Why We Study This Course?

- To give a basis understanding of computer operations, roles of processors, main memory, and input/output devices.
- Understanding the concept of programs as sequences of machine instructions.
- Understanding the relationship between assembly language and machine language.
- Understanding arithmetic and logical operations with integer operands, floating-point number systems and operations.
- Understanding memory organization, including cache structures and virtual memory schemes.

Computer Organization &Architecture

- **Computer Architecture :** is a set of disciplines that describes a computer system by specifying its parts and their relations.
- Visible to a programmer
- Have a direct impact on the logical execution of a program.
- Computer Architecture refers to attributes as:
  - ❑ Instruction set
  - ❑ the number of bit to represent various data types
  - ❑ I/O mechanisms
  - ❑ Memory addressing techniques

Computer Organization &Architecture

- **Computer Organization :** is the operational units and their interconnection that realize the architecture specification.
- Helps in optimizing computer programs performance
  - e.g. : software engineers need to know the processing ability of processors to optimize software in order to gain the best performance at the least expense. Which require quite detailed analysis of computer organization.
- You can change the organization of a computer without changing its architecture.

  E.G. → All Intel x86 family shares the same basic architecture but the organization differs.

Computer Architecture VS. Computer Organization

Computer's Architecture	Computer's Organization
is the abstract model and is the programmer's view in terms of instructions, addressing modes and registers.	expresses the realization of the architecture
describes what the computer does	describes how it does it
indicates computer hardware	Reveals computer performance
architecture is fixed first when designing computers	Is decided after the architecture is fixed

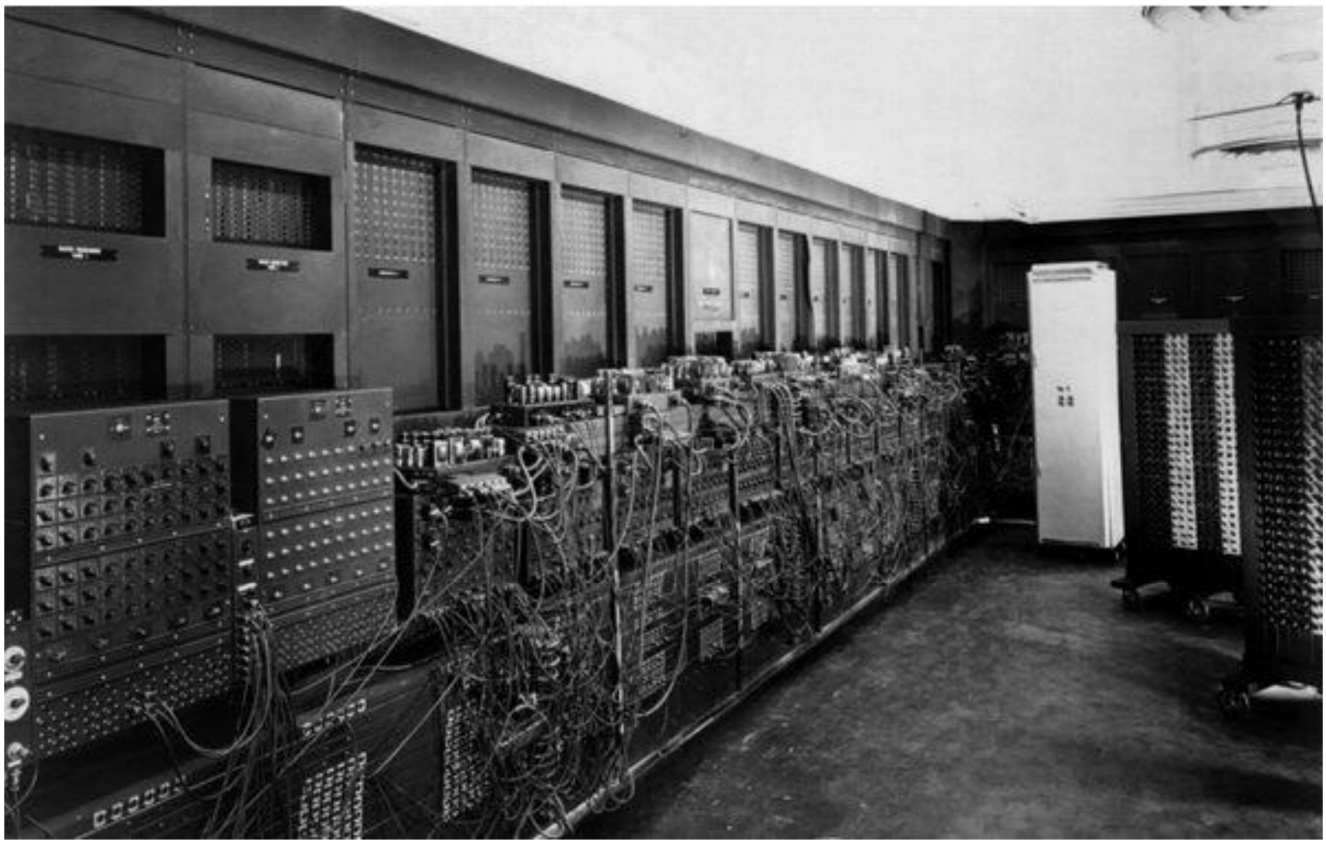
Historical Overview Of Computer Architecture

