Computer Structure and Computer Function

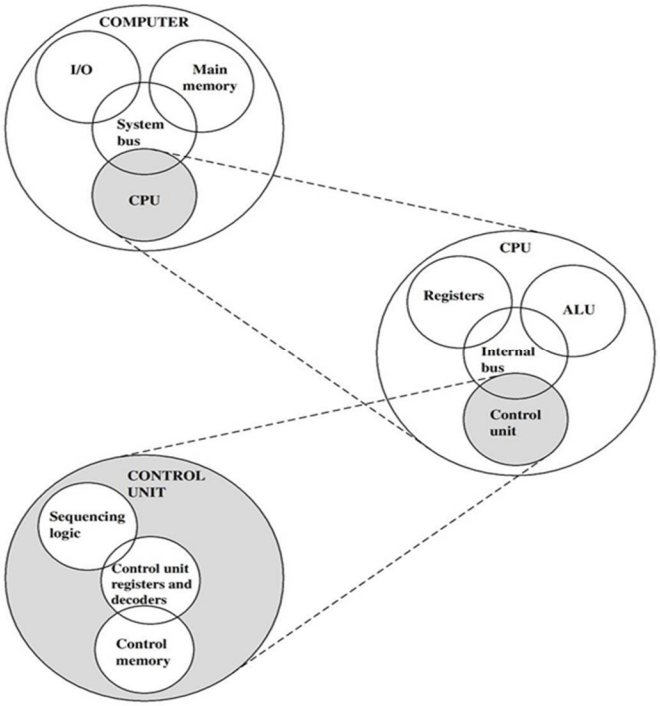
Contents of Lecture:

- Computer Components
- Computer Function
  - ✓ Instruction Fetch and Execute

References for This Lecture:

✓ William Stallings, Computer Organization and Architecture Designing For Performance, 9th Edition, Chapter 3 : *A Top-Level View of Computer Function and Interconnection* , pages 65 to 74

Computer Components:

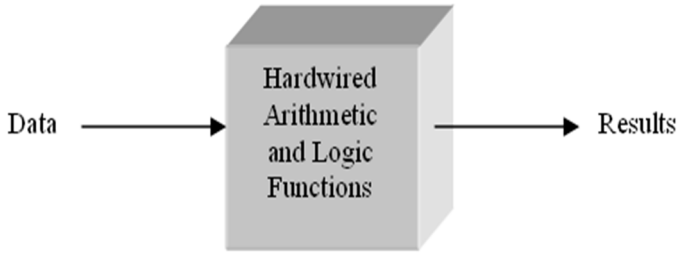


- Computer designs are based on concepts developed by **John von Neumann** at the Institute for Advanced Studies, Princeton. Such a design is referred to as the **von Neumann architecture (Top-Level)** and is based on three key concepts:

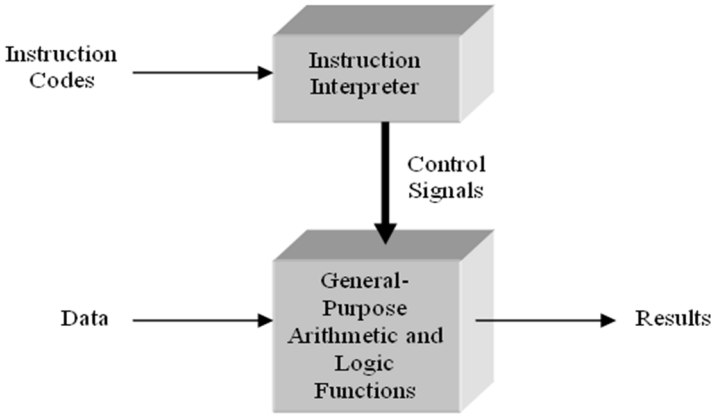
  - ✓ **Data** and **instructions** are stored in a single read–write memory.
  - ✓ The contents of this **memory are addressable by location**, without regard to the type of data contained there.
  - ✓ **Execution** occurs in a sequential fashion (unless explicitly modified) from one instruction to the next.

❖ **The reasoning behind these concepts are:**

- ✓ There is a small set of basic logic components that can be combined in various ways to store binary data and to perform arithmetic and logical operations on that data.
- ✓ If there is a particular computation to be performed, a configuration of logic components designed specifically for that computation could be constructed.
- ✓ The process of connecting the various components in the desired configuration as a form of programming. The resulting "program" is in the form of hardware and is termed **a hardwired program**.
- ✓ In the case of customized hardware, the system accepts **data** and produces **results** at the following Figure
- ✓ Suppose we construct a general-purpose configuration of arithmetic and logic functions. This set of hardware will perform various functions on data depending on control signals applied to the hardware.
- ✓ With general-purpose hardware, the system accepts data and control signals and produces results. The entire program is actually a sequence of steps. This new method of programming, a sequence of codes or instructions is called **software**.



