

Hardware

IGCSE Computer Science 0478 · Topic 3 revision notes

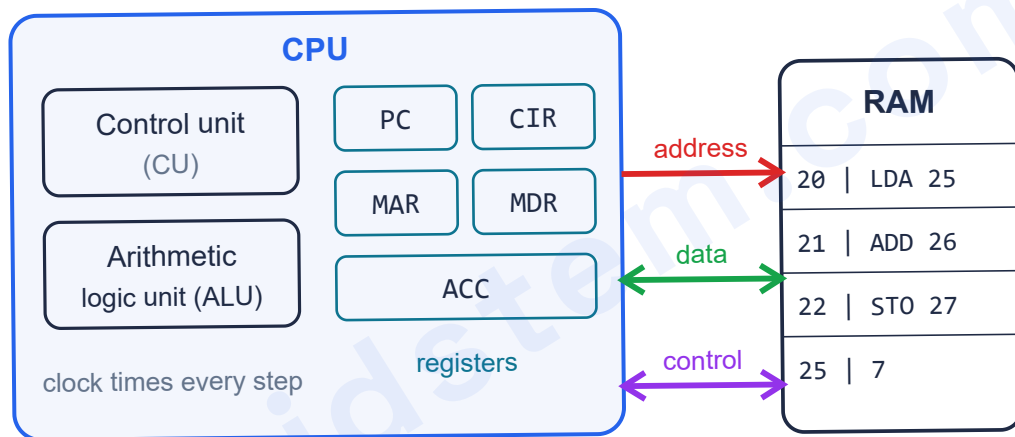
Everything from the 10 lessons of this topic on a few pages: the key idea of each lesson, the diagram to remember, the points to learn and the traps examiners set. Read it, cover it, test yourself.

- | | |
|---|--|
| 1. <u>The CPU and Von Neumann architecture</u> | 6. <u>Sensors and how they are used</u> |
| 2. <u>The fetch-decode-execute cycle</u> | 7. <u>Primary storage: RAM and ROM</u> |
| 3. <u>Cores, cache and clock speed</u> | 8. <u>Magnetic, optical and solid-state storage</u> |
| 4. <u>Instruction sets and embedded systems</u> | 9. <u>Virtual memory and cloud storage</u> |
| 5. <u>Input and output devices</u> | 10. <u>NICs, MAC addresses, IP addresses and routers</u> |

1 The CPU and Von Neumann architecture

Syllabus 3.1

The **central processing unit (CPU)** processes the instructions and data that are input into the computer so that the result can be output. A **microprocessor** is a type of integrated circuit with the CPU on a **single chip**. In a **Von Neumann architecture** the instructions and data are both stored in the same memory (RAM) and travel along the same **buses**. The CPU contains the **control unit (CU)**, the **arithmetic logic unit (ALU)** and **registers**: PC, MAR, MDR, CIR and ACC.



Von Neumann: instructions and data share one memory and the same buses

Fig. 1: the CPU's units and registers, linked to RAM by the address, data and control buses

Worked example: Name the register

Which register holds the address of the next instruction, and which holds the instruction being executed?

1. "Address of the next instruction" describes a pointer to where the next instruction is: that is the program counter.
2. "Instruction being executed" is the current instruction: that is the current instruction register.

Answer: PC and CIR

Where and what.

- ✓ **Addresses:** PC (next instruction) and MAR (location being accessed now).
- ✓ **Contents:** MDR (data or instruction moving to or from memory), CIR (current instruction), ACC (result).
- ✓ **Units:** CU decodes and controls; ALU does arithmetic and logic.
- ✓ **Buses:** address (one way), data and control (two way).

Watch out!

"The CPU stores all the files" is wrong. The CPU processes; files are kept in storage. Registers inside the CPU hold only a few values at a time.

Do not mix up the MAR and the MDR. The MAR holds an address (where); the MDR holds data or an instruction (what).

The data bus carries instructions as well as data, because in a Von Neumann computer both are stored in the same memory.

2 The fetch-decode-execute cycle

Syllabus 3.1

The CPU runs a program by repeating the **fetch-decode-execute (FDE) cycle**. **Fetch:** the address in the **PC** is copied to the **MAR** and sent along the **address bus**; the instruction at that address in RAM comes back along the **data bus** to the **MDR** and is copied to the **CIR**; the PC is incremented. **Decode:** the **CU** decodes the instruction in the CIR. **Execute:** the CU sends signals to carry it out, and the **ALU** does any calculation, with the result stored in the **ACC**.

