UNIT -1

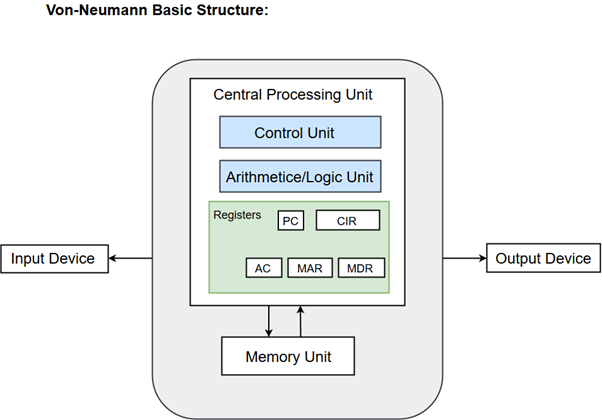
1. EXPLAIN VON NEWMANN MODEL WITH DIAGRAM?

ANS - Von-Neumann proposed his computer architecture design in 1945 which was later known as Von-Neumann Architecture. It consisted of a Control Unit, Arithmetic, and Logical Memory Unit (ALU), Registers and Inputs/Outputs.

Von Neumann architecture is based on the stored-program computer concept, where instruction data and program data are stored in the same memory. This design is still used in most computers produced today.

### A Von Neumann-based computer:

* Uses a single processor
* Uses one memory for both instructions and data.
* Executes programs following the fetch-decode-execute cycle



## Components of Von-Neumann Model:

* Central Processing Unit
* Buses
* Memory Unit

## Central Processing Unit

The part of the Computer that performs the bulk of data processing operations is called the Central Processing Unit and is referred to as the CPU.

The Central Processing Unit can also be defined as an electric circuit responsible for executing the instructions of a computer program.

The CPU performs a variety of functions dictated by the type of instructions that are incorporated in the computer.

The major components of CPU are Arithmetic and Logic Unit (ALU), Control Unit (CU) and a variety of registers.

### Arithmetic and Logic Unit (ALU)

The Arithmetic and Logic Unit (ALU) performs the required micro-operations for executing the instructions. In simple words, ALU allows arithmetic (add, subtract, etc.) and logic (AND, OR, NOT, etc.) operations to be carried out.

### Control Unit

The Control Unit of a computer system controls the operations of components like ALU, memory and input/output devices.

The Control Unit consists of a program counter that contains the address of the instructions to be fetched and an instruction register into which instructions are fetched from memory for execution.

### Registers

Registers refer to high-speed storage areas in the CPU. The data processed by the CPU are fetched from the registers.

Following is the list of registers that plays a crucial role in data processing.

|  |  |
| --- | --- |
| Registers | Description |
| MAR (Memory Address Register) | This register holds the memory location of the data that needs to be accessed. |
| MDR (Memory Data Register) | This register holds the data that is being transferred to or from memory. |
| AC (Accumulator) | This register holds the intermediate arithmetic and logic results. |
| PC (Program Counter) | This register contains the address of the next instruction to be executed. |
| CIR (Current Instruction Register) | This register contains the current instruction during processing. |

## Buses

Buses are the means by which information is shared between the registers in a multiple-register configuration system.

A bus structure consists of a set of common lines, one for each bit of a register, through which binary information is transferred one at a time. Control signals determine which register is selected by the bus during each particular register transfer.

Von-Neumann Architecture comprised of three major bus systems for data transfer.

|  |  |
| --- | --- |
| Bus | Description |
| Address Bus | Address Bus carries the address of data (but not the data) between the processor and the memory. |
| Data Bus | Data Bus carries data between the processor, the memory unit and the input/output devices. |
| Control Bus | Control Bus carries signals/commands from the CPU. |

## Memory Unit

A memory unit is a collection of storage cells together with associated circuits needed to transfer information in and out of the storage. The memory stores binary information in groups of bits called words. The internal structure of a memory unit is specified by the number of words it contains and the number of bits in each word.

