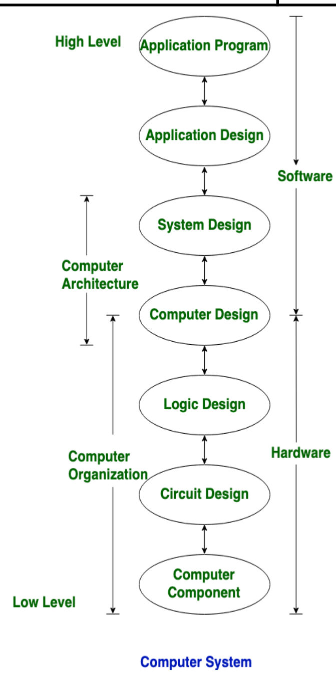
CSE-112 Computer Organization
Tutorial 1
Sujay Deb

---

1. What is the difference between computer organization and computer architecture?

Computer Organization	Computer Architecture
Computer Organization deals with a structural relationship.	Computer Architecture deals with the functional behavior of computer systems.
The actual implementation of a computer in hardware.	The view of a computer presented to software designers.
Computer Organization is what is referred to as microarchitecture.	Computer Architecture is also called Instruction Set Architecture (ISA).
Provides a deep understanding of how instructions are executed and how data is processed	Provides an understanding of how to create a computer system that meets specific performance goals
For designing a computer, an organization is decided after its architecture.	For designing a computer, its architecture is fixed first.



ImageSource: [Layering_of_Computer_Architecture]

---

2. **How does a C program run on your laptop? List as many steps as possible.**

The general flow for executing a C program is as follows:

C program -> Compiler -> Assembly language -> Assembler -> Binary (Executable) -> Loader -> Running Process

1. C program: It is a high-level description of computation. You create an ***application*** using a high-level language (C regarding this question). You will use various ***algorithms*** to implement your application. Moreover, you describe your application to the computer in a particular ***programming language*** with predefined syntax and semantics.
2. Compiler: Your application gets converted to assembly language by the compiler. A compiler uses the grammar of a programming language to interpret what the programmer means. Using this information, it generates assembly.
3. Assembly: At this stage, your program is no longer represented in a high-level language. It is now represented in the ***assembly***. The ISA of the target architecture defines the instructions in the assembly. Note that assembly is still a text file with human-readable instructions. We have yet to convert it to binary.
4. Assembler: Assembler converts the assembly to executable binary, i.e. ***machine code***. A computer can interpret this machine code.
5. Binary: The binary is a set of instructions that a computer can interpret easily. The binary must maintain the semantics of the ***Instruction Set Architecture*. ** ISA defines all instructions supported by the computer. It also defines the behavior of the computer in various situations. However, this is just an abstraction, an illusion the computer maintains to present a consistent view of the machine to the higher layers. The actual internal workings of the machine may not align with what the ISA exports. This is perfectly fine as long as the semantics of the ISA can be maintained. For example, a multiply instruction in the ISA must behave like a multiply operation, even if, internally, it is executed as repeated additions. Here the machine maintains an illusion of supporting multiplication.

NOTE: For more information, refer to

1. How the Compilation Process Works for C Programs
2. Loader (Computing)

Try to relate each step with the corresponding abstraction level described in the class. It's okay if you do not understand each step in detail. However, you must know what these steps are. This question aims to get acquainted with a general flow of how the high-level codes are executed on the machine.

3. Write down the interfaces and ports available in a computer.

1. Serial Port
2. Parallel Port
3. USB (Universal Serial Bus)
4. VGA (Video Graphics Array) Connector
5. Audio Plugs
6. PS/2 (Personal System/2) Port
7. SCSI (Small Computer Systems Interface) Port
8. High Definition Multimedia Interface (HDMI).

4. What is Von Neumann's architecture's main drawback over Harvard Architecture?

There are two types of Processor-Memory architecture possible. These are Von Neumann, Harvard. The Data and Instruction memory are kept in Harvard architecture as two different entities. In Von Neumann, the data and instructions are stored in a single memory. So, the processor can only read the data or the instruction at a transaction, reducing the processor performance in Von Neumann's architecture.