- The Von Neumann Architecture
- Transistors
- Integrated Circuits
- Microprocessors
- Artificial Intelligence Based

The Von Neumann Architecture

- UNIVAC and ENIAC computers were examples of first-generation computing devices.
  - Addition and subtraction were performed with 20 accumulators.
    - Temporary data produced and needed during computation could be stored in the accumulators or punched out and later reintroduced.
  - Multiplier
  - Divider,
  - Square root unit.
  - Input and output was given in the form of punch cards.
  - An electronic memory for storing
    - tabular functions
    - numerical constants.

UNIVAC



ENIAC

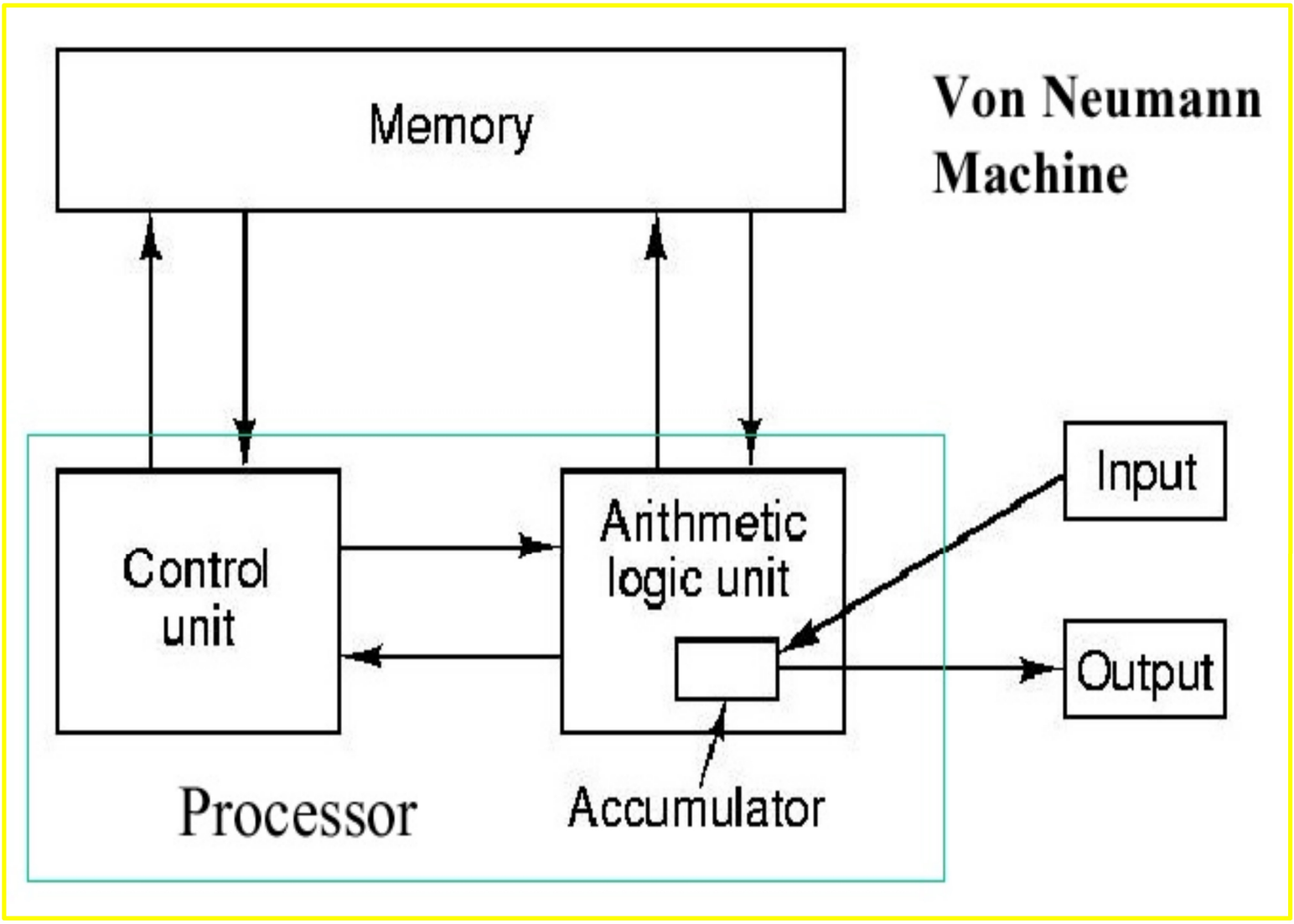
The Von Neumann Architecture

- Von Neumann learned about ENIAC in 1944 and became a consultant to its design team.
- His primary interest in this project was the logical structure and mathematical description of the new technology in contrast of the engineering view if its creators.