**Two major types of memories are used in computer systems:**

1. RAM (Random Access Memory)
2. ROM (Read-Only Memory)

2. TYPES OF ADDRESSING MODES?

ANS - Types of Addressing Modes

Below we have discussed different types of addressing modes one by one:

### Immediate Mode

In this mode, the operand is specified in the instruction itself. An immediate mode instruction has an operand field rather than the address field.

For example: ADD 7, which says Add 7 to contents of accumulator. 7 is the operand here.

### Register Mode

In this mode the operand is stored in the register and this register is present in CPU. The instruction has the address of the Register where the operand is stored.

### Register Indirect Mode

In this mode, the instruction specifies the register whose contents give us the address of operand which is in memory. Thus, the register contains the address of operand rather than the operand itself.

Auto Increment/Decrement Mode

In this the register is incremented or decremented after or before its value is used.

### Direct Addressing Mode

In this mode, effective address of operand is present in instruction itself.

* Single memory reference to access data.
* No additional calculations to find the effective address of the operand.

**For Example:** ADD R1, 4000 - In this the 4000 is effective address of operand.

### Indirect Addressing

In this, the address field of instruction gives the address where the effective address is stored in memory. This slows down the execution, as this includes multiple memory lookups to find the operand.

### Displacement Addressing Mode

In this the contents of the indexed register is added to the Address part of the instruction, to obtain the effective address of operand.

EA = A + (R), In this the address field holds two values, A(which is the base value) and R(that holds the displacement), or vice versa.

### Relative Addressing Mode

It is a version of Displacement addressing mode.

In this the contents of PC(Program Counter) is added to address part of instruction to obtain the effective address.

EA = A + (PC), where EA is effective address and PC is program counter.

The operand is A cells away from the current cell(the one pointed to by PC)

### Base Register Addressing Mode

It is again a version of Displacement addressing mode. This can be defined as EA = A + (R), where A is displacement and R holds pointer to base address.

### Stack Addressing Mode

In this mode, operand is at the top of the stack. For example: ADD, this instruction will *POP* top two items from the stack, add them, and will then *PUSH* the result to the top of the stack.

3. INSTRUCTION CYCLE ?

ANS - Instruction Cycle

An instruction cycle, also known as **fetch-decode-execute cycle** is the basic operational process of a computer. This process is repeated continuously by CPU from boot up to shut down of computer.

Following are the steps that occur during an instruction cycle:

### 1. Fetch the Instruction

The instruction is fetched from memory address that is stored in PC(Program Counter) and stored in the instruction register IR. At the end of the fetch operation, PC is incremented by 1 and it then points to the next instruction to be executed.

### 2. Decode the Instruction

The instruction in the IR is executed by the decoder.

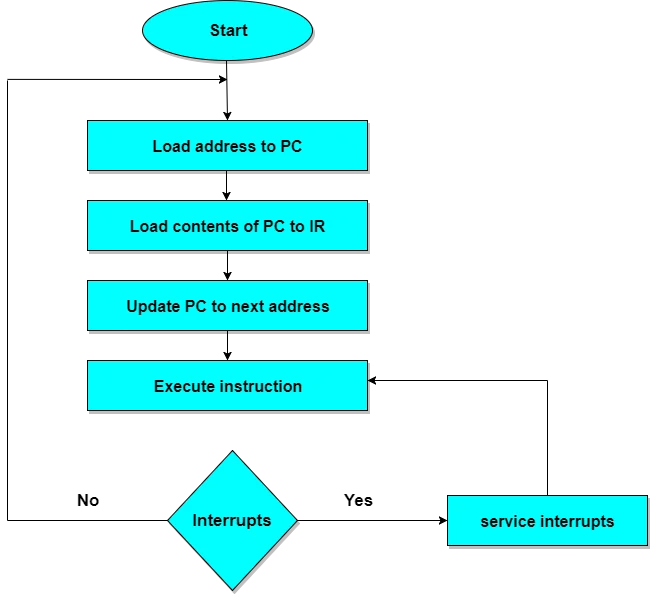
### 3. Read the Effective Address

If the instruction has an indirect address, the effective address is read from the memory. Otherwise operands are directly read in case of immediate operand instruction.

### 4. Execute the Instruction

The Control Unit passes the information in the form of control signals to the functional unit of CPU. The result generated is stored in main memory or sent to an output device.

