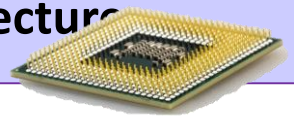
J277 - 1.1 Systems architecture

1.1 – Systems architecture

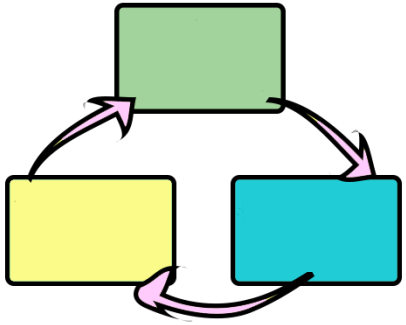
Sub topic	Guidance
1.1.1 Architecture of the CPU	
☐ The purpose of the CPU: ○ The fetch-execute cycle ☐ Common CPU components and their function: ○ ALU (Arithmetic Logic Unit) ○ CU (Control Unit) ○ Cache ○ Registers ☐ Von Neumann architecture: ○ MAR (Memory Address Register) ○ MDR (Memory Data Register) ○ Program Counter ○ Accumulator	**Required** ✓ What actions occur at each stage of the fetch-execute cycle ✓ The role/purpose of each component and what it manages, stores, or controls during the fetch-execute cycle ✓ The purpose of each register, what it stores (data or address) ✓ The difference between storing data and an address **Not required** ✗ Knowledge of passing of data between registers in each stage
1.1.2 CPU performance	
☐ How common characteristics of CPUs affect their performance: ○ Clock speed ○ Cache size ○ Number of cores	**Required** ✓ Understanding of each characteristic as listed ✓ The effects of changing any of the common characteristics on system performance, either individually or in combination
1.1.3 Embedded systems	
☐ The purpose and characteristics of embedded systems ☐ Examples of embedded systems	**Required** ✓ What embedded systems are ✓ Typical characteristics of embedded systems ✓ Familiarity with a range of different embedded systems

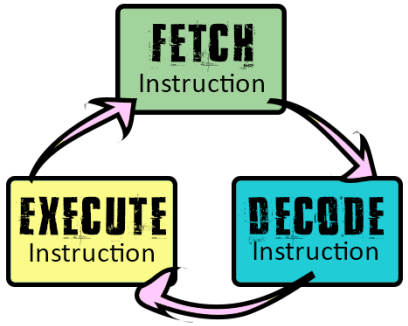
J277 - 1.1 Systems architecture



The Central Processing Unit (CPU)

This is where a computer processes all data and instructions.





The CPU has a small amount of memory called registers.

Registers:

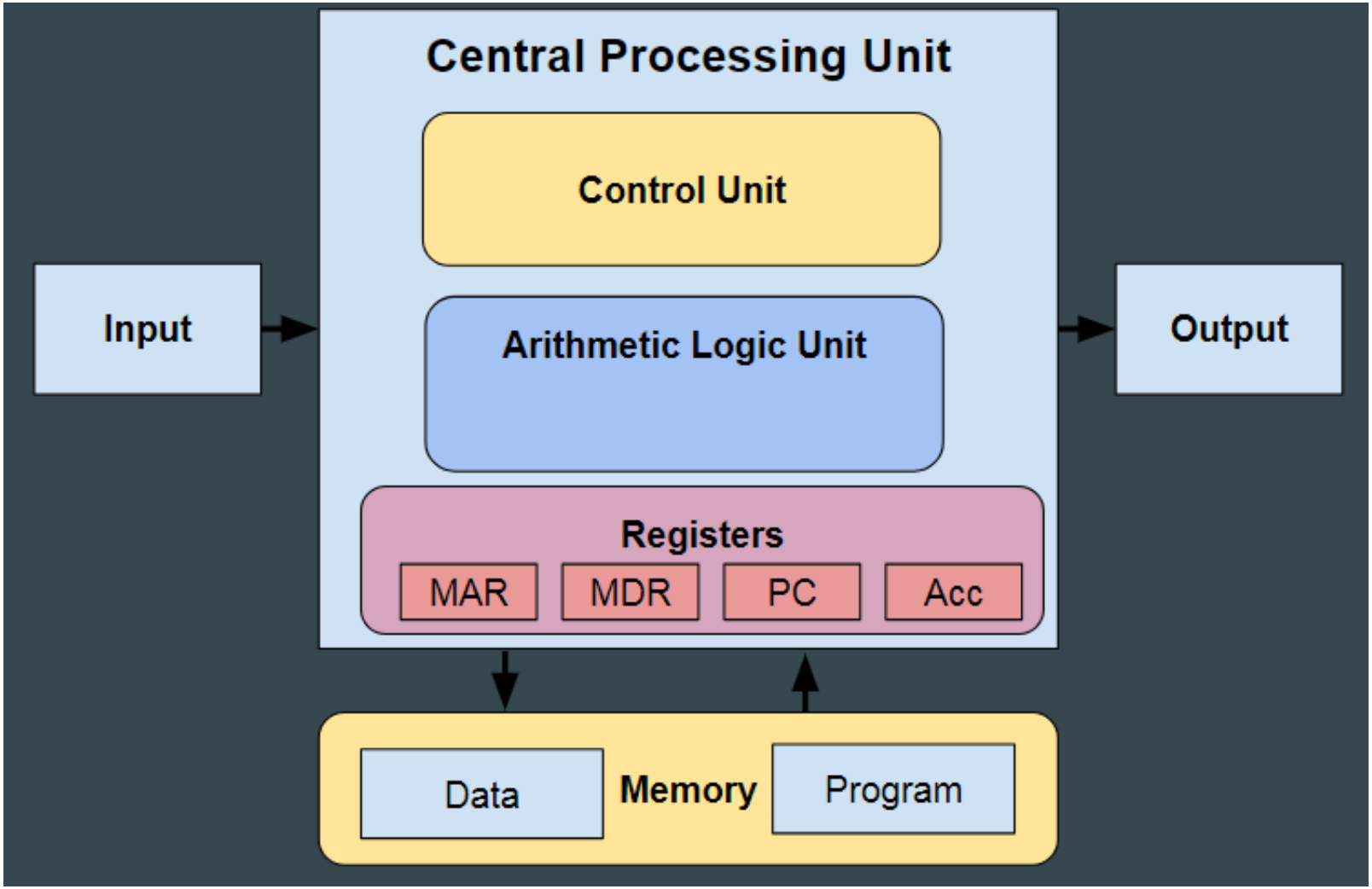
Memory Address Register (MAR)	Memory Data Register (MDR)	Program Counter (PC)	Accumulator (ACC)
Purpose: Stores the location in memory to be used by the MDR.	Holds the actual data or instruction fetched from, or to be written to the memory.	Holds the address of the next instruction.	Holds the results of calculations from the ALU.

