Computer Organization and Architecture

Central Processing Unit (CPU)

TOPIC TO BE COVERED:

Central Processing Unit (CPU)
General Register Organization
Stack Organization
Instruction Format
Addressing Modes

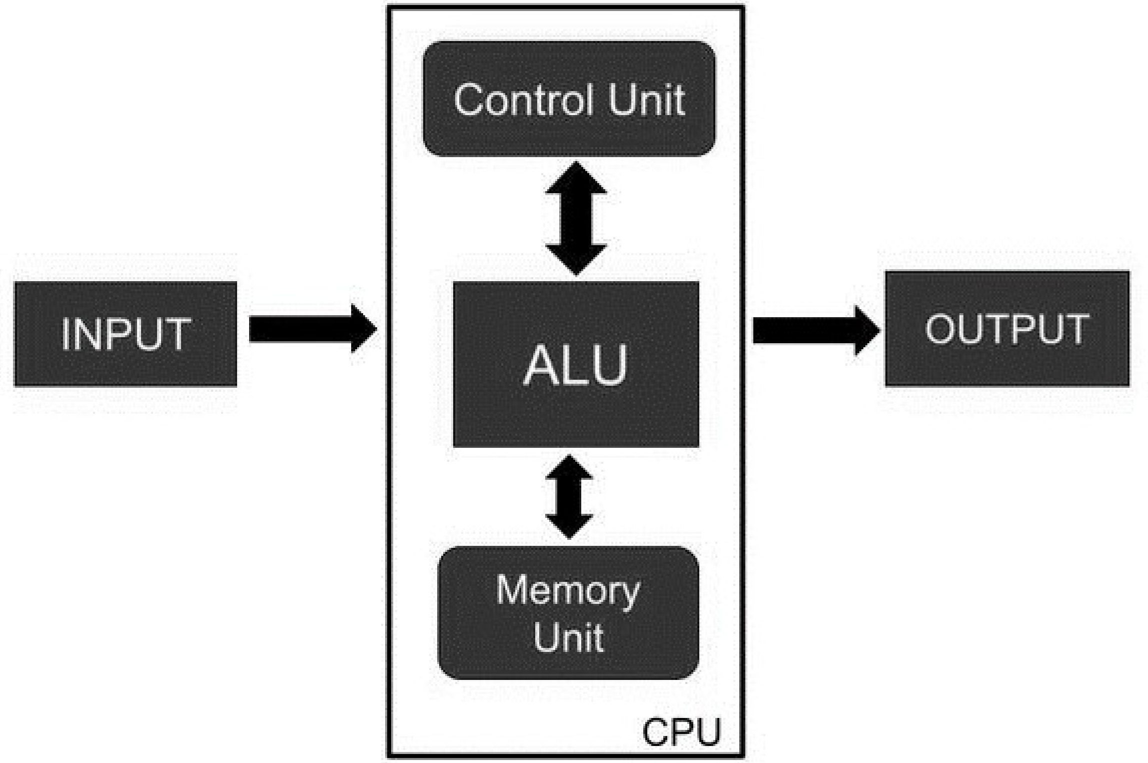
Central Processing Unit (CPU)

CPU is the brain of the computer. All types of data processing operations and all the important functions of a computer are performed by the CPU.

The CPU consists of 3 major units, which are:

1. Memory or Storage Unit
2. Control Unit
3. ALU(Arithmetic Logic Unit)

Central Processing Unit (Cont..)



Central Processing Unit (Cont..)

Memory or Storage Unit:

- Data and instructions are stored in memory units which are required for processing.
- It also stores the intermediate results of any calculation or task when they are in process.
- The final results of processing are stored in the memory units before these results are released to an output device for giving the output to the user.
- All sorts of inputs and outputs are transmitted through the memory unit.

Central Processing Unit (Cont..)

