



Gurukul Technicals Institute

Computer education · Robotics · Science projects · Delhi

FREE EBOOK

ROBOTICS AND ELECTRONICS

Sensors for School Projects

How common sensors work and how to use them with Arduino

INSIDE THIS BOOK

- What a sensor is
- Light sensors: the LDR and the flame sensor
- Infrared sensors: obstacles and motion
- Measuring distance: the ultrasonic sensor
- Temperature and humidity: LM35, DHT11 and DHT22
- ...and 5 more chapters, practice questions and a glossary

Beginner to intermediate

Students making science and robotics projects

engineergurukul.com

+91 97116 99511

Sensors for School Projects

Understand what sensors are, the difference between analog and digital outputs, and how to connect each common sensor to an Arduino Uno. For the LDR, IR, PIR, ultrasonic, temperature, humidity, soil, rain, gas, sound, flame, touch and accelerometer sensors you get the working principle, the pins, tested sample code and project ideas, with clear safety advice.

WHO IT IS FOR	Students making science and robotics projects
LEVEL	Beginner to intermediate
SUBJECT AREA	Robotics and electronics
WHAT IS INSIDE	10 chapters, practice questions with answers, and a glossary

CONTENTS

1	What a sensor is	4
2	Light sensors: the LDR and the flame sensor	5
3	Infrared sensors: obstacles and motion	7
4	Measuring distance: the ultrasonic sensor	9
5	Temperature and humidity: LM35, DHT11 and DHT22	11
6	Water sensors: soil moisture and rain	13
7	The MQ-2 gas sensor	15
8	Sound and touch sensors	17
9	Accelerometer basics	19
10	Planning a sensor project	21
	Practice questions	23

Free ebook: you may read, print and share it for learning; it may not be sold or changed. © 2026 Gurukul Technicals Institute. First edition, October 2026. Contact details are on the back cover.

What a sensor is

A **sensor** is a part that notices something in the world, such as light, heat, distance, movement or water, and turns it into an electrical signal. Your body has sensors too: eyes for light, skin for heat and touch, ears for sound. An Arduino has none of its own. Give it sensors, and it can react to the world.

Every project in this book follows one pattern: **sense, decide, act**. The sensor senses, the Arduino's code decides, and an LED, buzzer, motor or screen acts.

Analog and digital outputs

Output type	What it gives	How the Uno reads it	Example
Analog	any voltage from 0 V to 5 V	<code>analogRead()</code> on A0 to A5, as 0 to 1023	LDR, LM35, soil sensor
Digital	only HIGH (5 V) or LOW (0 V)	<code>digitalRead()</code> on any pin	PIR, IR obstacle sensor
Pulses and data	a timed pulse, or numbers sent as a stream of pulses	a timing function or a library	HC-SR04, DHT11, MPU-6050

An analog output tells you **how much**: how bright, how wet, how warm. A digital output tells you only **yes or no**: is there motion, is there an obstacle.

Sensor modules

Many sensors are sold on a small board called a **module**. A common design has the sensor, a chip called a **comparator** (often the LM393), a small blue **preset** (an adjustable resistor turned with a screwdriver) and two LEDs: one for power, one that lights when the sensor is triggered.

Such a module usually has these pins:

- **VCC**: power, usually 3.3 V to 5 V. On the Uno, use 5V.
- **GND**: ground.
- **DO**: digital out. The comparator sets it HIGH or LOW when the reading crosses a level set by the preset. The preset is the **threshold** knob.
- **AO**: analog out, the raw reading (not on every module).

Many modules are **active low**: DO goes LOW when the sensor is triggered. Always check yours by watching its signal LED and the Serial Monitor.

The voltage divider

Some sensors, such as the LDR and the thermistor, are simply resistors that change. The Uno cannot measure resistance directly, so we pair the sensor with a fixed resistor in series between 5V and GND. The voltage at the middle point changes as the sensor changes. This pair is a **voltage divider**, and the middle point goes to an analog pin.

REMEMBER

Every sensor needs three things joined correctly: power, ground and signal. A sensor with no GND connection gives nonsense readings.

TIP

Before you write any project code, first print the sensor's raw readings to the Serial Monitor. Knowing the real numbers in your room makes every later step easy.

SAFETY

All projects here run from a USB port or from batteries of 12 V or less. Never connect a sensor project to mains wiring.

CHAPTER 2

Light sensors: the LDR and the flame sensor

The LDR

How it works: an **LDR** (light-dependent resistor) has a wavy track of a special material on its face. Light frees tiny charges in that material so current flows more easily. In bright light its resistance may be a few hundred ohms or a few thousand; in darkness it rises to hundreds of thousands of ohms or more.