(a) Programming in Hardware



(b) Programming in Software

- Figure(b) indicates two major components of the system:
  - ✓ **An instruction interpreter**
  - ✓ And a module of **general-purpose arithmetic and logic functions.** These two constitute the CPU.

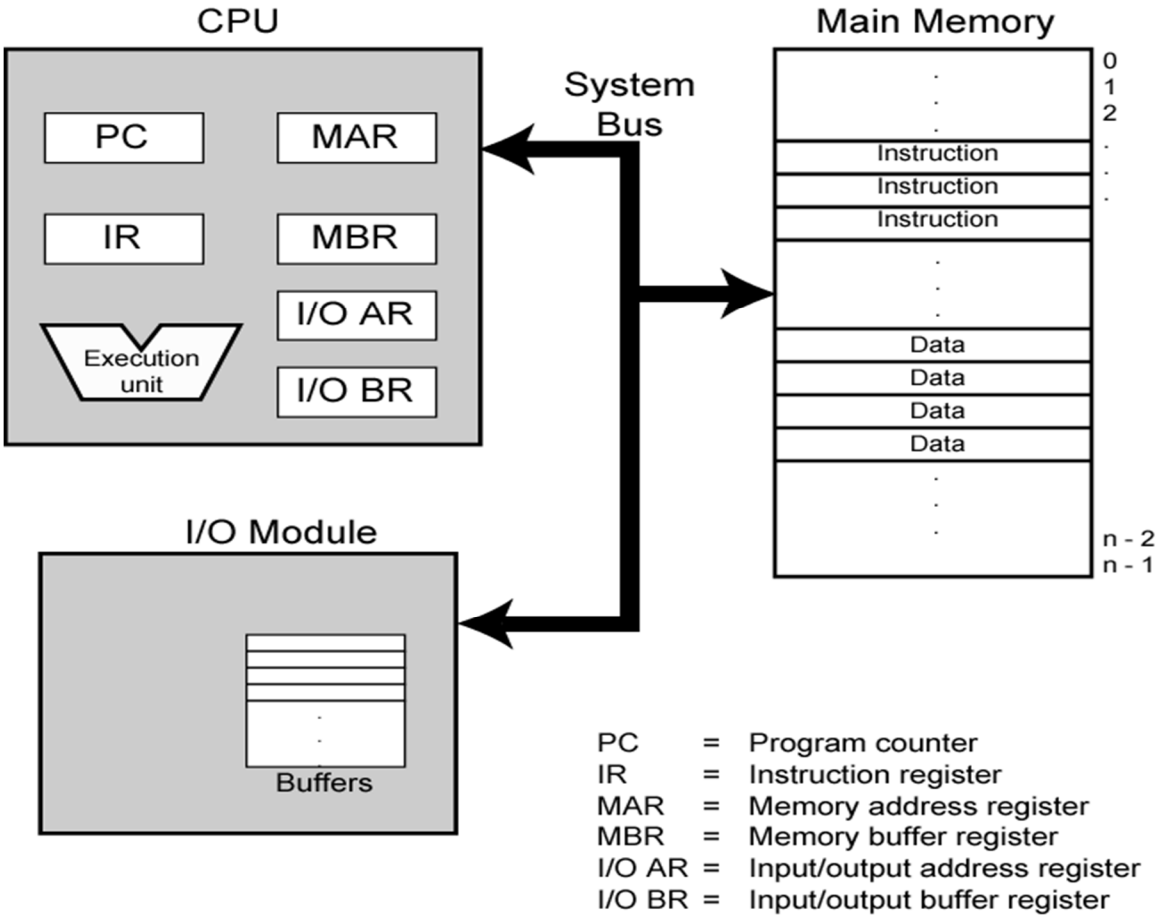
- This module contains basic components for accepting data and instructions in some form and converting them into an internal form of signals usable by the system. A means of reporting results is needed, and this is in the form of an output module; these are referred as ***I/O components.***

- An input device will bring instructions and data in sequentially. But a program is not invariably executed sequentially; it may jump around, operations on data may require access to more than just one element at a time in a predetermined sequence.

- Thus, there must be a place to store temporarily both instructions and data. That module is called ***memory***, or ***main memory***

- Von Neumann pointed out that the same memory could be used to store both instructions and data.

Computer components (Top-Level View):



- **Memory module or Main Memory:** consists of a set of locations, defined by sequentially numbered addresses. Each location contains a binary number that can be interpreted as either an instruction or data.

- **I/O module**: transfers data from external devices to CPU and memory, and vice versa. It contains internal buffers for temporarily holding these data until they can be sent on.