- The difference in point of view was the motive for Von Neumann to leave ENIAC and participate in the Development of EDVAC .

EDVAC

The Von Neumann Architecture

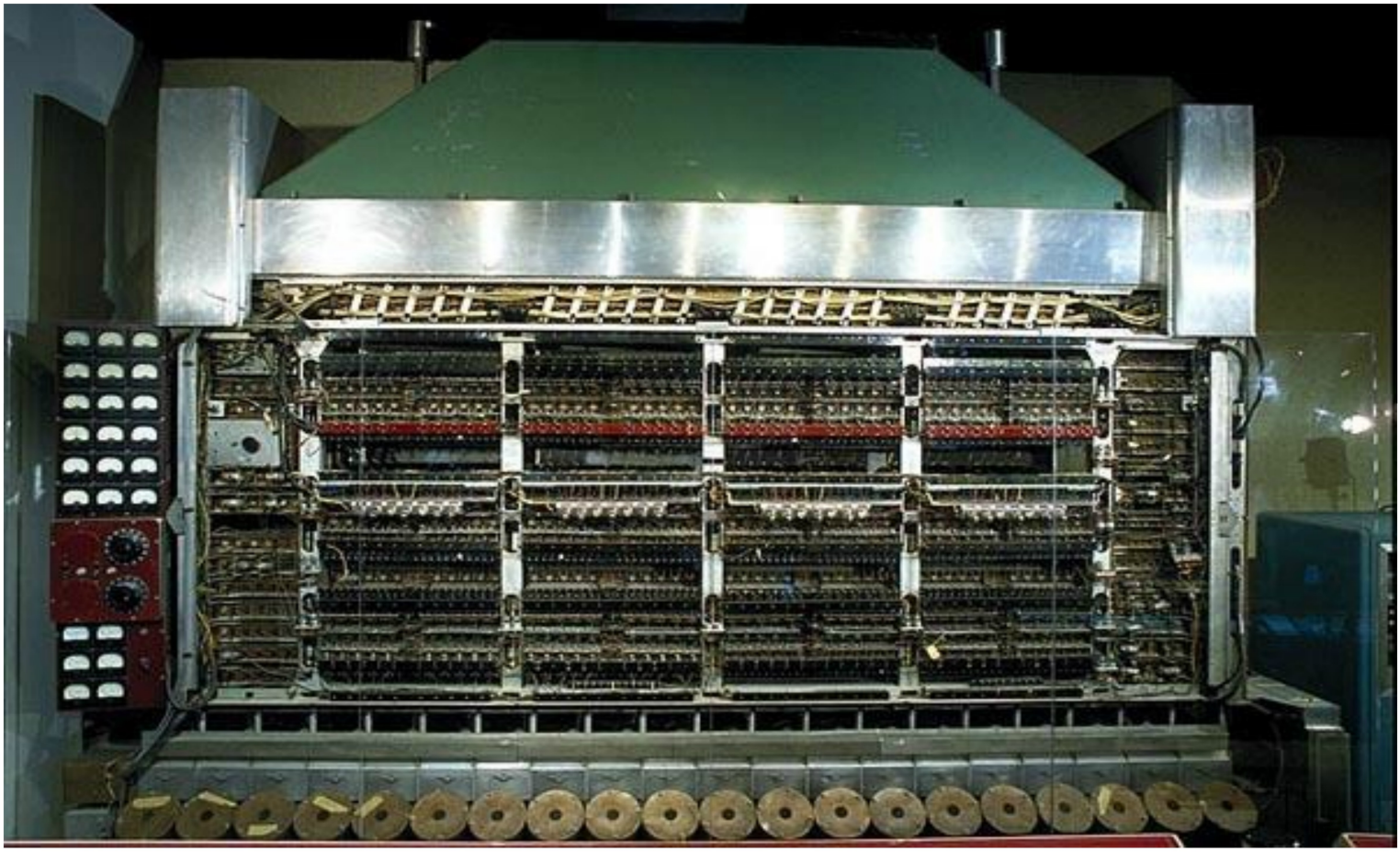
- In 1945, von Neumann wrote the paper "First Draft of a Report on the EDVAC", the report organized the computer system into four main parts:
  - Central Arithmetical unit (CA)
    - Four basic arithmetic operations (ADD , SUB , MULTPLY & DIVIDE)
    - roots, logarithms, trigonometric functions, and their inverses
  - Central Control unit (CU) : control the proper sequencing of operations and make the individual units act together to carry out the specific task programmed into the system.
  - Memory (M):store numerical data (variables, constant,….) and instructions
  - Input/Output devices (I/O): the interface between computer and user.