Pins: an LDR has two legs and no + or - side. Make a divider: 5V → LDR → A0, and A0 → 10 kΩ resistor → GND. Brighter light gives a higher reading.

```
// LDR: show the light level as a percentage
const int ldrPin = A0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  int raw = analogRead(ldrPin);           // 0 (dark) to 1023 (bright)
  int percent = map(raw, 0, 1023, 0, 100);
  Serial.print("Light: ");
  Serial.print(percent);
  Serial.println(" %");
  delay(500);
}
```

Project ideas: an automatic night lamp; a "sunrise alarm" that beeps when morning light arrives; a sun tracker that turns a small solar panel with a servo towards whichever of two LDRs is brighter.

DID YOU KNOW

LDRs react fairly slowly, taking a short moment to settle after the light changes. For fast work, such as counting objects that break a light beam quickly, makers use a photodiode or a phototransistor instead.

The flame sensor

How it works: a flame gives out a lot of **infrared** light, which our eyes cannot see. A flame sensor module has a small dark component, often a phototransistor, that is sensitive to near-infrared light of the kind flames give out. The LM393 comparator turns its signal into a DO output, and AO gives the raw level.

Pins: VCC → 5V, GND → GND, DO → pin 7, AO (if present) → A0, and a passive buzzer from pin 8 to GND. On most modules, DO goes LOW when a flame is seen.

```
// Flame sensor: warn when a flame is seen
const int flamePin = 7;      // module D0
const int buzzerPin = 8;

void setup() {
  pinMode(flamePin, INPUT);
  Serial.begin(9600);
}

void loop() {
  int level = analogRead(A0);      // lower usually means stronger
  infrared
  Serial.println(level);
  if (digitalRead(flamePin) == LOW) {    // LOW = flame on most modules
    tone(buzzerPin, 2000, 200);
  }
  delay(300);
}
```

SAFETY

Test a flame sensor first with a TV remote: most modules react to its invisible infrared beam when you press a button close to them. If a teacher wants to show a real flame, only an adult does it, with a single candle on a metal tray, at a safe distance, with water nearby. Never bring a flame close to the sensor or the wires.

Project ideas: a fire alarm model for a model house; a firefighting robot that turns towards a candle and blows it out with a small fan (adult supervision only); a model factory alarm panel with red and green LEDs.

CHAPTER 3

Infrared sensors: obstacles and motion

The IR obstacle sensor

How it works: the module has two small parts side by side, usually one clear and one dark. The clear one is an **IR LED** that shines invisible infrared light forward. The dark one is a **photodiode** that receives light. When an object comes close, the light bounces back into the photodiode, and the comparator switches DO.

Pins: VCC → 5V, GND → GND, OUT → pin 2. OUT goes LOW when an obstacle is near, and the signal LED lights. Turn the preset to set the distance; it usually works from a few centimetres to around 20 to 30 cm.

```
// IR obstacle sensor: report when something is close
const int irPin = 2;
const int ledPin = 13;

void setup() {
  pinMode(irPin, INPUT);
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  if (digitalRead(irPin) == LOW) {          // LOW means obstacle
    digitalWrite(ledPin, HIGH);
    Serial.println("Obstacle!");
  } else {
    digitalWrite(ledPin, LOW);
  }
  delay(100);
}
```

Good to know: black surfaces soak up infrared, so a black object may not be seen until it is very close. Strong sunlight contains infrared too and can confuse the sensor outdoors.

Project ideas: a touchless dustbin lid; a visitor counter at a door; the line sensors of a line-following robot (the same idea, pointing down at a black line on a white floor).

TRY THIS

Look at the IR LED through a phone camera while the module is powered. Many phone cameras show its invisible light as a faint purple glow. Your eyes see nothing.

The PIR motion sensor

How it works: every warm body, including yours, gives out infrared heat. A **PIR** (passive infrared) sensor has a special crystal that produces a tiny voltage when the infrared falling on it **changes**. The white dome is a **Fresnel lens**: it is divided into many small sections, so a person walking across the room moves from one section's view to the next, and the sensor sees the change. It is called passive because it sends nothing out; it only receives.

Pins: the popular HC-SR501 module has three pins, usually marked under the dome: VCC → 5V, OUT → pin 2, GND → GND. OUT goes HIGH (about 3.3 V, which the Uno reads as HIGH) when motion is seen.