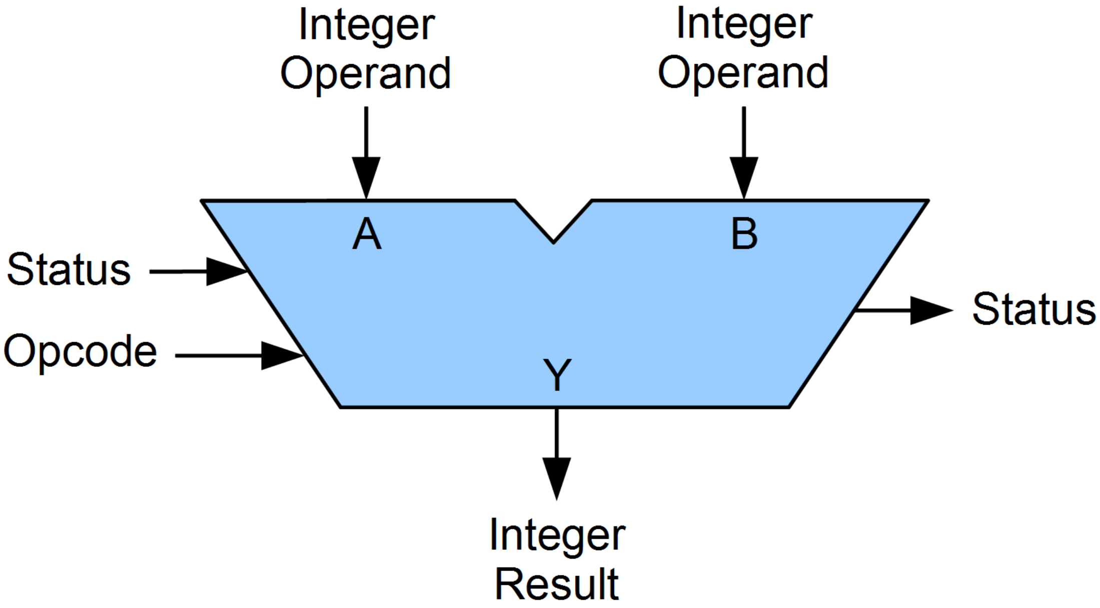
Control Unit:

- Controlling of data and transfer of data and instructions is done by the control unit among other parts of the computer.
- The control unit is responsible for managing all the units of the computer.
- The main task of the control unit is to obtain the instructions or data which is input from the memory unit, interprets them, and then directs the operation of the computer according to that.
- The control unit is responsible for communication with Input and output devices for the transfer of data or results from memory.
- The control unit is not responsible for the processing of data or storing data.

Central Processing Unit (Cont..)

ALU (Arithmetic Logic Unit) :

