Lecture Notes

On

Computer Architecture

UNIT-II

THE PROCESSOR ARCHITECTURE: VON-NEUMANN AND NON-VON-NEUMANN MACHINES, INTRODUCTION TO: (RISC & CISC); (DESIGNING A HYPOTHETICAL CPU) INSTRUCTION SET DESIGN: DATA REPRESENTATION, REGISTER SETS, TYPES OF INSTRUCTIONS, ADDRESSING TECHNIQUES; ALU/DATAPATH DESIGN: OPERATION CODES(OPCODES), ARITHMETIC OPERATIONS, LOGICAL OPERATIONS; CONTROL UNIT DESIGN: HARDWIRED AND MICRO-PROGRAMMED DESIGN APPROACHES.

VON-NEUMANN MACHINES:

1. The invention of stored program computers has been described by a mathematician, John von Neumann, who was a contemporary of Mauchley and Eckert.
2. Stored-program computers have become known as **von Neumann Architecture** systems.

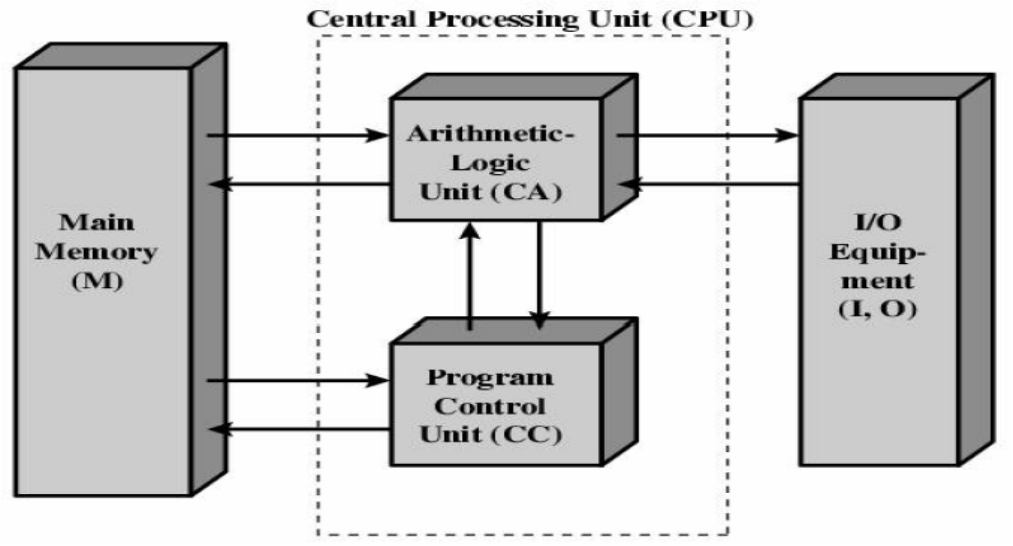
Today's stored-program computers have the following characteristics:

– Three hardware systems:

- A central processing unit (CPU)
- A main memory system
- An I/O system

-The capacity to carry out sequential instruction processing.

-A single data path between the CPU and main memory. This single path is known as the von Neumann bottleneck.



