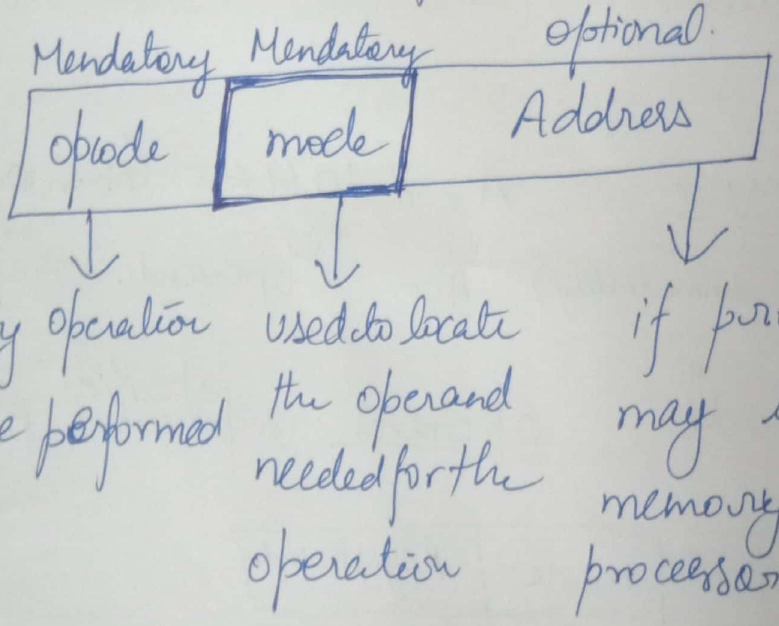
Addressing Modes.

These modes basically deals with the way how a digital computer fetches the operands from memory / peripheral / register.

To do any operation, operands can be fetched from

1. Memory 2. Peripheral (Keyboard) 3. Register.

A General Instruction format



Mendatory
Mendatory
optional.
opcode
mode
Address
Specify operation to be performed
used to locate the operand needed for the operation
if present, ~~tells~~ may specify a memory address or processor register.

Addressing Modes.

1. Implied Mode: Operands are implicitly specified in the definition of the instruction.

e.g. CMA (Compliment Acc. it implies operand is present in Acc.)

CLA (clear Acc).

In Stack organized system, instruction like

ADD (Addition)

SUB (Subtraction)

lie in the category of Implied Addressing mode. Since the operands are present implied to be on the top of the stack.

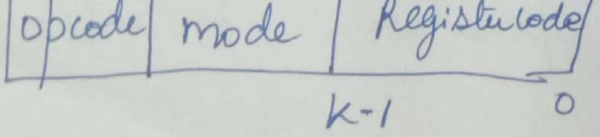
2. Immediate Mode: The operand is specified in the instruction.

e.g.

MVI A, #10H ← Given number is in Hex.

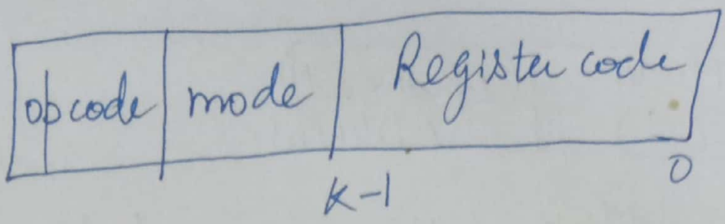
(move Immediately) Acc. operand.

3. Register Mode: Operand is ~~present~~ specified in register.



An K-bit field specified one of 2^k registers that hold the operand.

