Unit 3—Hardware...

HCCS STEM—Computer Science, IT, Engineering, Food/Nutrition, Business Studies

What do I need to be able to do?

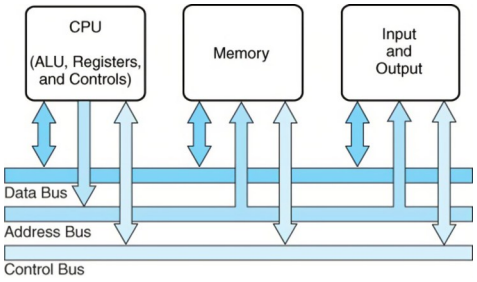
1.You must be able to identify the **main components** of a computer system, including the CPU and main memory, explain the Von Neumann architecture, and describe the function of CPU components like the Control Unit, ALU, Registers, and Clock, as well as the role of buses.

2.You must be able to explain the **fetch-decode-execute** cycle that the CPU performs and detail the **core functions of an operating system**, such as process management, memory management, peripheral management, and user management, alongside understanding various types of utility software.

3.You must also be able to compare different types of **secondary storage** based on their characteristics and uses, understand methods for developing **robust software** and identifying vulnerabilities like **validation** and **verification**, and distinguish between **high-level and low-level** programming languages and how they are translated by **compilers or interpreters**.

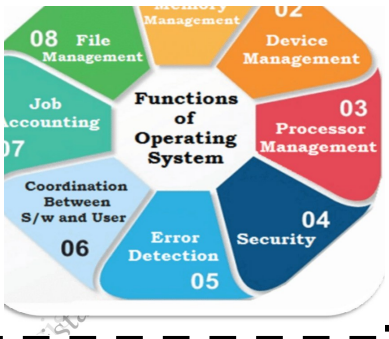
Keywords

1. **Central Processing Unit (CPU)**: This is the most important component of a computer and is responsible for processing instructions and running programs.
2. **Von Neumann Architecture**: This describes a computer design where program instructions and the data the programs are using are both stored in the same memory.
3. **FDE or FE Cycle**: This is the fundamental process the CPU repeats, involving fetching an instruction, decoding it, and then executing it.
4. **Operating System**: This is system software that manages computer hardware, users, and the resources used by software, providing essential functions like file and process management.
5. **Secondary Storage**: This refers to non-volatile storage that keeps data even when there is no power, used for storing programs and data.
6. **Validation**: This is a method for developing robust software by checking that data entered is reasonable and conforms to a set of rules.
7. **High-level Language**: This type of programming language uses statements that look a bit like English or Maths, making them easier to learn and understand for programmers.



In the **von Neumann architecture**, **program instructions and the data** programs use are stored together in the **same memory**. The **Central Processing Unit (CPU)** accesses both instructions and data from this shared memory. Key components within the CPU that facilitate this process include the **Control Unit (CU)**, the **Arithmetic Logic Unit (ALU)**, and **Registers**.

Secondary storage is where program instructions and the data used by programs are stored in a non-volatile way, meaning the data is kept even when there is no power. It is not directly accessible by the CPU, unlike main memory (RAM/ROM). Common types of secondary storage include magnetic hard drives/floppy disks, solid state drives (SSDs, USB sticks and SD cards), or optical storage like CDs, DVDs, and Blu-rays. These devices have various uses, such as storing large quantities of data, creating backups, distributing films, or transporting data.



A compiler translates the entire program written in a high-level language into machine code all at once. In contrast, an interpreter translates and executes the program line by line (like your use of Python). The object code produced by a compiler can be saved and run later (without the user seeing the code), while an interpreter translates and runs the code immediately without creating separate object code (but you need to leave the code visible to the user).

Developing robust software involves applying techniques like validation to ensure data is plausible and verification to double-check its correctness. Code quality practices, such as using meaningful variable names and subprograms, further enhance robustness by reducing errors. Audit trails, which are logs of system activities and problems, are used to identify the cause of issues, helping to improve software reliability.

10 Test your understanding questions

Here are 10 multiple-choice questions summarising the whole unit:

1. In the **von Neumann architecture**, where are **program instructions and the data** used by programs stored?

A) In separate memory locations. B) Together in the same memory. C) Only in the CPU's registers. D) Exclusively on secondary storage.

2. Which of the following are key components found within the **Central Processing Unit (CPU)**? A) Hard Drive, SSD, RAM. B) Monitor, Keyboard, Printer. C) Control Unit, Arithmetic Logic Unit (ALU), Registers. D) Compiler, Interpreter, Assembler.

3. What are the three main stages of the **fetch-decode-execute cycle**?

A) Input, Process, Output. B) Save, Load, Run. C) Fetch, Decode, Execute. D) Compile, Link, Run.

4. What is the function of a **bus** in computer architecture?

A) To perform arithmetic and logic operations. B) To store small amounts of data and results. C) To control the timing of the processor. D) A set of parallel wires connecting two or more components of the computer.

5. Which characteristic describes **secondary storage** data being kept when there is no power? A) Volatile. B) Non-volatile. C) Primary. D) Fast data rate.

6. Which of the following is an example of **optical storage**?

A) Hard Disk Drive. B) Solid State Drive. C) Blu-ray. D) USB memory stick.

7. Which of the following is a key function of an **operating system**?

A) Writing program code in a high-level language. B) Translating high-level code into machine code all at once. C) Managing computer hardware, users, and the resources used by software. D) Performing arithmetic and logic operations.

8. Which type of software performs **extra functionality** and housekeeping tasks that keep computers running efficiently?

A) Application software. B) Utility software. C) Operating system. D) High-level language.

9. What technique involves checking if **data entered is plausible**, for example, checking if a password is the correct length?

A) Verification. B) Debugging. C) Validation. D) Auditing.

10. What is a main difference in how a **compiler** and an **interpreter** translate a high-level language?

A) A compiler translates line by line, while an interpreter translates the whole program at once. B) A compiler translates the whole program to produce object code, while an interpreter translates and executes one line at a time. C) An interpreter requires the object code to be present to run, while a compiler does not. D) A compiler is used for low-level languages, while an interpreter is used for high-level languages.

10 Test your understanding answers

1. B, 2. C, 3. C, 4. D, 5. B, 6. C, 7. C, 8. B, 9. C, 10. B