computer model by Von Neumann's group

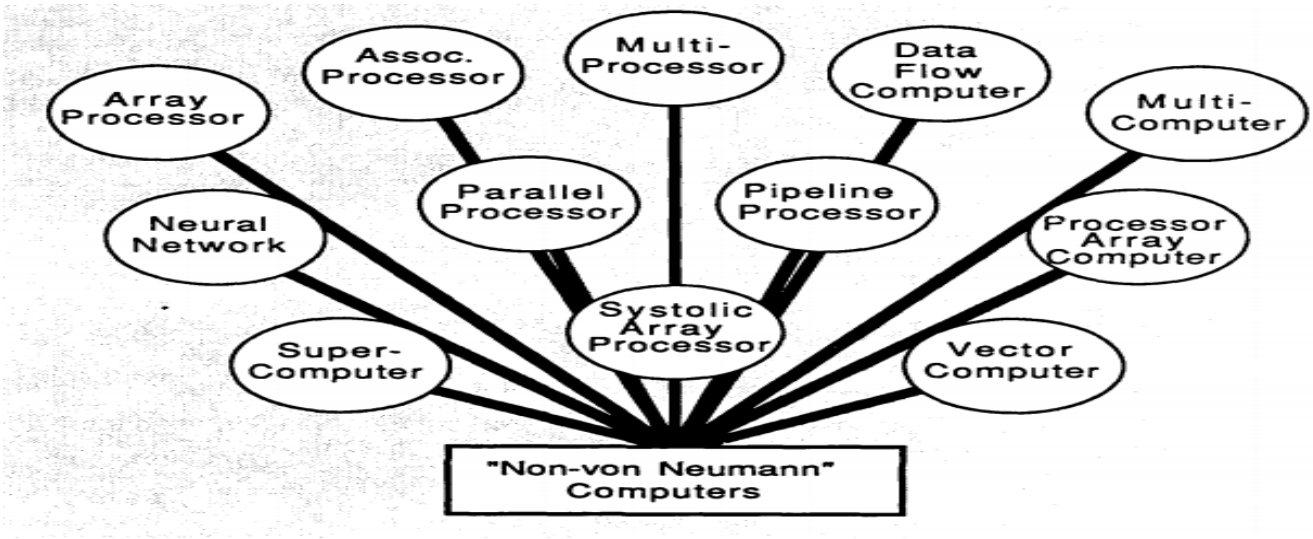
- o A main memory, which stores both data and instructions.
- o An arithmetic-logical unit (ALU) capable of operating on binary data.

- A control unit, which interprets the instructions in memory and causes them to be executed.
- Input and output (I/O) equipment operated by the control unit.

Non NON-VON-NEUMANN MACHINES:

- Conventional stored-program computers have undergone many incremental improvements over the years.
- These improvements include adding specialized buses, floating-point units, and cache memories etc.
- But enormous improvements in computational power require departure from the classic von Neumann architecture.
- Adding processors is one approach.

The term non-von Neumann computers refers to the class of machines that are not based upon the architecture of a sequential machine. A non von Neumann machine may thus be without the concept of sequential flow of control (i.e. without any register corresponding to a “program counter” that indicates the current point that has been reached in execution of a program) and/or without the concept of a variable (i.e. without “named” storage locations in which a value may be stored and subsequently referenced or changed).



"Non-von Neumann" Computers

RISC **(Reduced Instruction Set Computer)**:

- Microprocessor architecture.
- Designed to perform a set of smaller computer instructions so that it can operate at higher speeds.
- It is also called hard-wired approach
- Small, highly optimized set of instructions
- Uses a load-store architecture
- Short execution time
- Pipelining
- Many registers

Examples of RISC processors:

- IBM RS6000, MC88100.
- DEC's Alpha 21064, 21164 and 21264 processors

Features of RISC Processors:

The standard features of RISC processors are listed below:

- RISC processors use a small and limited number of instructions.
- RISC machines mostly uses hardwired control unit.
- RISC processors consume less power and are having high performance.
- Each instruction is very simple and consistent.
- RISC processors use simple addressing modes.
- RISC instruction is of uniform fixed length.

CISC **(Complex Instruction Set Computer):**

A complex instruction set computer (CISC) is a microprocessor instruction set architecture (ISA) in which each instruction can execute several low-level operations, such as a load from memory, an arithmetic operation, and a memory store, all in a single instruction.

Examples of CISC processors are:

- Intel 386, 486, Pentium, Pentium Pro, Pentium II, Pentium III
- Motorola's 68000, 68020, 68040, etc.