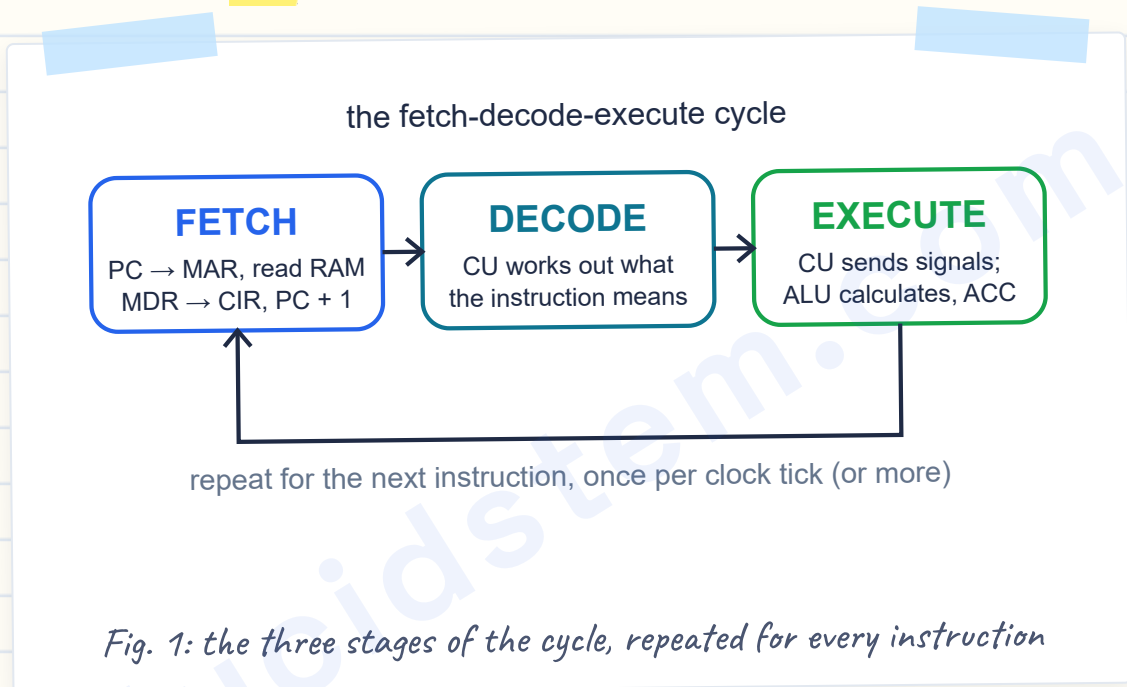


Fig. 1: the three stages of the cycle, repeated for every instruction

Worked example: Trace one instruction

The PC holds 20. Address 20 holds the instruction ADD 26, and address 26 holds the value 5. The ACC holds 7. Describe what happens.

1. Fetch: 20 is copied from the PC to the MAR and sent along the address bus.
2. ADD 26 comes back along the data bus to the MDR, then is copied to the CIR. The PC becomes 21.
3. Decode: the CU decodes ADD 26: add the value at address 26 to the ACC.
4. Execute: 26 goes to the MAR; the value 5 comes back into the MDR; the ALU adds $7 + 5$.

Answer: The ACC holds 12 and the PC holds 21

Follow the address.

- ✓ **Fetch:** $PC \rightarrow MAR \rightarrow \text{address bus}$; $\text{data bus} \rightarrow MDR \rightarrow CIR$; PC incremented.
- ✓ **Decode:** the CU decodes the instruction in the CIR.
- ✓ **Execute:** the CU sends signals; the ALU calculates; the result goes in the ACC.
- ✓ **Marks:** one per correct step, in order, naming the register or bus.

Watch out!

The PC is incremented during the fetch stage, ready for the next cycle. It is not incremented after the instruction is executed.

