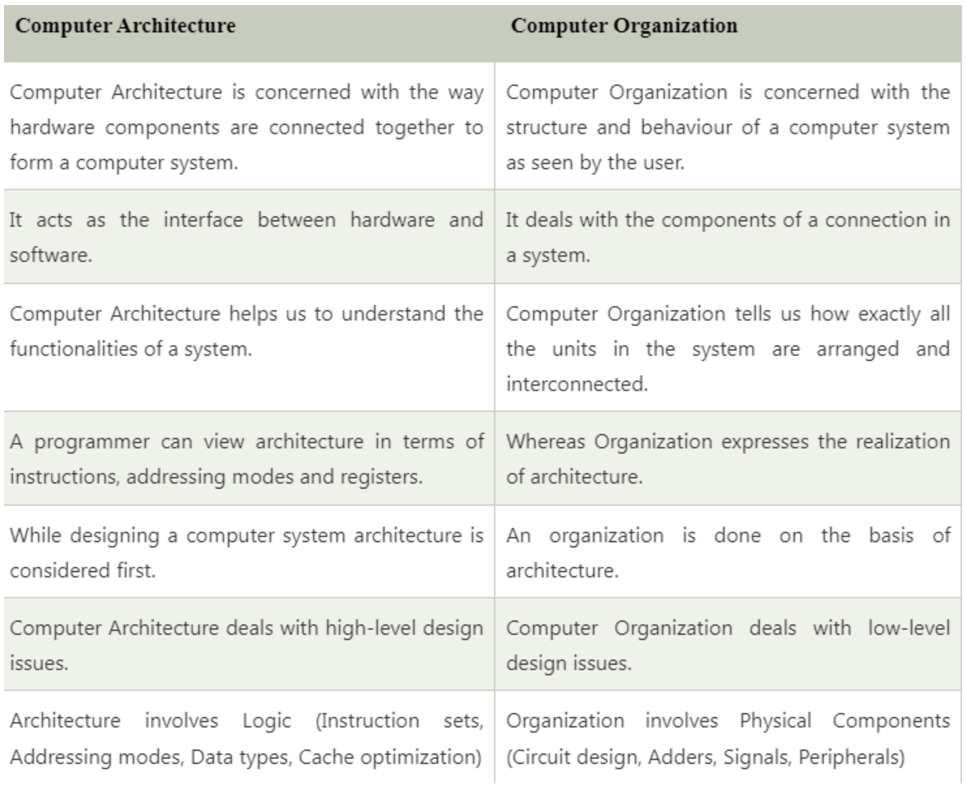
Computer Organization & Architecture (CSE 303)

Computer Architecture VS Computer Organization

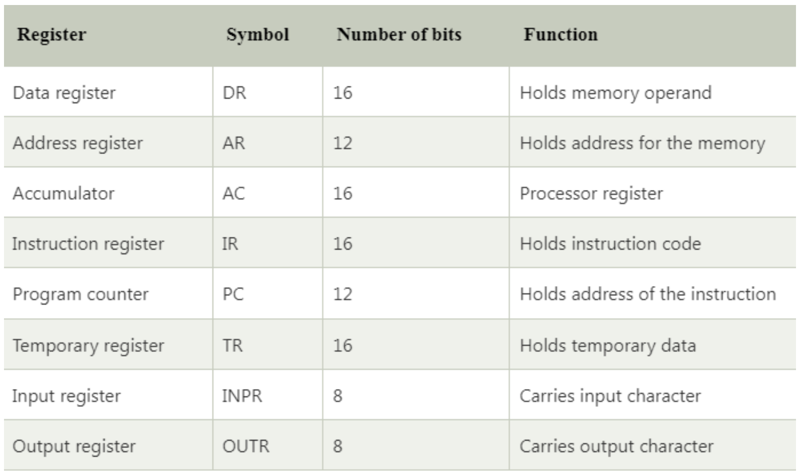


Computer Architecture	Computer Organization
Computer Architecture is concerned with the way hardware components are connected together to form a computer system.	Computer Organization is concerned with the structure and behaviour of a computer system as seen by the user.
It acts as the interface between hardware and software.	It deals with the components of a connection in a system.
Computer Architecture helps us to understand the functionalities of a system.	Computer Organization tells us how exactly all the units in the system are arranged and interconnected.
A programmer can view architecture in terms of instructions, addressing modes and registers.	Whereas Organization expresses the realization of architecture.
While designing a computer system architecture is considered first.	An organization is done on the basis of architecture.
Computer Architecture deals with high-level design issues.	Computer Organization deals with low-level design issues.
Architecture involves Logic (Instruction sets, Addressing modes, Data types, Cache optimization)	Organization involves Physical Components (Circuit design, Adders, Signals, Peripherals)