Features of CISC Processors:

The standard features of CISC processors are listed below:

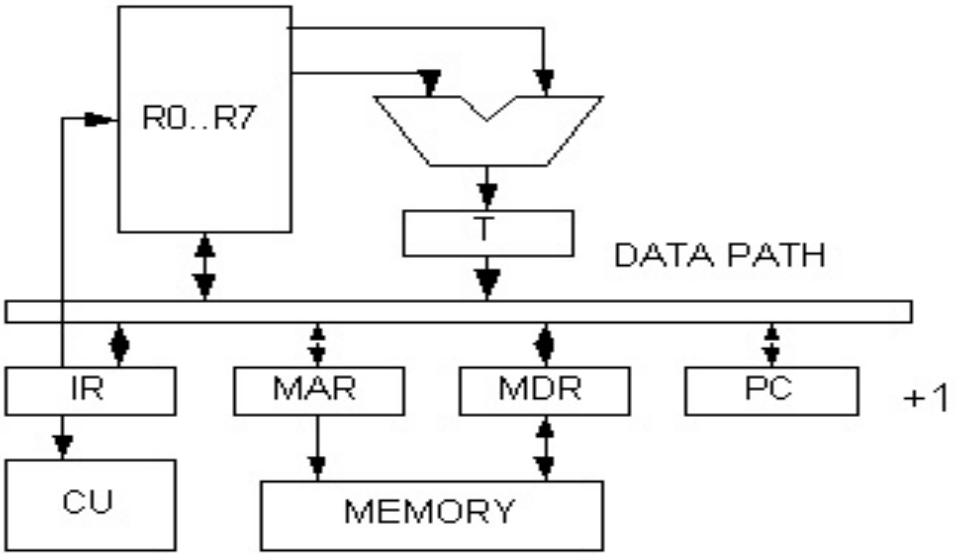
- CISC chips have a large amount of different and complex instructions.
- CISC machines generally make use of complex addressing modes.
- Different machine programs can be executed on CISC machine.
- CISC machines uses micro-program control unit.
- CISC processors are having limited number of registers.

Difference between CISC and RISC

CISC	RISC
▪ Primary goal is to complete a task in as few lines of assembly as possible ▪ Emphasis on hardware ▪ Includes multi-clock complex instructions ▪ Memory-to-memory: "LOAD" and "STORE" incorporated in instructions ▪ Difficult to apply pipelining. ▪ Small code sizes, high cycles per second	▪ Primary goal is to speedup individual instruction ▪ Emphasis on software ▪ Single-clock, reduced instruction only ▪ Register to register: "LOAD" and "STORE" are independent instructions ▪ Easy to apply pipelining. ▪ Low cycles per second, large code sizes

Designing a hypothetical CPU:

Consider a simple hypothetical CPU which contains all the important elements of a real processor. The architectural description of a simple hypothetical CPU is its organization (structure), its instruction set (ISA) and its behavior.



Instruction set design:

An instruction set is a collection of all instructions of a CPU. Therefore, if we define all the instructions of a computer, we can say we have defined the instruction set. Each instruction consists of several elements. An instruction element is a unit of information required by the CPU for execution.

Elements of an instruction:

An instruction has the following elements.

- An operation code also termed as OP code which specifies the operation to be performed.
- A reference to the operands on which data processing is to be performed.
- A reference to the operands which may store results of data processing operation performed by the instruction.
- A reference for the next instruction to be fetched and executed.

