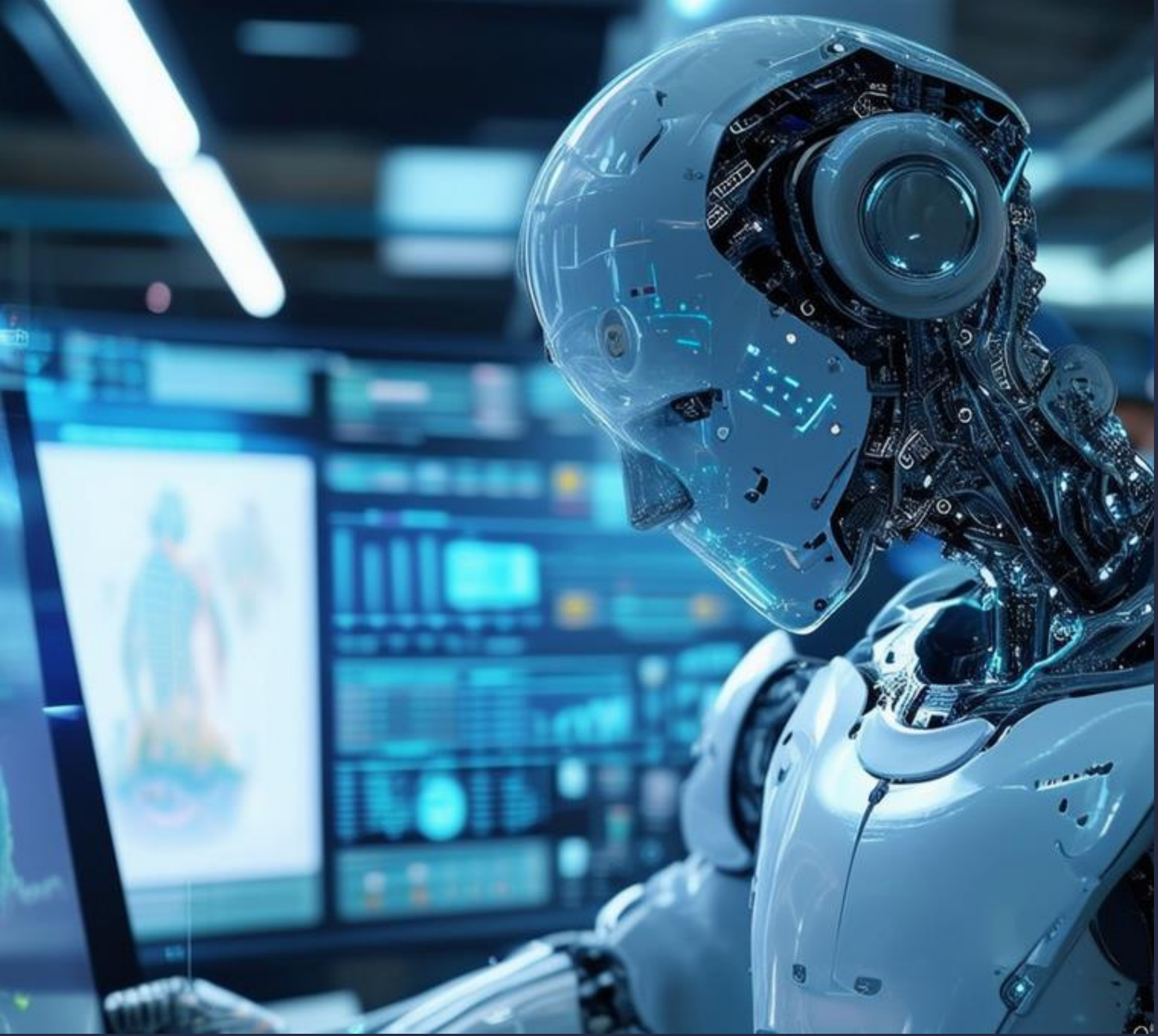


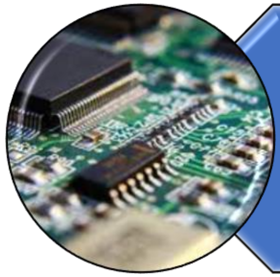
# Basic Robotics Workshop

Organised by **KURC**

# Course Outline

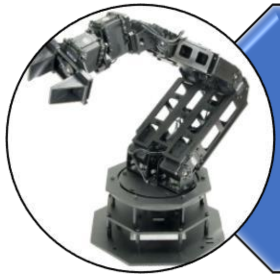
- Electronics basics
- Arduino Basics
- Bluetooth and Communication
- Motors and Actuators
- Robotics Project

# Different Systems in Robotics



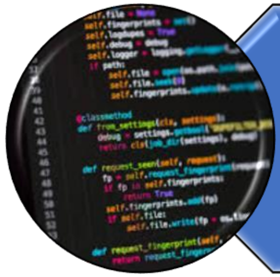
Electrical

Acts as its nervous system, collecting sensor data, powering its movements, and enabling control.



Mechanical

Responsible for the physical structure, movement, joints, actuators, crucial for achieving desired functionalities and tasks.



Programming

Provides instructions to the robot's control system, enabling it to interpret sensor data, make decisions, and control actuators, ultimately defining the robot's behavior, functionality, and response to its environment.

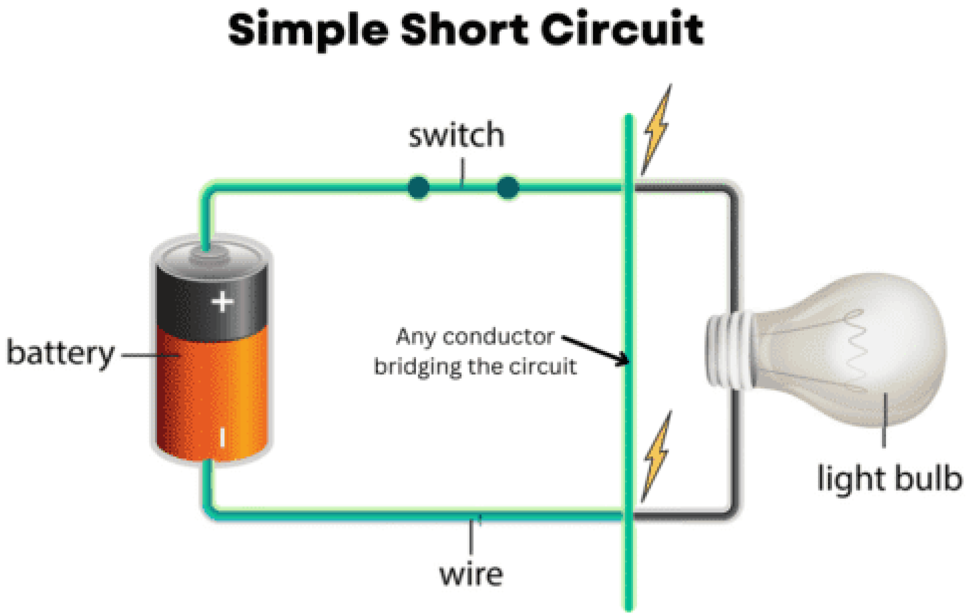
# Tools and Devices

- Multimeter
- Soldering Iron
- Wire stripper
- Glue Gun
- Bench Vice

# SAFETY OF CIRCUIT

Double check for shorts before plugging power

Don’t turn on if there is water spillage



The electricity will primarily take the path of least resistance, creating a shorter circuit, and eliminating the light bulb

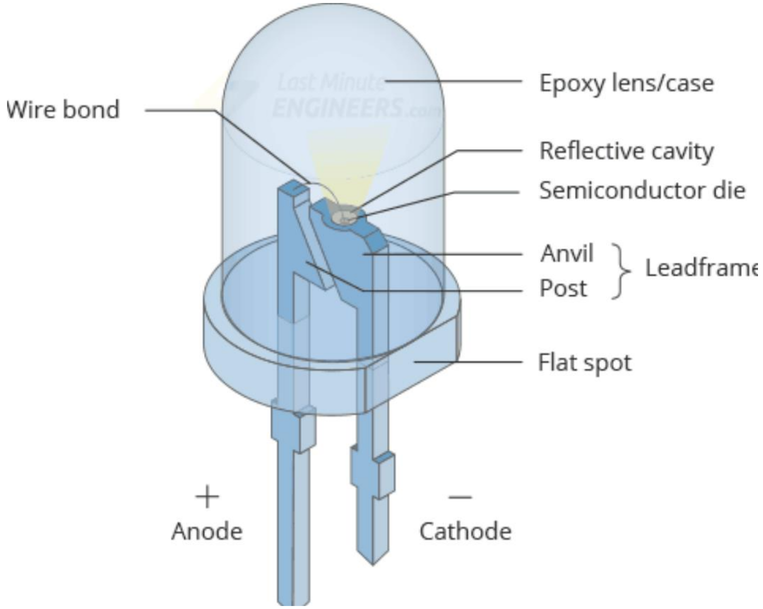
shutterstock.com · 2376956949

# SAFETY

No touching naked wires

Loose wiring, the neutral and hot wire

# LEDS: Light Emitting Component

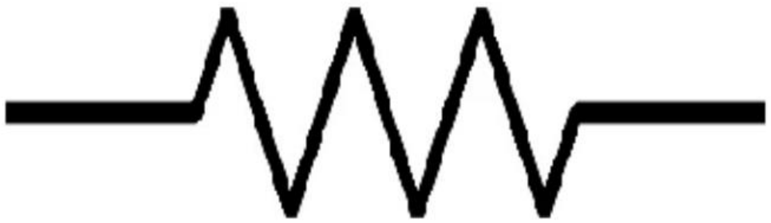


LEDs (Light-Emitting Diodes) are semiconductor devices that emit light when an electric current is applied, commonly used for indicators, displays, and lighting applications.

