

# Hardware

AS

Read online at [pastopic.com/cambridge/computer-science-9618/notes/hardware](https://pastopic.com/cambridge/computer-science-9618/notes/hardware)

Last checked 30 September 2026

© 2026 Pastopic. Free for personal study. Please share the link, not the file.

---

## What you need to know

By the end of this topic you should be able to:

1. **Explain why a computer needs input and output devices, primary memory and secondary storage**, including removable storage.
2. **Explain what an embedded system is**, with its benefits and drawbacks.
3. **Describe how these devices work**: laser printer, 3D printer, microphone, speakers, magnetic hard disk, solid state (flash) memory, optical disc reader/writer, touchscreen and virtual reality (VR) headset.
4. **Explain how and why buffers are used**.
5. **Explain the differences between RAM and ROM**, and what each is used for in different devices.
6. **Explain the differences between SRAM and DRAM**, and why a device uses one rather than the other.
7. **Explain the differences between PROM, EPROM and EEPROM**.
8. **Explain monitoring and control systems**: the difference between them, how sensors (temperature, pressure, infrared, sound and others) and actuators are used, and why feedback matters.
9. **Use the symbols for the NOT, AND, OR, NAND, NOR and XOR gates**, say what each gate does and write its truth table. Every gate except NOT has exactly two inputs.
10. **Build a logic circuit** from a problem statement, a logic expression or a truth table.
11. **Build a truth table** from a problem statement, a logic circuit or a logic expression.
12. **Write a logic expression** from a problem statement, a logic circuit or a truth table.

Simplifying expressions with Boolean algebra and Karnaugh maps is A Level (Paper 3), not this topic.

---

## Understand it

### The parts of a computer system

A computer takes data in, processes it, stores it and sends results out, and each job needs its own hardware:

- **Input devices** (keyboard, microphone, touchscreen, sensors) turn the outside world into data; **output devices** (screen, printer, speakers, actuators) turn results into something people or machines can use.

- **Primary memory** (RAM and ROM) is memory the processor can reach **directly**. Programs and data must be in RAM while they run.
- **Secondary storage** (hard disk, SSD, optical disc) is **non-volatile**: it keeps files, the operating system and applications when the power is off, and holds far more than RAM for less money per gigabyte. The processor cannot use it directly; data is loaded into RAM first.
- **Removable storage** (USB flash drive, memory card, optical disc) lets you carry data between computers, keep a backup away from the computer, or install software.

## Embedded systems

An **embedded system** is a small computer (a microprocessor or microcontroller with its own memory and input/output) **built into a larger device** to do **one dedicated job**. A washing machine's controller, for example, runs only the wash programs. Its program is **firmware** stored in ROM (often EEPROM or flash), and it usually has no general-purpose operating system and needs little processing power.

**Benefits:** small, cheap to make in large numbers, low power use, and reliable and fast at its one task.

**Drawbacks:** the user can't easily update or change the firmware; faults are hard to find and repair (a specialist job, often costing more than a new device); it can't be adapted for a different task; and faulty or out-of-date devices tend to be thrown away, adding to electronic waste.

## How the devices work

**Laser printer.** A rotating drum is given an electric charge. A laser, steered by a mirror, scans across the drum and changes the charge wherever the page is to be printed, leaving an electrostatic "image" of the page on the drum. Charged toner powder sticks only to that image. The paper is given a charge that attracts the toner, so the toner is pulled off the drum onto the paper. A **fuser** (heated rollers) melts the toner and presses it into the paper, and the drum is discharged and cleaned for the next page.

**3D printer.** It makes a solid object by **additive manufacturing**: it builds the object **one thin layer at a time**, from the bottom up, following a digital model made in CAD (computer-aided design) software. The head or laser moves in x, y and z to form each layer, fused to the one below. Common methods:

- **fused deposition modelling (FDM)**: plastic filament is heated and pushed through a nozzle (extruder), then cools and hardens;
- **stereolithography (SLA)**: a UV laser hardens (cures) liquid resin layer by layer;
- **selective laser sintering (SLS)**: a laser fuses powdered plastic or metal.

A temperature sensor keeps the nozzle or bed hot enough without overheating.

**Microphone.** Sound waves make a thin **diaphragm** vibrate. The diaphragm moves a coil past a magnet (or, in a condenser microphone, changes the capacitance between two plates), producing a varying **analogue electrical signal** that matches the sound wave. It is then sampled and converted to digital.

**Speakers** work the other way round. The digital sound is converted to an amplified analogue signal. This varying current flows through a coil attached to a **cone** and sitting in a permanent magnet's field, so the coil and cone move in and out, pushing the air to make sound waves.

**Magnetic hard disk.** One or more **platters** coated in a magnetisable material are mounted on a **spindle** and spun at high speed inside a sealed case. Each surface has a **read/write head** on an **arm** that moves across it, floating just above the surface. The surface is divided into concentric **tracks** and **sectors**. To write, a current in the head creates a magnetic field that magnetises a tiny area one way or the other (1 or 0). To read, the magnetised areas passing under the head change the magnetic field, which produces a changing current in the head.