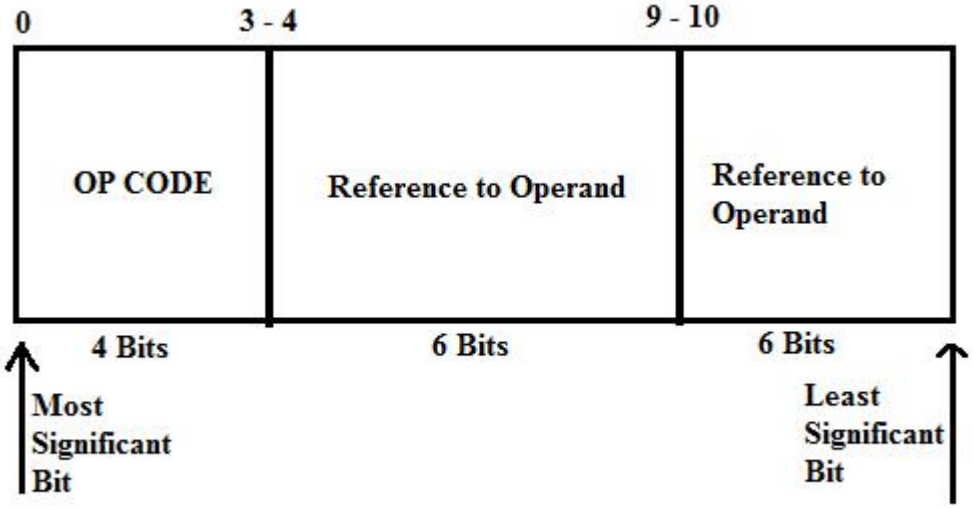
The next instruction which is to be executed is normally the next instruction in the memory. Therefore, no explicit reference to the next instruction is provided.

Instruction set design

- ✓ address space 1024 words
- ✓ load, store architecture
- ✓ 16 bit word, fixed length
- ✓ 8 registers, R7 as stack pointer
- ✓ condition code Z,S zero, sign

How an instruction is represented:

Instructions are represented as sequence of bits. An instruction is divided into number of fields. Each of these fields corresponds to a constituent element of instructions. A layout of instruction is termed as instruction format i.e. following is the instruction format for IAS computers. It uses four bits for OP code and only two operand references are provided. Here, no explicit reference is provided for the next instruction to be executed.



In most instruction sets, many instruction formats are used. An instruction first reads into an instruction register (IR), and then the CPU decodes the instruction and extracts the required operand on the basis of reference mode through the instruction fields and processes it.

Types of instructions:

The instructions can be categorized under the following categories.

1. Data processing instruction: These instructions are used for arithmetic and logic operation in a machine. Examples of data processing instructions are: arithmetic, Boolean, Shift, Character and string processing instructions, stacks and registers, manipulation instructions, vector instructions etc.

2. Memory/ Register referenced instructions (Data storage / Retrieval Instructions): Since the data instruction processing operations are normally performed on the data storage in CPU registers. Thus we need an instruction to bring data to and from memory to register. These are called data storage/ retrieval instructions. Example of data storage and retrieval instructions is load and store instructions.

3. Input output instructions (Data movement instruction): These are basically input output instructions. These are required to bring in programs and data from various devices to memory or to communicate results to the input output devices. Some of these instructions can be: start input / output, Halt input/ output, test input/ output, etc.

4. Control Instructions: These instructions are used for testing the status of computation through processor status word (PSW). Another of such instruction is the branch instruction used for transfer of control.

5. Miscellaneous Instructions: The instruction does not fit in any of the above categories.

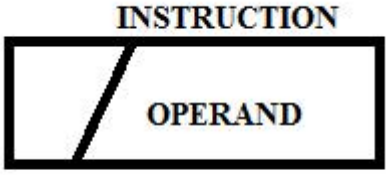
Some of these instructions are:

- Interrupts or supervisory call, swapping return from interruptions, halt instructions or some privileged instructions of operating system.

Addressing techniques:

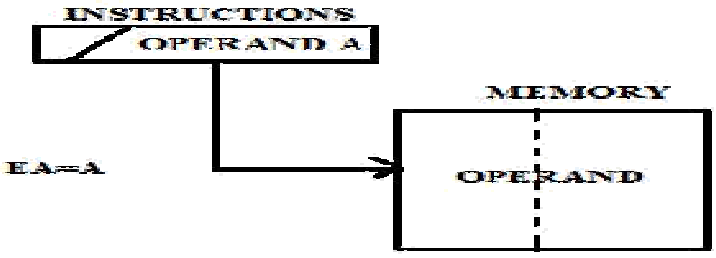
It is the way of specifying the operand part of instruction.

1. **Immediate Addressing Mode:** The simplest form of addressing is immediate addressing in which the operand is actually present in the instructions I.e. data is contained in the instructions itself (I.e. Operand =A).