Name the registers in full or by their correct initials. "It goes to the register" does not earn the mark: say "copied from the MDR to the CIR".

thelucidstem.com

3 Cores, cache and clock speed

Syllabus 3.1

A **core** is a complete processing unit that can fetch, decode and execute instructions; more cores mean more instructions can be processed **at the same time**. **Cache** is a small amount of very fast memory inside or next to the CPU that stores frequently used instructions and data; a bigger cache means fewer slow trips to RAM. The **clock** sends regular pulses that time each FDE cycle; a higher **clock speed** (measured in hertz, for example 3.5 GHz) means more cycles per second.

three things that change how fast a CPU works

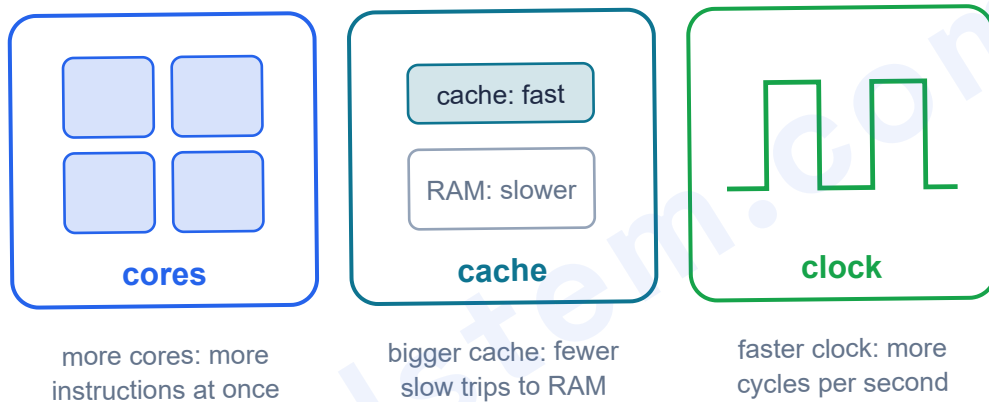


Fig. 1: cores, cache and clock speed each change how many instructions a CPU can process per second

Worked example: Compare two CPUs

CPU A: 4 cores, 3.0 GHz, 8 MiB cache. CPU B: 2 cores, 3.0 GHz, 4 MiB cache. Which is likely to perform better, and why?

1. The clock speeds are the same, so compare the other two factors.
2. A has more cores, so it can process more instructions at the same time.
3. A has a bigger cache, so it needs fewer slower fetches from RAM.

Answer: CPU A, because of its extra cores and larger cache

Factor, then effect.

- ✓ **Clock:** faster clock, more FDE cycles per second.
- ✓ **Cache:** bigger cache, more frequently used data near the CPU, fewer slow trips to RAM.
- ✓ **Cores:** more cores, more instructions processed at the same time.
- ✓ **Limits:** heat at high clock speeds; not all tasks split across cores; cache is expensive.

Watch out!

Clock speed is not a measure of memory. 3.5 GHz describes how often the CPU's clock ticks, not how much it can store.

Always say how the factor improves performance: "more cores, so more instructions can be processed at the same time", not just "more cores make it faster".

4 Instruction sets and embedded systems

Syllabus 3.1

An **instruction set** is the list of all the commands that a CPU can process. The commands are **machine code** (binary), each usually made of an **opcode** (the operation) and an **operand** (the data or address to use). An **embedded system** is a microprocessor-based system built into a larger device to perform a **dedicated function**, for example in a washing machine, car, security system, lighting system or vending machine. A **general purpose computer**, such as a PC or laptop, performs many different functions.

one machine code instruction (8 bits in this simple CPU)

0 0 1 1

opcode: what to do

1 0 0 1

operand: data or address

embedded system

one dedicated function
washing machine, car ABS,
vending machine, alarm

general purpose

many different functions
PC or laptop: install
and run any software

Fig. 1: a machine code instruction, and an embedded system compared with a general purpose computer

Worked example: Embedded or not?

Explain why the computer in a vending machine is an embedded system.

1. It is built into a larger device (the vending machine).
2. It performs one dedicated function: taking payment, reading the selection and releasing the item.
3. It runs a fixed program; the user cannot use it for anything else.

Answer: It is a dedicated system built into the machine

Dedicated or general.

- ✓ **Instruction set:** all the machine code commands a CPU can process.
- ✓ **Instruction:** opcode (what to do) and operand (data or address).
- ✓ **Embedded:** dedicated function, built into a device, small, cheap, low power, fixed program.
- ✓ **General purpose:** PC or laptop, many functions, user installs software.

Watch out!

Different types of CPU can have different instruction sets, so machine code written for one type of processor may not run on another.