LED Bulb

# Resistors

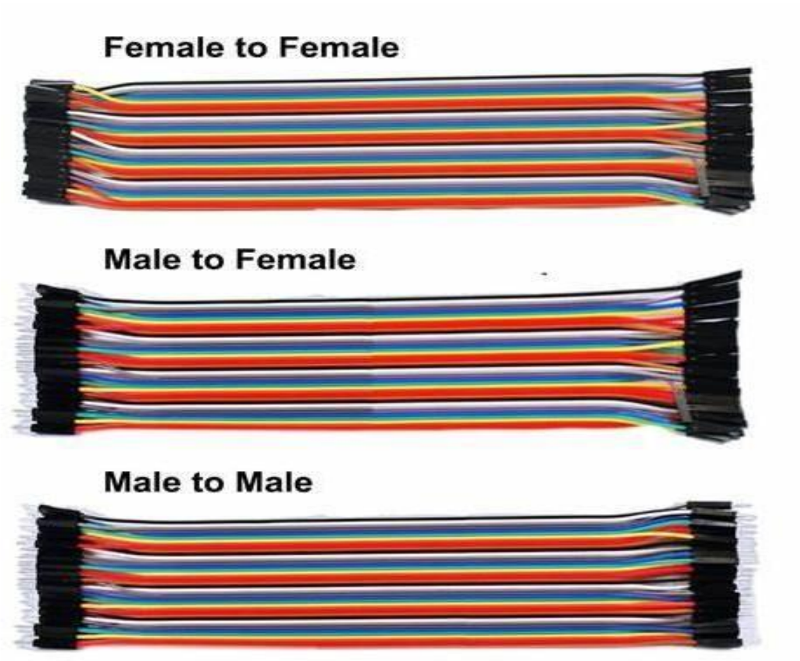
Resistors are passive electronic components that limit or regulate current flow in a circuit. They are used to control voltage, protect components, and adjust signal levels.



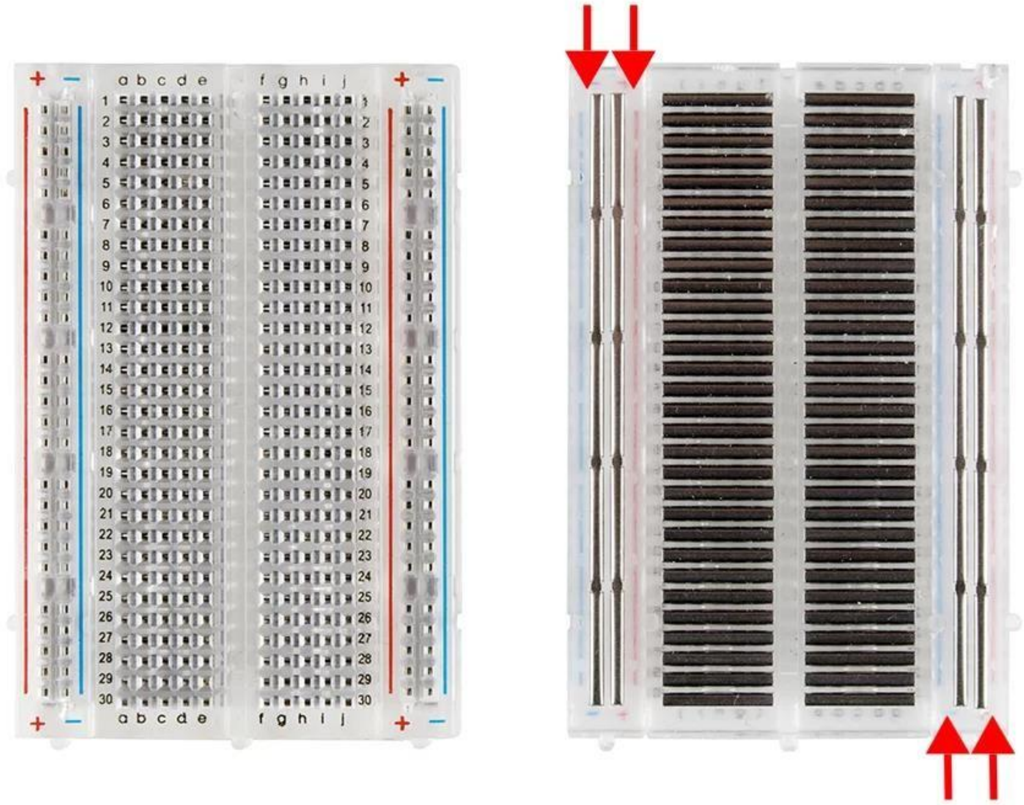
Symbol

Resistor

# Basic prototyping materials



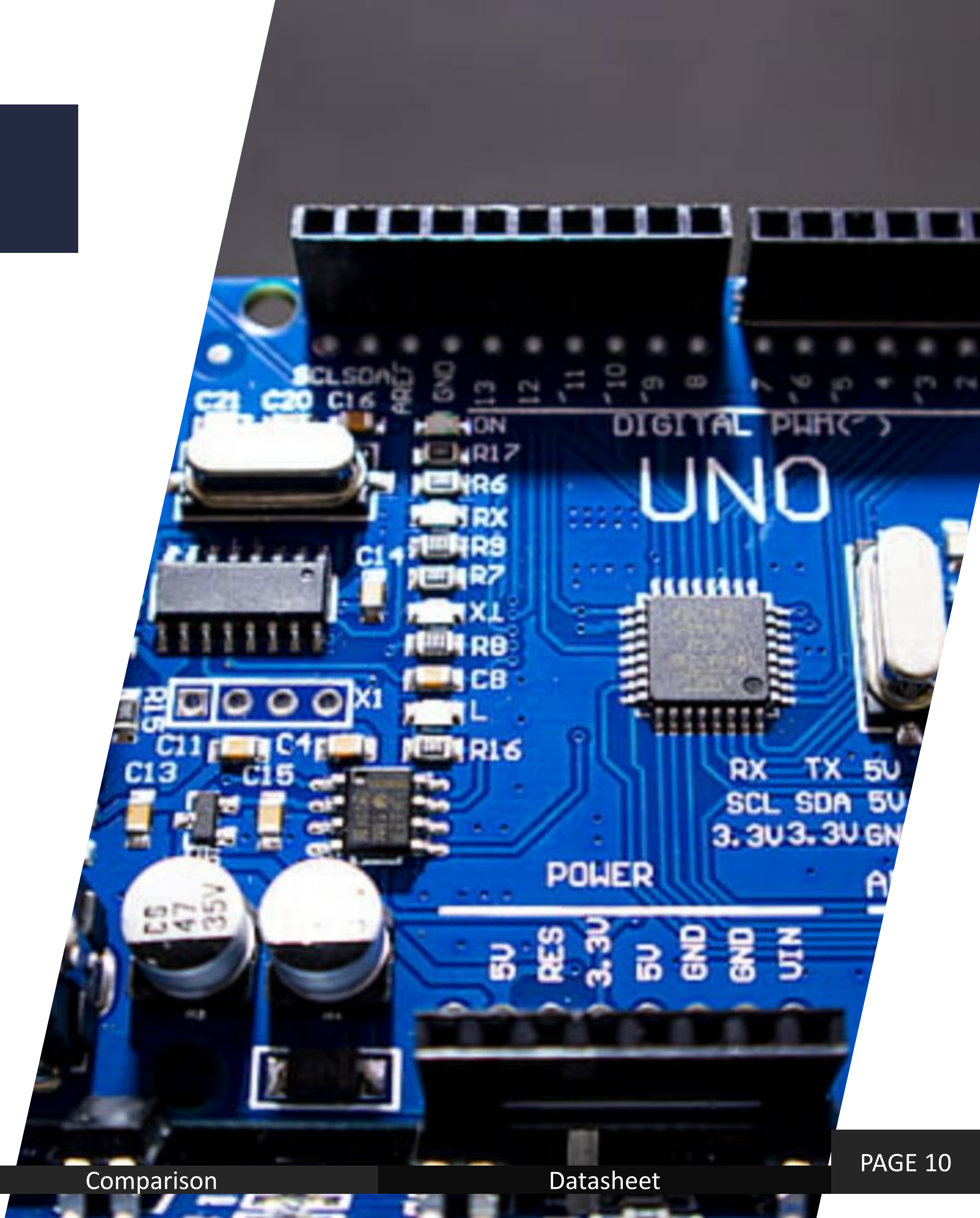
Jumper Wire



Breadboard

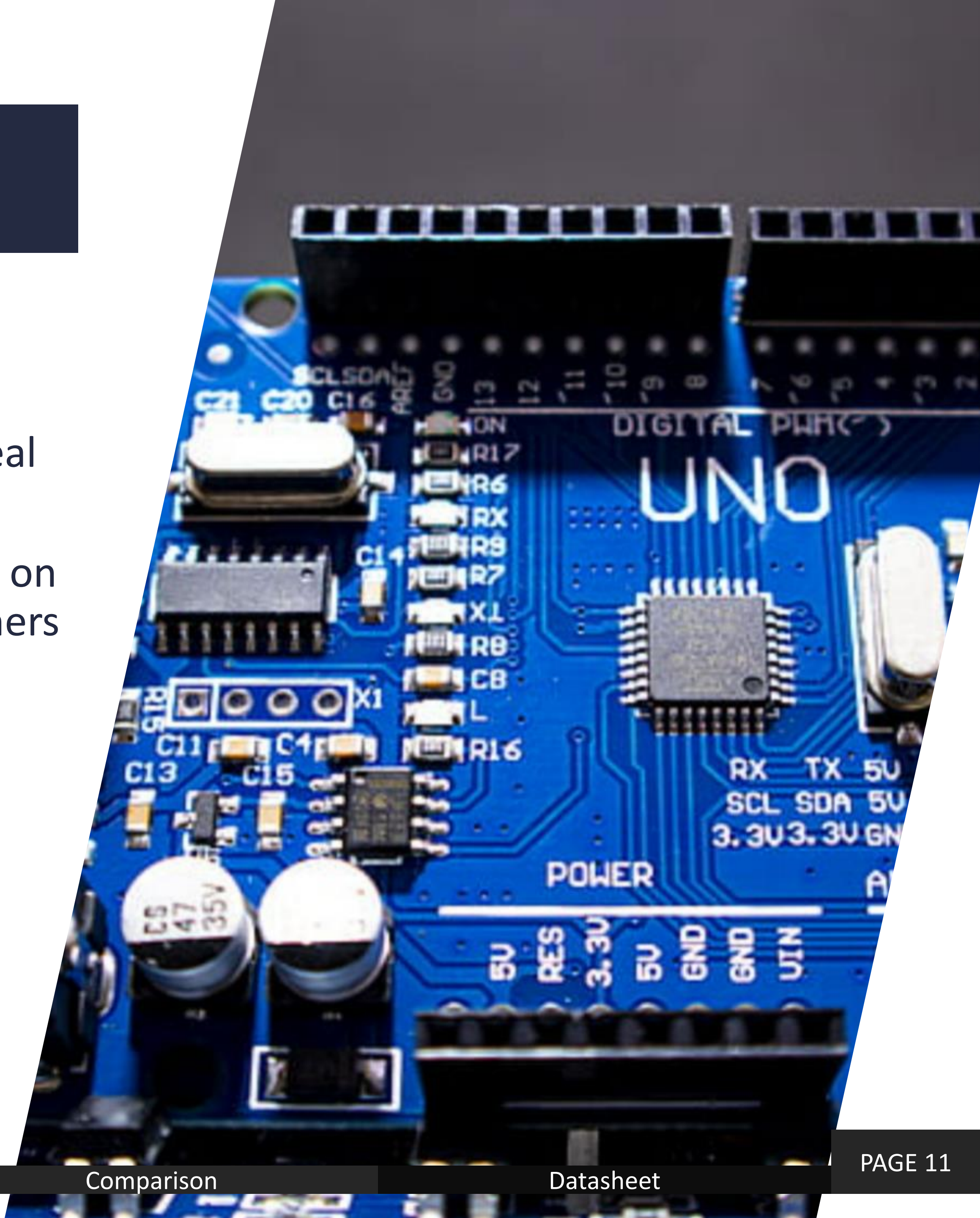
# Task: 1

- Light up an led with provided power source and breadboard.
- Do not forget to use Resistors.