Von Neumann Machine
Memory
Control unit
Arithmetic logic unit
Input
Output
Processor
Accumulator

The Von Neumann Architecture

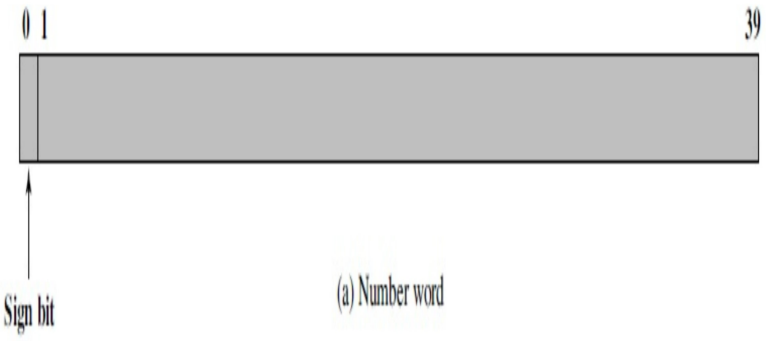
- Manufacturers went from building an EDVAC clone to building an EDSAC clone to building an IAS clone all on the basis of Von Neumann's ideas.
- IAS was built from late 1945 until 1951 under the directions of Von Neumann and based on his architecture, even though it was both originated and implemented by others.
- IAS can be called the father of all modern computers as it was one of the first computers which stored both program and data in the computer's memory
- IAS was the prototype for the IBM family of computers.