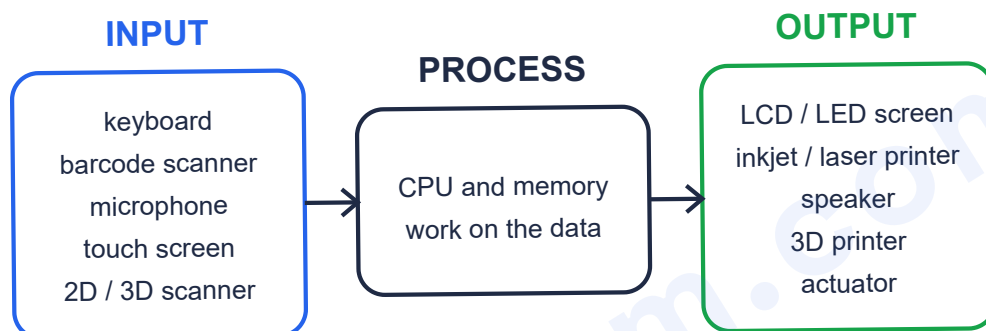
A laptop is not an embedded system even though it contains a microprocessor. Being general purpose, able to run many different programs, is what makes it different.

thelucidstem.com

5 Input and output devices

Syllabus 3.2

An **input device** allows data to be entered into a computer so that it can be processed; it is needed because the computer cannot process data it does not have. An **output device** takes the results of processing and presents them in a form people (or other devices) can use: on a screen, on paper, as sound or as movement. Input devices **capture** data; output devices **display, produce or act on** it.



input: data into the computer • output: results out of it

Fig. 1: input devices capture data, the CPU processes it and output devices present the results

Worked example: Choose the device

A school office prints 500 letters every morning. Which printer is more suitable, and why?

1. The task is high volume and needs speed.
2. A laser printer prints pages quickly and its toner cartridge lasts for many pages.
3. An inkjet is slower and its cartridges run out faster.

Answer: A laser printer

Into or out of.

- ✓ **Input:** enters data for processing (keyboard, mouse, microphone, camera, scanners, touch screen).
- ✓ **Output:** presents or acts on results (screens, projectors, printers, speaker, actuator).
- ✓ **Touch screens:** resistive (pressure, two layers), capacitive (charge, finger), infra-red (beams).
- ✓ **Choosing:** link the device's feature to the scenario's need.

Watch out!

Say why a device is an input device: it sends data *into* the computer for processing. A touch screen is both an input and an output device.

An actuator is an *output* device: it receives a signal from the processor and makes something move.

thelucidstem.com

6 Sensors and how they are used

Syllabus 3.2

A **sensor** is an input device that **measures a physical quantity** (such as temperature, light or pressure) in its surroundings and sends the reading to a processor. Most sensors produce an **analogue** signal, which an **analogue-to-digital converter (ADC)** turns into digital data. Sensors are used in **monitoring** systems (record and warn) and **control** systems (the processor compares readings with stored values and makes an **actuator** act). Sensors only measure; they never make decisions or act on their own.

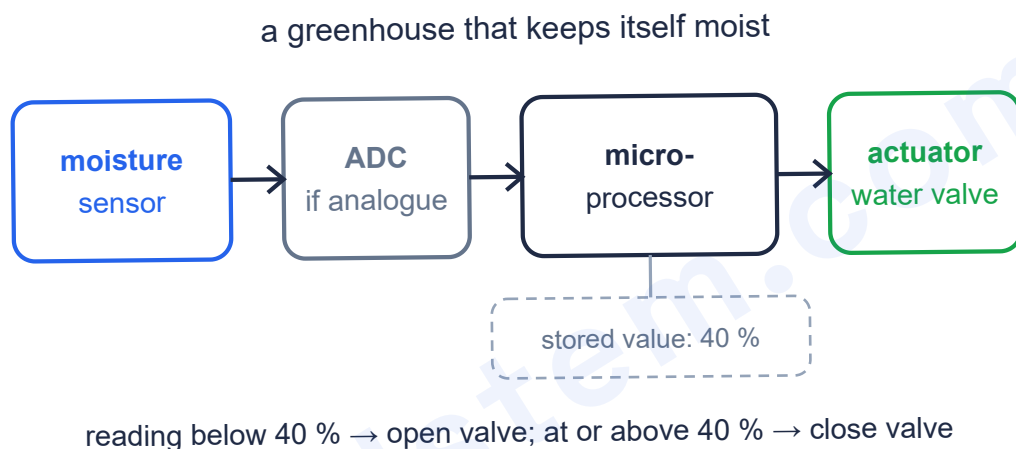


Fig. 1: a moisture sensor feeds a microprocessor, which decides whether an actuator should open the water valve

Worked example: Automatic greenhouse

A greenhouse keeps the soil damp, the air warm and opens a window if the air becomes too humid. Identify three suitable sensors.