**Solid state (flash) memory** has no moving parts. It is a grid of cells built from **NAND** or **NOR** gates. The mark schemes model each cell as **two transistors**: a **control gate** and a **floating gate**, separated by a thin insulating layer. A voltage on the control gate lets electrons pass into the floating gate, where they stay trapped **even without power**; trapped or not trapped gives the bit's value. The control gate decides whether current can flow through the cell when it is read. Data is erased in blocks by applying a voltage that removes the electrons.

**Optical disc reader/writer.** The disc spins at high speed and a **laser** on a movable optical head follows a single **spiral track** from the centre outwards. To write, a strong laser burns **pits** into the surface; the unburnt parts are **lands**. To read, a weaker laser shines on the track and a sensor measures the reflected light: pits and lands reflect differently, and the changes are read as 1s and 0s.

**Touchscreen.** A **resistive** screen has two flexible conductive layers with a small gap between them. Pressing makes the layers touch, **completing a circuit** at that point, so any object (finger, stylus, gloved hand) works. A **capacitive** screen has a conductive layer that holds an electric charge; a finger draws some charge, causing a **change in the electric current** at that point, so it needs a finger or special stylus. Other types detect a broken grid of **infrared** beams, or a shadow seen by cameras. In every case the processor calculates the **x and y coordinates** of the touch and matches them to whatever is displayed there (a button or menu item), then carries out that action.

**VR headset.** One or two small **screens** (LCD or OLED), one per eye, viewed through **lenses**, show slightly different images so the scene looks 3D. Speakers or headphones give surround sound. A **gyroscope** and **accelerometer** detect the direction and speed of head movement; the processor uses this to redraw the scene to match where you are looking. Some headsets have cameras that track eye movements.

## Buffers

A **buffer** is an area of memory that holds data **temporarily** while it moves between two devices that work at **different speeds**. The processor is far faster than a printer or disk. Without a buffer it would have to wait for the slow device after every small piece of data.

With a buffer, the processor fills it quickly and then goes back to other tasks. The device empties it at its own slower speed. When the buffer is empty (or nearly so) the device sends an **interrupt** asking for more data, and the process repeats. The buffer also works the other way, for input: keystrokes, sensor readings or a streamed video are held until the processor or player is ready.

## RAM and ROM

|                | RAM (random access memory)                                                                   | ROM (read only memory)                                                                                       |
|----------------|----------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------|
| Volatile?      | <b>Yes:</b> contents lost when power is off                                                  | <b>No:</b> contents kept without power                                                                       |
| Can it change? | Read and written all the time                                                                | Normally only read; fixed or changed rarely                                                                  |
| Holds          | Data and instructions <b>currently in use:</b> running programs, open files, parts of the OS | Instructions that must be there at switch-on: the <b>bootstrap / start-up</b> program, BIOS, <b>firmware</b> |
| Size           | Large (GB in a PC)                                                                           | Small                                                                                                        |

**More RAM** improves performance: more of the running programs and data fit in RAM, so the computer uses less virtual memory and fetches less from slower secondary storage.

In an embedded device: the washing machine's ROM holds the program that runs the wash cycles; its RAM holds the choices the user just made, the current sensor readings and the time left.

## SRAM and DRAM

Both are kinds of RAM, so both are volatile.

|                     | SRAM (static RAM)                                  | DRAM (dynamic RAM)                                            |
|---------------------|----------------------------------------------------|---------------------------------------------------------------|
| How a bit is stored | Transistors arranged as a <b>flip-flop</b> (latch) | A <b>capacitor</b> holding a charge, plus a transistor        |
| Refreshing          | <b>Not needed</b> while powered                    | Must be <b>refreshed</b> constantly, as the charge leaks away |
| Speed               | <b>Faster</b> access                               | Slower access                                                 |
| Cost                | More expensive per bit                             | <b>Cheaper</b> per bit                                        |
| Density             | Fewer bits per chip (more transistors per bit)     | <b>More bits per chip</b> (simpler cell)                      |
| Power               | Uses less power (no refreshing)                    | Uses more power (refresh cycles)                              |
| Typical use         | Processor <b>cache</b> , where speed matters       | <b>Main memory</b> , where large, cheap capacity matters      |

So choose **SRAM** when speed is most important (cache, a device that must respond quickly) and **DRAM** when you need a lot of memory at low cost and top speed isn't needed (main memory, a printer storing page data).