Computer Registers

Registers are a type of computer memory used to quickly accept, store, and transfer data and instructions that are being used immediately by the CPU. The registers used by the CPU are often termed as Processor registers.
Following are the different types of registers involved in each instruction cycle:

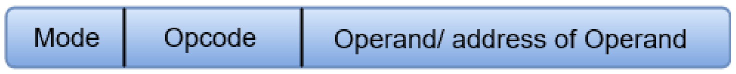
1. **Memory address registers(MAR):** It is connected to the address lines of the system bus. It specifies the address in memory for a read or write operation.
2. **Memory Buffer Register(MBR):** It is connected to the data lines of the system bus. It contains the value to be stored in memory or the last value read from the memory.
3. **Program Counter(PC):** Holds the address of the next instruction to be fetched.
4. **Instruction Register(IR):** Holds the last instruction fetched.



Register	Symbol	Number of bits	Function
Data register	DR	16	Holds memory operand
Address register	AR	12	Holds address for the memory
Accumulator	AC	16	Processor register
Instruction register	IR	16	Holds instruction code
Program counter	PC	12	Holds address of the instruction
Temporary register	TR	16	Holds temporary data
Input register	INPR	8	Carries input character
Output register	OUTR	8	Carries output character

Computer Instructions
Computer instructions are a set of machine language instructions that a particular processor understands and executes. A computer performs tasks on the basis of the instruction provided.

An instruction comprises of groups called fields. These fields include:

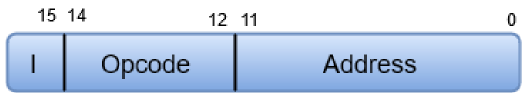


- The Operation code (Opcode) field which specifies the operation to be performed.
- The Address field which contains the location of the operand, i.e., register or memory location.
- The Mode field which specifies how the operand will be located.

A basic computer has three instruction code formats which are:

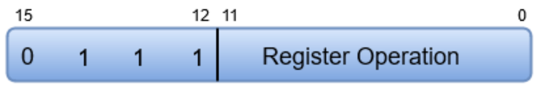
1. Memory - reference instruction
2. Register - reference instruction
3. Input-Output instruction

Memory - reference instruction



(Opcode = 000 through 110)

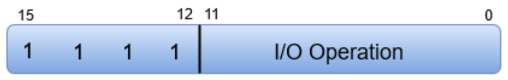
In Memory-reference instruction, 12 bits of memory is used to specify an address and one bit to specify the addressing mode 'I'.
Register - reference instruction



(Opcode = 111, I = 0)

The Register-reference instructions are represented by the Opcode 111 with a 0 in the leftmost bit (bit 15) of the instruction. A Register-reference instruction specifies an operation on or a test of the AC (Accumulator) register.

Input-Output instruction

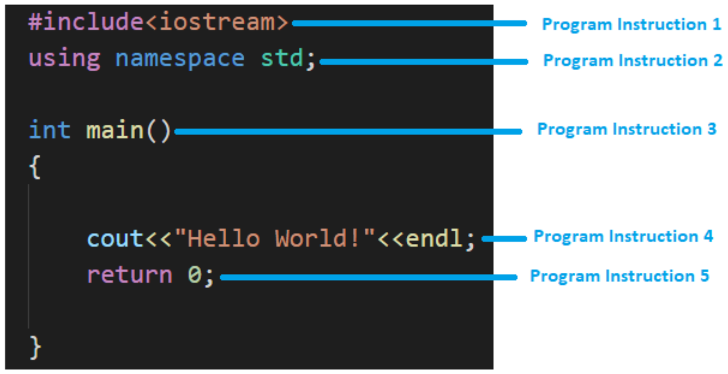


(Opcode = 111, I = 1)

Just like the Register-reference instruction, an Input-Output instruction does not need a reference to memory and is recognized by the operation code 111 with a 1 in the leftmost bit of the instruction. The remaining 12 bits are used to specify the type of the input-output operation or test performed.