It also has two small orange presets. One sets **sensitivity** (how far away, up to about 7 metres). The other sets the **delay**: how long OUT stays HIGH after motion, from a few seconds to a few minutes. Turn the delay fully to its shortest while testing.

```
// PIR: light an LED and print a message when someone moves
const int pirPin = 2;
const int ledPin = 13;

void setup() {
  pinMode(pirPin, INPUT);
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
  Serial.println("Warming up. Please keep still for a minute.");
  delay(60000); // the sensor settles after power-on
  Serial.println("Ready.");
}

void loop() {
  int motion = digitalRead(pirPin);
  digitalWrite(ledPin, motion); // HIGH = motion
  if (motion == HIGH) {
    Serial.println("Motion detected");
  }
  delay(200);
}
```

DID YOU KNOW

A PIR sensor cannot see you through a glass window. Ordinary glass blocks the kind of infrared heat that warm bodies give out, even though it lets visible light through.

REMEMBER

A PIR senses movement of warm bodies, not presence. A person sitting perfectly still may stop triggering it, which is why some automatic lights switch off in a quiet room.

Project ideas: an automatic stair light model; an intruder alarm for a cupboard or a model shop; an energy-saving small battery fan that runs only when someone is in the room.

CHAPTER 4

Measuring distance: the ultrasonic sensor

How it works: the **HC-SR04** has two round "eyes". One is a tiny speaker that sends a short burst of **ultrasound**, sound at 40 kHz, far too high for humans to hear. The other is a microphone that listens for the echo. The module measures how long the echo takes and keeps its Echo pin HIGH for that time.

Pins: VCC → 5V, Trig → pin 9, Echo → pin 10, GND → GND. It measures from about 2 cm to about 4 m, best within a narrow cone in front of it.

The maths: sound travels about 343 metres per second in air at 20 °C, which is 0.0343 cm per microsecond. The echo time covers the trip there and back, so:

distance in cm = echo time in microseconds $\times$ 0.0343 $\div$ 2

CPP

```
// HC-SR04: measure distance with a reusable function
const int trigPin = 9;
const int echoPin = 10;

float readDistanceCm() {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);          // a 10-microsecond pulse starts a reading
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    unsigned long t = pulseIn(echoPin, HIGH, 30000UL); // 0 if no echo
    if (t == 0) return -1;                // -1 means "nothing in range"
    return t * 0.0343 / 2;
}

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

void loop() {
    float cm = readDistanceCm();
    if (cm < 0) {
        Serial.println("Out of range");
    } else {
        Serial.print(cm, 1);
        Serial.println(" cm");
    }
    delay(200);
}
```

EXAMPLE

Harsh in Ahmedabad fixed the sensor at the top of a 100 cm deep bucket, facing down at the water. It read 35 cm. So the water was 100 - 35 = 65 cm deep. His water tank level model works the same way.

Temperature changes the speed of sound

Sound travels faster in warm air. A handy rule is speed $\approx 331 + 0.6 \times$ temperature in °C, in metres per second. At 20 °C that gives 343 m/s; on a 40 °C summer afternoon in Nagpur it gives 355 m/s. For school projects the difference is small, but for accurate work you can measure the temperature with another sensor and correct the speed in code.

TIP

Soft surfaces such as cushions and curtains soak up sound, and sloping surfaces bounce it away. Test your project facing a flat, hard surface first.

Project ideas: a parking helper that beeps faster as a toy car nears a wall; a smart dustbin that opens when a hand comes near; a height measurer fixed above a doorway; the eyes of an obstacle-avoiding robot.

CHAPTER 5

Temperature and humidity: LM35, DHT11 and DHT22

The LM35

How it works: the LM35 is a small chip in a three-legged plastic body, like a transistor. Its output voltage rises by 10 millivolts for every degree Celsius: 0.25 V at 25 °C and 0.40 V at 40 °C. In the simple three-wire circuit it measures from about 2 °C up to 150 °C.

Pins: hold it with the flat face towards you and the legs down. Left → 5V, middle (Vout) → A0, right → GND.

CPP

```
// LM35: average ten readings for a steadier temperature
const int sensorPin = A0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  long total = 0;
  for (int i = 0; i < 10; i++) {
    total += analogRead(sensorPin);
    delay(10);
  }
  float reading = total / 10.0;
  float tempC = reading * (5.0 / 1023.0) * 100.0;    // 10 mV per degree
  Serial.print("Temperature: ");
  Serial.print(tempC, 1);
  Serial.println(" C");
  delay(1000);
}
```

Averaging smooths out small jumps between readings, because the random ups and downs cancel out.

SAFETY

If the LM35 becomes hot a few seconds after powering up, it is connected the wrong way round. Unplug at once and let it cool before touching it.