- Arithmetic Section
- Logic Section



General Register Organization

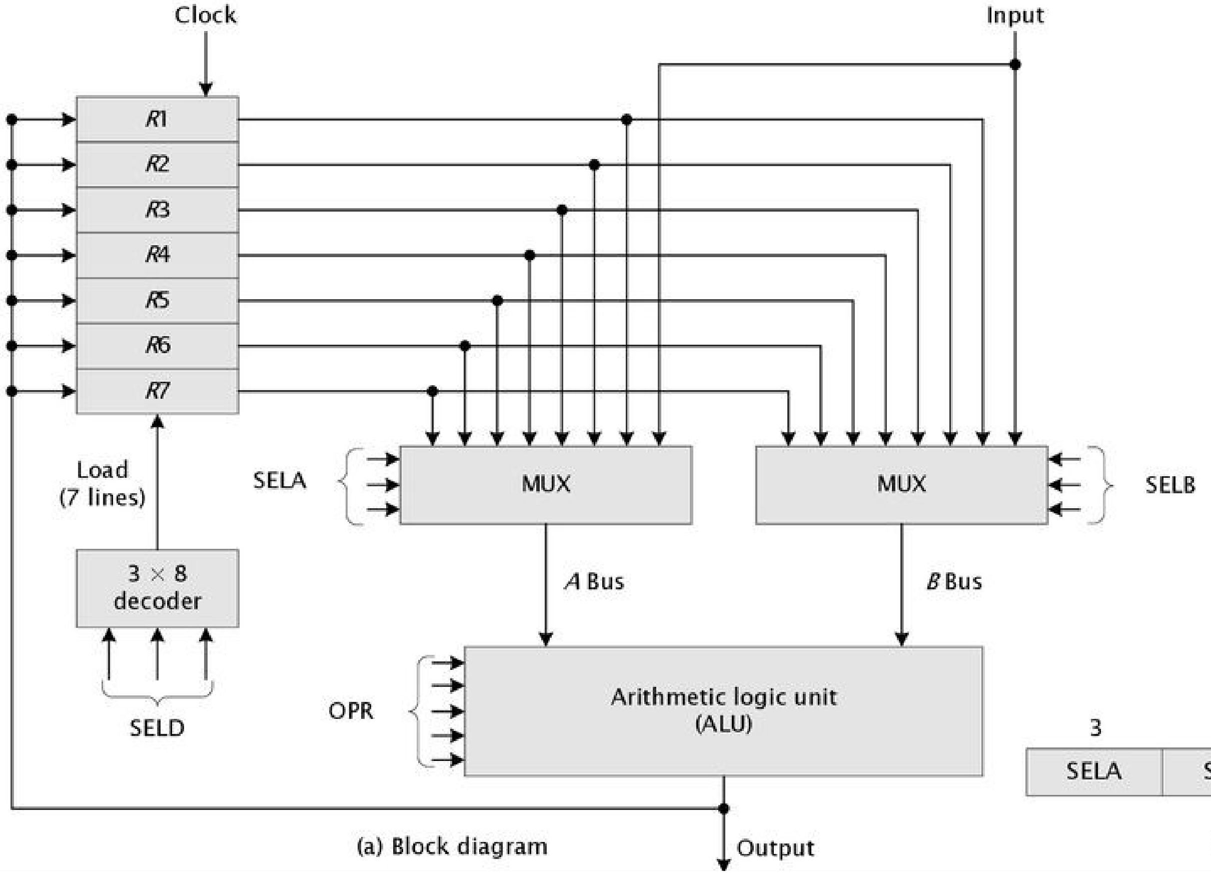
A set of flip-flops forms a register. A register is a unique high-speed storage area in the CPU.

Registers implement two important functions in the CPU operation are as follows −

- It can support a temporary storage location for data. This supports the directly implementing programs to have fast access to the data if required.
- It can save the status of the CPU and data about the directly implementing program.

General Register Organization (Cont..)

- Register set with common ALU



(a) Block diagram

3	3	3	5
SELA	SELB	SELD	OPR

(b) Control word

General Register Organization (Cont..)

The CPU bus system is managed by the control unit. The control unit explicit the data flow through the ALU by choosing the function of the ALU and components of the system.

Consider R1 ← R2 + R3, the following are the functions implemented within the CPU −

MUX A Selector (SELA) − It can place R2 into bus A.

MUX B Selector (SELB) − It can place R3 into bus B.

ALU Operation Selector (OPR) − It can select the arithmetic addition (ADD).

Decoder Destination Selector (SELD) − It can transfers the result into R1.

The multiplexers of 3-state gates are performed with the buses. The state of 14 binary selection inputs determines the control word. The 14-bit control word defines a micro-operation.

General Register Organization (Cont..)

The encoding of register selection fields is specified in the table.

Encoding of Register Selection Field

Binary Code	SELA	SELB	SELD
000	Input	Input	None
001	R1	R1	R1
010	R2	R2	R2
011	R3	R3	R3
100	R4	R4	R4
101	R5	R5	R5
110	R6	R6	R6
111	R7	R7	R7

Stack Organization

What is Stack:

The stack is a storage device, used for storing information or data in a manner of LIFO (Last In First Out). Whenever we enter the data in the form of LIFO manner, the element that has to be deleted first is the last inserter element, so the last inserted element is taken out first. It is the memory unit within an address register called stack pointer (SP). The stack pointer always indicates the top element in the stack that means which location the data has to be inserted.