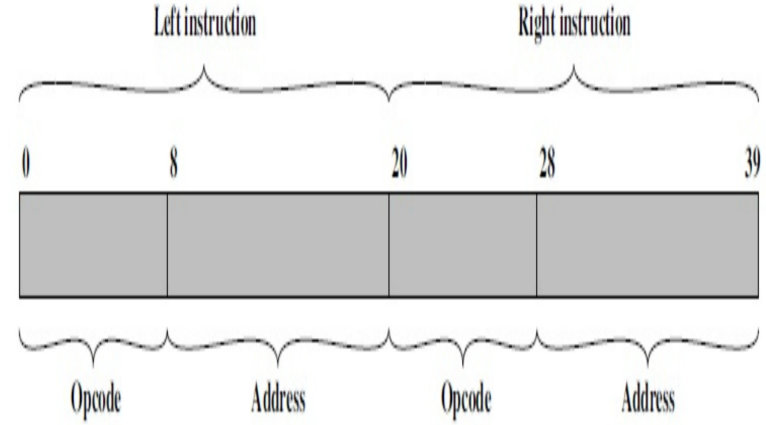
IAS



IAS

- The IAS has
  - an accumulator register (AC)
  - an arithmetic register (AR)
  - memory (RAM) called Selectron that has up to 1024 (40-bit memory locations) words.
  - IAS machine language instructions are 20 bits while one data value are 40 bits.





The following table lists the complete IAS instruction set. There are 22 machine instructions for the IAS computer, each associated with an 8-bit op-code.

inst name	opcode	description
S(x)->Ac+	1	copy the number in Selectron location x into AC
S(x)->Ac-	2	same as #1 but copy the negative of the number
S(x)->AcM	3	same as #1 but copy the absolute value
S(x)->Ac-M	4	same as #1 but subtract the absolute value
S(x)->Ah+	5	add the number in Selectron location x into AC
S(x)->Ah-	6	subtract the number in Selectron location x from AC
S(x)->AhM	7	same as #5, but add the absolute value
S(x)->Ah-M	8	same as #7, but subtract the absolute value
S(x)->R	9	copy the number in Selectron location x into AR
R->A	10	copy the number in AR to AC
S(x)*R->A	11	Multiply the number in Selectron location x by the number in AR. Place the left half of the result in AC and the right half in AR.
A/S(x)->R	12	Divide the number in AC by the number in Selectron location x. Place the quotient in AR and the remainder in AC.
Cu->S(x)	13	Continue execution at the left-hand instruction of the pair at Selectron location x
Cu`->S(x)	14	Continue execution at the right-hand instruction of the pair at Selectron location x
Cc->S(x)	15	If the number in AC is >= 0, continue as in #13. Otherwise, continue normally.
Cc`->S(x)	16	If the number in AC is >= 0, continue as in #14. Otherwise, continue normally.
At->S(x)	17	Copy the number in AC to Selectron location x
Ap->S(x)	18	Replace the right-hand 12 bits of the left-hand instruction at Selectron location x by the right-hand 12 bits of AC
Ap`->S(x)	19	Same as above, but modifies right-hand instruction
L	20	Shift the number in AC to the left 1 bit (new bit on the right is 0)
R	21	Shift the number in AC to the right 1 bit (leftmost bit is copied)
halt	0	Halt the program (see paragraph 6.8.5 of IAS report)