The CPU also has these components:

Arithmetic logic unit (ALU)	Control Unit	Cache
Does calculations including addition, subtraction, multiplication and division. Also performs binary shifts and logical decisions.	Controls the flow of data in and out of the CPU. Manages fetching, decoding and executing of instructions.	This memory provides fast access to regularly used instructions and data without having to go to the main memory (RAM).

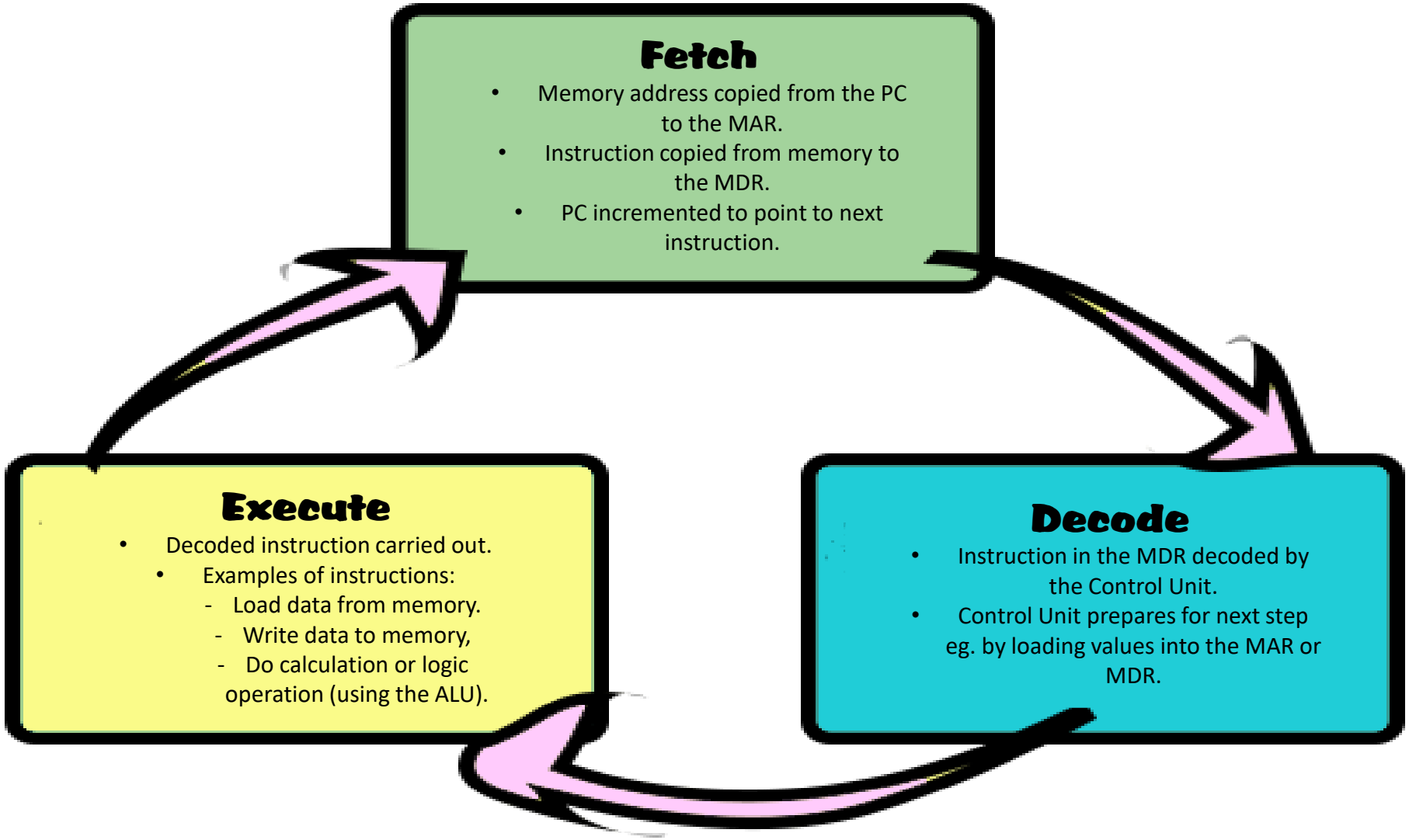
How the CPU works: the Von Neumann architecture