The DHT11 and DHT22

How they work: each has a tiny humidity-sensing part, whose electrical property changes as it takes up water vapour from the air, plus a temperature sensor and a small chip. The chip sends both readings as digital data along one wire. **Humidity** means how much water vapour is in the air; "relative humidity" compares it with the most the air could hold at that temperature.

	DHT11 (usually blue)	DHT22 (usually white)
Temperature range	0 to 50 °C	-40 to 80 °C
Temperature accuracy	about ± 2 °C	about ± 0.5 °C
Humidity range	about 20 to 80 %	0 to 100 %
Humidity accuracy	about ± 5 %	about ± 2 to 5 %
Fastest reading	once a second	once every two seconds

Pins (three-pin module): + $\rightarrow$ 5V, - $\rightarrow$ GND, OUT $\rightarrow$ pin 2. A bare four-pin sensor also needs a 10 k Ω resistor from its data pin to 5V; modules have it built in.

Install "DHT sensor library" by Adafruit from **Sketch $\rightarrow$ Include Library $\rightarrow$ Manage Libraries**, and agree when it offers to install the libraries it depends on.

```
// DHT11 or DHT22: temperature, humidity and "feels like" heat index
#include <DHT.h>

#define DHTPIN 2
#define DHTTYPE DHT11      // change to DHT22 if you have one

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  delay(2000);
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  if (isnan(h) || isnan(t)) {
    Serial.println("Read failed. Check the wiring.");
    return;
  }
  float feels = dht.computeHeatIndex(t, h, false);  // false = Celsius
  Serial.print(t, 1);
  Serial.print(" C, ");
  Serial.print(h, 0);
  Serial.print(" % humidity, feels like ");
  Serial.print(feels, 1);
  Serial.println(" C");
}
```

DID YOU KNOW

On a humid day, sweat dries slowly, so your body cannot cool itself well and the air feels hotter than the thermometer says. The **heat index** is a "feels like" temperature that combines heat and humidity, which is why a humid 34 °C day in Chennai can feel worse than a dry 38 °C day in Jaipur.

Project ideas: a classroom weather station on an LCD; an automatic small battery fan that switches on above a set temperature; a study of how humidity changes from morning to night through the monsoon.

CHAPTER 6

Water sensors: soil moisture and rain

Both these sensors work on the same idea: water with dissolved salts conducts electricity, and dry material does not. More water between two metal tracks means less resistance.

The soil moisture sensor

How it works: a fork with two metal legs goes into the soil. Current flows from one leg to the other through the damp soil. A small module turns this into a voltage on AO, and a comparator gives DO.

Pins: VCC → 5V, GND → GND, AO → A0 (DO is optional). With most of these modules, the AO reading **falls** as the soil gets wetter.

CPP

```
// Soil moisture as a percentage, using your own calibration
const int soilPin = A0;
const int dryValue = 850;    // reading with the fork in dry soil
const int wetValue = 400;    // reading just after watering

void setup() {
  Serial.begin(9600);
}

void loop() {
  int raw = analogRead(soilPin);
  int percent = map(raw, dryValue, wetValue, 0, 100);
  percent = constrain(percent, 0, 100);    // keep it between 0 and 100
  Serial.print("Raw: ");
  Serial.print(raw);
  Serial.print("    Moisture: ");
  Serial.print(percent);
  Serial.println(" %");
  delay(1000);
}
```

Calibrating: note the raw reading in dry soil and again in freshly watered soil, and put your two numbers in `dryValue` and `wetValue`. `map()` then turns any reading into a percentage, and `constrain()` stops it going below 0 or above 100.

REMEMBER

Bare metal legs corrode in damp soil, because current flows through them all the time. A **capacitive** soil sensor senses water through a coated board with no bare metal, so it lasts much longer.

The rain sensor

How it works: the rain board has two comb-shaped metal tracks that fit between each other without touching. A raindrop that lands across two tracks lets a little current flow. The more of the board is wet, the lower the resistance.

Pins: the board connects to its module with two wires (either way round). Module VCC → 5V, GND → GND, DO → pin 7, AO → A0. On most modules, DO goes LOW when the board is wet.

```
// Rain sensor: tell dry, light rain and heavy rain apart
const int rainDigital = 7;
const int rainAnalog = A0;

void setup() {
  pinMode(rainDigital, INPUT);
  Serial.begin(9600);
}

void loop() {
  int wetness = analogRead(rainAnalog);    // lower = wetter
  if (digitalRead(rainDigital) == HIGH) {
    Serial.println("Dry");
  } else if (wetness > 500) {
    Serial.println("Light rain");
  } else {
    Serial.println("Heavy rain");
  }
  delay(1000);
}
```

The level 500 is only a starting point; test with a few drops and then with a wet cloth, and choose your own.