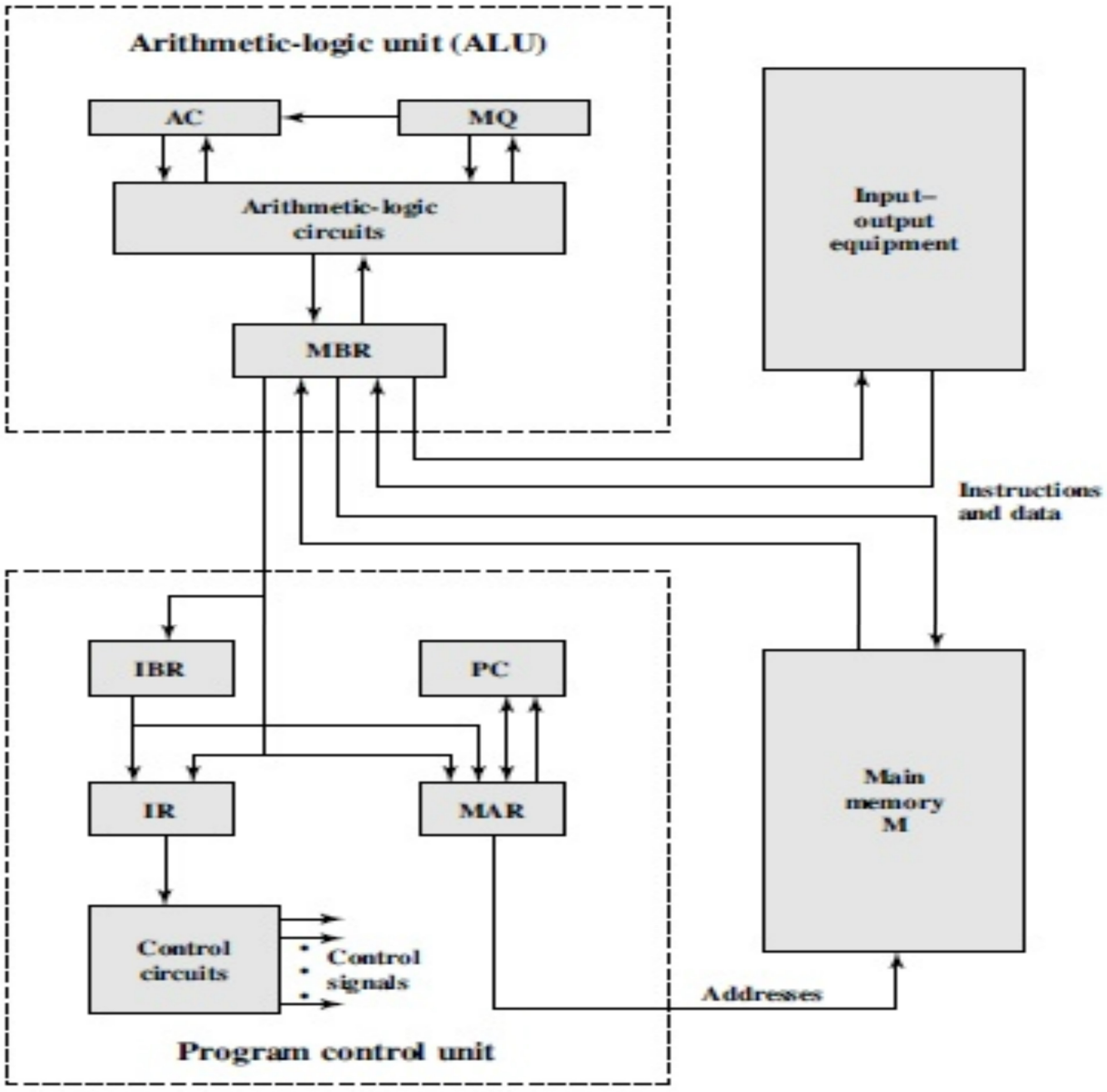
AC: Accumulator
AR: Arithmetic Register

IAS

- ALU contains storage locations, called registers as:
  - **Memory Buffer Register (MBR):** Contains the word to be stored in memory or sent to the I/O unit, or is used to receive a word from memory or from the I/O unit.
  - **Memory Address Register (MAR):** Specifies the address in memory of the word to be written from or read into the MBR.
  - **Instruction Register (IR):** Contains the 8-bit op-code instruction being executed.
  - **Instruction Buffer Register (IBR):** Employed to hold temporarily the right-hand instruction from a word in memory.