The cycle is then repeated by fetching the next instruction. Thus in this way the instruction cycle is repeated continuously.



4 BUS STRUCTURE?

ANS - A Bus is a collection of wires that connects several devices.

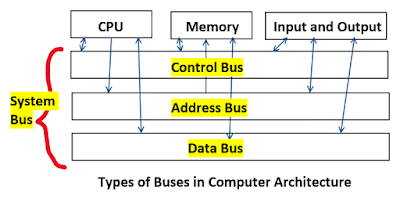
Buses are used to send control signals and data between the processor and other components

This is to achieve a  reasonable speed of operation.

In computer system all the peripherals are connected to microprocessor through Bus.

Types of Bus structure:

1. Address bus
2. Data bus
3. Control bus



1. Address Bus:

1. Address bus carry the memory address while reading from writing into memory.
2. Address bus caary I/O post address or device address from I/O port.
3. In uni-directional address bu only the CPU could  send address and other units could not address the microprocessor.
4. Now a days computers are haing bi-directional address bus.

2. Data Bus:

1. Data bus carry the data.
2. Data bus is a bidirectional bus.
3. Data bus fetch the instructions from memory.
4. Data bus used to store the result of an instruction into memory.
5. Data bus carry commands to an I/O device controller or port.
6. Data bus carry data from a device controller or port.
7. Data bus issue data to a device controller or port.

3. Control Bus:

Different types of control signals are used in a bus:

1. Memory Read: This signal, is issued by the CPU or DMA controller when performing  a read operation with the memory.
2. MemoryWrite: This signal isissued by the CPU or DMAcontroller when performing  a write operation with the memory.
3. I/O Read: This signal isissued by the CPU when it is reading from an input port.
4. I/O Write: This signal is issued by the CPU when writing into an output port.
5. Ready: The ready is an input signal to the CPU

5. WHAT IS MAR AND PC?

ANS - the Memory Address Register (MAR) is the CPU register that either stores the memory address from which data will be fetched to the CPU or theaddress to which data will be sent and stored. In other words, MAR holds the memory location of data that needs to be accessed.

A program counter (PC) is a [register](https://www.techtarget.com/whatis/definition/register) in a computer [processor](https://www.techtarget.com/whatis/definition/processor) that contains the address (location) of the [instruction](https://www.techtarget.com/whatis/definition/instruction) being executed at the current time. As each instruction gets [fetched](https://searchsqlserver.techtarget.com/definition/fetch), the program counter increases its stored value by 1. After each instruction is fetched, the program counter points to the next instruction in the sequence. When the computer restarts or is reset, the program counter normally reverts to 0

UNIT -2

1. CONTROL UNIT ORGANIZATION?

**Control Unit Organization**

**The Control Unit is the unit in the CPU, which controls the various components like input  & output devices, logic unit and memory. The Control Unit is the circuitry that controls or directs the component’s operations and tells them how to respond to the instruction received from the program.**

### Functions of the control unit

**CU's main function is to direct the operations during the execution of a program by the processor.**

**The control unit converts the input into control signals sent to different components and registers to execute its functioning.**

**The Control Unit cannot store or process the data; it just directs the data signals.**

**Control unit directs working of all its components. It provides direction to the control signals and send them to the appropriate component to execute the instruction. Control unit components are:- control signal within the CPU, instruction register, input flags, control bus, clock signals, control signals to/from the bus.**

### Design of Control Unit

**Every CPU can be designed in a different- different way, same with the control unit. Control unit can be designed in two possible best ways.**

1. **Hardwired Control Unit**
2. **Micro-programmed Control Unit**

### Hardwired Control Unit

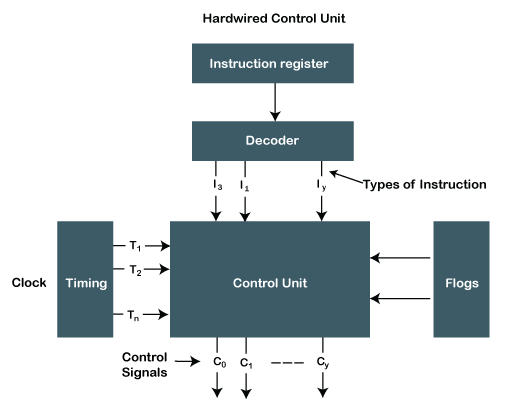
**As the name defined hardwired control unit, going to design the control unit by using the hardware (circuits). The control logic is implemented with Gates, flip-flops, decoders, and other digital circuits in this type.**