EXAMPLE

Pooja in Kolkata made a "bring in the clothes" alarm for the terrace. The rain board sits outside on a slope so water runs off, and a buzzer indoors sounds at the first drops. Her family now rushes upstairs before the clothes get soaked.

Project ideas: an automatic plant-watering model (a small pump needs a relay or transistor module and its own battery, never an Arduino pin); a model car wiper that starts when rain falls; a soil-dryness reminder for the school garden.

CHAPTER 7

The MQ-2 gas sensor

How it works: inside the steel mesh cap is a small tube coated with **tin dioxide**, with a tiny heater inside it. When the coating is hot, gases such as LPG, methane, hydrogen, alcohol vapour and smoke change its resistance. The module turns that change into a voltage on AO, and the comparator gives DO.

The heater uses roughly 150 mA at 5 V and makes the sensor warm. A USB-powered Uno can supply this from its 5V pin. The readings drift at first and settle after the heater has been on for a few minutes; a brand-new sensor settles better after many hours of running.

Pins: VCC → 5V, GND → GND, AO → A0 (DO is optional), and a passive buzzer from pin 8 to GND.

CPP

```
// MQ-2: compare the reading with the clean-air level found at start-up
const int gasPin = A0;
const int buzzerPin = 8;
int baseline = 0;

void setup() {
  Serial.begin(9600);
  Serial.println("Heating up for one minute...");
  delay(60000); // longer is better
  long total = 0;
  for (int i = 0; i < 20; i++) {
    total += analogRead(gasPin);
    delay(100);
  }
  baseline = total / 20;
  Serial.print("Clean-air reading: ");
  Serial.println(baseline);
}

void loop() {
  int value = analogRead(gasPin);
  Serial.println(value);
  if (value > baseline + 150) { // a big rise above clean air
    tone(buzzerPin, 2000, 300);
  }
  delay(500);
}
```

What the numbers mean: a hobby module can show that a reading has risen, but it cannot tell you which gas it is or exactly how much. Turning readings into real amounts needs careful lab calibration.

SAFETY

Never test this sensor with a real gas leak, a cylinder or a stove. The safest way is simply to explain how it works and show the readings settling as the heater warms. If a teacher wants a demonstration, only an adult does it: a single short puff of gas from an unlit lighter near the sensor, in an airy room, with no flame, spark or heater anywhere near. Do not touch the mesh cap; it gets hot.

REMEMBER

A school project is not a safety device. Homes and shops need properly certified gas detectors. If you ever smell gas at home: do not operate any electrical switch, do not light anything, open doors and windows, turn off the cylinder regulator only if it is safe to reach, leave the room and tell an adult at once.

Project ideas: a chart of how the reading settles during warm-up; a comparison of readings by an open window and inside a closed cupboard; a model kitchen showing why an LPG detector is fitted low on the wall (LPG is heavier than air, so a leak collects near the floor).

CHAPTER 8

Sound and touch sensors

The sound sensor

How it works: the module has a small round **microphone**. Sound waves make a thin layer inside it vibrate, which produces a tiny changing voltage. The comparator compares it with the level set by the preset and switches DO when the sound is louder. AO carries the microphone's signal, which changes hundreds or thousands of times a second as the sound wave goes up and down.

Pins: VCC → 5V, GND → GND, DO → pin 7, AO → A0. Modules differ: some set DO LOW on a loud sound, others HIGH. Watch the signal LED while you clap, and turn the preset until it flickers only for a clap, not for talking.

CPP

```
// Clap switch: each clap turns the LED on or off
const int soundPin = 7;      // module DO
const int ledPin = 13;
const int ACTIVE = LOW;      // change to HIGH if your module works the other way

bool ledOn = false;

void setup() {
  pinMode(soundPin, INPUT);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  if (digitalRead(soundPin) == ACTIVE) {
    ledOn = !ledOn;
    digitalWrite(ledPin, ledOn ? HIGH : LOW);
    delay(300);           // ignore the rest of the clap and its echo
  }
}
```

Good to know: one `analogRead()` of AO catches the wave at a random moment, so single readings jump about. To measure loudness, read AO many times over a few milliseconds and use the difference between the highest and lowest values: a bigger difference means a louder sound.

Project ideas: a clap-controlled lamp; a "silence please" meter for the library that lights a red LED when the room gets noisy; a sound-activated toy that dances when music plays.