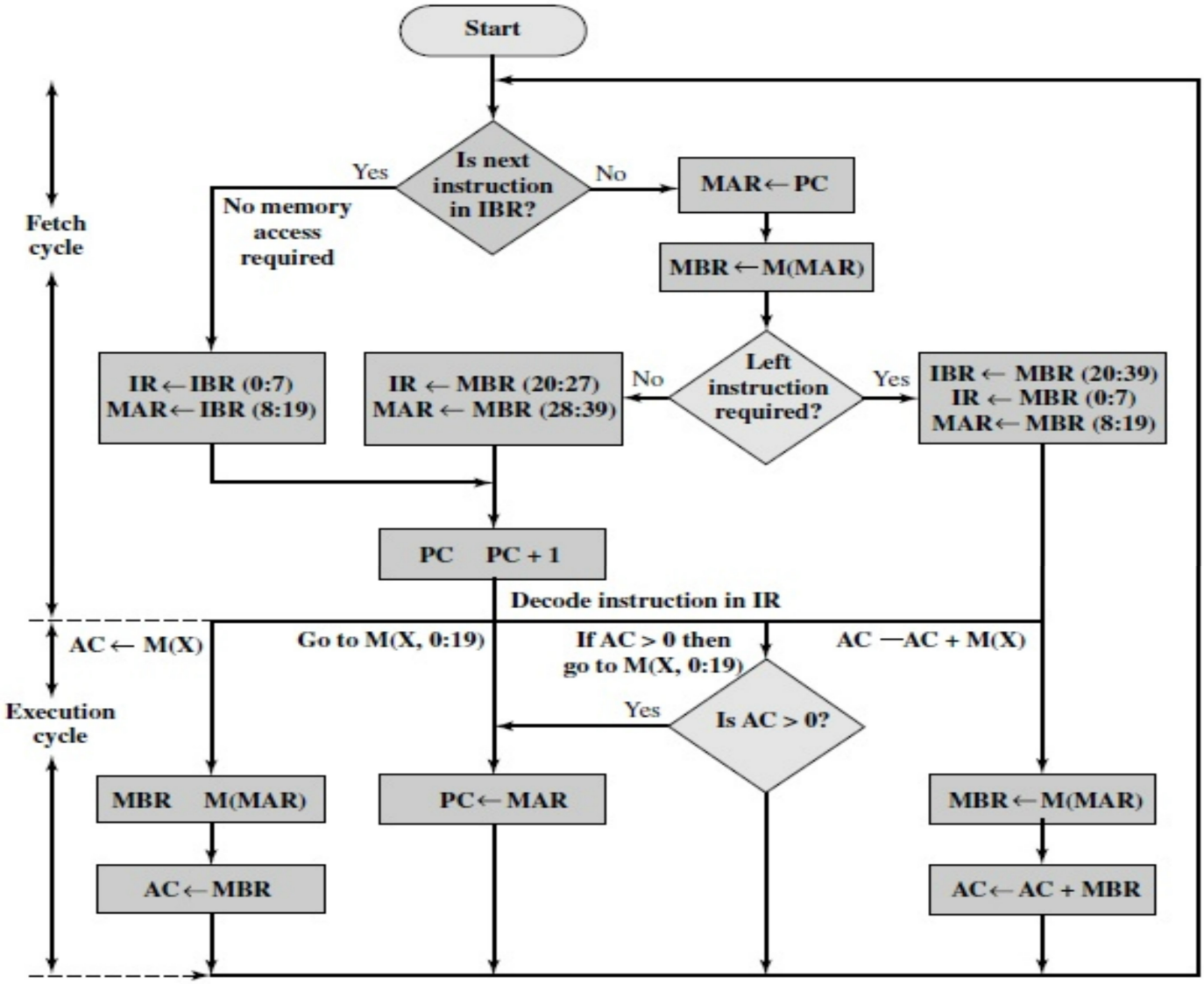
IAS

- **Program Counter (PC):** Contains the address of the next instruction-pair to be fetched from memory.
- **Accumulator (AC) :** holds
  - output of the ALU after an arithmetic operation
  - temporarily operands loaded from memory
  - the most-significant digits of a product
  - the divisor for division
- **Multiplier Quotient (MQ):** holds
  - temporarily operands
  - results of ALU operations
  - least-significant bits of the product as multiplication proceeds
  - quotient from division


Arithmetic-logic unit (ALU)
AC
MQ
Arithmetic-logic circuits
MBR
Input–output equipment
Instructions and data
IBR
PC
IR
MAR
Main memory M
Control circuits
Control signals
Addresses
Program control unit

IAS

- The IAS operates through CU by performing an instruction cycle repetitively which consists of two sub-cycles as:
  - Fetch Cycle:
    - the op-code of the next instruction is loaded into the IR and the address is loaded into the MAR.
    - The instruction may be taken from the IBR, or it can be obtained from memory by loading a word into the MBR, and then down to the IBR, IR, and MAR.
  - Execute Cycle*:* is performed when the op-code is in IR.
    - The Control circuitry send out the appropriate control signals to cause data to be moved or an operation to be performed by the ALU.