## PROM, EPROM and EEPROM

These are kinds of ROM that can be programmed after manufacture:

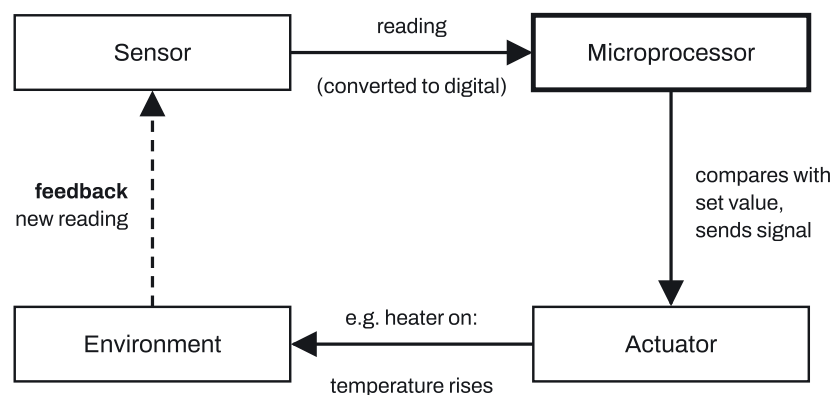
- **PROM** (programmable ROM): written **once** after it is made, with a special device that burns fuses. It can never be changed.
- **EPROM** (erasable PROM): erased with **ultraviolet light** through a small window. The chip must be **removed** from the device, and the **whole chip** is erased before it is rewritten. It can be reused many times.
- **EEPROM** (electrically erasable PROM): erased with an **electrical signal (voltage)** while it stays **in the device**. It can erase and rewrite **single bytes** or the whole chip, many times. Flash memory is a kind of EEPROM.

EEPROM suits devices whose firmware may need updating: the maker can release an update and the user installs it without special equipment or taking the device apart.

## Monitoring and control systems

A **monitoring system** uses sensors to **measure and record** conditions, and may **display** readings or **warn** (a light, a sound, a message). It **does not act** to change what it measures, so its outputs don't affect its next inputs, and it has no actuators.

A **control system** also uses sensors, but it **acts** automatically to change the conditions, using **actuators**: motors, valves, pumps, heaters, switches. What it does changes what the sensor measures next. That loop is **feedback**.



The microprocessor compares each reading with a **set value** or range. Outside the range, it signals the actuator to act; inside, it turns the actuator off. **Feedback matters** because the output affects the next input: the system sees whether its action worked and adjusts again, keeping conditions within the set limits automatically.

**Sensors** measure physical quantities, usually as an analogue signal that is converted to digital for the processor.

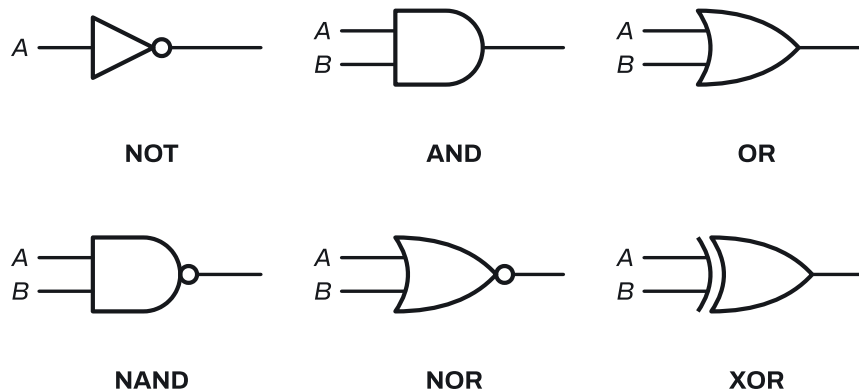
| Sensor      | Measures             | Example use                                 |
|-------------|----------------------|---------------------------------------------|
| Temperature | How hot something is | Oven, greenhouse heating, 3D printer nozzle |

| Sensor                 | Measures                                         | Example use                                                                        |
|------------------------|--------------------------------------------------|------------------------------------------------------------------------------------|
| Pressure               | Force pressing on it (or weight)                 | Detects an item lifted off a shelf, a person sitting on a seat, a vehicle's weight |
| Infrared               | Heat given off by a body, or a beam being broken | Detects a person nearby, counts objects passing, measures height                   |
| Sound                  | Noise level                                      | Detects breaking glass, a voice or a clap                                          |
| Light                  | Brightness                                       | Turns lights on at dusk                                                            |
| Ultrasonic / proximity | Distance to an object                            | Parking sensors, robots avoiding obstacles                                         |

An **actuator** turns an electrical signal into physical movement or action: a motor opens a window, a solenoid pushes a rejected item off a conveyor belt, a valve lets water through.

## Logic gates

A logic gate takes binary inputs (1 = on/true, 0 = off/false) and produces one binary output.



| Gate | Output is 1 when ...                                  | In words                       |
|------|-------------------------------------------------------|--------------------------------|
| NOT  | the input is 0                                        | reverses the input             |
| AND  | <b>both</b> inputs are 1                              | both                           |
| OR   | <b>at least one</b> input is 1                        | either or both                 |
| NAND | <b>not both</b> inputs are 1 (output 0 only for 1, 1) | NOT (AND)                      |
| NOR  | <b>both</b> inputs are 0 (output 1 only for 0, 0)     | NOT (OR)                       |
| XOR  | the inputs are <b>different</b>                       | one or the other, but not both |

| A | B | AND | OR | NAND | NOR | XOR |
|---|---|-----|----|------|-----|-----|
| 0 | 0 | 0   | 0  | 1    | 1   | 0   |
| 0 | 1 | 0   | 1  | 1    | 0   | 1   |

| A | B | AND | OR | NAND | NOR | XOR |
|---|---|-----|----|------|-----|-----|
| 1 | 0 | 0   | 1  | 1    | 0   | 1   |
| 1 | 1 | 1   | 1  | 0    | 0   | 0   |

