

Topic 3: Hardware

One gate to build them all, how memory cells really work, and why SSDs wear out.

Optional stretch material. This goes beyond the 9618 syllabus, so you will not be asked about it directly in the exam. Its purpose is to deepen ideas you already know. **Builds on:** 3.1 Computers and their components, 3.2 Logic gates and logic circuits. Try the challenge questions on the last pages before you read the answers.

NAND: the universal gate

Every logic circuit can be built from **NAND gates alone** (NOR has the same property). Chip designers like this because one repeated transistor pattern is cheaper to manufacture and optimise.

Function	Built from NAND	Gates
NOT A	NAND(A, A)	1
A AND B	NAND(NAND(A,B), NAND(A,B))	2
A OR B	NAND(NAND(A,A), NAND(B,B))	3

Why do they work? NAND(A, A) is true only when A is false, so it is a NOT. AND is just NAND followed by NOT. OR follows because $A \text{ OR } B = \text{NOT}(\text{NOT } A \text{ AND NOT } B)$.

SRAM and DRAM: two ways to remember a bit

	SRAM (cache)	DRAM (main memory)
Cell	About 6 transistors forming a latch	1 transistor + 1 capacitor
Holds a bit by	Feedback loop; stable while powered	Charge on a capacitor, which leaks
Refresh needed?	No	Yes, every few tens of ms; reading also drains the charge so it must be written back
Speed / size / cost	Fastest, large cell, expensive	Slower, tiny cell, cheap per GB

This is why a CPU has a few MB of SRAM cache but gigabytes of DRAM: the small DRAM cell lets billions fit on a chip.

Inside an SSD

Flash memory stores bits as trapped charge. Its awkward rule: data is **written a page at a time** (a few KB) but can only be **erased a whole block at a time** (hundreds of pages), and each block survives only a limited number of erase cycles (from a few thousand for consumer drives up to ~100 000 for the best cells).

So an SSD never overwrites in place. An edited page is written to a fresh page and the old copy is marked stale. Later, **garbage collection** copies the still-valid pages out of a mostly-stale block and erases it. These extra internal writes are called **write amplification**. **Wear levelling** spreads erases evenly so no block dies early, and **TRIM** lets the operating system tell the drive which blocks are free.

Challenge questions

1. Build XOR ($A \text{ XOR } B$) from NAND gates only. How many gates do you need?
2. Explain why a CPU uses SRAM for its cache but DRAM for main memory.
3. A file occupying one SSD page is edited. Describe what the drive does and why this causes write amplification.

Worked answers

Only look once you have had a proper go at each question.

- 1. 4 gates:** $n1 = \text{NAND}(A, B)$; $n2 = \text{NAND}(A, n1)$; $n3 = \text{NAND}(B, n1)$; $\text{output} = \text{NAND}(n2, n3)$.
Check $A=1, B=0$: $n1 = 1$; $n2 = \text{NAND}(1, 1) = 0$; $n3 = \text{NAND}(0, 1) = 1$; $\text{output} = \text{NAND}(0, 1) = 1$. Correct.
- 2.** Cache must keep up with the processor, and SRAM needs no refresh and can be read immediately, so it is far faster. But an SRAM cell is about six transistors, so gigabytes of it would be enormous and very costly. DRAM's one-transistor cell is cheap and dense, and its slower, refresh-limited speed is acceptable when the cache hides most accesses.
- 3.** The drive cannot overwrite the page, so it writes the new version to a free page and marks the old one invalid. Eventually a block is mostly invalid pages; garbage collection copies its few valid pages to another block and erases it. Those copies are writes the user never asked for, so the drive writes more data internally than the OS sent, using up erase cycles.