# About Microcontrollers

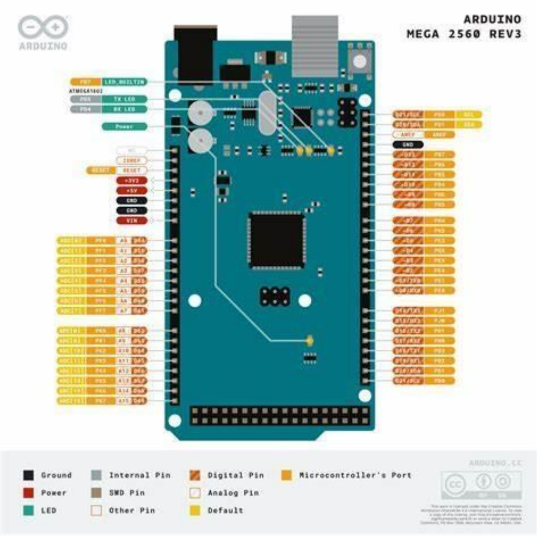
- A microcontroller is a compact integrated circuit designed to control specific devices. It includes a processor, memory, and I/O peripherals, making it ideal for embedded systems.
- Arduino is an open-source electronics platform based on easy-to-use hardware and software. It enables beginners to build interactive projects like robots, sensors, and automation systems.



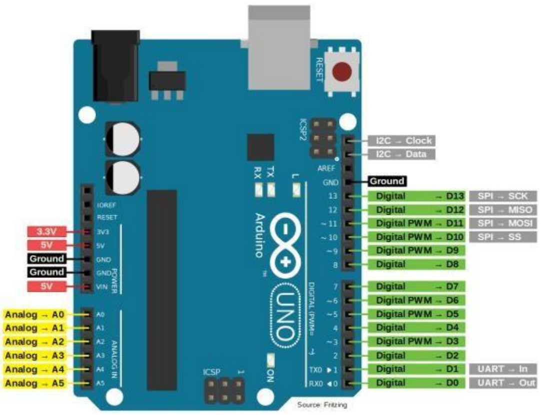
# Microcontroller

- **Processor core**: Executes instructions and controls other components.
- **Memory: Stores** data and program code ,typically includes both volatile (RAM) and non-volatile(Flash or EEPROM)memory.
- **I/O Peripherial**: Allows interaction with other devices, like sensors and actuators.
- **Communication Interfaces**: Such as SPI and I2C for interacting with other microcontroller or devices.

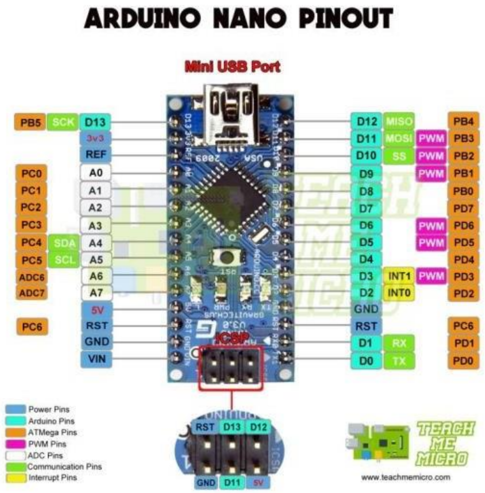
## Arduino mega



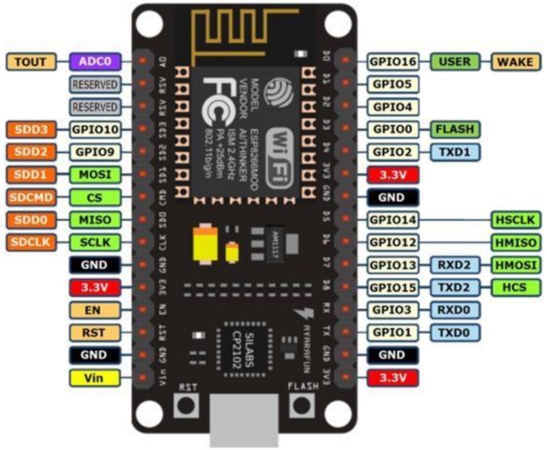
## Arduino Uno R3



## Arduino Nano

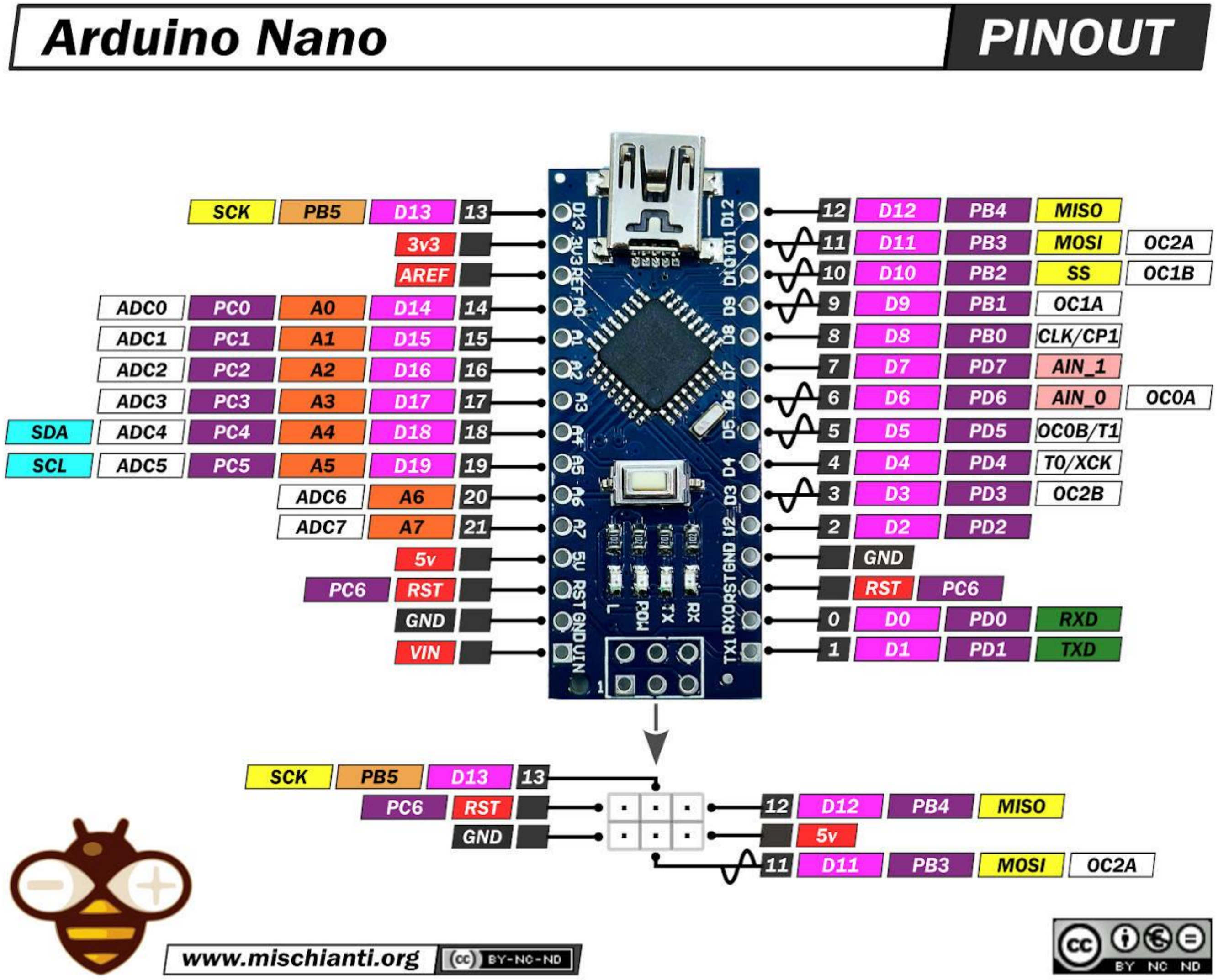


## Node MCU



# • Datasheet Nano

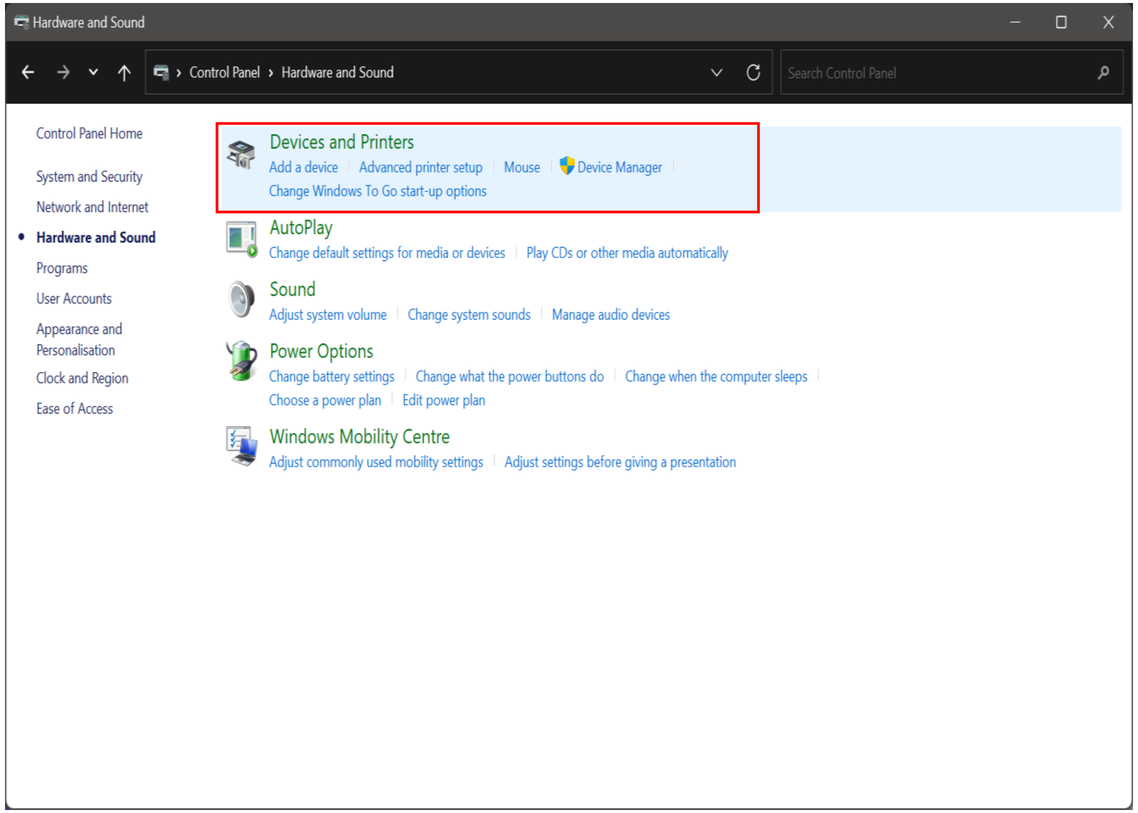
−GPIO pins
-3.3v
-GND
-Analog Pin
-Reset and Flash