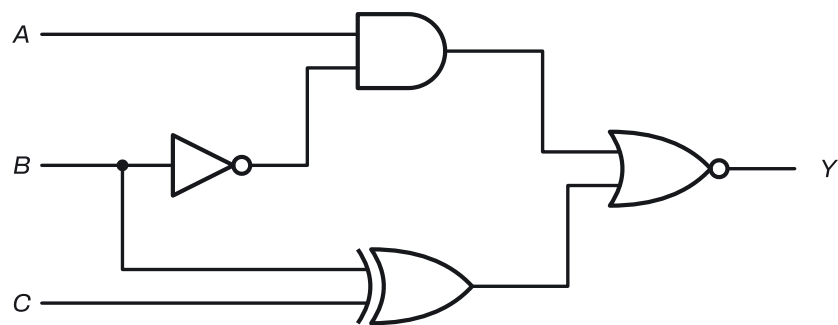
The small circle (the "bubble") always means NOT. NAND and NOR are AND and OR with the output reversed; compare their columns.

## Circuits, expressions and truth tables

The same logic can be written three ways, and you must be able to turn any one into the others.

- A **logic expression** uses gate names and brackets, for example  $Y = (A \text{ AND NOT } B) \text{ NOR } (B \text{ XOR } C)$ . The brackets show what is worked out first; NOT applies to what comes straight after it (a single input, or a whole bracket).
- A **logic circuit** draws each operator as a gate. Wires carry values from left to right. A **dot** where wires meet means the wire splits and carries the same value both ways; wires that cross without a dot are not joined.
- A **truth table** lists every combination of inputs and the output for each. With 3 inputs there are  $2^3 = 8$  rows, written in binary order 000, 001, ..., 111.

Here is the circuit for  $Y = (A \text{ AND NOT } B) \text{ NOR } (B \text{ XOR } C)$ . The innermost parts become the gates on the left, and the last operator applied is the gate on the right.



To find its truth table, add a **working column** for each gate before the last:

| A | B | C | A AND NOT B | B XOR C | Y |
|---|---|---|-------------|---------|---|
| 0 | 0 | 0 | 0           | 0       | 1 |
| 0 | 0 | 1 | 0           | 1       | 0 |
| 0 | 1 | 0 | 0           | 1       | 0 |
| 0 | 1 | 1 | 0           | 0       | 1 |
| 1 | 0 | 0 | 1           | 0       | 0 |
| 1 | 0 | 1 | 1           | 1       | 0 |

| A | B | C | A AND NOT B | B XOR C | Y |
|---|---|---|-------------|---------|---|
| 1 | 1 | 0 | 0           | 1       | 0 |
| 1 | 1 | 1 | 0           | 0       | 1 |

The NOR column is 1 only where **both** working columns are 0.

Going **from a truth table to an expression**, use **sum of products**: for each row where the output is 1, write an AND of all the inputs, putting NOT in front of each input that is 0 in that row. Then join these terms with OR. The output of Y above is 1 in rows 000, 011 and 111, so

$$Y = (\text{NOT } A \text{ AND NOT } B \text{ AND NOT } C) \text{ OR } (\text{NOT } A \text{ AND } B \text{ AND } C) \text{ OR } (A \text{ AND } B \text{ AND } C).$$

Each term is 1 in exactly one row, so the OR of them gives the same truth table as the original expression.

## Key facts and formulas

There is no formula list in the 9618 exams, so every fact here is **Learn it**.

| Formula                                                                                                                                          | Status   |
|--------------------------------------------------------------------------------------------------------------------------------------------------|----------|
| NOT: output = 1 only when the input = 0                                                                                                          | Learn it |
| AND: output = 1 only when both inputs = 1; NAND is the reverse (output = 0 only for 1, 1)                                                        | Learn it |
| OR: output = 1 when at least one input = 1; NOR is the reverse (output = 1 only for 0, 0)                                                        | Learn it |
| XOR: output = 1 only when the inputs are different                                                                                               | Learn it |
| Truth table rows = $2^n$ for $n$ inputs: 3 inputs $\rightarrow$ 8 rows, in binary order 000 to 111                                               | Learn it |
| Sum of products: one AND term per output-1 row (NOT on each 0 input), terms joined by OR                                                         | Learn it |
| RAM: volatile, read/write, current programs and data. ROM: non-volatile, start-up instructions and firmware                                      | Learn it |
| SRAM: flip-flops, no refresh, faster, dearer, cache. DRAM: capacitors, refreshed, cheaper, denser, main memory                                   | Learn it |
| PROM: write once. EPROM: UV, removed, whole chip. EEPROM: voltage, in place, byte by byte                                                        | Learn it |
| Control system: sensor $\rightarrow$ processor compares with set value $\rightarrow$ actuator $\rightarrow$ feedback (output changes next input) | Learn it |

## How to do it

We have tagged 60 Paper 1 questions for this topic (2021–2025; 57 different questions, 3 repeated on a second paper). 45 of the 60 mix more than one type, so the counts add up to more than 60.