The touch sensor

How it works: the popular TTP223 module has a small pad and a chip that senses **capacitance**, the ability to hold a tiny electric charge. Your finger, close to the pad, adds capacitance, and the chip notices the change. It works even through thin plastic, paper or glass, with no moving parts to wear out. Phone screens sense your fingers in a similar way.

Pins: VCC → 5V (it works from about 2 V to 5.5 V), GND → GND, SIG (or I/O) → pin 4. By default, SIG goes HIGH while you touch the pad. Small solder pads on the back can change it to toggle mode or reverse the output; check your module's notes before changing them.

CPP

```
// Touch sensor: a touch lamp that toggles on each touch
const int touchPin = 4;
const int ledPin = 13;

bool lampOn = false;
int lastTouch = LOW;

void setup() {
  pinMode(touchPin, INPUT);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  int touch = digitalRead(touchPin);
  if (touch == HIGH && lastTouch == LOW) { // a new touch has started
    lampOn = !lampOn;
    digitalWrite(ledPin, lampOn ? HIGH : LOW);
  }
  lastTouch = touch;
  delay(20);
}
```

TIP

Stick the module behind a thin plastic sheet or the wall of your project box. The pad still senses your finger through it, and your project looks neat and finished.

Project ideas: a touch lamp; a quiz buzzer where the first team to touch its pad wins; a hidden touch button that opens a model door lock with a servo.

Accelerometer basics

How it works: an **accelerometer** measures acceleration, which is how quickly speed changes. Inside the chip are microscopic structures that bend slightly when the chip speeds up, slows down or tilts, and the chip measures that bending. It measures along three directions, called the **axes**: X (left-right), Y (forward-back) and Z (up-down).

Even when the chip is still, it feels **gravity**. Lying flat on a table, it reads about 1 g on the Z axis and about 0 on X and Y. (1 g is the pull of Earth's gravity, about 9.8 metres per second per second.) Tilt it, and gravity shares itself between the axes. That is how phones know which way up they are and turn the screen.

The MPU-6050

A common choice is the **MPU-6050** on a small module, often labelled GY-521. It holds an accelerometer and a **gyroscope** (which measures turning speed) in one chip. It talks over **I2C**, a way for chips to share data over two wires: SDA (data) and SCL (clock).

Pins: VCC → 5V (the module has its own regulator), GND → GND, SDA → A4, SCL → A5. Its I2C address is 0x68.

This sketch uses only the Wire library, which comes with the IDE. It wakes the chip, then reads the three acceleration values. With the chip's default setting, 16,384 counts mean 1 g.

```
// MPU-6050: read acceleration on X, Y and Z in g
#include <Wire.h>

const int MPU = 0x68;          // the module's I2C address

int16_t readWord() {
    uint8_t high = Wire.read();
    uint8_t low = Wire.read();
    return (int16_t)(((uint16_t)high << 8) | low);
}

void setup() {
    Serial.begin(9600);
    Wire.begin();
    Wire.beginTransmission(MPU);
    Wire.write(0x6B);          // the power register
    Wire.write(0);              // 0 wakes the chip up
    Wire.endTransmission();
}

void loop() {
    Wire.beginTransmission(MPU);
    Wire.write(0x3B);          // the first acceleration register
    Wire.endTransmission(false);
    Wire.requestFrom(MPU, 6);  // two bytes each for X, Y and Z

    float x = readWord() / 16384.0;
    float y = readWord() / 16384.0;
    float z = readWord() / 16384.0;

    Serial.print("X: ");
    Serial.print(x, 2);
    Serial.print(" Y: ");
    Serial.print(y, 2);
    Serial.print(" Z: ");
    Serial.println(z, 2);
    delay(200);
}
```

Reading the output: lay the module flat and Z shows about 1.00. Tip it on one edge and X or Y moves towards 1.00 or -1.00 while Z falls. Shake it and all three jump.

EXAMPLE

Kabir in Dehradun made a tilt alarm for his bicycle model. If X or Y goes above 0.5 or below -0.5, the bicycle has fallen over, and a buzzer sounds. He tested it by gently tipping the model on his desk.

DID YOU KNOW

Some modules, such as those using the ADXL335 chip, give the three axes as three analog voltages instead of I2C data. You read them with `analogRead()` on A0, A1 and A2. The chip itself runs on 3.3 V, so use a module that has a regulator.

Project ideas: a tilt alarm; a gesture-controlled robot car that moves the way you tilt your hand; a simple step counter; an "earthquake" shake-table model that shows how strongly a table moves.