How a Program Instruction is Executed in a Computer System?

The CPU starts the program execution by fetching them one by one. The control unit decodes the machine instructions as per the instruction format.
The computer program uses different types of program instructions as per the program logic and algorithm.



```
#include<iostream>
using namespace std;

int main()
{

    cout<<"Hello World!"<<endl;
    return 0;

}
```

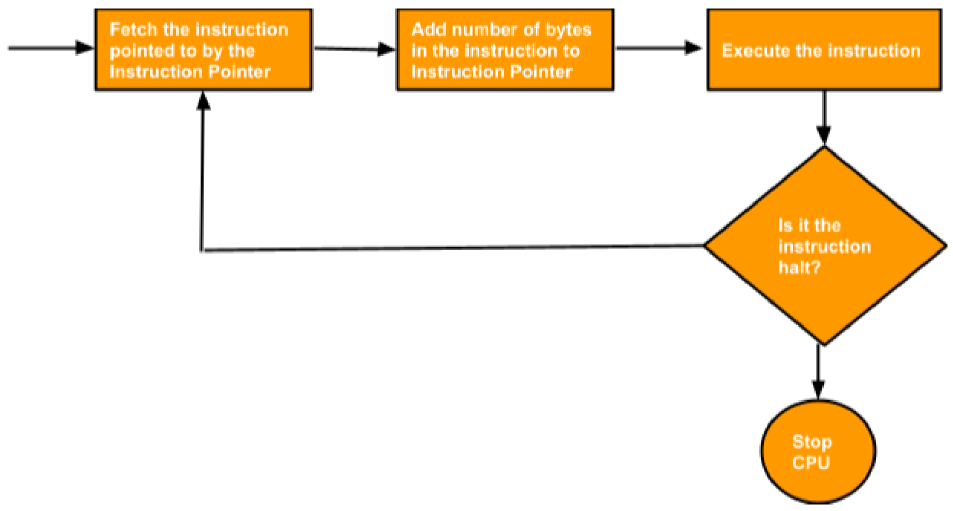
During the program compilation process, each program instruction is converted into machine instruction in binary. Depending upon the programming language, the program compiler converts the entire program into an executable code (set of machine instructions). In the case of an interpreted language, this conversion takes place line by line.

The executable code consists of a set of machine instructions in binary that can be directly decoded and executed by the CPU. A central processor of a computer is programmed to carry out the instructions provided to the CPU. It contains a special register — the instruction register — whose bit pattern determines what the central processor unit can do. Once that action has been completed, the bit pattern within the instruction register may be modified, and also the central processor unit can perform the operation nominally by this next bit pattern.

Since directions are simply bit patterns, they will be kept in memory. The instruction pointer register continuously has the memory address of (points to) the next instruction to be executed. so as for the management unit to execute this instruction, it's derived into the instruction register. the case is as follows:

1. A sequence of instructions is stored in memory.
2. The memory address wherever the first instruction is found is copied to the instruction pointer.
3. The CPU sends the address within the instruction pointer to memory on the address bus.
4. The CPU sends a "read" signal to the control bus.
5. Memory responds by sending a copy of the state of the bits at that memory location on the data bus, which the CPU then copies into its instruction register.
6. The instruction pointer is automatically incremented to contain the address of the next instruction in memory.
7. The CPU executes the instruction within the instruction register.
8. Go to step 3
9. Steps 3, 4, and 5 are called instruction fetch. Notice that steps 3 – 8 constitute a cycle, the instruction execution cycle.

It is shown graphically below.