In the Von Neumann architecture, data and instructions are both stored in the same memory.



How the CPU works: Fetch-Execute Cycle

The Fetch-Execute cycle repeats continuously while the computer is running.



J277 - 1.1 Systems architecture

1.1 – Systems architecture

Sub topic	Guidance
1.1.1 Architecture of the CPU	
☐ The purpose of the CPU: ○ The fetch-execute cycle ☐ Common CPU components and their function: ○ ALU (Arithmetic Logic Unit) ○ CU (Control Unit) ○ Cache ○ Registers ☐ Von Neumann architecture: ○ MAR (Memory Address Register) ○ MDR (Memory Data Register) ○ Program Counter ○ Accumulator	**Required** ✓ What actions occur at each stage of the fetch-execute cycle ✓ The role/purpose of each component and what it manages, stores, or controls during the fetch-execute cycle ✓ The purpose of each register, what it stores (data or address) ✓ The difference between storing data and an address **Not required** ✗ Knowledge of passing of data between registers in each stage
1.1.2 CPU performance	
☐ How common characteristics of CPUs affect their performance: ○ Clock speed ○ Cache size ○ Number of cores	**Required** ✓ Understanding of each characteristic as listed ✓ The effects of changing any of the common characteristics on system performance, either individually or in combination
1.1.3 Embedded systems	
☐ The purpose and characteristics of embedded systems ☐ Examples of embedded systems	**Required** ✓ What embedded systems are ✓ Typical characteristics of embedded systems ✓ Familiarity with a range of different embedded systems

J277 - 1.1 Systems architecture

Factors affecting the speed of the CPU

Generally, CPUs with more cores, higher clock speeds and larger caches will have better performance, but will cost more.

Clock speed

The number of instructions a single processor core can carry out per second, measured in hertz.

Number of cores

Each core processes data independently, so more cores means more instructions carries out per second.

Cache size

Cache is a small amount of memory in the CPU used to store frequently used programs so that they can be fetched quicker (rather than from RAM).

J277 - 1.1 Systems architecture

1.1 – Systems architecture

Sub topic	Guidance
1.1.1 Architecture of the CPU	
☐ The purpose of the CPU: ○ The fetch-execute cycle ☐ Common CPU components and their function: ○ ALU (Arithmetic Logic Unit) ○ CU (Control Unit) ○ Cache ○ Registers ☐ Von Neumann architecture: ○ MAR (Memory Address Register) ○ MDR (Memory Data Register) ○ Program Counter ○ Accumulator	**Required** ✓ What actions occur at each stage of the fetch-execute cycle ✓ The role/purpose of each component and what it manages, stores, or controls during the fetch-execute cycle ✓ The purpose of each register, what it stores (data or address) ✓ The difference between storing data and an address **Not required** ✗ Knowledge of passing of data between registers in each stage
1.1.2 CPU performance	
☐ How common characteristics of CPUs affect their performance: ○ Clock speed ○ Cache size ○ Number of cores	**Required** ✓ Understanding of each characteristic as listed ✓ The effects of changing any of the common characteristics on system performance, either individually or in combination
1.1.3 Embedded systems	
☐ The purpose and characteristics of embedded systems ☐ Examples of embedded systems	**Required** ✓ What embedded systems are ✓ Typical characteristics of embedded systems ✓ Familiarity with a range of different embedded systems

J277 - 1.1 Systems architecture

Embedded systems

Embedded systems are:

Computer systems that are built into other devices, with a dedicated function within a larger control system.

Examples of embedded systems:

Washing machine
User can select a wash program.
A sequence of fill, turn, drain cycles wash the clothes.

Sat Nav
User can select a route.
A sequence of directions are outputs driving to a destination.

Digital watches
User can select the app, the process will be completed on the app and then the output will be displayed.

They're usually **easier** to design, **cheaper** to produce, and more **efficient** at their task than general purpose systems.