CHAPTER 10

Planning a sensor project

Choosing the right sensor

You want to sense	Good choice	Output
Day or night, brightness	LDR	analog
Something close in front	IR obstacle sensor	digital
A person moving in a room	PIR	digital
An exact distance	HC-SR04	timed pulse
Temperature only	LM35	analog
Temperature and humidity	DHT11 or DHT22	data
Dry or wet soil	soil moisture sensor	analog and digital
Rain starting	rain sensor	analog and digital
Smoke or gas (demonstration only)	MQ-2	analog and digital
A clap or loud noise	sound sensor	digital and analog
A finger touch	TTP223	digital
A flame	flame sensor	digital and analog
Tilt, shake or fall	MPU-6050	data

Plan your pins before you wire

When a project uses several sensors, write a pin plan first. Keep A4 and A5 free for I2C if you need an LCD or an MPU-6050, keep pins 0 and 1 free for USB, and use the ~ pins for anything that needs PWM.

EXAMPLE

Riya in Bengaluru planned her smart garden: soil sensor on A0, LDR on A1, DHT11 on pin 2, buzzer on pin 8, LCD on A4 and A5. Writing the plan first meant no two parts fought over one pin.

Calibrate, smooth and test

- 1. **Print raw readings** in the real place where the project will work, not just on your desk.
- 2. **Calibrate:** note the readings at the two extremes (dry and wet, dark and bright) and set your levels between them.
- 3. **Smooth:** average several readings, as in the LM35 sketch, when values jump about.
- 4. **Add a gap:** use two levels, one to switch on and one to switch off, so outputs do not flicker.
- 5. **Test one sensor at a time,** then join them.

When a sensor misbehaves

Problem	Likely cause
Reading stuck at 0 or 1023	signal wire loose, or VCC or GND missing
Digital output never changes	preset turned fully one way; turn it slowly while watching the signal LED
Readings jump randomly	loose breadboard contact, or nothing connected to the pin
Works on the desk, fails outside	sunlight (IR sensors), wind noise (sound) or water on the board
Module gets hot (an MQ-2 cap is meant to be warm)	VCC and GND reversed: unplug at once

REMEMBER

A good project explains itself. At a science exhibition, show the raw readings on the Serial Monitor or an LCD, and tell the judges how the sensor works and how you calibrated it.

SAFETY

Keep every sensor project on batteries of 12 V or less or on USB. Ask an adult to help with soldering, cutting and hot glue, and with any test that involves flame, gas or heat.

CHECK YOURSELF

Practice questions

1. What is a sensor? Give one example from your own body.
2. What is the difference between an analog output and a digital output?
3. What do the VCC, GND, DO and AO pins of a sensor module do?
4. What does "active low" mean for a module's DO pin?
5. Why must an LDR be used with a fixed resistor to be read by an Arduino?
6. In the LDR divider in this book, does the reading go up or down in brighter light?
7. How can you safely test a flame sensor without any flame?
8. Why might an IR obstacle sensor fail to notice a black object?
9. What does the white dome on a PIR sensor do?
10. Why can a PIR sensor not detect a person behind a glass window?
11. An HC-SR04's echo lasts 1,749 microseconds. How far away is the object?
12. Using speed $\approx 331 + 0.6 \times \text{temperature}$, what is the speed of sound at 30 °C?
13. An LM35 gives 0.36 V. What is the temperature?
14. Give two differences between the DHT11 and the DHT22.
15. What is the heat index?
16. Why does a bare soil sensor's fork corrode, and what kind of sensor avoids this?
17. What do `map()` and `constrain()` do in the soil moisture sketch?
18. Why must you never test an MQ-2 with a real gas leak, and what is the safest way to show it?
19. Why does the clap switch sketch wait 300 ms after detecting a sound?
20. How does a TTP223 touch sensor know that your finger is there?
21. What does an accelerometer read when it lies still and flat on a table, and why?
22. Which Uno pins are used for I2C, and which two sensors or displays in this book use them?
23. A digital output never changes, whatever you do. What should you try first?