# Setup Arduino IDE

- Connect Arduino
- Make sure device is detected
- Choose port and board
- Upload code

# Step 1 : Check Connection



Control Panel Home

System and Security

Network and Internet

**Hardware and Sound**

Programs

User Accounts

Appearance and Personalisation

Clock and Region

Ease of Access

Devices and Printers
Add a device | Advanced printer setup | Mouse | Device Manager
Change Windows To Go start-up options

AutoPlay
Change default settings for media or devices | Play CDs or other media automatically

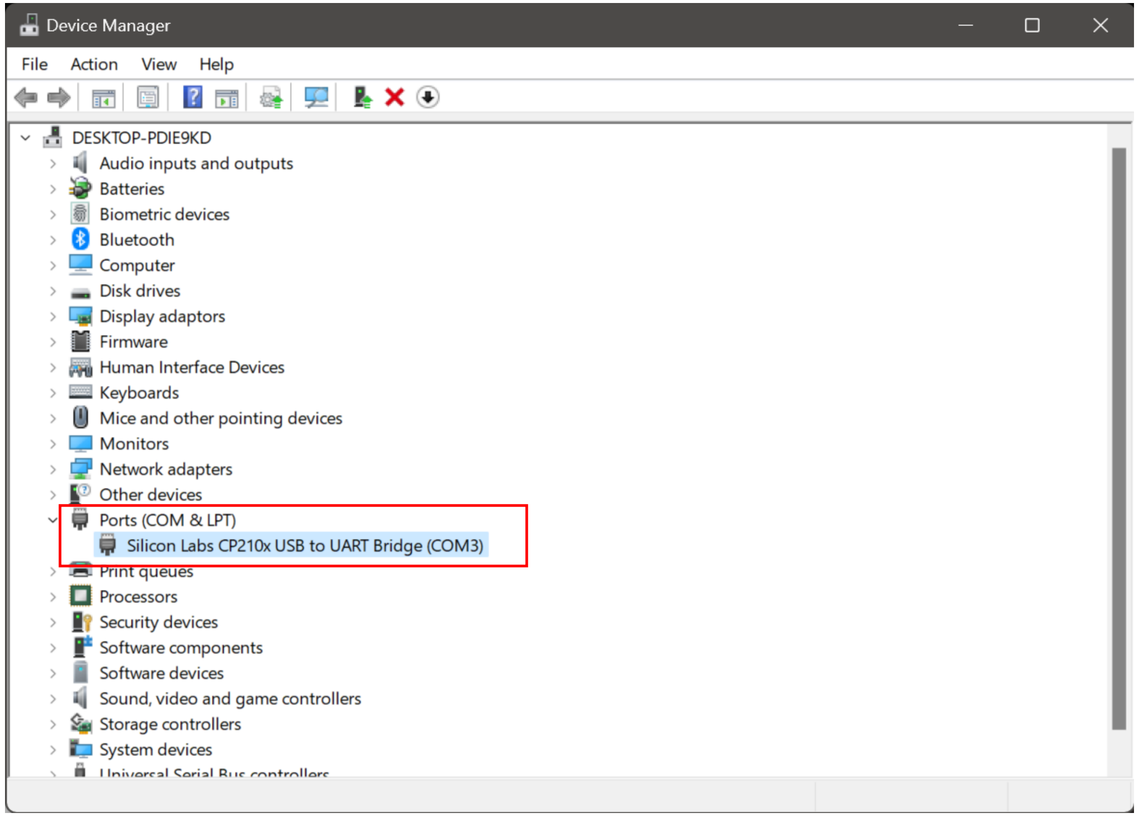
Sound
Adjust system volume | Change system sounds | Manage audio devices

Power Options
Change battery settings | Change what the power buttons do | Change when the computer sleeps
Choose a power plan | Edit power plan

Windows Mobility Centre
Adjust commonly used mobility settings | Adjust settings before giving a presentation

Go to device manager

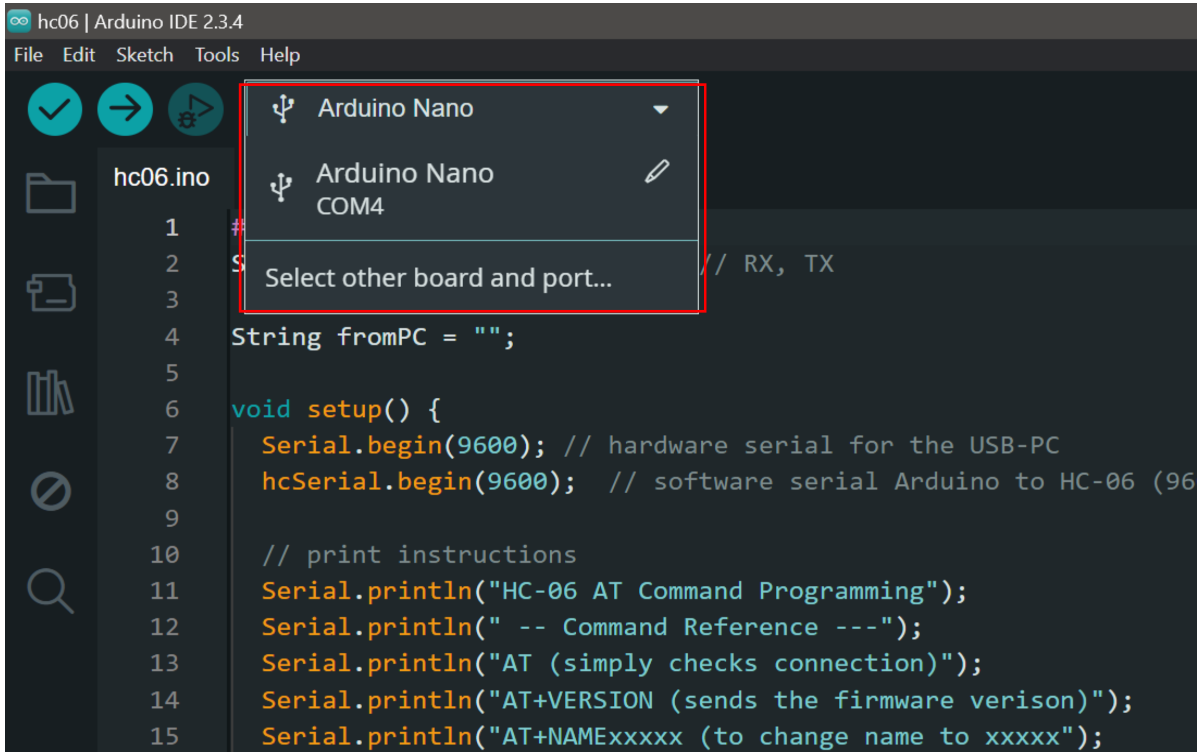
Check for Board



DESKTOP-PDIE9KD
- Audio inputs and outputs
- Batteries
- Biometric devices
- Bluetooth
- Computer
- Disk drives
- Display adaptors
- Firmware
- Human Interface Devices
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Other devices
- Ports (COM & LPT)
  - Silicon Labs CP210x USB to UART Bridge (COM3)
- Print queues
- Processors
- Security devices
- Software components
- Software devices
- Sound, video and game controllers
- Storage controllers
- System devices
- Universal Serial Bus controllers

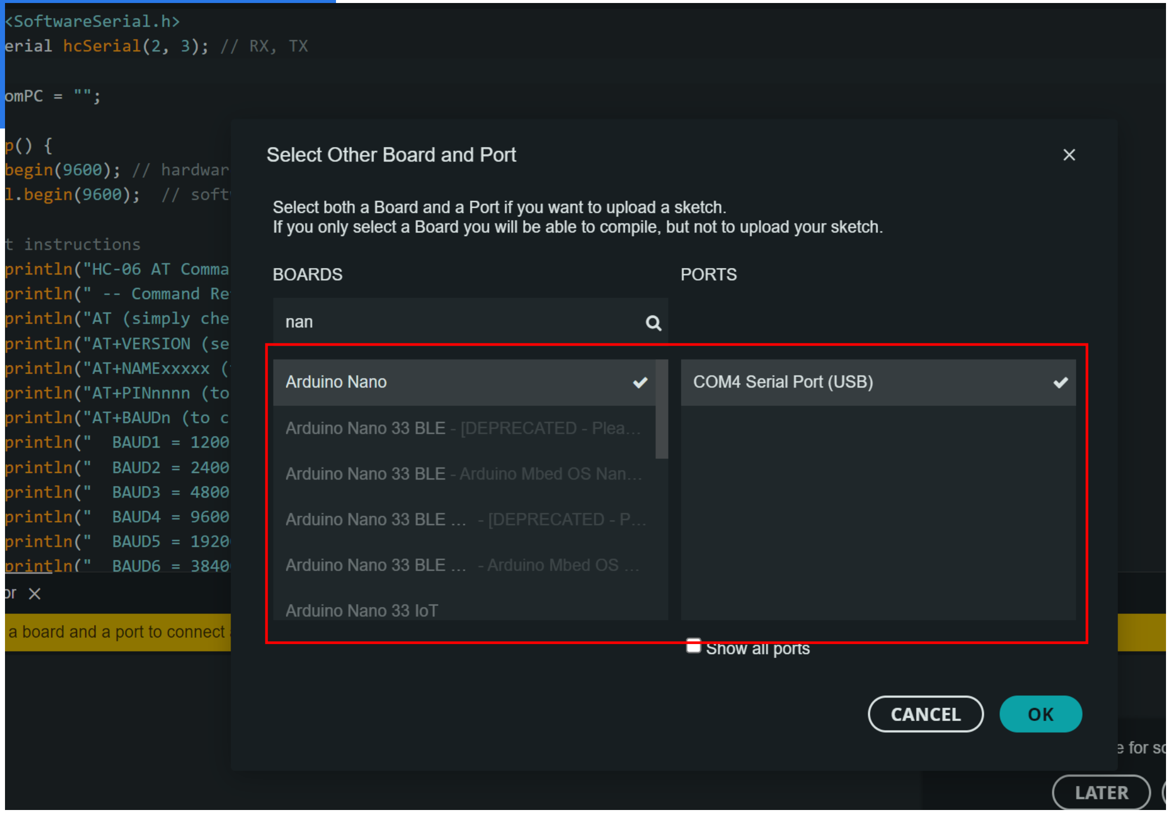
# Choose port and board

Go to boards dropdown ->select other board and port



```
#
S                                      // RX, TX

String fromPC = "";

void setup() {
  Serial.begin(9600); // hardware serial for the USB-PC
  hcSerial.begin(9600);  // software serial Arduino to HC-06 (96

  // print instructions
  Serial.println("HC-06 AT Command Programming");
  Serial.println(" -- Command Reference ---");
  Serial.println("AT (simply checks connection)");
  Serial.println("AT+VERSION (sends the firmware verison)");
  Serial.println("AT+NAMExxxxx (to change name to xxxxx");
```

