Exercise walk-through

Central Processing Unit (CPU) Architecture ▪ 4.1

eXercise

You must be able to

- explain the **Von Neumann 冯 · 诺依曼** model and the **stored-program 存储程序** concept
- state the role of the **registers**, the **ALU 算术逻辑单元**, the **control unit 控制单元** and the clock
- explain the **address**, **data** and **control buses**
- describe the stages of the **fetch–execute cycle** and the purpose of **interrupts 中断**

Method — Von Neumann model and registers

In the **Von Neumann** model one memory holds **both program and data** (the **stored-program** idea), and the CPU runs the instructions one at a time, in order, unless a branch changes the flow. **Registers** 寄存器 are tiny, very fast stores inside the CPU. Each special-purpose register has a fixed job:

Register	Holds
PC (program counter)	address of the **next** instruction
MAR (memory address register)	the address being read/written
MDR (memory data register)	the data going to/from memory
CIR (current instruction register)	the instruction being decoded
ACC (accumulator)	the value the ALU is working on
IX (index register)	a value added to a base address (indexed addressing)

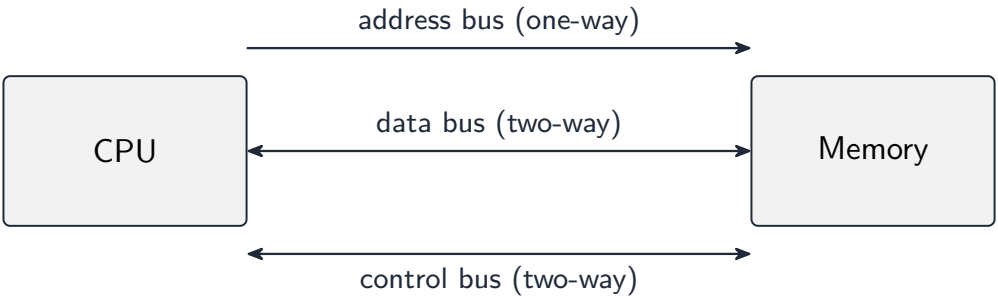
Method — Von Neumann model and registers

The **ALU** does arithmetic and logic; the **control unit** decodes instructions and sends control signals; the **clock** keeps everything in step.

Method — The three buses

The **address bus** is **one-way** (the CPU only ever *sends* an address to memory); the **data** and **control** buses are **two-way**. The **width** of a bus matters: a wider **data bus** carries more bits per transfer (faster), and a wider **address bus** can address **more** memory locations (each extra line doubles the number of addresses).

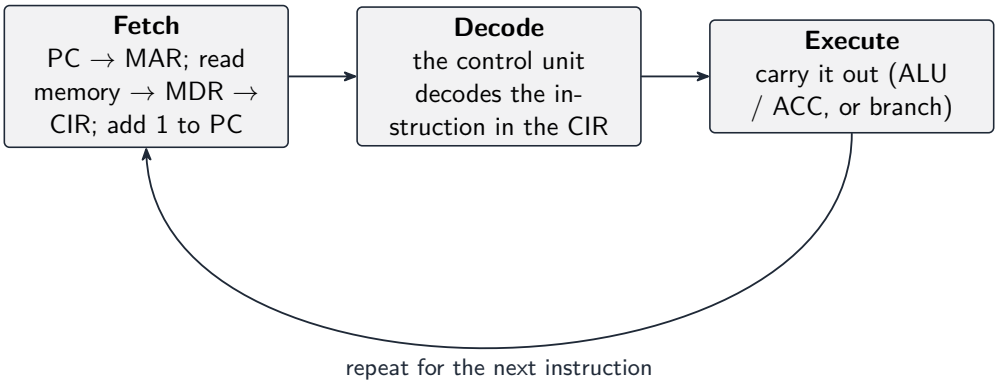
Method — The three buses



Method — The fetch–execute cycle

Every instruction goes through the same three stages, then the cycle repeats:
An **interrupt** is a signal that makes the CPU pause the current program, run a service routine, then return — so urgent events are handled quickly.

Method — The fetch–execute cycle



Practice 2.1 ▪ 1 mark

State what the **stored-program concept** means.

Solution 2.1

Method: Von Neumann model and registers

Worked steps

Both the **program (instructions) and the data** are held together in the **same memory** B1.

Practice 2.2 ▪ 2 marks

State the role of the **PC** and the **MAR**.