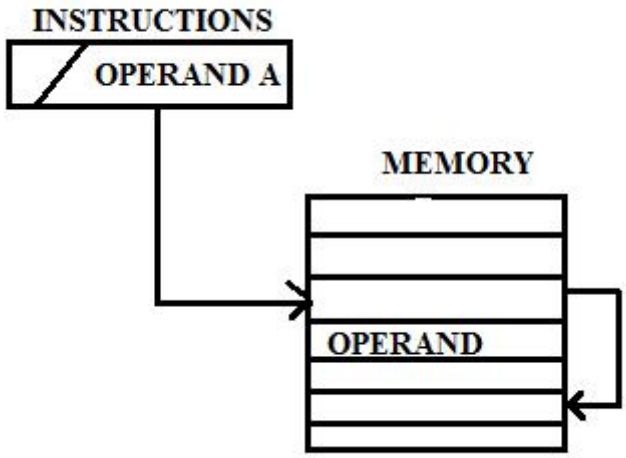
This mode can be used to define and use constants or set initial values of variables. The advantages of immediate addressing is that no memory reference other than the instruction fetch is required to obtain the operand, thus saving one memory or cache cycle. This advantage is that the size of the number is restricted to the size of the address field which will determine the maximum magnitude of the data. Here no reference is used.

2. **Direct Addressing Mode:** A very simple form of addressing is direct addressing, in which the address field contains the effective address of the operand. In this mode the address of the data is supplied with the instruction I.e. EA=A.

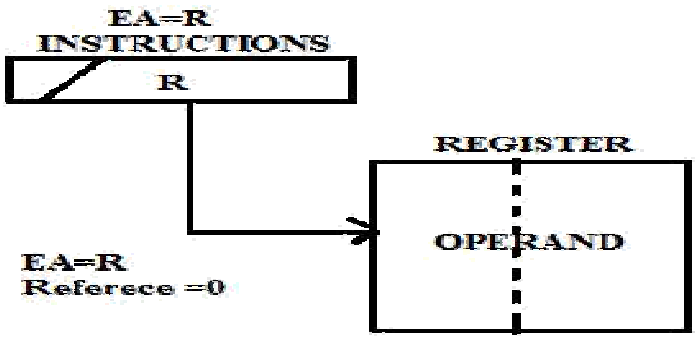


In this mode addressing domain would be limited. Here the memory reference is 1.

3. **Indirect Addressing Mode**: Instructions contain the address of memory locations which in turn refers to the data. The problem with direct addressing is that the length of the address fields is usually less than the word length, thus limited the address range. One solution is to have the address field refer to the address of the word memory, which in turn contains a full length address of the operand. This is known as indirect addressing I.e. EA= (A). Here two reference variables are used.