# Choose port and board



```
<SoftwareSerial.h>
erial hcSerial(2, 3); // RX, TX

omPC = "";

p() {
begin(9600); // hardwar
l.begin(9600);  // soft

t instructions
println("HC-06 AT Comma
println(" -- Command Re
println("AT (simply che
println("AT+VERSION (se
println("AT+NAMExxxxx (
println("AT+PINnnnn (to
println("AT+BAUDn (to c
println("  BAUD1 = 1200
println("  BAUD2 = 2400
println("  BAUD3 = 4800
println("  BAUD4 = 9600
println("  BAUD5 = 1920
println("  BAUD6 = 3840
```

or ×

a board and a port to connect

Select Other Board and Port ×

Select both a Board and a Port if you want to upload a sketch.
If you only select a Board you will be able to compile, but not to upload your sketch.

BOARDS: nan

- Arduino Nano ✓
- Arduino Nano 33 BLE - [DEPRECATED - Plea...
- Arduino Nano 33 BLE - Arduino Mbed OS Nan...
- Arduino Nano 33 BLE ... - [DEPRECATED - P...
- Arduino Nano 33 BLE ... - Arduino Mbed OS ...
- Arduino Nano 33 IoT

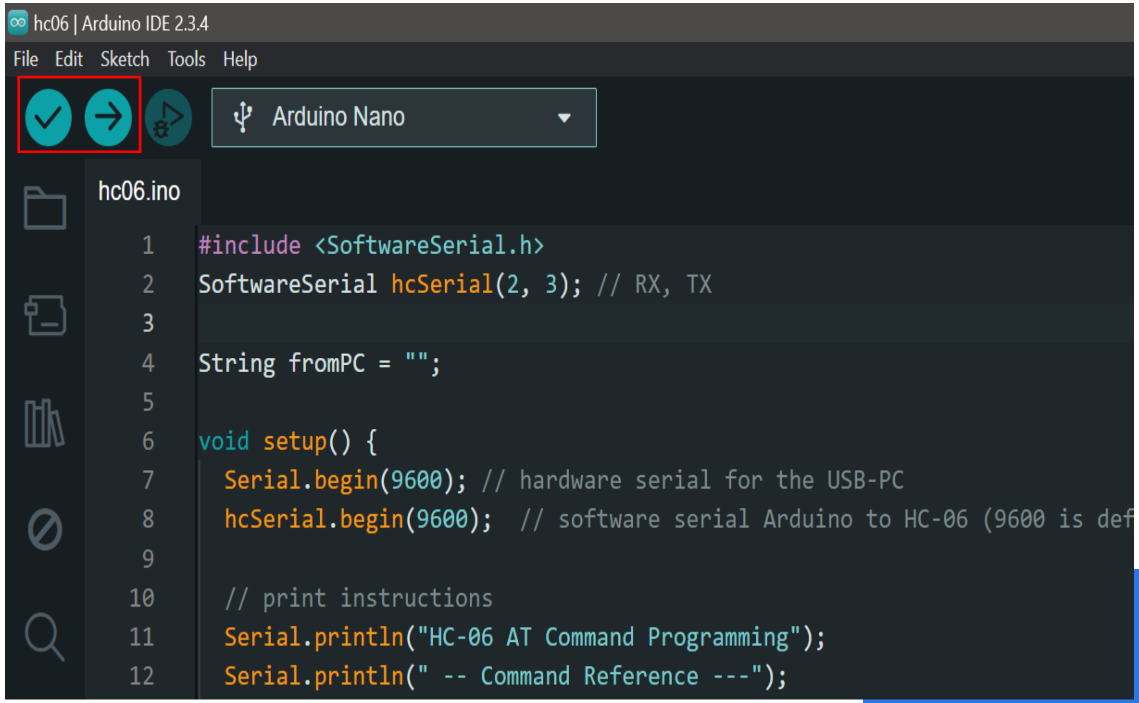
PORTS:

- COM4 Serial Port (USB) ✓

□ Show all ports

Search for NodeMCU 1.0 and select

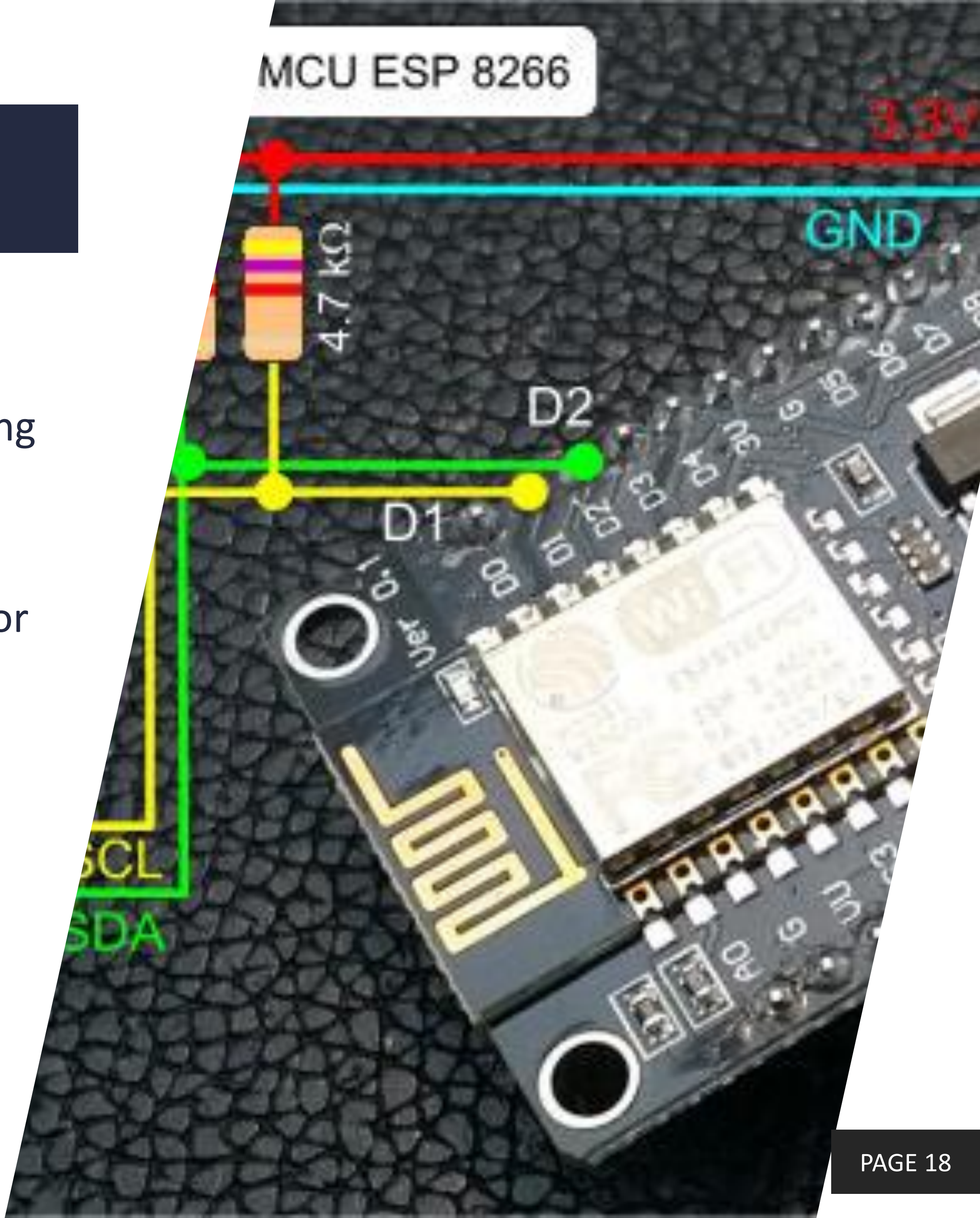
Click upload to upload your code



```
#include <SoftwareSerial.h>
SoftwareSerial hcSerial(2, 3); // RX, TX

String fromPC = "";

void setup() {
  Serial.begin(9600); // hardware serial for the USB-PC
  hcSerial.begin(9600);  // software serial Arduino to HC-06 (9600 is def

  // print instructions
  Serial.println("HC-06 AT Command Programming");
  Serial.println(" -- Command Reference ---");
```

# Blinking LED

- Your first task will be to build a blinking LED setup using Arduino
- The most basic code one can write in Arduino
- Blinking LED is practically printing “Hello world” but for hardware
- The Arduino has an inbuilt led that can be accessed using LED_BUILTIN.
- Find the code at https://github.com/kurc2014/Basic_robotics_2025