Solution 2.2

Worked steps

PC holds the address of the **next** instruction B1; MAR holds the address currently being read from or written to memory B1.

Practice 2.3 ▪ 2 marks

State the difference between a **general-purpose** and a **special-purpose** register.

Solution 2.3

Method: Von Neumann model and registers

Worked steps

A special-purpose register has a **fixed role** in the cycle (e.g. PC, MAR); a general-purpose register holds **temporary values** chosen by the program B1 B1.

Practice 2.4 ▪ 2 marks

State which bus is **one-way**, and explain why.

Solution 2.4

Worked steps

The **address bus** is one-way B1; the CPU only ever **sends** an address to memory — memory never sends an address back B1.

Exam-style 3.1 ▪ 4 marks

Describe the stages of the **fetch–execute cycle**.

Solution 3.1

Method: The fetch–execute cycle

Worked steps

Fetch: the address in the PC is copied to the MAR; the instruction is read from memory into the MDR, then the CIR; the PC is increased by 1 B1 B1. **Decode:** the control unit decodes the instruction in the CIR B1. **Execute:** the instruction is carried out (ALU/ACC, or a branch) B1.

Exam-style 3.2 ▪ 2 marks

(a) State the role of the **ALU** and the **control unit**.
(b) State **two** factors that affect the performance of a CPU.

Solution 3.2

Worked steps

(a) The ALU performs **arithmetic and logic** operations; the control unit **decodes instructions and sends control signals** B1 B1.

(b) Any two of: clock speed; number of cores; cache size; bus width B1 B1.

Exam-style 3.3 ▪ 2 marks

Explain the purpose of an **interrupt**.

Solution 3.3

Worked steps

An interrupt is a signal that makes the CPU **pause** the current program B1, run a routine to handle an urgent event, then **return** to where it left off B1.

Exam-style 3.4 ▪ 2 marks

During the fetch stage, describe the roles of the **MAR** and the **MDR**.

Solution 3.4

Worked steps

The MAR holds the **address** of the instruction to fetch B1; the MDR holds the **instruction/data** read back from that address B1.

Exam-style 3.5 ▪ 2 marks

A computer is redesigned with a **wider address bus**. Explain the effect this has on the computer.

Solution 3.5

Method: The three buses

Worked steps

A wider address bus has more lines, so it can carry **more distinct addresses** B1, letting the CPU directly address **more memory** (each extra line doubles the addressable memory) B1.

Exam-style 3.6 ▪ 4 marks

A processor has four cores, a clock speed of 3.2 GHz and a small amount of cache memory.
(a) **Complete** the table by describing the role of each register.

register	**role**
Program Counter (PC)	
Memory Address Register (MAR)	
Memory Data Register (MDR)	
Current Instruction Register (CIR)	

Exam-style 3.6 ▪ 4 marks

(b) **Explain** how increasing the **number of cores** can improve performance, and **give** one reason why four cores rarely make a program four times faster.
(c) **Explain** how increasing the amount of **cache** improves performance.
(d) **State** three differences between **Dynamic RAM** and **Static RAM**, and **identify** which is normally used for cache.
(e) A second processor has the same clock speed but a **wider** data bus. **Explain** the effect on performance.

Solution 3.6

Method: Von Neumann model and registers

Worked steps

(a) **PC** — holds the address of the **next** instruction to be fetched B1. **MAR** — holds the address currently being read from or written to B1. **MDR** — holds the data or instruction just fetched from, or about to be written to, that address B1. **CIR** — holds the instruction currently being decoded and executed B1.
(b) Each core can fetch and execute its own instructions, so several instructions run **at the same time** B1. But the program must be written to divide its work between cores, and parts of it are inherently sequential B1; the cores also compete for the same memory and bus, so the gain is less than the core count suggests B1.

Solution 3.6

(c) Cache holds recently and frequently used instructions and data close to the processor B1; it is far faster to access than main memory, so more cache means more of what the processor needs is found there and fewer slow main-memory fetches are needed B1.
(d) Any three: DRAM stores each bit in a **capacitor** that must be **refreshed**, SRAM uses flip-flops and does not B1; SRAM is **faster** B1; SRAM is more expensive per bit and takes more space, so DRAM gives more capacity for the money B1. **SRAM** is used for cache B1.
(e) A wider data bus carries **more bits per transfer** B1, so each fetch moves more data and fewer transfers are needed for the same work, raising throughput without any change in clock speed B1.