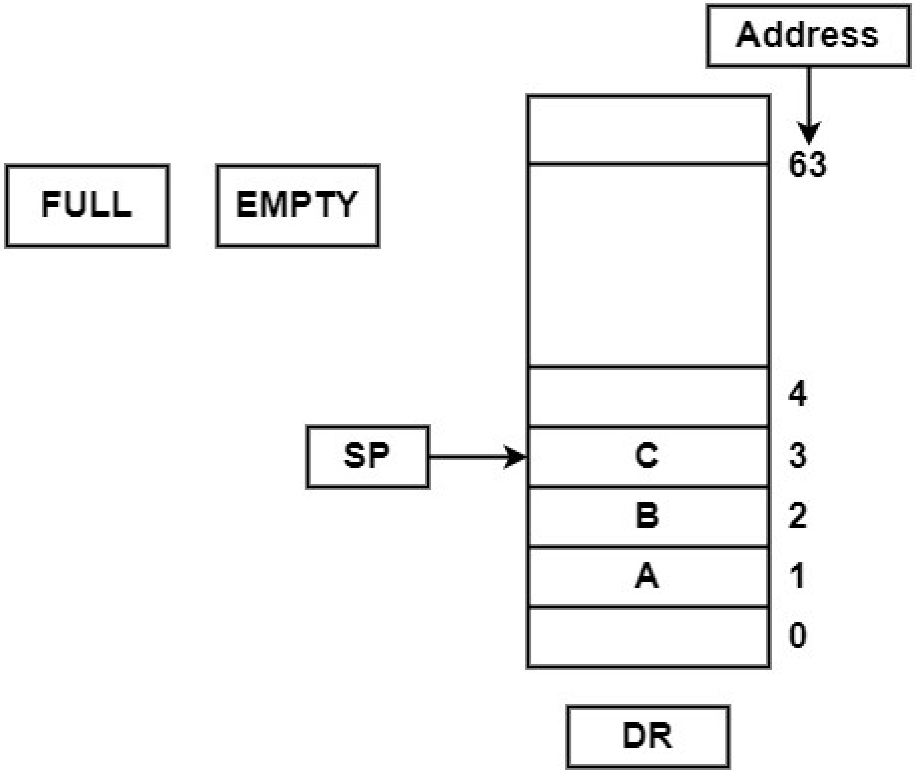
Type of Stack:

Register Stack

Memory Stack

Stack Organization (Cont..)

Register Stack



Stack Organization (Cont..)

Register Stack

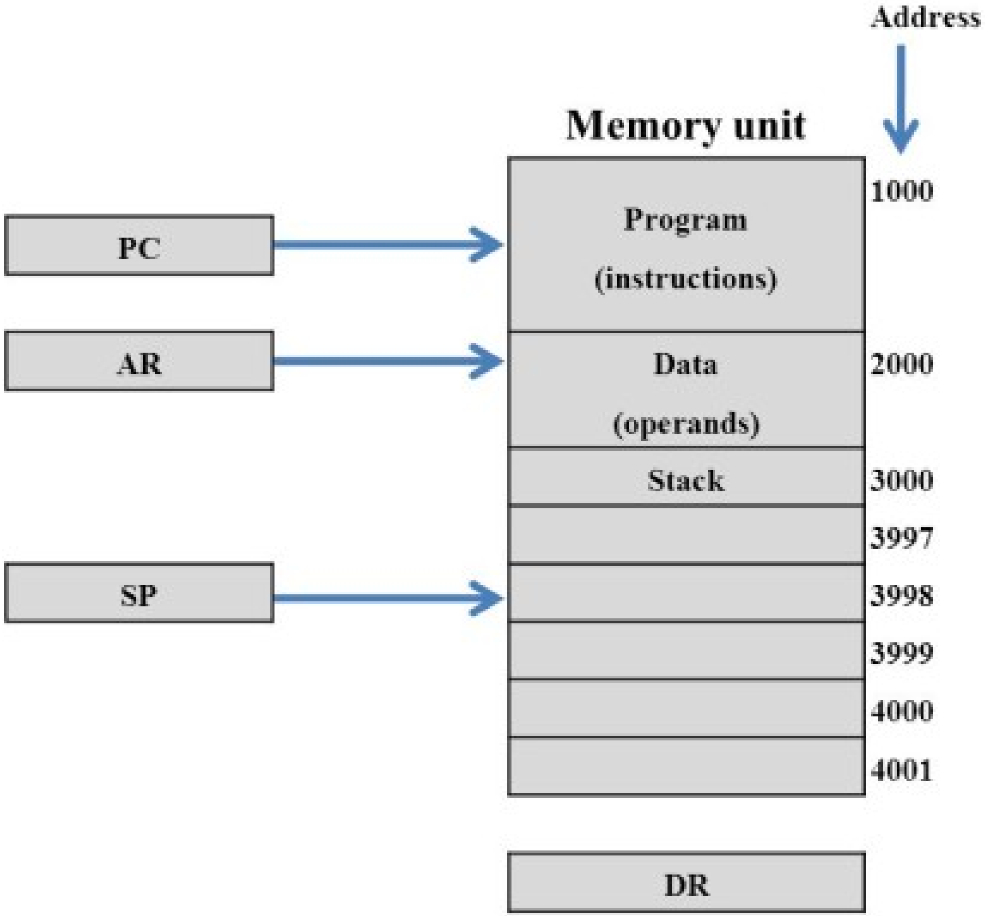
The push operation is executed as follows −