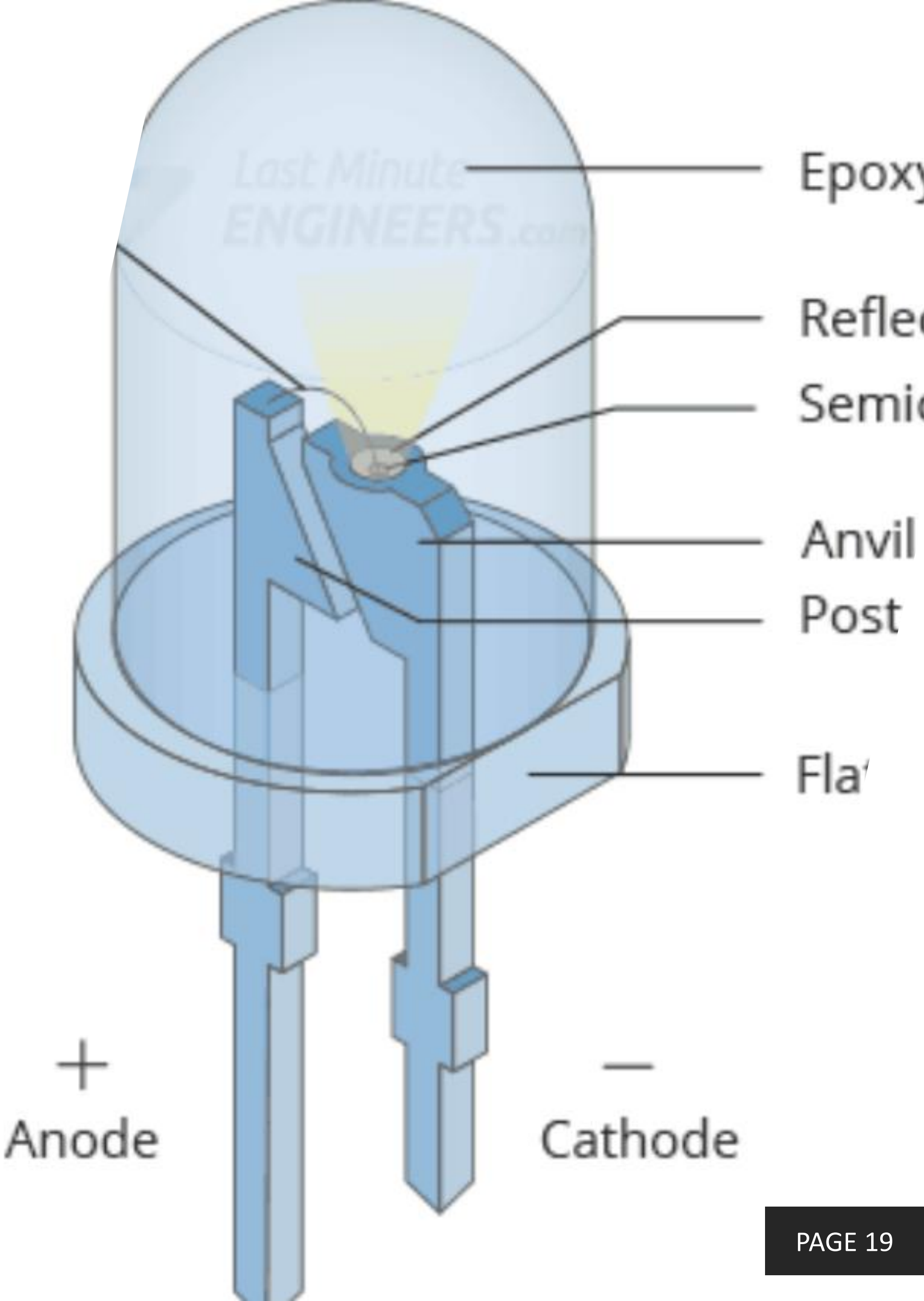


# Blinking LED

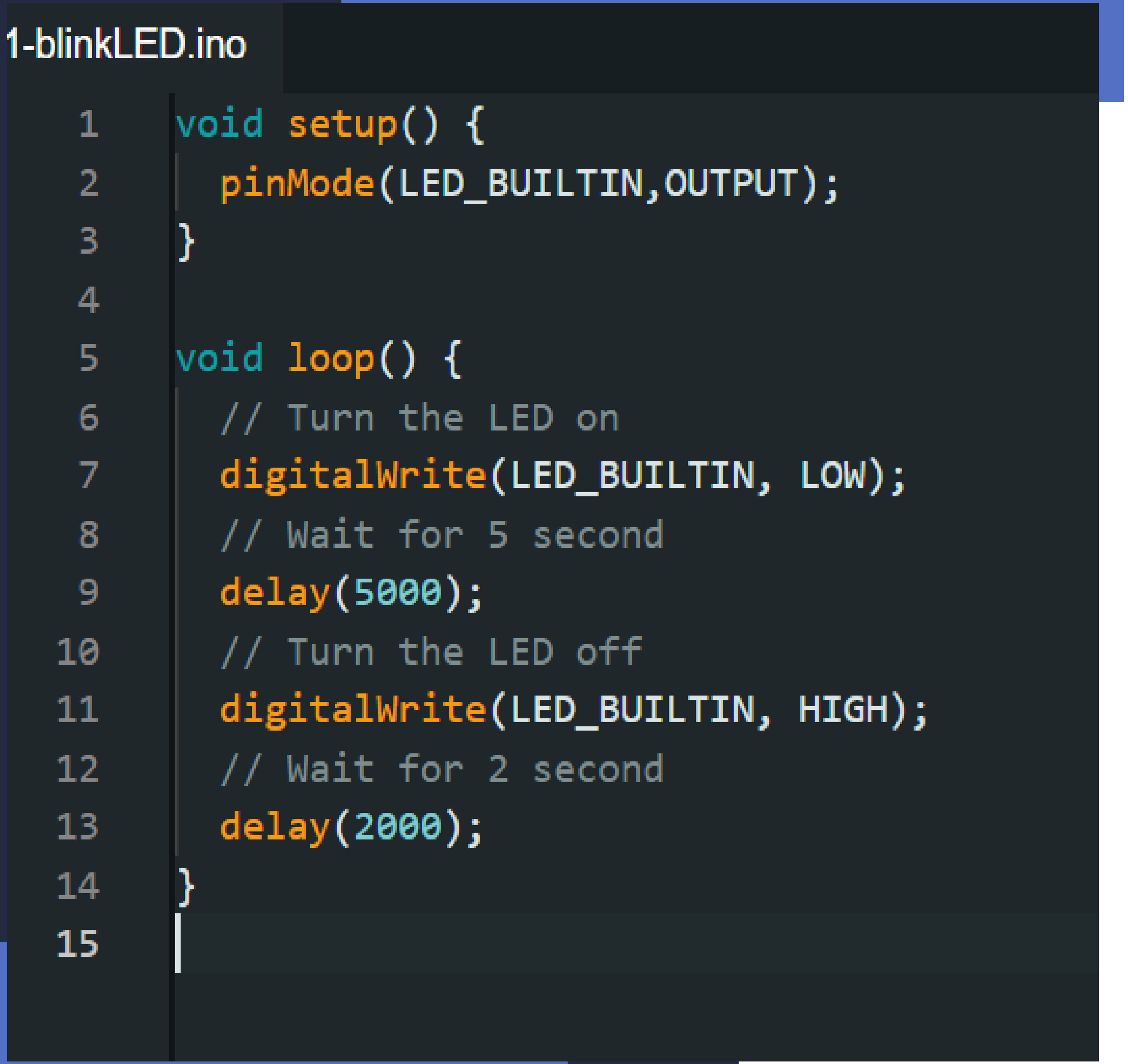
- To understand basic Arduino programming, you need to understand the basic structure.
- Libraries are used by the #include keyword and are at the top of program
- void setup{}

  void setup{} is used to define pinModes, setup serial communication, initialize libraries, and setting initial state
- void loop{}

  The loop() function contains the main code that runs repeatedly after the setup() function has completed. Used to read sensors, control actuators, implement logic and control

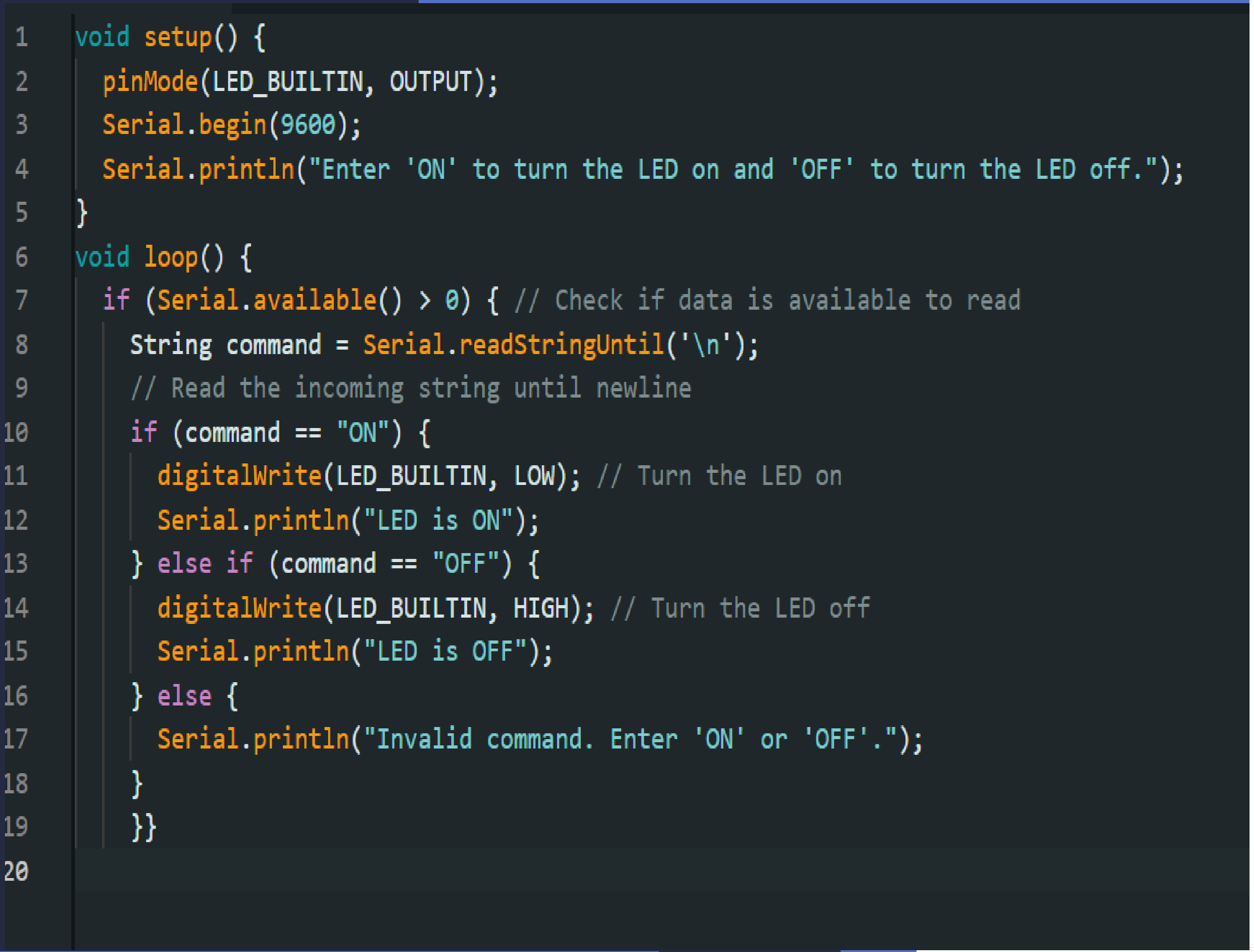


# Simple Blinking LED



```
void setup() {
  pinMode(LED_BUILTIN,OUTPUT);
}

void loop() {
  // Turn the LED on
  digitalWrite(LED_BUILTIN, LOW);
  // Wait for 5 second
  delay(5000);
  // Turn the LED off
  digitalWrite(LED_BUILTIN, HIGH);
  // Wait for 2 second
  delay(2000);
}