- **The CPU:** exchanges data with memory.
- **System bus**: Interconnection structures.

- **Registers:-**
  - ✓ Memory addresses register **(MAR),** which specifies the address in memory for the next read or write.
  - ✓ Memory buffer register **(MBR),** which contains the data to be written into memory or receives the data read from memory.
  - ✓ I/O address register **(I/OAR),** specifies a particular I/O device.
  - ✓ I/O buffer **(I/OBR)** register, is used for the exchange of data between an I/O module and the CPU.
  - ✓ Program counter **(PC)**, holds the address of the instruction to be fetched next.
  - ✓ Instructions register **(IR),** the fetched instruction is loaded into a register in the processor.
  - ✓ Accumulator **(AC),** the processor contains a single data register to store the result.

Computer function:

- The basic function performed by a computer is execution of a program, which consists of a set of instructions stored in memory.

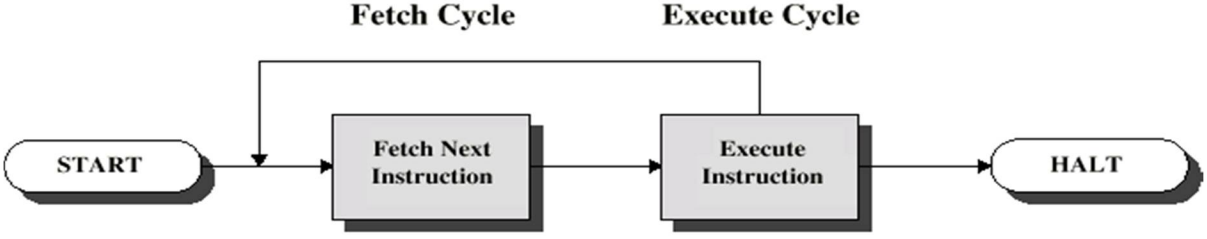
- In simplest form instruction processing consists of two steps:
  - ✓ The processor reads (**fetches**) instructions from memory one at a time and executes each instruction.
  - ✓ Program execution consists of repeating the process of instruction fetch and instruction execution. The instruction execution may involve several operations and depends on the nature of the instruction

- The processing required for a single instruction is called an **instruction cycle**.

- The two steps are referred to as the **fetch cycle** and the **execute cycle**.

- Program execution halts only if the machine is turned off, some sort of unrecoverable error occurs, or a program instruction that halts the computer is encountered.

- The Basic Instruction Cycle in the next Figure.

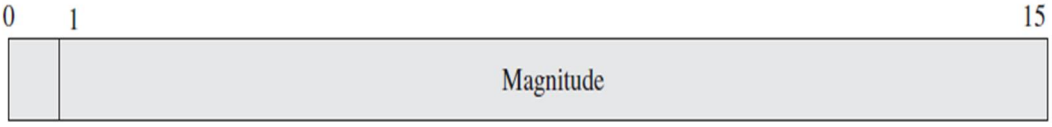


❖ **Fetch cycle:-**
- ✓ Program Counter (PC) holds address of next instruction to fetch.
- ✓ Processor fetches instruction from memory location pointed to by PC.
- ✓ Increment PC (unless told otherwise).
- ✓ Instruction loaded into Instruction Register (IR).
- ✓ Processor interprets instruction and performs required actions.

❖ **Execute cycle:-**
- ✓ **Processor-memory:** Data may be transferred from processor to memory or from memory to processor.
- ✓ **Processor-I/O:** Data may be transferred to or from a peripheral device by transferring between the processor and an I/O module.
- ✓ **Data processing:** The processor may perform some arithmetic or logic operation on data.
- ✓ **Control:** An instruction may specify that the sequence of execution be altered. *(An instruction's execution may involve a combination of these actions).*
- ✓ **Combination of above**: An instruction's execution may involve a combination of these actions.

❖ **Example:-**
- ✓ Consider a simple example using a hypothetical machine that includes the characteristics listed in following Figure.
- ✓ The processor contains a single data register, called an accumulator (AC).
- ✓ Both instructions and data are 16 bits long. Thus, it is convenient to organize memory using 16 -bit words.

(a) Instruction format



(b) Integer format

❖ **How many Opcodes can be representing in 4 bits?**
❖ **What is the size of memory that can be directly accessed?**

❖ **Solve:**

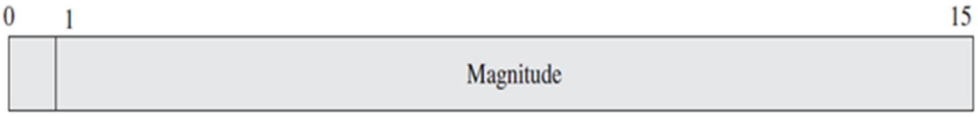
(a) Instruction Format:

- ✓ The instruction format provides 4 bits for the opcode, so that there can be as many as 2^4 =16 different opcodes, and up to 2^{12} = 4096 (4K) words of memory can be directly addressed.