“***Control logic is the unit which generates the control signals which are implemented using the hardware that’s why it is called as hardwired control unit.”***

**Advantage:  Hardware is always faster. Due to this, it can be optimized to produce a faster mode of operation. It generates the fastest control signal.**

**Disadvantage: a) If CPU design is very complicated or very much modern, hardwired control is difficult to implement**

**b) Rearranging the wires among various components is difficult.**

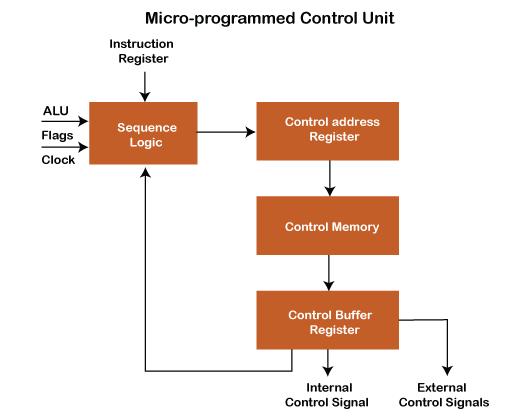


### Micro-programmed Control Unit

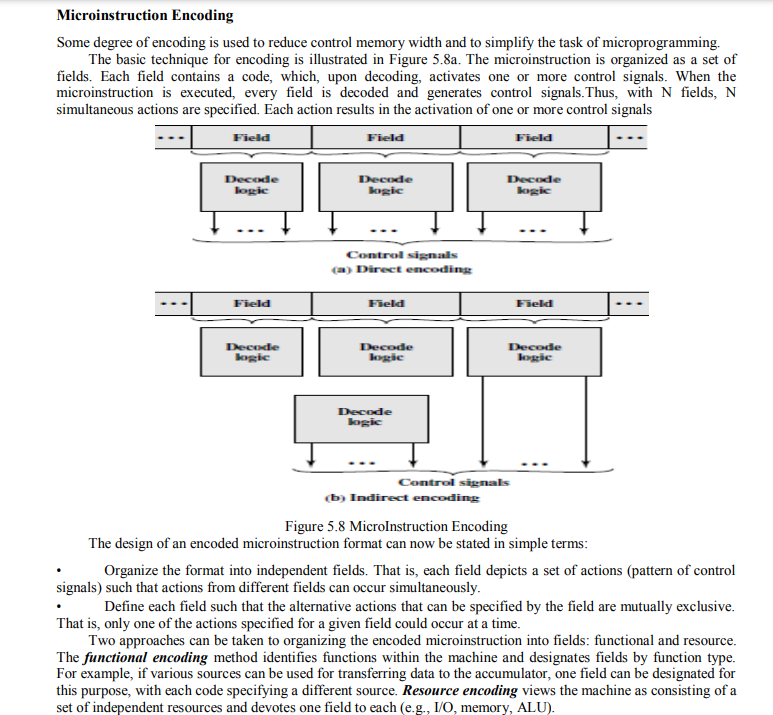
**In this control, logic is implemented with micro-programs. That's why it is called a micro-programmed control unit. It’s a midway between hardware and software.**

“**All possible control words are stored in a memory because it is a part of control unit) and based on the requirements, the specific control word is fetched from memory, then its signals are sent to different components.”**

**In this type, the control word is used to store the control signals encoded during the execution of a program and control words are stored in the memory. Memory can be easily changed; we can change the value (0 or 1) in the control word, enabling and disabling signals. We do not need to change the whole control word format as it can work on a variable format of 16-64 bit, or we can easily update the control logic.**



2. micro instruction encoding?