```

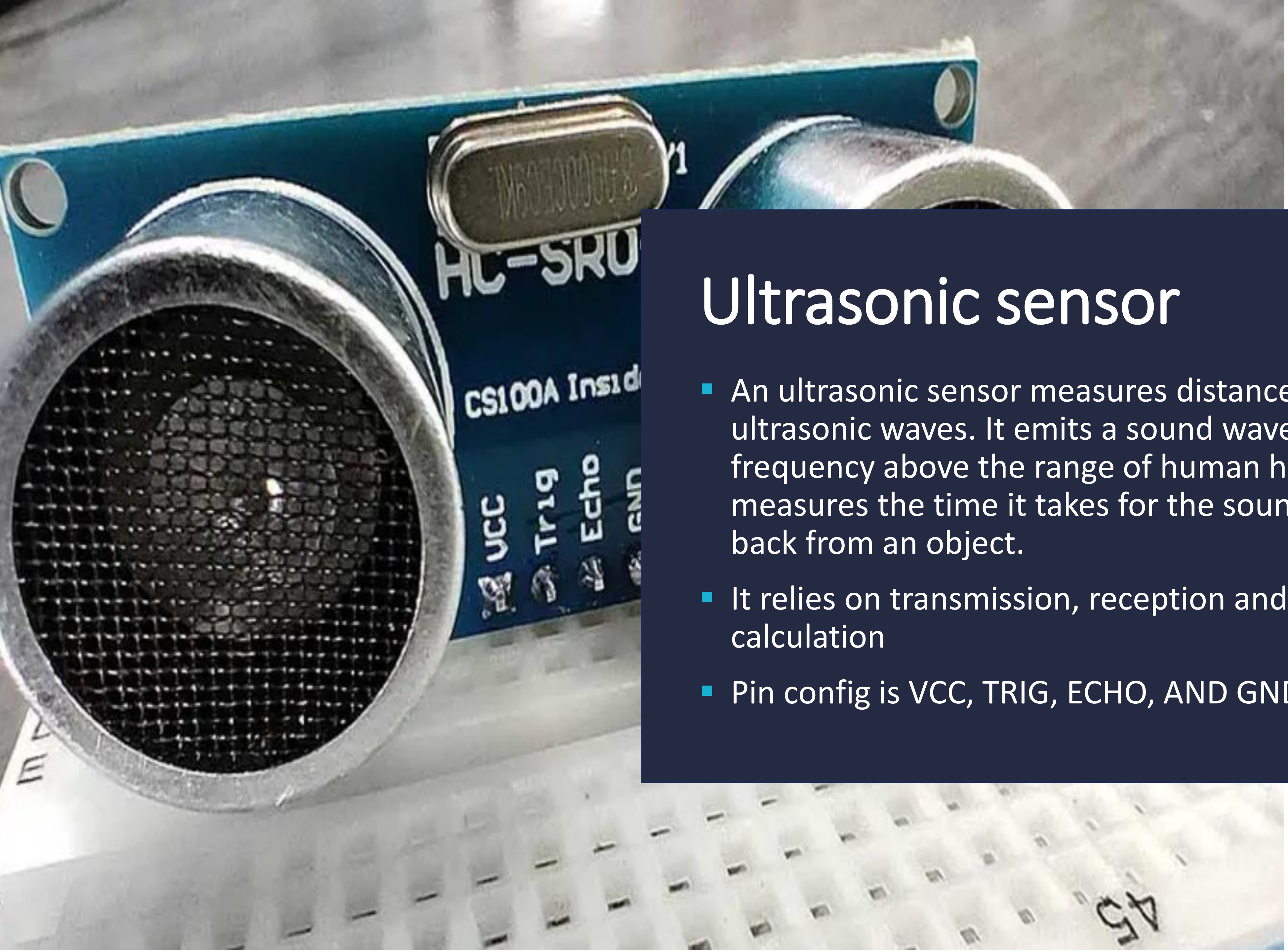
- This simple code toggles the built-in LED on and off with a one-second interval, demonstrating how to control digital outputs on the ESP8266. By using pinMode and digitalWrite, you can manipulate the state of pins to create various interactive projects.



```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);
  Serial.println("Enter 'ON' to turn the LED on and 'OFF' to turn the LED off.");
}
void loop() {
  if (Serial.available() > 0) { // Check if data is available to read
    String command = Serial.readStringUntil('\n');
    // Read the incoming string until newline
    if (command == "ON") {
      digitalWrite(LED_BUILTIN, LOW); // Turn the LED on
      Serial.println("LED is ON");
    } else if (command == "OFF") {
      digitalWrite(LED_BUILTIN, HIGH); // Turn the LED off
      Serial.println("LED is OFF");
    } else {
      Serial.println("Invalid command. Enter 'ON' or 'OFF'.");
    }
    }}

```

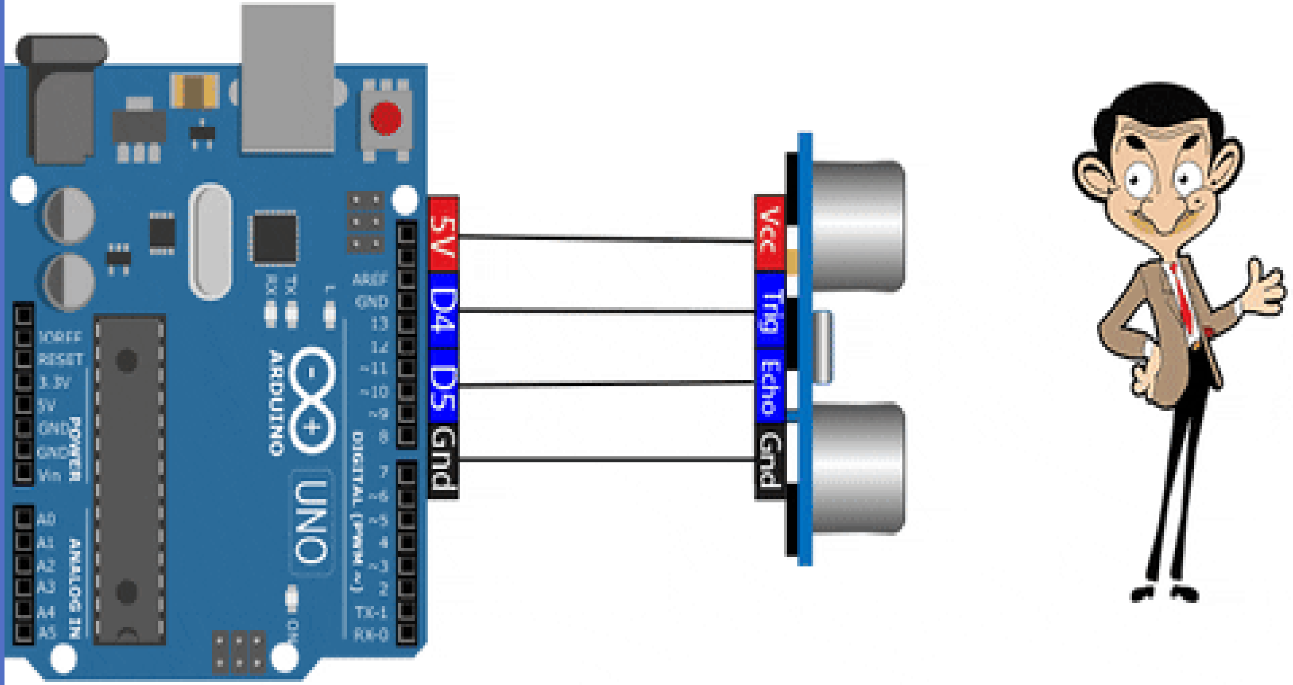
# Blink LED using commands

- This simple code toggles the built-in LED on and off with a one-second interval, demonstrating how to control digital outputs on the Arduino. By using pinMode and digitalWrite, you can manipulate the state of pins to create various interactive projects.