SP←SP + 1	It can increment stack pointer
K[SP] ← DR	It can write element on top of the stack
If (SP = 0) then (FULL ← 1)	Check if stack is full
EMTY ← 0	Mark the stack not empty

Stack Organization (Cont..)

Memory Stack

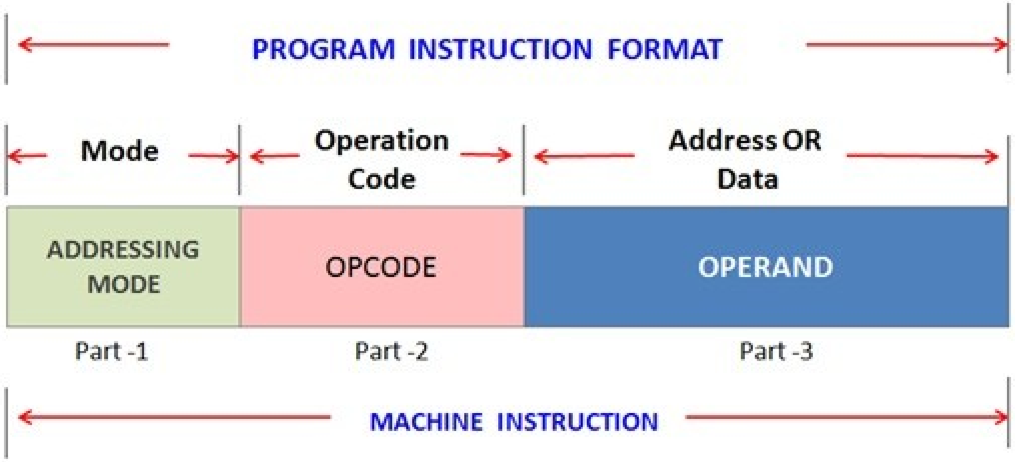
Step1: SP←SP+1
Step2: M[SP]←DR
Pop operation in Memory Stack
Step1: DR←M[SP]
Step2: SP←SP-1



Instruction Format

Based on the number of address, instructions are classified as:

- Zero address instructions
- One address instructions
- Two address instructions
- Three address instructions