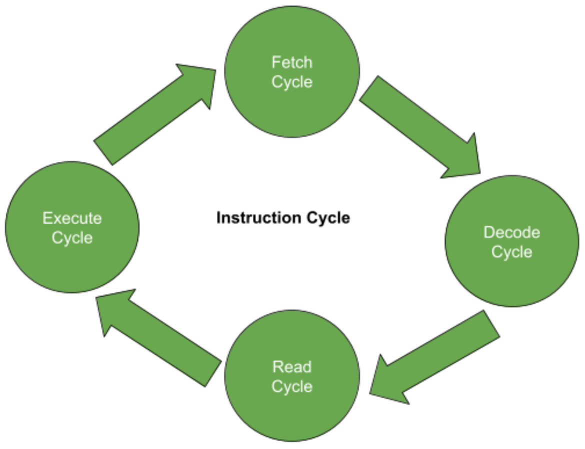

Fetch the instruction pointed to by the Instruction Pointer
Add number of bytes in the instruction to Instruction Pointer
Execute the instruction
Is it the instruction halt?
Stop CPU

Instruction Cycle In Computer Architecture

Each computer's CPU can have different cycles based on different instruction sets, but will be similar to the following cycle:

Fetch Stage: The next instruction is fetched from the memory address that is currently stored in the program counter and stored in the instruction register. At the end of the fetch operation, the PC points to the next instruction that will be read in the next cycle.

- **Fetch Cycle:** Fetch the instruction from memory
- **Decode Cycle:** Decode the instruction
- **Read Cycle:** Read the effective address from the memory
- **Execute Cycle:** Execute the instruction



The following are the main components of every instruction cycle:

1. Fetch Cycle

The fetching of instruction is the first phase. The fetch instruction is common for each instruction executed in a central processing unit. In this phase, the central processing unit sends the PC to MAR and then sends the READ command into a control bus.

After sending a read command on the data bus, the memory returns the instruction, which is stored at that particular address in the memory. Then, the CPU copies data from the data bus into MBR and then copies the data from MBR to registers.

After all this, the pointer is incremented to the next memory location so that the next instruction can be fetched from memory.

2. Decode Cycle

The decoding of instruction is the second phase. In this phase, the CPU determines which instruction is fetched from the instruction and what action needs to be performed on the instruction. The opcode for the instruction is also fetched from memory and decodes the related operation which needs to be performed for the related instruction.

3. Read Cycle

The reading of an effective address is the third phase. This phase deals with the decision of the operation. The operation can be of any type of memory type non-memory type operation. Memory instruction can be categorized into two categories: direct memory instruction and indirect memory instruction.

4. Execute Cycle

The execution of the instruction is the last phase. In this stage, the instruction is finally executed. The instruction is executed, and the result of the instruction is stored in the register. After the execution of an instruction, the CPU prepares itself for the execution of the next instruction. For every instruction, the execution time is calculated, which is used to tell the processing speed of the processor.

Types of Instruction Set

Generally, there are two types of instruction set used in computers.

1. Reduced Instruction Set Computer (RISC)
2. Complex Instruction Set Computer (CISC)

1. Reduced Instruction Set Computer

A number of computer designers recommended that computers use fewer instructions with simple constructs so that they can be executed much faster within the CPU without having to use memory as often. This type of computer is called a Reduced Instruction Set Computer.

The concept of RISC involves an attempt to reduce execution time by simplifying the instruction set of computers.

Characteristics of RISC

The characteristics of RISC are as follows:

- Relatively few instructions.
- Relatively few addressing modes.

- Memory access is limited to loading and storing instructions.
- All operations are done within the register of the CPU.
- Single-cycle instruction execution.
- Fixed length, easily decoded instruction format.
- Hardwired rather than microprogrammed control.

A characteristic of RISC processors' ability is to execute one instruction per clock cycle. This is done by overlapping the fetch, decode and execute phases of two or three instructions by using a procedure referred to as pipelining.

2. Complex Instruction Set Computer

CISC is a computer where a single instruction can perform numerous low-level operations like a load from memory and a store from memory, etc. The CISC attempts to minimize the number of instructions per program but at the cost of an increase in the number of cycles per instruction.

The design of an instruction set for a computer must take into consideration not only machine language constructs but also the requirements imposed on the use of high-level programming languages.

The goal of CISC is to attempt to provide a single machine instruction for each statement that is written in a high-level language.

Characteristics of CISC

The characteristics of CISC are as follows:

- A large number of instructions typically from 100 to 250 instructions.
- Some instructions perform specialized tasks and are used infrequently.
- A large variety of addressing modes- typically from 5 to 20 different modes.
- Variable length instruction formats.
- Instructions that manipulate operands in memory.

Example: For performing an ADD operation.

- CISC will execute a single ADD command which will execute all the required load and store operations.
- RISC will execute each operation for loading data from memory, adding values, and storing data back to memory using different low-level instructions.