| Question type                                                       | Questions |
|---------------------------------------------------------------------|-----------|
| Complete a truth table from a logic expression or circuit           | 19        |
| Draw a logic circuit from an expression or problem statement        | 15        |
| Describe the principal operation of a hardware device               | 12        |
| Identify and justify monitoring vs control; explain feedback        | 11        |
| Write a logic expression from a logic circuit                       | 10        |
| Describe gate functions, truth tables and symbols                   | 9         |
| Define embedded systems; give features, benefits, drawbacks         | 9         |
| Compare SRAM and DRAM; choose one for a device                      | 8         |
| State what RAM, ROM and secondary storage hold and why              | 8         |
| Explain how and why a buffer is used                                | 7         |
| Choose a sensor (or actuator) for a scenario and say how it is used | 7         |
| Distinguish PROM, EPROM and EEPROM; justify EEPROM                  | 6         |
| Write a logic expression from a truth table or problem statement    | 5         |
| Justify magnetic hard disk vs solid state storage                   | 2         |

The ideas are all in "Understand it". This section is about turning them into marks.

### 1. Complete a truth table from a logic expression or circuit (19 questions)

1. Write the input rows in binary order (000 to 111), if they aren't printed already.
2. Split the expression at its **last** operator, the one outside all brackets. Make a working column for each side, and for any smaller part you might get wrong (a NOT on an input, an inner NAND).
3. Fill each working column **top to bottom**, using the gate rules. Then combine the columns with the last operator.

For a **circuit**, label each gate's output with a letter and make that the working column. Follow each wire back to its source, remembering that a dot means a split.

#### Variations:

- **Chained operators** such as  $P \text{ XOR } Q \text{ XOR } R$ : work left to right,  $(P \text{ XOR } Q)$  first, then  $\text{XOR } R$ . A chain of XORs is 1 when an odd number of its inputs are 1.
- **NOT in front of a bracket** reverses the whole bracket: work out the bracket as a column, then flip it.

- **Find the wrong rows** in a given table: work out the full output column yourself, then list the rows that differ.
- Usually 1 mark per block of four rows, so one slip costs a mark. Write clear 0s and 1s; don't overwrite digits.

## 2. Draw a logic circuit from an expression or problem statement (15 questions)

1. Find the **last** operator; that is the gate on the far right, whose output is the answer.
2. Work inwards: each bracket becomes a gate feeding it. Draw the innermost gates first, on the left, next to the inputs.
3. Use **two-input gates only** (NOT has one input). A three-way AND is two AND gates.
4. Put a **NOT gate** exactly where the expression has NOT: "NOT A AND B" has the NOT on input A only; "NOT (A AND B)" has it after the AND gate (or use NAND if the expression says NAND).
5. Where one input goes to two gates, draw a **dot** at the split. Label the inputs and the output.

**Don't simplify.** Draw exactly the gates in the expression. The mark scheme gives marks for groups of gates (for example "the first two gates" and "the last two gates"), and a clever shortcut may not match it.

**From a problem statement:** first write the logic expression, one condition at a time, then draw it.

1. Turn each condition into a small expression using the parameter table: "too high" = T, "acceptable" = NOT T.
2. Read the **joining words**: "and" → AND, "or" → OR, "either ... or ... but not both" → XOR.
3. Group the conditions with brackets the way the bullets group them. Then draw it as above (see Example 2).

## 3. Describe the principal operation of a hardware device (12 questions)

1. Look at the marks: a 4-mark "describe" needs four separate steps, in order, from input to result.
2. Use the device's **technical words** (platter, spindle, read/write head; pits and lands; diaphragm; drum, toner, fuser; additive, layer by layer; floating gate, control gate), and say what each part does.
3. For **fill-the-gap** and **true/false** versions, the missing words are the key terms in "Understand it", so learn them exactly.

**Variations:** *touchscreen*: one or two types, then that the processor finds the **coordinates** and matches them to the item shown. *3D printer*: the general steps (CAD file, additive, layer by layer) plus one named method.

## 4. Identify and justify monitoring vs control; explain feedback (11 questions)

1. Decide: does the system **act to change** the thing it measures, using an **actuator**? Then it is **control**. Does it only record, display or warn? Then it is **monitoring**. The choice alone usually scores **no mark**.
2. Justify with the scenario: name the sensor and the actuator; say what the output does to the next input.
  - Control: "the actuator (motor) opens the doors, and when the person has walked through, the sensor's next reading changes, so the doors close".

- Monitoring: "turning on the warning sign does not change the vehicle's height, so there is no feedback, and there is no actuator".
3. "Explain the importance of feedback": output affects the next input, so the system can adjust automatically and **keep conditions within the set limits**.

Don't just repeat the question as your reason.

## 5. Write a logic expression from a logic circuit (10 questions)

1. Label the output of every gate with the expression it produces, working **left to right**: the first gates use the inputs; later gates combine the labels.
2. Put **brackets** round each gate's result before it goes into the next gate.
3. Write the final gate's result, using the gate names (AND, NOR ...), not symbols like + or ·, and **don't simplify**. Either order of inputs to a gate is accepted.