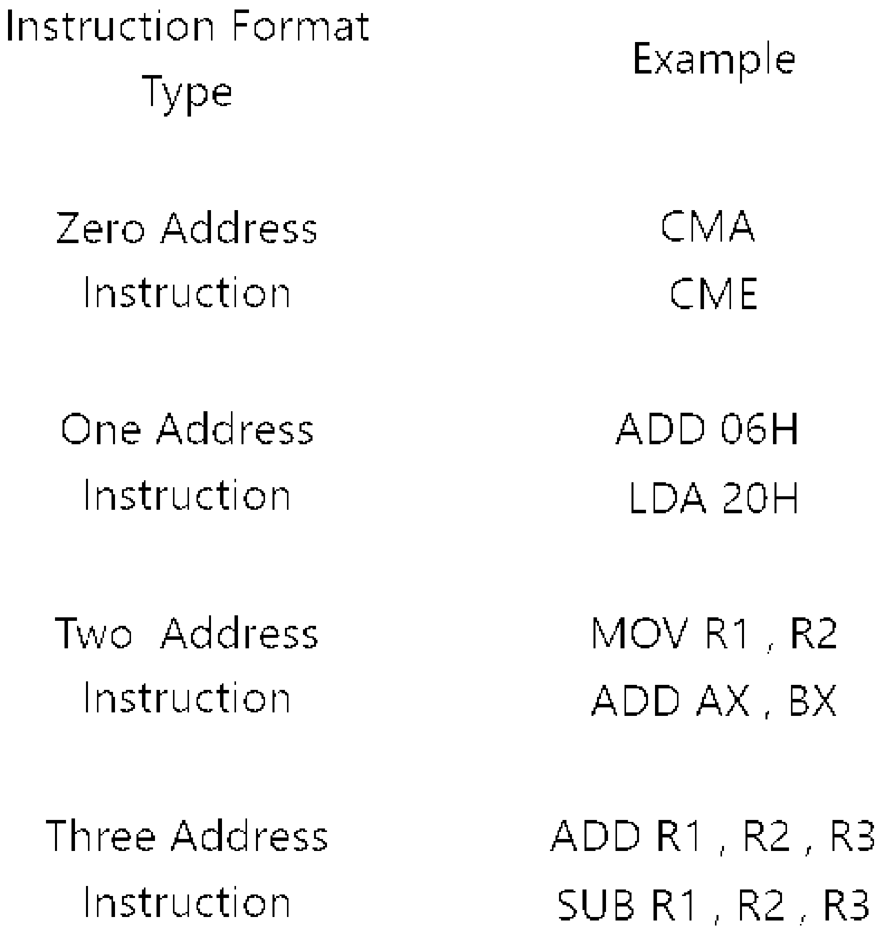


PART 1 – Addressing Mode - Rule For Operand - Data Or Address

PART 2 – OPCODE - For Control Unit - Which Operation To Perform.

PART 3 – OPERAND - For ALU - On Data Operation To Be Performed.

Instruction Format (Cont..)



Instruction Format Type	Example
Zero Address Instruction	CMA CME
One Address Instruction	ADD 06H LDA 20H
Two Address Instruction	MOV R1 , R2 ADD AX , BX
Three Address Instruction	ADD R1 , R2 , R3 SUB R1 , R2 , R3

Instruction Format (Cont..)

Zero address instruction

Expression: X = (A+B)*(C+D)

Postfixed : X = AB+CD+*

TOP means top of stack

M[X] is any memory location

PUSH	A	TOP = A
PUSH	B	TOP = B
ADD		TOP = A+B
PUSH	C	TOP = C
PUSH	D	TOP = D
ADD		TOP = C+D
MUL		TOP = (C+D)*(A+B)
POP	X	M[X] = TOP

Instruction Format (Cont..)

One address instruction

Expression: X = (A+B)*(C+D)

AC is accumulator

M[] is any memory location

M[T] is temporary location

LOAD	A	AC = M[A]
ADD	B	AC = AC + M[B]
STORE	T	M[T] = AC
LOAD	C	AC = M[C]
ADD	D	AC = AC + M[D]
MUL	T	AC = AC * M[T]
STORE	X	M[X] = AC

Instruction Format (Cont..)

Two address instruction

Expression: X = (A+B)*(C+D)

R1, R2 are registers

M[] is any memory location

MOV	**R1, A**	**R1 = M[A]**
ADD	R1, B	R1 = R1 + M[B]
MOV	R2, C	R2 = C
ADD	R2, D	R2 = R2 + D
MUL	R1, R2	R1 = R1 * R2
MOV	X, R1	M[X] = R1

Instruction Format (Cont..)