1. Keeping the soil damp: the soil's water content must be measured: moisture sensor.
2. Keeping the air warm: temperature sensor.
3. Air too humid: water vapour in the air: humidity sensor.
4. The microprocessor compares each reading with its stored value and controls the sprinkler, heater and window motor (actuators).

Answer: Moisture, temperature and humidity sensors

Measure, do not decide.

- ✓ **Sensor:** input device that measures a physical quantity.
- ✓ **ADC:** converts analogue readings to digital for the processor.
- ✓ **Processor:** compares readings with stored values and decides; the actuator acts.
- ✓ **Names:** use the syllabus names, e.g. infra-red, moisture, humidity, proximity.

Watch out!

Never write "the sensor turns on the heater" or "the sensor decides". The sensor only sends readings; the microprocessor compares and decides; the actuator acts.

Moisture and humidity are different. Moisture is water in the soil (or a material); humidity is water vapour in the air.

Answer with sensor names from the syllabus list. "Motion sensor" or "heat sensor" may not earn the mark: say infra-red or temperature.

7 Primary storage: RAM and ROM

Syllabus 3.3

Primary storage is storage that is **directly accessed by the CPU**. **RAM** (random access memory) holds the programs and data **currently in use**; it can be read and written, and it is **volatile**, so its contents are lost when the power is off. **ROM** (read only memory) holds instructions that must never be lost, such as the **start-up (bootstrap) instructions**; it can only be read, and it is **non-volatile**. A computer needs both: ROM to start up, RAM to run programs.

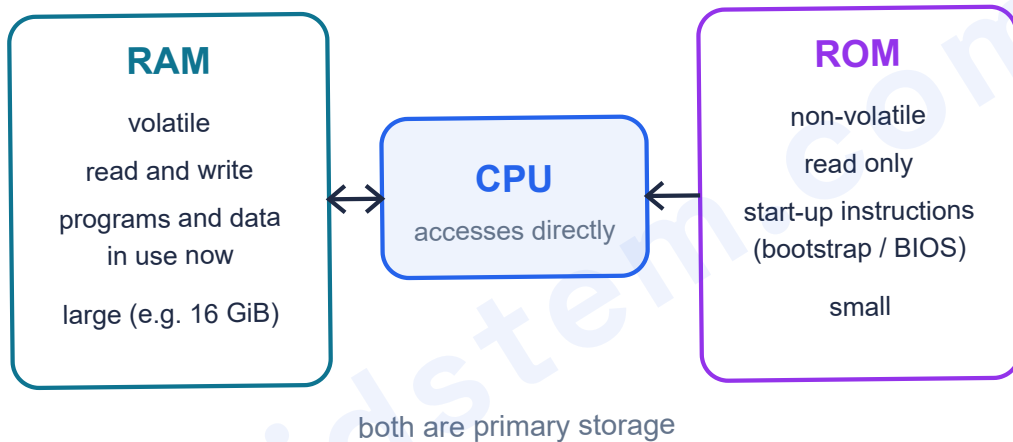


Fig. 1: RAM and ROM are both primary storage, accessed directly by the CPU

Worked example: Where is it stored?

State whether each is stored in RAM or ROM: (a) a document being edited; (b) the boot-up instructions; (c) the web browser you are using.

1. (a) Being edited means it is in use and changing: RAM.
2. (b) Needed with no power and must never change: ROM.
3. (c) A program currently running: RAM.

Answer: RAM, ROM, RAM

Compare in pairs.

- ✓ **Primary:** directly accessed by the CPU.
- ✓ **RAM:** volatile, read and write, programs and data in use.
- ✓ **ROM:** non-volatile, read only, start-up instructions.
- ✓ **Differences:** always give both sides of each difference.

Watch out!

Cache and registers are also inside or close to the CPU, but in this topic the two named types of primary storage are RAM and ROM.