Example 3 is read this way. Two small circuits (X and Y) are usually 1 mark each.

## 6. Describe gate functions, truth tables and symbols (9 questions)

1. Say **exactly when the output is 1**, and that otherwise it is 0. "NAND is the opposite of AND" is not enough.
2. "Give inputs that make the output 1": read the truth table. AND needs 1, 1; NAND any pair but 1, 1; NOR only 0, 0; XOR 0, 1 or 1, 0.
3. Symbols: AND flat-backed; OR curved-backed and pointed; XOR with the extra curve; a bubble on NOT, NAND and NOR.

## 7. Define embedded systems; give features, benefits, drawbacks (9 questions)

1. **Define** with the features from "Understand it": processor, memory and I/O **built into** a larger device, **one dedicated task**, **firmware** the user can't easily change.
2. **Apply** each feature to the named device, such as "the TV's processor only decodes the signal and shows the picture".
3. Benefits and drawbacks: one per mark, about the device if one is named.

## 8. Compare SRAM and DRAM; choose one for a device (8 questions)

1. Give each difference as a **pair**: "DRAM must be refreshed; SRAM does not need refreshing".
2. **Why this device uses X**: pick the table rows that matter. Speed for a cache or a laptop's processor (SRAM); capacity and cost for a printer's page memory (DRAM).
3. **Advantages of DRAM**: cheaper per bit, more data per chip. **Disadvantages**: needs refreshing, slower access.

## 9. State what RAM, ROM and secondary storage hold and why (8 questions)

1. **RAM:** data and instructions **in use now**, named for the device (current sensor readings, the user's settings, the current video).
2. **ROM:** what must be there at switch-on and must never change: start-up (bootstrap) instructions, BIOS, firmware.
3. **Secondary:** anything kept after power-off, such as saved videos or a car's total miles (the current trip's miles are in primary memory).

## 10. Explain how and why a buffer is used (7 questions)

1. **Why** (often 1 mark): the processor and the device work at different speeds, so the buffer frees the processor.
2. **How** (3–4 marks), in order: the processor sends data to the buffer at **its** speed → the device takes it at **its own** speed → the processor carries on with other work → an **interrupt** asks for more when the buffer is empty.
3. Name the real devices (laptop to printer, computer to hard disk, doorbell to phone).

## 11. Choose a sensor (or actuator) for a scenario and say how it is used (7 questions)

1. Name the physical type (pressure, infrared, temperature, sound, light, ultrasonic), not "motion sensor" or "camera".
2. Say **what it detects and what change triggers the action**: "a pressure sensor under the seat detects the extra weight when someone sits down".
3. For an **actuator**, say what it physically does (a motor turns, an arm pushes the item off the belt).

## 12. Distinguish PROM, EPROM and EEPROM; justify EEPROM (6 questions)

1. Base each difference on one of four points: **how many times** it can be written, **how** it is erased (UV or voltage), **where** (removed or in place), **how much** (whole chip or single bytes).
2. **Why EEPROM**: the firmware can be **updated** many times, **without removing the chip** or special equipment, so users get new features or fixes.

## 13. Write a logic expression from a truth table or problem statement (5 questions)

1. **Truth table**: sum of products, as in "Understand it"; usually 1 mark per correct term.
2. **Problem statement**: one bracket per condition, as in type 2: "the system is on and a person is detected" → (E AND C).
3. **Tick or match** versions: test each expression on a few rows and rule out any that disagree.

## 14. Justify magnetic hard disk vs solid state storage (2 questions)

A reason, then its application, for 2 marks each. **Hard disk for a video server:** lower cost **per gigabyte**, so the huge capacity costs less; and it lasts longer under constant writing, as flash cells wear out. **SSD for a portable device:** no moving parts, faster, lighter, less power. "Magnetic is cheaper" alone is too vague.

---

## Worked examples

---

The bold codes show where marks are earned, as in Cambridge mark schemes. The number is how many marks.

- **B1** mark for a correct result on its own, with no method needed.
- 

### Example 1

(a) Describe the operation of a two-input NOR gate. **[1]**

(b) Complete the truth table for the logic expression

$$Y = (A \text{ AND NOT } B) \text{ NOR } (B \text{ XOR } C) \text{ [2]}$$

(c) Draw a logic circuit for the same expression. **[2]**

(a) The output is 1 only when both inputs are 0; for every other combination the output is 0. **B1**

(b) Working columns, rows 000 to 111 (the full table is in "Circuits, expressions and truth tables"):

- A AND NOT B: 0, 0, 0, 0, 1, 1, 0, 0
- B XOR C: 0, 1, 1, 0, 0, 1, 1, 0
- Y is 1 only where both are 0: **1, 0, 0, 1, 0, 0, 0, 1**

**B1** for the first four rows, **B1** for the last four.

(c) The drawn circuit is in "Circuits, expressions and truth tables". **B1** for the NOT on B and the AND gate with A, and the XOR gate with B and C. **B1** for the final NOR gate taking both outputs. The dot on B shows it feeds two gates.