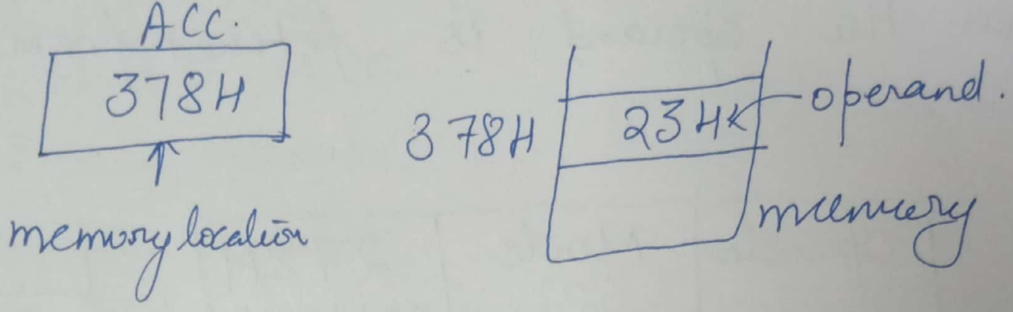
4. Register Indirect Mode: Instruction specify a register that gives the address of the operand in memory.



Let K-bit address denote Reg. ACC.

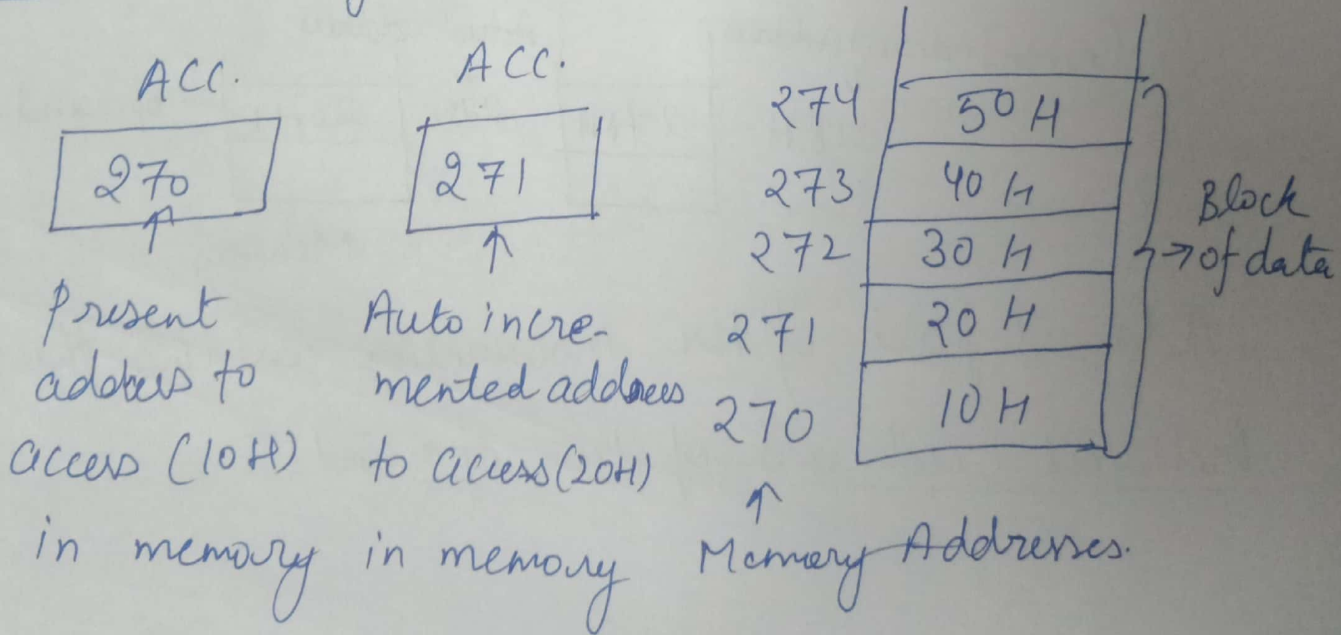
Now acc. will contain add of the operand

e.g.



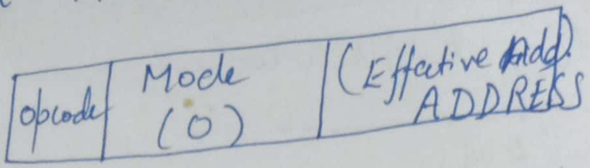
5. Autoincrement or Autodecrement mode:

This mode is similar to Register Indirect mode except. that the register is incremented or decremented after its value is used to access memory.