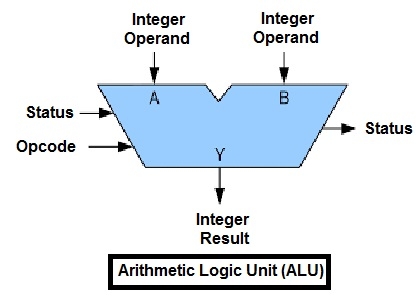
3. differnec between horizontal and vertical micro instructions?

Ans - Difference between Horizontal and Vertical micro-programmed Control Unit:

| **S. No** | **Horizontal µ-programmed CU** | **Vertical µ-programmed CU** |
| --- | --- | --- |
| 1. | It supports longer control word. | It supports shorter control word. |
| 2. | It allows a higher degree of parallelism. If degree is n, then n Control Signals are enabled at a time. | It allows a low degree of parallelism i.e., the degree of parallelism is either 0 or 1. |
| 3. | No additional hardware is required. | Additional hardware in the form of decoders is required to generate control signals. |
| 4. | It is faster than a Vertical micro-programmed control unit. | it is slower than a Horizontal micro-programmed control unit. |
| 5. | It is less flexible than a Vertical micro-programmed control unit. | It is more flexible than a Horizontal micro-programmed control unit. |
| 6. | A horizontal micro-programmed control unit uses horizontal micro-instruction, where every bit in the control field attaches to a control line. | A vertical micro-programmed control unit uses vertical micro-instruction, where a code is used for each action to be performed and the decoder translates this code into individual control signals. |
| 7. | The horizontal micro-programmed control unit makes less use of ROM encoding than the vertical micro-programmed control unit. |  |

UNIT -3

1. DESIGN ARITHMETIC LOGIC UNIT

ANS - Inside a computer, there is an Arithmetic Logic Unit (ALU), which is capable of performing logical operations (e.g. AND, OR, Ex-OR, Invert etc.) in addition to the arithmetic operations (e.g. Addition, Subtraction etc.). The control unit supplies the data required by the ALU from memory, or from input devices, and directs the ALU to perform a specific operation based on the instruction fetched from the memory. ALU is the “calculator” portion of the computer.

An arithmetic logic unit(ALU) is a major component of the central processing unit of the a computer system. It does all processes related to arithmetic and logic operations that need to be done on instruction words. In some microprocessor architectures, the ALU is divided into the arithmetic unit (AU) and the logic unit (LU).

An ALU can be designed by engineers to calculate many different operations. When the operations become more and more complex, then the ALU will also become more and more expensive and also takes up more space in the CPU and dissipates more heat. That is why engineers make the ALU powerful enough to ensure that the CPU is also powerful and fast, but not so complex as to become prohibitive in terms of cost and other disadvantages.

ALU is also known as an Integer Unit (IU). The arithmetic logic unit is that part of the CPU that handles all the calculations the CPU may need. Most of these operations are logical in nature. Depending on how the ALU is designed, it can make the CPU more powerful, but it also consumes more energy and creates more heat. Therefore, there must be a balance between how powerful and complex the ALU is and how expensive the whole unit becomes. This is why faster CPUs are more expensive, consume more power and dissipate more heat.

Different operation as carried out by ALU can be categorized as follows –

* **logical operations** − These include operations like AND, OR, NOT, XOR, NOR, NAND, etc.
* **Bit-Shifting Operations** − This pertains to shifting the positions of the bits by a certain number of places either towards the right or left, which is considered a multiplication or division operations.
* **Arithmetic operations** − This refers to bit addition and subtraction. Although multiplication and division are sometimes used, these operations are more expensive to make. Multiplication and subtraction can also be done by repetitive additions and subtractions respectively.

2. CISC AND RISC

ANS - Reduced Instruction Set Architecture (RISC) –   
The main idea behind this is to make hardware simpler by using an instruction set composed of a few basic steps for loading, evaluating, and storing operations just like a load command will load data, a store command will store the data.

Complex Instruction Set Architecture (CISC) –   
The main idea is that a single instruction will do all loading, evaluating, and storing operations just like a multiplication command will do stuff like loading data, evaluating, and storing it, hence it’s complex.

Both approaches try to increase the CPU performance

* RISC: Reduce the cycles per instruction at the cost of the number of instructions per program.
* CISC: The CISC approach attempts to minimize the number of instructions per program but at the cost of an increase in the number of cycles per instruction.