### Example 2

A garden watering pump (X) is controlled by this parameter table.

| Parameter | Description | Binary value | Condition   |
|-----------|-------------|--------------|-------------|
| S         | Soil        | 1 / 0        | Dry / Moist |

---

| Parameter | Description      | Binary value | Condition               |
|-----------|------------------|--------------|-------------------------|
| R         | Rain             | 1 / 0        | Detected / Not detected |
| M         | Manual override  | 1 / 0        | On / Off                |
| T         | Water tank level | 1 / 0        | Low / Not low           |

The pump switches on ( $X = 1$ ) if:

- the soil is dry and rain is not detected, or the manual override is on
- and the water tank level is not low.

(a) Write the logic expression for the pump. **[3]**

(b) Describe the logic circuit for your expression, gate by gate. **[2]**

(c) State the value of  $X$  when  $S = 1$ ,  $R = 1$ ,  $M = 1$  and  $T = 0$ . **[1]**

**(a)** "Soil dry and rain not detected"  $\rightarrow S \text{ AND NOT } R$ . **B1**

"... or the manual override is on"  $\rightarrow (S \text{ AND NOT } R) \text{ OR } M$ . **B1**

"and the tank is not low" applies to everything above, so the final gate is an AND with NOT T:

$X = ((S \text{ AND NOT } R) \text{ OR } M) \text{ AND NOT } T$  **B1**

**(b)** A NOT gate on R, and an AND gate taking S and NOT R; an OR gate taking that AND output and M. **B1**

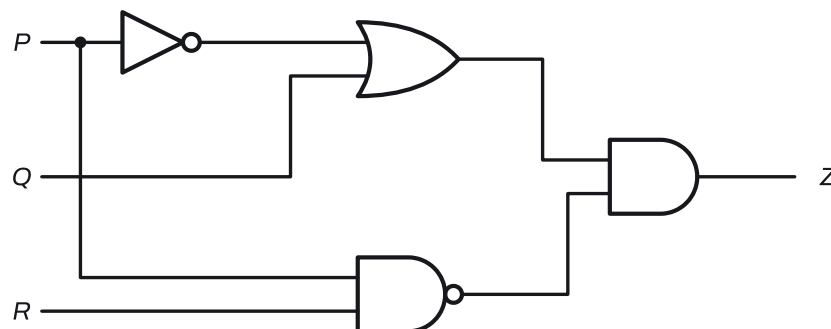
A NOT gate on T, and a final AND gate taking the OR output and NOT T. **B1**

**(c)**  $S \text{ AND NOT } R = 1 \text{ AND } 0 = 0$ . Then  $0 \text{ OR } M = 0 \text{ OR } 1 = 1$ . Then  $1 \text{ AND NOT } T = 1 \text{ AND } 1 = 1$ , so  $X = 1$ . **B1**

(The override is on and the tank has water, so the pump runs even though it is raining.)

### Example 3

(a) Write the logic expression for this logic circuit. Do not simplify. **[2]**



(b) Complete the truth table for the circuit. **[2]**

(a) Top gates: NOT P, then (NOT P OR Q). Bottom gate: (P NAND R). The final AND joins them:

$$Z = (\text{NOT } P \text{ OR } Q) \text{ AND } (P \text{ NAND } R)$$

**B1** for each bracket with the AND. Writing (P NAND R) AND (NOT P OR Q) is equally correct.

(b)

| P | Q | R | NOT P OR Q | P NAND R | Z |
|---|---|---|------------|----------|---|
| 0 | 0 | 0 | 1          | 1        | 1 |
| 0 | 0 | 1 | 1          | 1        | 1 |
| 0 | 1 | 0 | 1          | 1        | 1 |
| 0 | 1 | 1 | 1          | 1        | 1 |
| 1 | 0 | 0 | 0          | 1        | 0 |
| 1 | 0 | 1 | 0          | 0        | 0 |
| 1 | 1 | 0 | 1          | 1        | 1 |
| 1 | 1 | 1 | 1          | 0        | 0 |

**B1** for the first four rows, **B1** for the last four. When P = 0, NOT P = 1 makes the OR 1, and a NAND with a 0 input is 1, so the first four rows are all 1.

## Example 4

A factory's embedded controller keeps a drying oven between 60 °C and 65 °C. It switches a heater on when the oven is too cold and off when it is too hot. Every hour it sends a report to a laser printer.

(a) Explain why the oven system is a control system and not a monitoring system. **[3]**

(b) Give two reasons why the controller's firmware is stored in EEPROM and not EPROM. **[2]**

(c) Explain how a buffer is used when the report is sent to the laser printer. **[3]**

(d) Describe how the laser printer transfers the report onto paper. **[4]**

(a) It uses an actuator (the heater) to change the temperature, not just to record it. **B1** The heater's output changes the oven temperature, which is the next input read by the temperature sensor, so there is feedback.