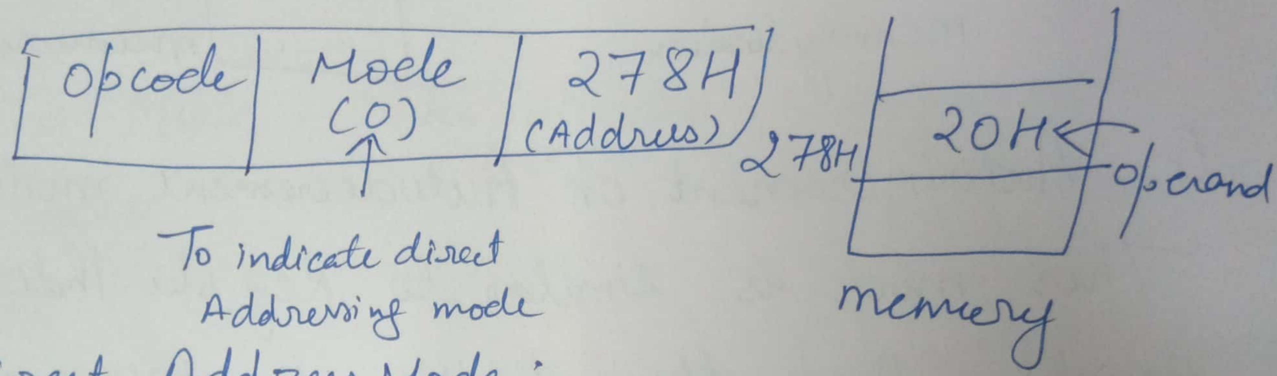
ACC.
270
Present address to access (10H) in memory
ACC.
271
Auto incre-mented address to access (20H) in memory
274
273
272
271
270
50 H
40 H
30 H
20 H
10 H
Block of data
Memory Addresses.

6 Direct Address Mode:

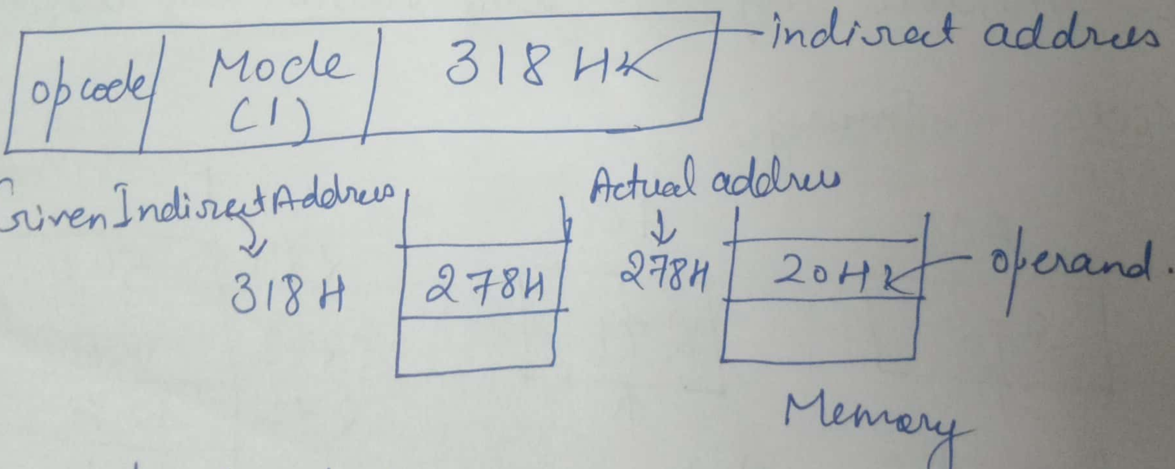


Here effective ~~ADD~~ address of the Operand is specified in the instruction in address part. ~~from~~ Using this address the operand is fetched from memory.