Earlier when programming was done using assembly language, a need was felt to make instruction do more tasks because programming in assembly was tedious and error-prone due to which CISC architecture evolved but with the uprise of high-level language dependency on assembly reduced RISC architecture prevailed.

Characteristic of RISC –

1. Simpler instruction, hence simple instruction decoding.
2. Instruction comes undersize of one word.
3. Instruction takes a single clock cycle to get executed.
4. More general-purpose registers.
5. Simple Addressing Modes.
6. Fewer Data types.
7. A pipeline can be achieved.

Characteristic of CISC –

1. Complex instruction, hence complex instruction decoding.
2. Instructions are larger than one-word size.
3. Instruction may take more than a single clock cycle to get executed.
4. Less number of general-purpose registers as operations get performed in memory itself.
5. Complex Addressing Modes.
6. More Data types.

3. INSTRUCTION PIPELINING?

ANS - Instruction Pipeline

Pipeline processing can occur not only in the data stream but in the instruction stream as well.

Most of the digital computers with complex instructions require instruction pipeline to carry out operations like fetch, decode and execute instructions.

In general, the computer needs to process each instruction with the following sequence of steps.

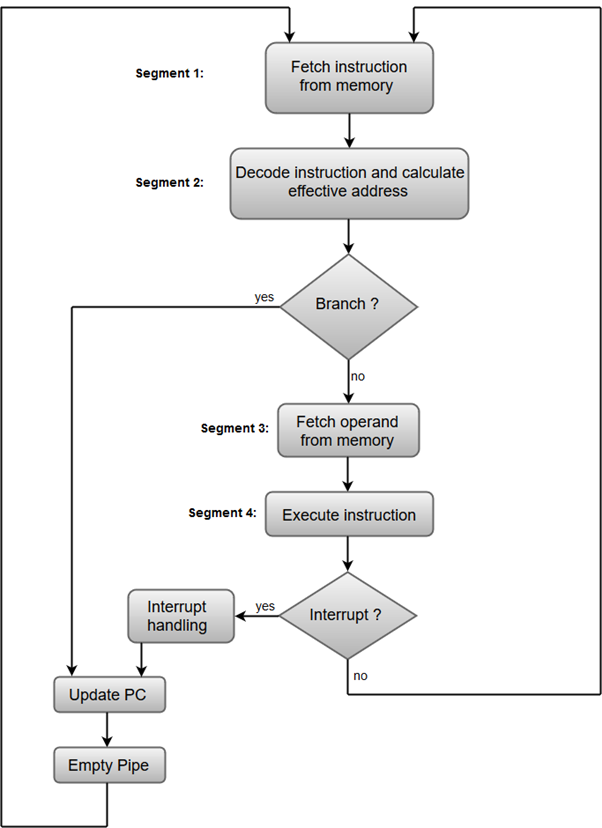
1. Fetch instruction from memory.
2. Decode the instruction.
3. Calculate the effective address.
4. Fetch the operands from memory.
5. Execute the instruction.
6. Store the result in the proper place.

Each step is executed in a particular segment, and there are times when different segments may take different times to operate on the incoming information. Moreover, there are times when two or more segments may require memory access at the same time, causing one segment to wait until another is finished with the memory.

The organization of an instruction pipeline will be more efficient if the instruction cycle is divided into segments of equal duration. One of the most common examples of this type of organization is a **Four-segment instruction pipeline.**

A **four-segment instruction** pipeline combines two or more different segments and makes it as a single one. For instance, the decoding of the instruction can be combined with the calculation of the effective address into one segment.

The following block diagram shows a typical example of a four-segment instruction pipeline. The instruction cycle is completed in four segments.



#### Segment 1:

The instruction fetch segment can be implemented using first in, first out (FIFO) buffer.

#### Segment 2:

The instruction fetched from memory is decoded in the second segment, and eventually, the effective address is calculated in a separate arithmetic circuit.

#### Segment 3:

An operand from memory is fetched in the third segment.

#### Segment 4:

The instructions are finally executed in the last segment of the pipeline organization.