Start
Is next instruction in IBR?
Yes
No
No memory access required
MAR ← PC
MBR ← M(MAR)
Left instruction required?
No
Yes
IR ← IBR (0:7)
MAR ← IBR (8:19)
IR ← MBR (20:27)
MAR ← MBR (28:39)
IBR ← MBR (20:39)
IR ← MBR (0:7)
MAR ← MBR (8:19)
PC PC + 1
Decode instruction in IR
Fetch cycle
Execution cycle
AC ← M(X)
Go to M(X, 0:19)
If AC > 0 then go to M(X, 0:19)
AC —AC + M(X)
Is AC > 0?
Yes
MBR M(MAR)
PC ← MAR
MBR ← M(MAR)
AC ← MBR
AC ← AC + MBR

IAS

- The instruction executed by IAS originated from Von Neumman is classified into three types :
    - Move
    - ALU
    - Control (Unconditional and conditional branching)

MOVE INSTRUCTIONS	
• AC ← MQ	Move the number held in the MQ into the AC.
• M(*x*) ← AC	Move the number in the AC to location *x* in memory. The memory address *x* is found in the 12 least-significant bits of the instruction.
• M(*x*,28:39) ← AC(28:39)	Replace the right-hand 12 bits of the right-hand instruction located at position *x* in the memory with the right-hand 12 bits in the AC.
• M(*x*,8:19) ← AC(28:39)	Replace the right-hand 12 bits of the left-hand instruction in location *x* in the memory with the right-hand 12 bits in the AC.

Control Instructions	
• Go to M(*x*, 20:39)	Shift the control to the right-hand instruction of the pair in M(*x*).
• JUMP M(X,0:19)	Take next instruction from left half of M(X)
• If AC >= 0, then PC ← M(*x*, 0:19)	If the number in the AC is >= 0, go to the left-hand instruction in M(*x*).
• If AC >= 0, then PC ← M(*x*, 20:39)	If the number in the AC is >= 0, go to the right-hand instruction in M(*x*)

ALU INSTRUCTIONS	
• $AC \leftarrow M(x)$	Clear the AC and add the number from location x in the memory.
• $AC \leftarrow AC - M(x)$	Clear the AC and subtract the number at location x in the memory.
• $AC \leftarrow AC_c + \lvert M(x)\rvert$	Clear the AC and add the absolute value of the number at location x in the memory.
• $AC \leftarrow AC_c - \lvert M(x)\rvert$	Clear the AC and subtract the absolute value of the number at location x in the memory.
• $AC \leftarrow AC + M(x)$	Add the number at location x in the memory into the AC.
• $AC \leftarrow AC - M(x)$	Subtract the number at location x in the memory from the AC.
• $AC \leftarrow AC + \lvert M(x)\rvert$	Add the absolute value of the number at location x in the memory to the AC.
• $AC \leftarrow AC - \lvert M(x)\rvert$	Subtract the absolute value of the number at location position x in the memory into the AC.
• $MQ \leftarrow M(x)$	Clear the MQ register and add the number at location x in the memory into it.
• $AC, MQ \leftarrow M(x) \times MQ$	Clear the AC and multiply the number at location x in the memory by the number in the MQ, placing the most- significant 39 bits of the answer in the AC and the least-significant 39 bits of the answer in the MQ.
• $MQ, AC \leftarrow AC / M(x)$	Clear the register and divide the number in the AC by the number at location x of the memory, leaving the remainder in the AC and placing the quotient in MQ.
• $AC \leftarrow AC \times 2$	Multiply the number in the AC by 2.
• $AC \leftarrow AC / 2$	Divide the number in the AC by 2.

Historical Overview Of Computer Architecture (cont.)

- **Transistors**: Transistor is a device composed of semiconductor material that amplifies a signal or opens or closes a circuit. Transistors replaced vacuum tubes but still generate a great deal of heat that causes the computer to damage
- **Integrated Circuits** : are placed on silicon chips, called semiconductors. Computers consist of many chips placed on electronic boards called printed circuit boards that can take less space, faster and require less energy
- **Microprocessors** : the terms microprocessor and CPU are used interchangeably. Three basic characteristics differentiate microprocessors:
  - Instruction Set: The set of instructions that the microprocessor can execute.
  - Bandwidth: The number of bits processed in a single instruction.
  - Clock Speed: Given in megahertz (MHz), the clock speed determines how many instructions per second the processor can execute.
- **Artificial Intelligence Based**