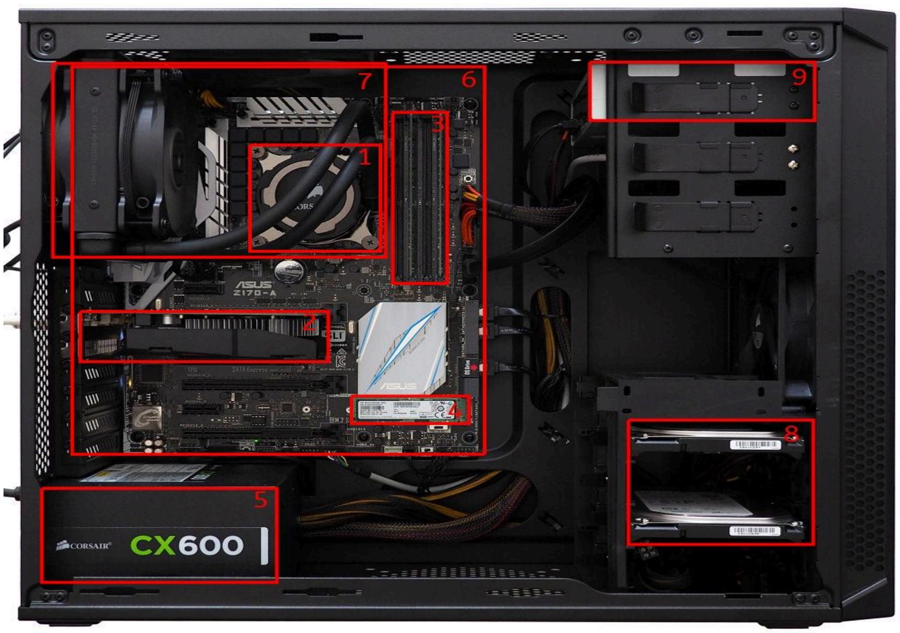
5. What are the main components of Modern desktop systems? Classify them as storage, computation, interconnect, or power. List some of the key defining features of these components.

Components	Classification	Key features
CPU	Computation	core count, SMT support, clock frequency, IPC
GPU	Computation	clock frequency, amount of GDDR memory
RAM	Volatile Storage	size, latency
HDD	Non-Volatile Storage	size, latency, robustness (how long before it likely fails)
SSD	Non-Volatile Storage	size, latency, robustness (how long before it likely fails, often denoted in terms of read/write cycles), type (NVME/SATA)
Motherboard	Interconnect	chipset, IO ports, form-factor
Power Supply	Power	power, wattage, efficiency

6. List out some ISAs.
The list of ISA:

Reduced Instruction Set Computer type	Complex Instruction Set Computer type
Alpha RISC V Arm	X86 Z80 8051

7. Label all the components that you see in this picture.



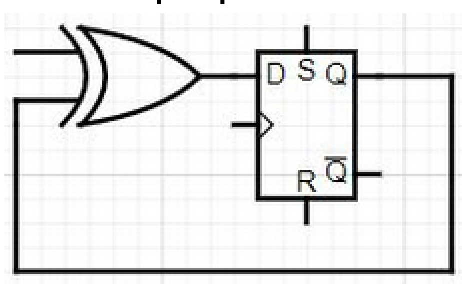
Some of the major components shown in the picture are
1. CPU (Under the heatsink+pump of the cooling system)
2. GPU
3. RAM
4. SSD (M.2 type)
5. Power supply
6. Motherboard (The main PCB holding everything together)
7. Cooling system (Water cooling in this case)
8. HDD (might also be SATA SSDs, unclear from the picture)
9. CD/DVD

Please note that it is not essential to know all the acronyms. However, you must know what different components look like.
ImageSource: https://images.app.goo.gl/KbiZEWuFoBqa7d539

8. Differentiate between Asynchronous and Synchronous Sequential circuits.
Ans:

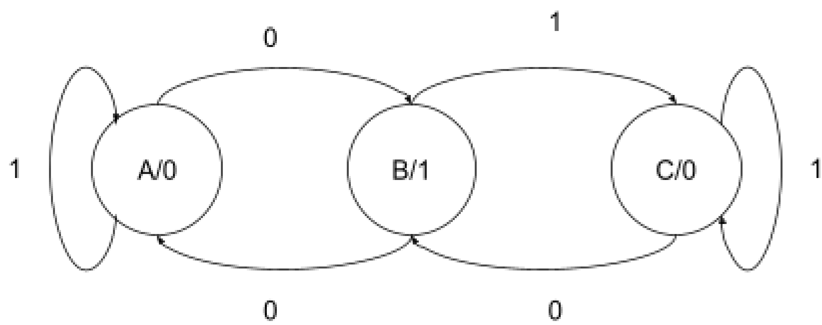
Asynchronous Sequential Circuit	Synchronous Sequential Circuit
1. These circuits do not use clock signals. 2. They are faster as they change the state when there is a change in the input signal. 3. These circuits are more difficult to design, and their output is uncertain. 4. Eg asynchronous counter.	1. These circuits use clock signals. 2. They are slower as they wait for the next clock pulse to perform the next operation. 3. They are easier to design and debug. 4. Eg. Synchronous counter, flip flops.

9. Which flip-flop is this?



Ans: T-flip flop

10. Design a digital circuit using D-Flip flops for the Moore machine mentioned below.



Ans:
Assign States as follows:
A(00), B(01), C(10), D(11).

Old State		Input	Next State	
x1	x0	a	x1	x0
0	0	0	0	1
0	0	1	0	0
0	1	1	1	0
0	1	0	0	0
1	0	0	0	1
1	0	1	1	0
1	1	0	0	0
1	1	1	0	0

x1 = (x1_bar * x0 * a) + (x1 * x0_bar * a)

x0 = (x1_bar * x0_bar * a_bar) + (x1 * x0 * a_bar)

Remark: If we do not account state D(11) the system may turn into lockup state.