**B1** The system compares each reading with 60–65 °C and acts automatically, without a person. **B1**

(b) EEPROM can be erased and rewritten with a voltage while it stays in the controller, so the firmware can be updated without removing the chip. **B1** It can be rewritten byte by byte, not only by erasing the whole chip with UV light. **B1**

(c) The controller sends the report into the buffer at its own high speed. **B1** The printer takes data from the buffer at its slower speed, so the processor is free to do other tasks meanwhile. **B1** When the buffer is empty, the printer sends an interrupt to request more data. **B1**

(d) The drum is charged, and a laser changes the charge where the report is to be printed, leaving an electrostatic image of the page. **B1** Charged toner sticks only to the image on the drum. **B1** The paper is charged so that it pulls the toner off the drum. **B1** The fuser heats and presses the paper so the toner melts and sticks. **B1**

---

## Common mistakes

- **Confusing XOR with NOR**, in symbols and truth tables; reports mention it in several years. XOR is 1 when the inputs **differ**; NOR is 1 only for **0, 0**.
- **Merging NOTs into the wrong gate**. (NOT A) AND (NOT B) is **not** A NAND B, and (NOT R) OR (NOT M) is **not** R NOR M. Draw the NOT gates exactly where the expression puts them.
- **Missing or misplaced NOT gates**: leaving off the final NOT, putting a NOT on each input instead of on the output, or drawing a NOT without its bubble.
- **Gates with more than two inputs**, or untidy shapes: if the examiner can't tell your AND from your OR, the mark is lost.
- **Simplifying when it wasn't asked**. "Write the expression for this circuit" and "sum of products" want the unsimplified form.
- **Expanding the acronym as an answer**. "ROM is read only" or "EEPROM is erased electrically" repeats the question. Say what that means for the device.
- **Vague comparisons**. "DRAM is faster" (it is the other way round) or "magnetic is cheaper" earn nothing. Say what each type does, and "cost per unit of storage".
- **Mixing up magnetic and optical disks** (lasers, pits and lands are optical), or primary memory with secondary storage.
- **Generic embedded-system answers**. "It is embedded in the car" or "it is a control system" scores nothing. Apply each feature to the device.
- **Feedback as user opinions**. In a control system, feedback means the output affects the next input.
- **Describing a buffer without saying why**, which is the speed difference that would otherwise leave the processor waiting.

---

## How to get the marks

- **One mark per point**. Give at least as many separate points as marks, each on its own line, and apply them to the scenario when there is one.
- **"Describe the principal operation"**: the steps in order, with the technical names of the parts. A 4-mark answer needs four distinct steps.
- **"Explain"**: a point and **why** or **how**, for example "a buffer ... so the processor is not held up waiting for the slower printer".
- **"Identify" / "State" / "Give"**: a short answer is enough: a name, a sensor type, a gate.
- **"Justify"** (monitoring or control, a sensor, a port, a storage choice): the choice usually earns nothing by itself; the marks are for reasons linked to the scenario.

- **Differences** must be two-sided: say what each thing does. Don't just describe one of them.
- **Logic questions:** truth-table marks come in blocks of rows, circuit marks in groups of connected gates, and expressions must use gate names and brackets, unsimplified.

---

## Quick check

1. (a) Which gate outputs 1 only when both inputs are 0? (b) Give the inputs that make a NAND gate output 0.  
(c) What is the output of an XOR gate with inputs 1 and 0?
2. Complete the output column of the truth table for  $X = (A \text{ OR } C) \text{ AND } (B \text{ NAND } C)$ , inputs in binary order 000 to 111.
3. A truth table with inputs A, B, C has output 1 only in rows 000 and 101. Write its logic expression as a sum of products.
4. A microwave oven is an embedded system with RAM and EEPROM. State one item each would hold, and one benefit of EEPROM over PROM for this oven.
5. A weather station records temperature and rainfall every 10 minutes and shows them on a website. Is it a monitoring or control system? Give two reasons.

## Answers

1. (a) NOR. (b)  $A = 1$  and  $B = 1$  (every other pair gives 1). (c) 1, because the inputs are different.
2. Working:  $A \text{ OR } C = 0, 1, 0, 1, 1, 1, 1, 1$ ;  $B \text{ NAND } C = 1, 1, 1, 0, 1, 1, 1, 0$ . AND them together:  $X = 0, 1, 0, 0, 1, 1, 1, 0$ .
3. Row 000  $\rightarrow$  NOT A AND NOT B AND NOT C; row 101  $\rightarrow$  A AND NOT B AND C. So  $X = (\text{NOT A AND NOT B AND NOT C}) \text{ OR } (A \text{ AND NOT B AND C})$ .
4. RAM: the cooking time and power level just entered, or the time left. EEPROM: the firmware that runs the cooking programs. Benefit: EEPROM can be rewritten in place, so the maker can update the firmware; PROM can be written only once.
5. Monitoring. It only records and displays readings and has no actuator. Nothing it outputs changes the temperature or rainfall it measures next, so there is no feedback.

## Past questions to try

- **9618/13/M/J/25 Q4**: an expression from each of two circuits, then a truth table with a chain of XORs.
- **9618/12/O/N/22 Q3**: a circuit from a problem statement (greenhouse window), then why ROM is used in an embedded system.
- **9618/13/O/N/24 Q2**: SRAM against DRAM, then how a buffer is used with a hard disk.
- **9618/11/O/N/24 Q5**: sensors for a security system, expressions from a parameter table, and monitoring or control.