Answers

1. A part that notices something in the world and turns it into an electrical signal. The eyes sense light, the skin senses heat and touch, the ears sense sound.
2. An analog output can be any voltage in a range and tells how much; a digital output is only HIGH or LOW and tells yes or no.
3. VCC is power, GND is ground, DO is the digital output set by the comparator and preset, and AO is the raw analog reading.
4. The pin goes LOW when the sensor is triggered, and stays HIGH otherwise.
5. The Uno measures voltage, not resistance. The LDR and fixed resistor form a voltage divider whose middle voltage changes with light.
6. It goes up, because the LDR's resistance falls in bright light, so the AO voltage rises.
7. Point a TV remote at it and press a button; most flame sensors react to the remote's infrared.
8. Black surfaces absorb infrared, so very little light bounces back to the photodiode.
9. It is a Fresnel lens divided into sections, so a moving warm body passes from one section's view to another and the sensor sees a change.
10. Ordinary glass blocks the infrared heat that warm bodies give out.
11. $1,749 \times 0.0343 \div 2 =$ about 30 cm.
12. $331 + 0.6 \times 30 = 349$ metres per second.
13. $0.36 \text{ V} \div 0.010 \text{ V per degree} = 36 \text{ }^{\circ}\text{C}$.
14. The DHT22 has a wider range (-40 to $80 \text{ }^{\circ}\text{C}$ against 0 to $50 \text{ }^{\circ}\text{C}$) and is more accurate (about $\pm 0.5 \text{ }^{\circ}\text{C}$ against $\pm 2 \text{ }^{\circ}\text{C}$); the DHT11 can be read once a second, the DHT22 once every two seconds.
15. A "feels like" temperature that combines heat and humidity, because humid air slows the cooling effect of sweat.
16. Current flows through its metal legs in damp soil all the time, slowly eating the metal. A capacitive sensor, with no bare metal in the soil, avoids this.
17. `map()` turns the raw reading, between the dry and wet values, into 0 to 100 per cent; `constrain()` keeps the result between 0 and 100 .
18. A real leak can catch fire or explode. The safest way is to explain how the sensor works and show its readings settling as it warms; any demonstration is done only by an adult, with no flame anywhere near.
19. To ignore the rest of the same clap and its echo, which would otherwise switch the LED again.
20. It senses capacitance. A finger near the pad adds capacitance, and the chip notices the change.
21. About 1 g on the Z axis and about 0 on X and Y, because it feels Earth's gravity pulling down.
22. A4 (SDA) and A5 (SCL); the MPU-6050 and the I2C LCD use them.

23. Turn the module's preset slowly while watching its signal LED, because the threshold may be set fully one way.

WORD LIST

Glossary

Accelerometer

a sensor that measures acceleration, including the pull of gravity, along three axes.

Active low

an output that goes LOW when the sensor is triggered.

Analog output

a signal that can take any voltage in a range.

Axis

one of the three directions, X, Y and Z, along which an accelerometer measures.

Calibration

finding a sensor's real readings at known conditions so code can use them.

Capacitance

the ability to hold a small electric charge; a finger changes it near a touch pad.

Comparator

a chip that compares two voltages and switches its output when one passes the other.

Digital output

a signal that is only HIGH or LOW.

Fresnel lens

a flat, grooved lens, like a PIR sensor's dome, that gathers light from many directions.

Heat index

a "feels like" temperature that combines heat and humidity.

Humidity

the amount of water vapour in the air.

I2C

a way for chips to share data over two wires, SDA and SCL.

Infrared

light just beyond red that our eyes cannot see; warm bodies and flames give it out.

LDR

light-dependent resistor; its resistance falls in bright light.

Module

a sensor mounted on a small board with the extra parts it needs.

Photodiode

a part that lets more current pass when light falls on it.

PIR

passive infrared sensor, which senses moving warm bodies.

Preset

a small adjustable resistor, turned with a screwdriver, used to set a threshold.

Sensor

a part that notices a change in the world and turns it into an electrical signal.

Threshold

the level at which a digital output switches from one state to the other.

Ultrasound

sound too high-pitched for humans to hear.

Voltage divider

two resistances in series that share a voltage; used to read resistive sensors.



Gurukul Technicals Institute

Computer education institute in Delhi, with authorised member centres, robotics classes and science projects, working models and DIY kits.

COURSES

Computer & Office Skills · Accounting & Business Software · Programming & Web Development · Design & Digital Skills · Hardware & Networking · Robotics & Coding · Other Programmes

PROJECTS AND KITS

Ready-made school projects · DIY and STEM kits · Robotics and coding · Components and spares · Engineering projects

CONTACT

Head office: A-10, Chief Commissioner Colony, opposite Rana Pratap Bagh, Delhi 110007

Phone: 011 2721 1512 · **Mobile and WhatsApp:** +91 97116 99511

Email: engineergurukul@yahoo.co.in

Website: engineergurukul.com · **Instagram:** [@science_projects_club](https://www.instagram.com/science_projects_club)

More free ebooks for computer courses, robotics and science projects: engineergurukul.com/resources/ebooks/

Free to read, print and share. Not for sale.