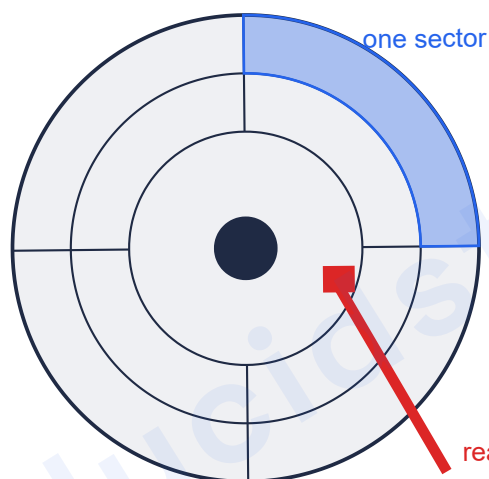
"RAM is temporary" on its own is vague. Say *volatile*: its contents are lost when the power is turned off.

ROM does not store the operating system in a PC. ROM holds the start-up instructions, which load the operating system into RAM.

8 Magnetic, optical and solid-state storage

Syllabus 3.3

Secondary storage is **not directly accessed by the CPU** and is **non-volatile**, so it is needed for **more permanent storage** of data and programs. **Magnetic** storage (HDD) uses spinning **platters** divided into **tracks and sectors**; data is read and written by **electromagnets** on a moving head. **Optical** storage (CD, DVD, Blu-ray) uses a **laser** to create and read **pits and lands**. **Solid-state** (flash) storage (SSD, SD card, USB drive) uses **NAND or NOR** technology, with transistors acting as **control gates and floating gates**, and has no moving parts.



platter: tracks (rings) and sectors

magnetic (HDD)

electromagnet magnetises spots

optical (CD, DVD, Blu-ray)

laser reads pits and lands

solid-state (SSD, SD, USB)

NAND/NOR flash: electrons held in floating gates, no moving parts

Fig. 1: a hard disk platter divided into tracks and sectors, and the three types of secondary storage

Worked example: Choose the storage

A student wants storage for a laptop that is carried every day. Recommend a type and justify it.

1. The laptop is moved often, so it may be knocked or dropped.
2. Solid-state storage has no moving parts, so it is more robust.
3. It also uses less power (longer battery life) and gives faster access.

Answer: Solid-state (an SSD)

How, then examples.

- ✓ **Magnetic:** platters, tracks and sectors, electromagnet on a read/write head. HDD.
- ✓ **Optical:** laser creates and reads pits and lands. CD, DVD, Blu-ray.
- ✓ **Solid-state:** NAND/NOR flash, control and floating gates, no moving parts. SSD, SD card, USB drive.
- ✓ **Choosing:** link a feature (robust, capacity, cost, speed, portable) to the scenario.

Watch out!

Data on a hard disk is stored magnetically, not with a laser.

Lasers belong to optical storage.

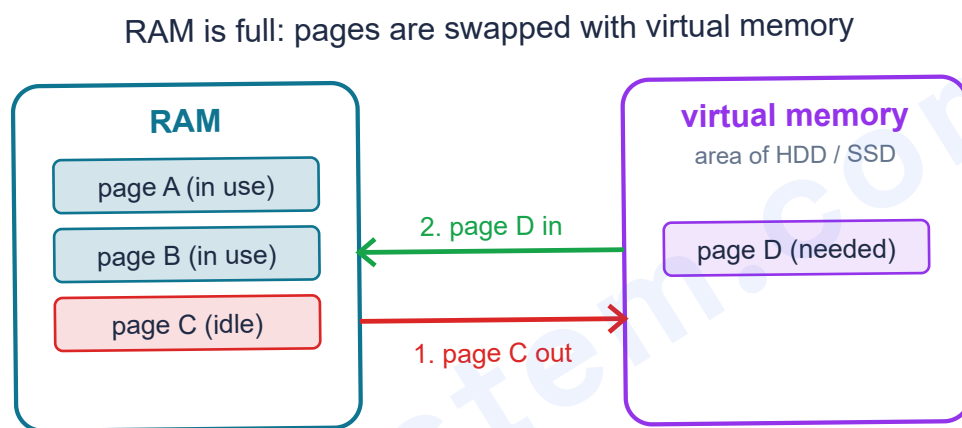
Solid-state does not use magnetism or lasers. Do not write "the SSD spins"; it has no moving parts.

thelucidstem.com

9 Virtual memory and cloud storage

Syllabus 3.3

Virtual memory is an area of **secondary storage** (HDD or SSD) used as if it were extra RAM when **RAM is full**. Data is split into **pages**; pages not currently needed are moved from RAM to virtual memory, and moved back into RAM **when needed**. It lets a computer run programs that need more memory than its RAM holds. **Cloud storage** keeps data on **remote physical servers**, owned by a provider, that are **accessed over the internet**, in comparison to storing data **locally** on the user's own devices.



too much swapping slows the computer down (disk thrashing)

Fig. 1: when RAM is full, an idle page is moved to virtual memory so a needed page can be brought into RAM

Worked example: Explain the slowdown

A computer with 4 GiB of RAM becomes very slow when many programs are open. Explain why, and suggest an improvement.