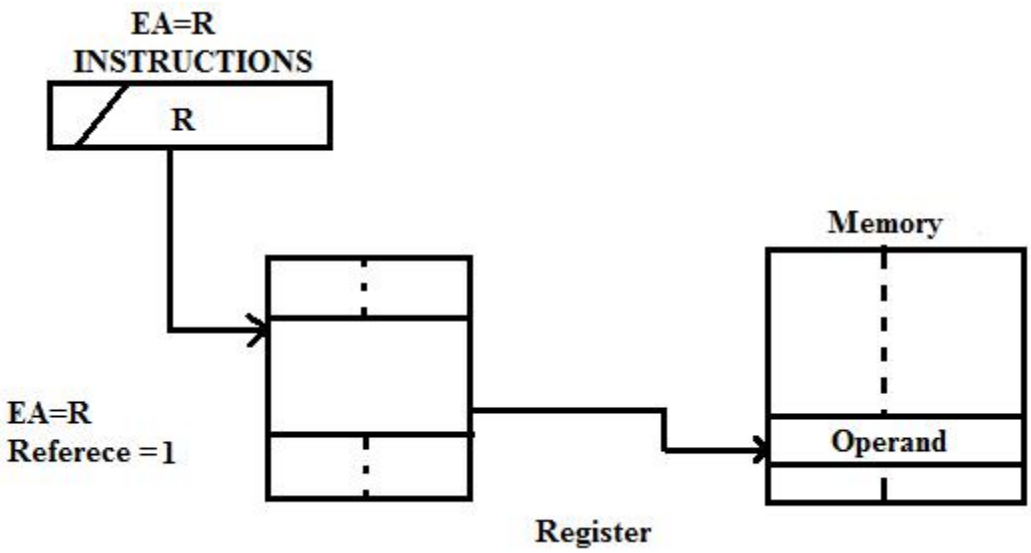


4. **Register addressing mode:** Register addressing is similar to direct addressing mode. The only difference is that the address field refers to a register rather than a main memory address EA=R.



In this mode the data present in the register is supplied with the instruction. Here no reference variables are used.

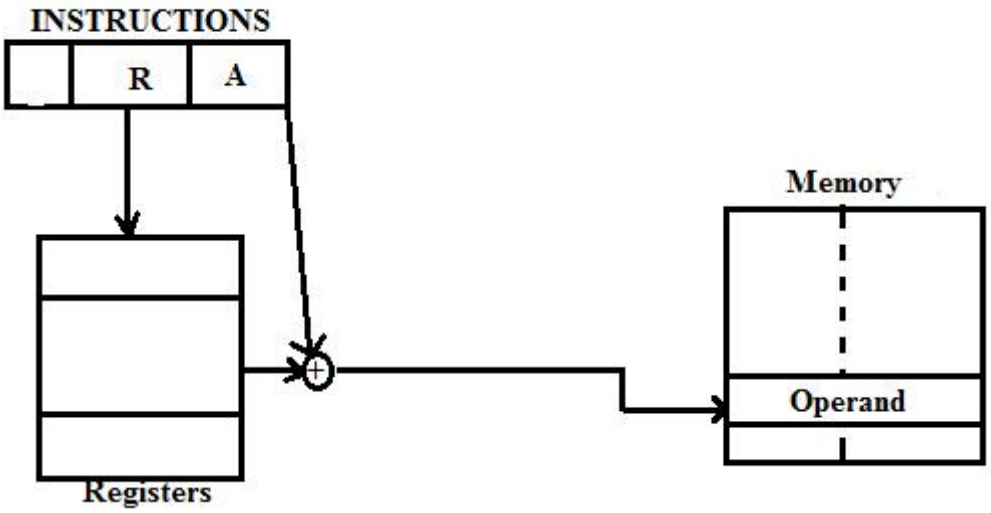
5. **Register Indirect Addressing Mode:** Register addressing mode is analogous to direct addressing mode similarly register indirect addressing mode is analogous to the indirect addressing mode. In both cases the only difference is whether the address field refers to a memory location in a register. Thus for register indirect addressing mode. EA=(R).



The advantage of register indirect addressing mode over direct addressing mode is that it uses one less memory reference than indirect addressing mode. Other advantages are same as that of indirect addressing mode. On this mode registers mentioned contain the address of memory location contain the data.

6. **Displacement Addressing Mode:** A very powerful mode of addressing combines the capability of direct addressing and register indirect addressing. It is known as a variety of names depending upon the content of its use, but the basic mechanism is the same. We will refer to this as displacement addressing. EA=A+(R).

   Displacement addressing requires that the instruction have two address fields, at least one of which is explicit. The value contained in one address field (Value=A) is used directly.



The other address field or an implicit reference based on OP code refers to a register whose contents are added to A to produce the effective address.

ALU/Data path design:

- **ALU ORGANIZATION:** An ALU is that part of the computer that mainly performs arithmetic, logic + shift operations. All the other components control unit, memory registers. Input output is there mainly to bring the data into ALU to process it and then take the result back out. The complexity of ALU depends on the type of instructions set that has been realized for it. The simple ALU can be constructed for fixed point numbers. On the other hand floating point arithmetic implementation requires more complex control logic and data processing capabilities i.e. the hardware for floating point arithmetic or other complex functions, an auxiliary special purpose unit is used. This unit is called arithmetic co-processor.