Three address instruction

Expression: X = (A+B)*(C+D)

R1, R2 are registers

M[] is any memory location

ADD	R1, A, B	R1 = M[A] + M[B]
ADD	R2, C, D	R2 = M[C] + M[D]
MUL	X, R1, R2	M[X] = R1 * R2

Addressing Modes

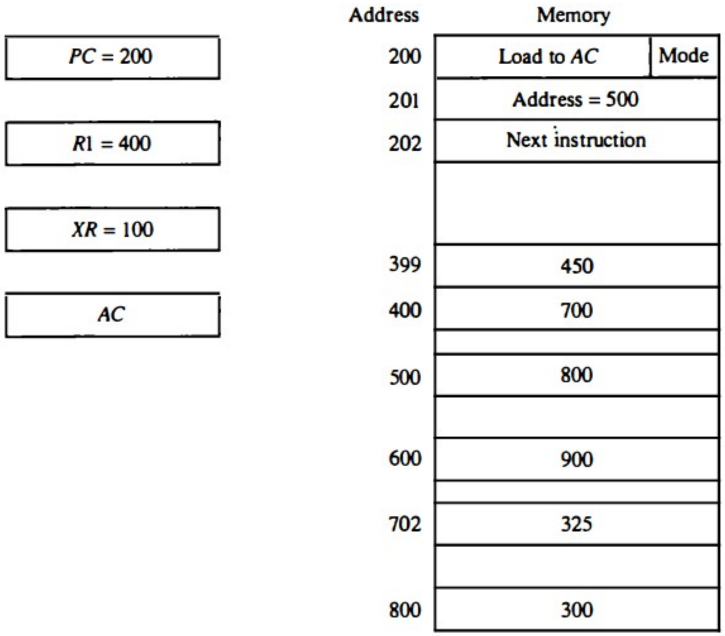
Addressing Modes– The term addressing modes refers to the way in which the operand of an instruction is specified. The addressing mode specifies a rule for interpreting or modifying the address field of the instruction before the operand is actually executed.

The various kind of addressing modes:

- Implied Mode
- Immediate Mode
- Register Mode
- Register Indirect Mode
- Autodecrement Mode
- Autoincrement Mode
- Direct Address Mode
- Indirect Address Mode
- Indexed Addressing Mode

Addressing Modes (Cont..)

Numerical Example for addressing Mode



PC = 200

R1 = 400

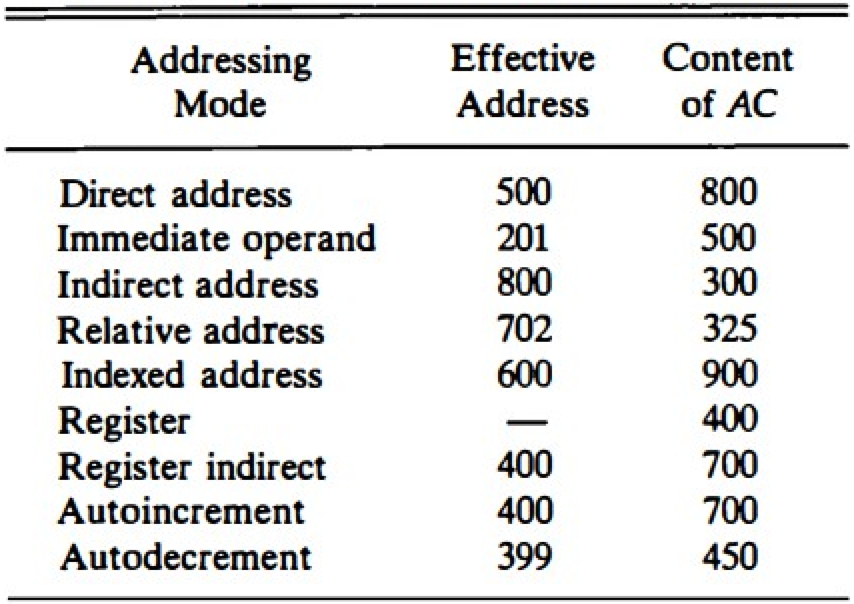
XR = 100

AC

Address	Memory	
200	Load to AC	Mode
201	Address = 500	
202	Next instruction	
399	450	
400	700	
500	800	
600	900	
702	325	
800	300	

Figure 7 Numerical example for addressing modes.

TABLE 4 Tabular List of Numerical Example



Addressing Mode	Effective Address	Content of AC
Direct address	500	800
Immediate operand	201	500
Indirect address	800	300
Relative address	702	325
Indexed address	600	900
Register	—	400
Register indirect	400	700
Autoincrement	400	700
Autodecrement	399	450

Thank You