1. The programs need more than 4 GiB, so RAM is full.
2. Pages are transferred between RAM and virtual memory on secondary storage.
3. Secondary storage is much slower than RAM, so each transfer wastes time.
4. More RAM would let more pages stay in RAM, so fewer transfers are needed.

Answer: Too much use of virtual memory; install more RAM

Pages and servers.

- ✓ **Virtual memory:** area of secondary storage used when RAM is full.
- ✓ **Pages:** moved out of RAM when idle, back into RAM when needed.
- ✓ **Cloud:** remote physical servers owned by a provider, accessed over the internet.
- ✓ **Compare:** say "compared with local storage" and give both sides.

Watch out!

The CPU never processes data directly from virtual memory. A page must be moved back into RAM before it can be used.

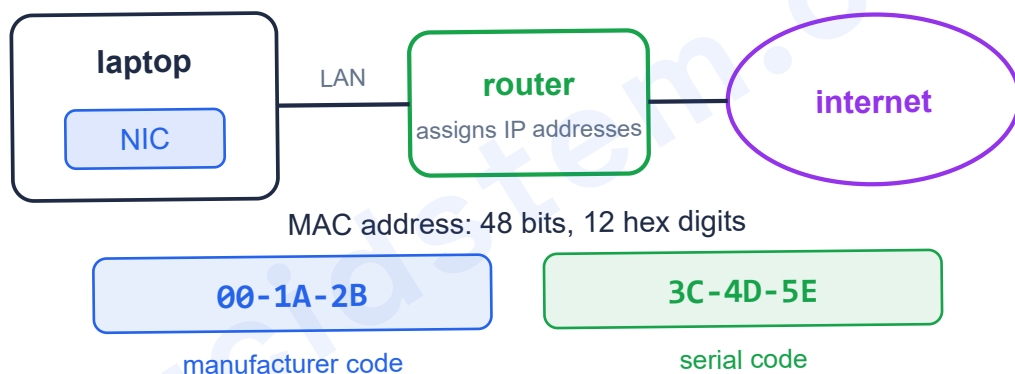
Do not say cloud storage "does not use physical storage". Cloud data is still stored on physical servers; the user simply does not own or look after them.

thelucidstem.com

10 NICs, MAC addresses, IP addresses and routers

Syllabus 3.4

A computer needs a **network interface card (NIC)** to access a network. Each NIC is given a unique **MAC address** at the point of manufacture: 48 bits, usually written as 12 **hexadecimal** digits, made of the **manufacturer code** and the **serial code**. An **IP address** is **allocated by the network** to identify a device on it; it can be **static** (stays the same) or **dynamic** (can change each time the device connects), and is either **IPv4** (32 bits) or **IPv6** (128 bits). A **router** sends data to a specific destination on a network, can assign IP addresses, and can connect a local network to the internet.



MAC: fixed in the NIC when made • IP: given by the network, can change

Fig. 1: a laptop's NIC connects to a router; its MAC address is made of a manufacturer code and a serial code

Worked example: Valid or not?

Is 192.168.300.4 a valid IPv4 address?

1. IPv4 has 4 numbers, each stored in 8 bits.
2. 8 bits can only hold 0 to 255.
3. 300 is greater than 255.

Answer: Not valid

Fixed or allocated.

- ✓ **NIC:** needed to access a network; given a MAC address when manufactured.
- ✓ **MAC:** 48 bits, 12 hex digits, manufacturer code + serial code.
- ✓ **IP:** allocated by the network; static or dynamic; IPv4 32 bits, IPv6 128 bits.
- ✓ **Router:** sends data to a specific destination, assigns IP addresses, connects a LAN to the internet.

Watch out!

The MAC address belongs to the NIC and is set when it is made. It is not given by the network or the router; that is the IP address.

A router does more than "connect to the internet". For full marks, give its specific roles: forwarding data to a destination using IP addresses and assigning IP addresses.

thelucidstem.com