What is ALU? An arithmetic logic unit (ALU) represents the fundamental building block of the central processing unit of a computer. An arithmetic logic unit (ALU) is a digital circuit used to perform arithmetic and logic operations. Modern CPUs contain very powerful and complex ALUs. In addition to ALUs, modern CPUs contain a control unit (CU).

Most of the operations of a CPU are performed by one or more ALUs, which load data from input registers. A register is a small amount of storage available as part of a CPU. The control unit tells the ALU what operation to perform on that data and the ALU stores the result in an output register. The control unit moves the data between these registers, the ALU, and memory.

Control unit Design: The memory, arithmetic and logic unit, input and output unit's store and process information and perform input and output operations. The operation of these units must be coordinated in same way. This is the task of the control unit. The control unit is effectively the nerve center that sends control signals to other units and senses their states.

I/O transfers consisting of input and output operations and are controlled by the instructions of I/O programs that identify the devices involved and the information to be transferred. However the actual timing signal that lowers the transfer are generated by the control circuits. Timing signals are signals that determine when a given action is to take place. Data transfers between the processor and memory and are also controlled by the control unit through timing signals. It is reasonable to think of a control unit as a well-defined physical separate unit that interacts with other parts of the machine.

Much of the control circuitry is physically distributed through the machine. A large set of control line wires caries the signals used for timing and synchronization of events in all units. The operation of a computer can be summarized as follows:

- The computer accepts information in the form of programs and data through an input unit and stores in the memory.
- Information stored in the memory is fetched under program control into an arithmetic and logic unit, where it is processed.
- Processed information leaves the computer through an output unit.
- All activities inside the machine are directed by the control unit.

1. STRUCTURE OF CONTROL UNIT: A control unit has set of input values on the basis of which it produces an output control signal which in turn performs micro-operation. The output signal controls the execution of a program.

The input of the control unit is as:

- **The master clock signal:** The signal causes micro-operation to be performed in a single clock cycle with a single or a set of simultaneously micro-operations. The time taken in performing a single micro-operation is also termed as Processor cycle time in some machines. However, some micro-operations such as memory read may require more than one click cycle if it is greater than 1.

- **The Instruction Register:** The operation code (OP code) which normally include addressing mode bits of the instruction helps determining the various cycles to be performed an hence determines the related micro-operations which are needed to be performed.

- **Flags:** Flags are used by control unit for determining the status of the CPU.In case the zero flag is set the control unit will issue control signals which will cause program counter (PC) to be incremented by one.

- **Control Signals from Control Bus:** Some of the control signals are provided to the control unit through the control bus. These signals are issued from outside the CPU. Some of these signals interrupt and acknowledge. On the basis of input signals the control unit activates certain output control signals which in turn are responsible for the execution of an instruction.

These output control signals are:

- ✓ **Control signals which are required within CPU:** These control signals causes two types of micro-operation viz for data transfer form one register to another and for performing an ALU operation using input and output registers.

- ✓ **Control signals to control bus:** The basic purpose of these control signals is to bring or to transfer data from CPU register to memory or I/O modules. The control signal is issued to the control bus to activate the data bus.

2. **Control Unit Organization:** The basic responsibilities of the control unit is to control:
   - ✓ Data exchange of CPU with the memory or I/O modules.
   - ✓ Internal operation in the CPU such as:
     1. Moving data between registers (register transfer operation).
     2. Making ALU to perform a particular operation on the data.
     3. Regulating other internal operation.

3. Functional Requirements of a Control Unit: A control unit must know about the:

O. Basic components of the CPU.

1. Micro operation CPU performs.
2. CPU of a computer consists of the following basic functional components.
   1. The arithmetic logic unit (ALU) which performs the basic arithmetic and logical operations.
   2. Registers which are used for information storage within the CPU.