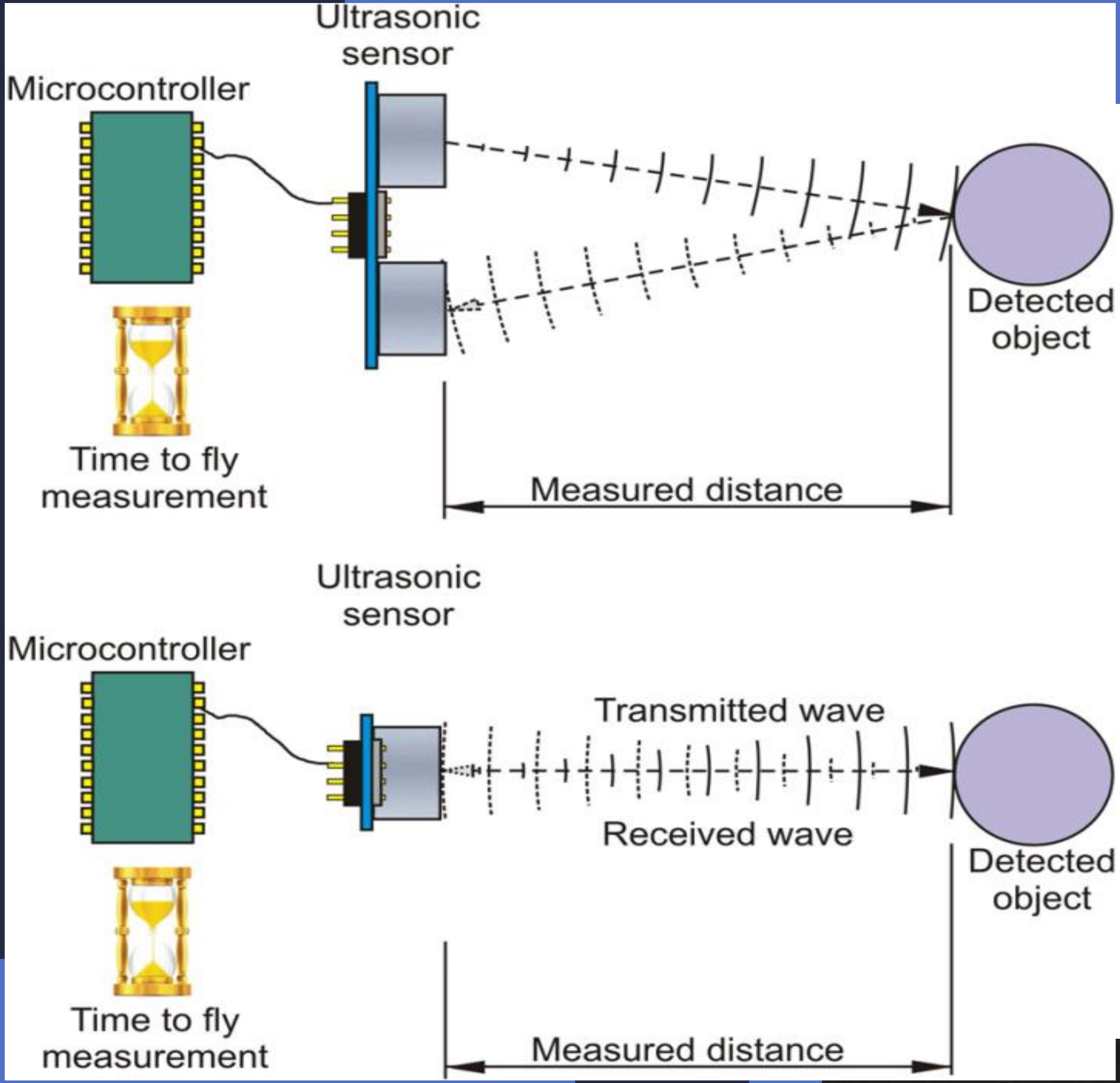
# Ultrasonic sensor

- An ultrasonic sensor measures distance by using ultrasonic waves. It emits a sound wave at a frequency above the range of human hearing and measures the time it takes for the sound to bounce back from an object.
- It relies on transmission, reception and distance calculation
- Pin config is VCC, TRIG, ECHO, AND GND



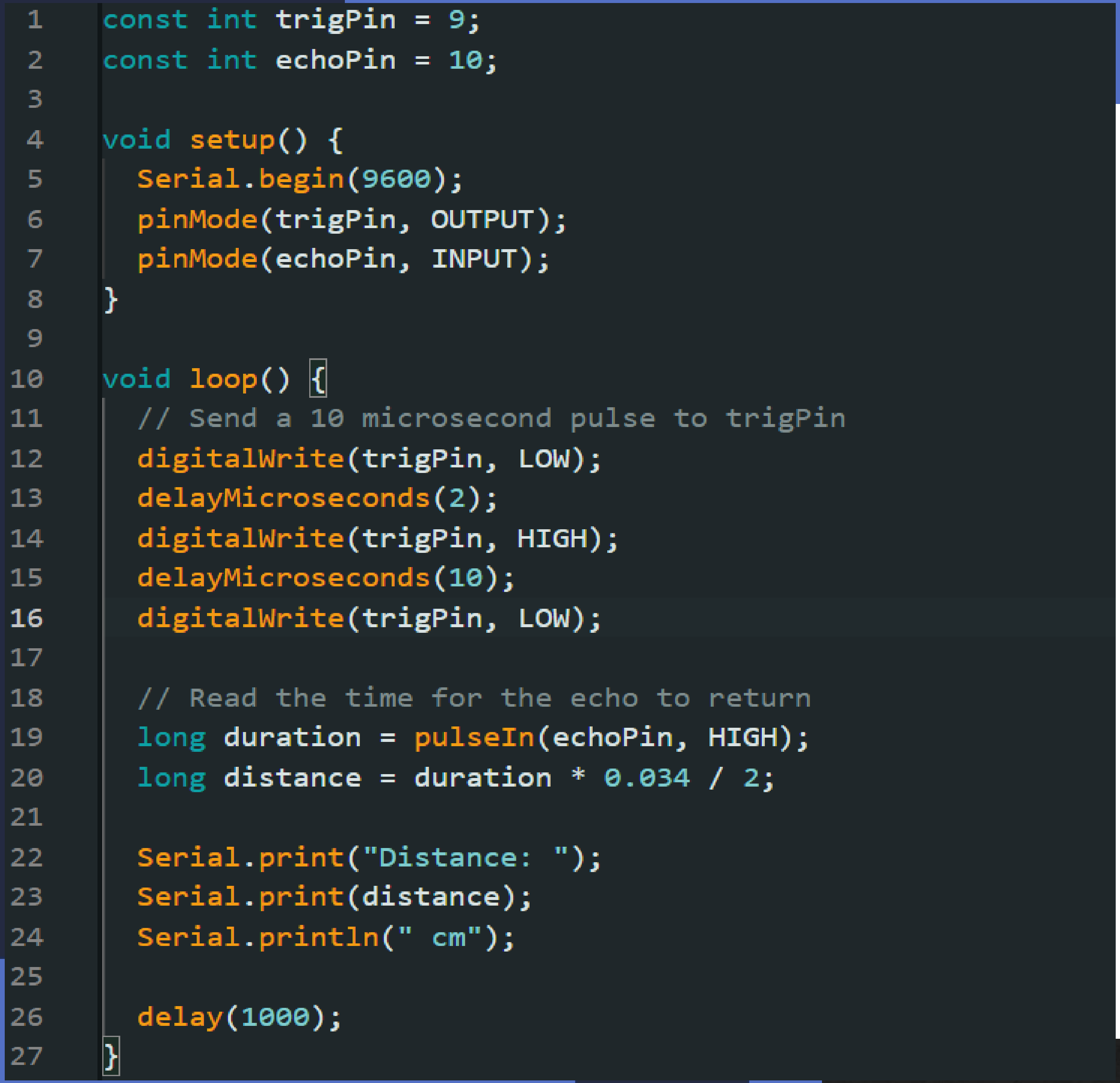
# How it works:

- The sensor's transmitter (trigger pin) sends a high-frequency ultrasonic pulse.
- The ultrasonic pulse travels through the air.
- When it hits an object, it reflects back towards the sensor.
- The sensor's receiver (echo pin) detects the reflected wave.
- The echo pin goes HIGH and stays HIGH for the duration of the time it takes for the pulse to travel to the object and back.



# Distance calculation

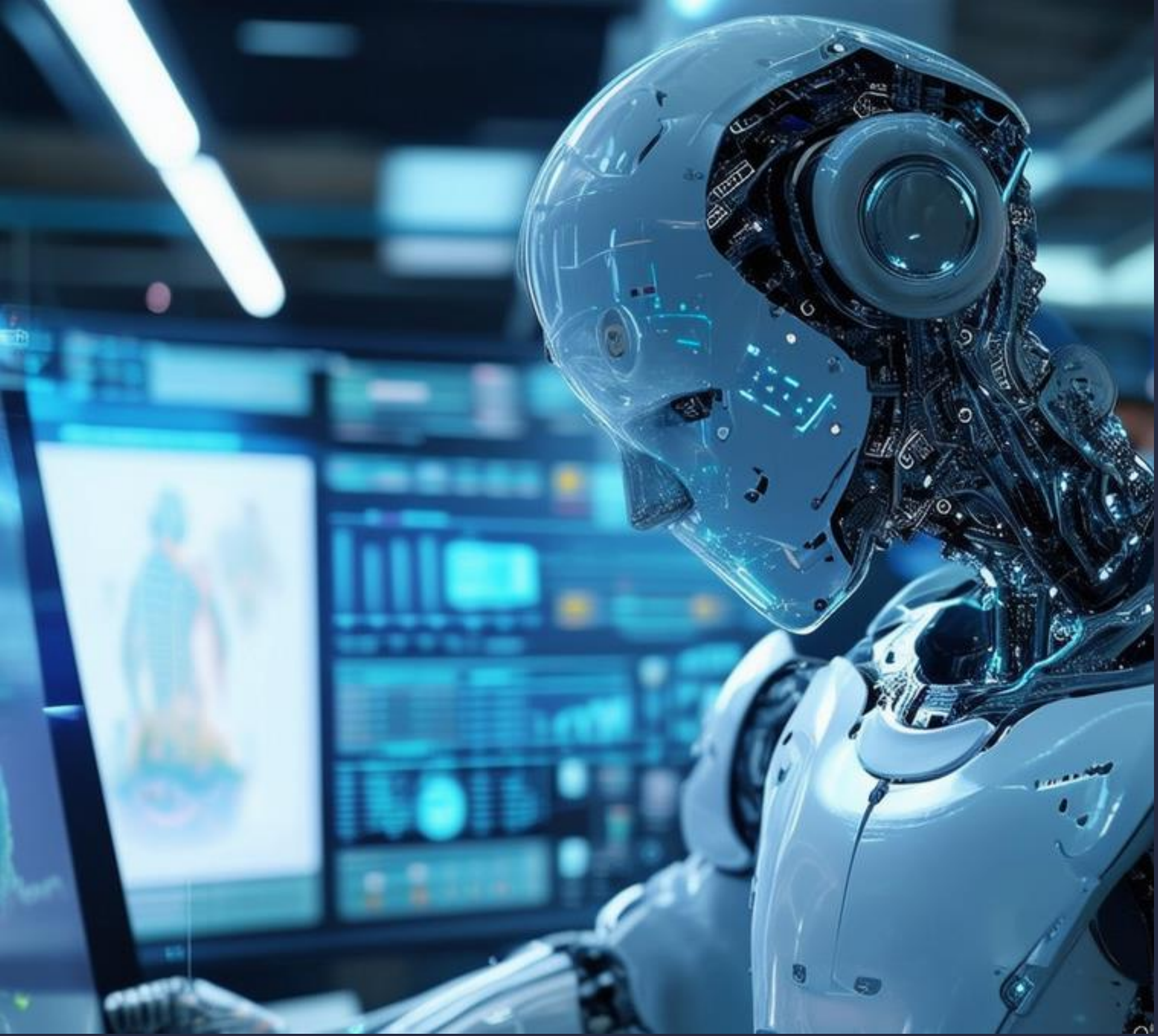
- The speed of sound in air is approximately 343 meters per second (0.0343 cm/µs).
- Distance is calculated using the formula:
- Distance=(Time taken × Speed of Sound)/2
- The division by 2 accounts for the pulse traveling to the object and back.
- Distance =  (time in µs×0.0343 cm/µs)/2



```
const int trigPin = 9;
const int echoPin = 10;

void setup() {
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
  // Send a 10 microsecond pulse to trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Read the time for the echo to return
  long duration = pulseIn(echoPin, HIGH);
  long distance = duration * 0.034 / 2;

  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");

  delay(1000);
}
```

# Distance calculation

- By sending an ultrasonic pulse and measuring the time it takes for the echo to return, the distance to an object can be calculated and displayed on the Serial Monitor.
- Trigger sensor
- Measure echo duration
- Calculate distance
- Display distance



# End of Day 1

Organised by **KURC**