(a) Instruction format

- Opcodes=2^4=16
- Memory= 2^{12}=4096 words = 4K words
  —directly addressed

(b) Data Representation (Integer Format):



(b) Integer format

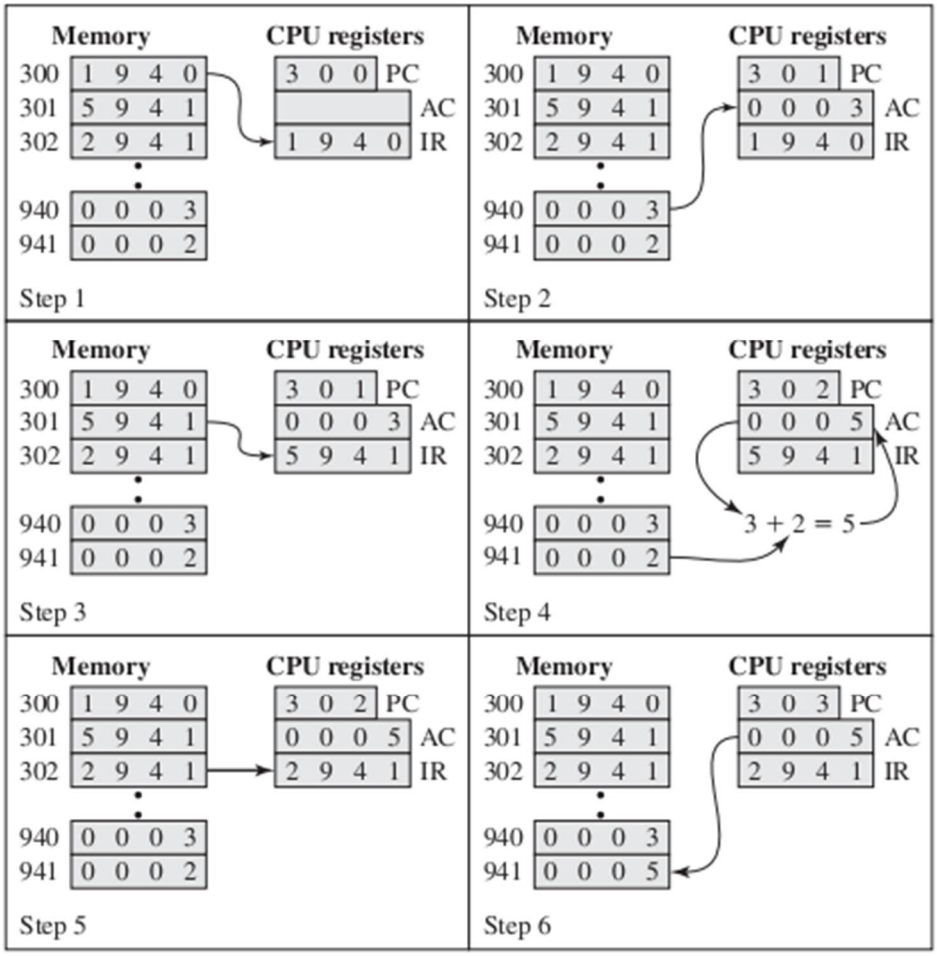
(c) Internal CPU registers:

Program counter (PC) = Address of instruction
Instruction register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

(d) Partial list of opcodes:

0001 = Load AC from memory
0010 =Store AC to memory
0101= Add to AC from memory

- Illustrates a partial program execution, showing the relevant portions of memory and processor registers.
- The program fragment shown adds the contents of the memory word at address 940 to the contents of the memory word at address 941 and stores the result in the latter location.



Example of Program Execution (contents of memory and registers in hexadecimal)

- Three instructions, which can be described as three fetch and three execute cycles, are required:
  1- The PC contains 300, the address of the first instruction. This instruction (the value 1940 in hexadecimal) is loaded into the instruction register IR and the PC is incremented.
     - ☒ ***Note that*** *this process involves the use of a memory address register (MAR) and a memory buffer register (MBR). For simplicity, these intermediate registers are ignored.*

  2- The first 4 bits (first hexadecimal digit) in the IR indicate that the AC is to be loaded. The remaining 12 bits (three hexadecimal digits) specify the address (940) from which data are to be loaded.

  3- The next instruction (5941) is fetched from location 301 and the PC is incremented.
  4- The old contents of the AC and the contents of location 941 are added and the result is stored in the AC.

  5- The next instruction (2941) is fetched from location 302 and the PC is incremented.
  6- The contents of the AC are stored in location 941.