3. Internal data paths: these paths are useful for moving the data between two registers or between a register and ALU.
4. External data paths: the role of these data paths is normally to link the CPU registers with the memory or I/O modules. This role is normally fulfilled by the system bus.
5. The control unit which causes all the operations to happen in the CPU.

3. The micro-operations performed by the CPU can be classified as:
    1. Micro-operations for register to external interface i.e. (in most cases system bus data transfer).
    2. Micro-operations for external interface to register data transfer.
    3. Micro-operations for performing arithmetic and logic operations. These micro-operations involve use of registers for input and output.
4. The basic responsibilities of the control unit lie in the fact that the control unit must be able to guide the various components of CPU to perform a specific sequence of micro operation to achieve the execution of an instruction.

Thus, the control unit must perform two basic functions:

1. Cause the execution of a micro-operation.
2. Enable the CPU to execute a proper sequence of micro-operation which is determined by the instruction to be executed.

Hardwired and Micro-programmed design approaches.

1. **Hardwired Control Unit:**

A hardwired control unit is implemented as logic circuits in the hardware. The inputs to control unit are: the instruction registers, flags, timing signals and control bus signals. On the basis of these inputs the output signal sequences are generated. Output control signals are functions of

inputs. Thus, we can derive a logical function for each control unit. The implementation of all the combinational circuits may be very difficult. Therefore a new approach microprogramming was used.

2. **Micro-programmed Control Unit:** The hardwired control unit lack flexibility in design. In addition it is quite difficult to design, test and implement as in many computers the number of control lines is in hundreds. Micro-program will consist of instruction, with each of the instruction describing.
    1. One or more micro-operations to be executed.
    2. The information about the micro-instruction to be executed next.

   Such an instruction is termed as microinstruction and such a program is termed as a micro-program or firmware. The firmware is a midway between hardware and software. Firmware in comparison to hardware is easier to design, whereas in comparison to software is difficult to write. A control unit provides a set of control signal lines, distributed throughout the CPU, with each of the lines representing a zero or one. Therefore a microinstruction is made responsible for generating control signals for desired control lines to implement a desired micro-operation eg. to implement a register to register transfer operation, output of the source register and input of destination register need to be enabled by the respective control signal line via a microinstruction. Thus, each microinstruction can generate a set of control signals on the control lines which in turn implement one or more micro-operation. A set of control signals with each bit representing a single control line is called a control word. Thus, a microinstruction can cause execution of one or more micro-operations and a sequence of microinstructions that is micro-program can cause execution of an instruction. The microprograms are mostly stored in read only memory, known as control store or control memory as alteration in control store is generally not required after the production of the control

unit. In RAM such a memory instruction set of a computer can be changed by simply modifying the microprograms for various op-codes. Such a computer can be tailored for specific applications on the basis of microprograms. A computer having writable control memory are termed as "dynamically micro programmable" as the content of control memory for such computers can be changed under program control. The control memory is a word organized unit with each word representing a microinstruction. The computers which have micro programmed control unit have two separate memories: a main memory and the control memory.

Advantages of Micro Programmed Control Unit: since the microprograms can be changed relatively easily. Therefore micro-programmed control unit are very flexible in comparison to hardwired control unit.

Disadvantage:

1. Hardware cost is more because of the control memory and its access circuitry.
2. This is slower than hardwired control unit because the microinstruction is to be fetched from the control memory which is time consuming.

Reference Books:

1. "Computer Organization and Design: The Hardware/Software Interface", David A. Patterson and John L. Hennessy: Elsevier.
2. "Computer Organization and Architecture: Designing for Performance", William Stallings: Pearson Education.
3. "Computer Systems Design and Architecture", Vincent P. Heuring and Harry F. Jordan: Pearson Education. !

Online References:

1. http://www.encyclopedia.com/computing/dictionaries-thesauruses-pictures-and-press-releases/non-von-neumann-architecture
2. http://www.dtic.mil/dtic/tr/fulltext/u2/a207268.pdf
3. https://www.cp.eng.chula.ac.th/~piak/teaching/ca/s1.htm