e.g

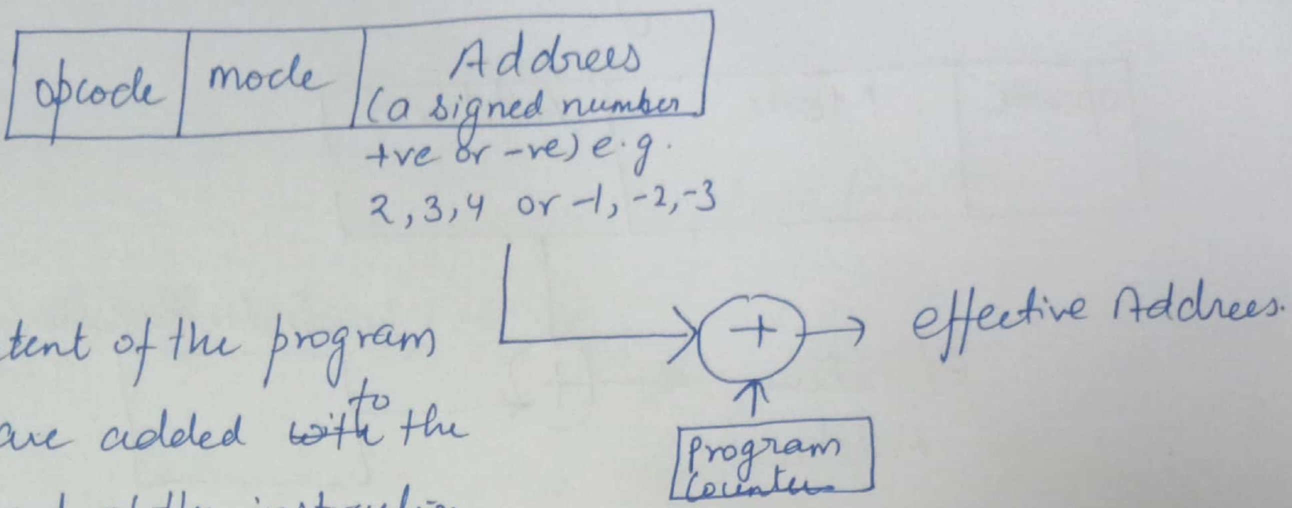


7. Indirect Address Mode:



Address part of the instruction contains the indirect address of the operand.

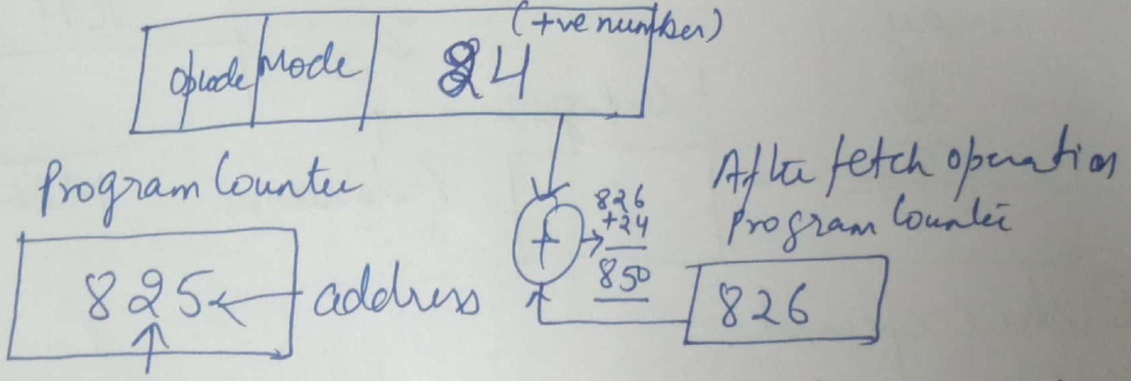
Relative Address Mode:



The content of the program Counter are added ~~with~~ to the address part of the instruction to get effective address.

The address part of the Instruction contains a signed number -ve or +ve this number is added to Program counter.

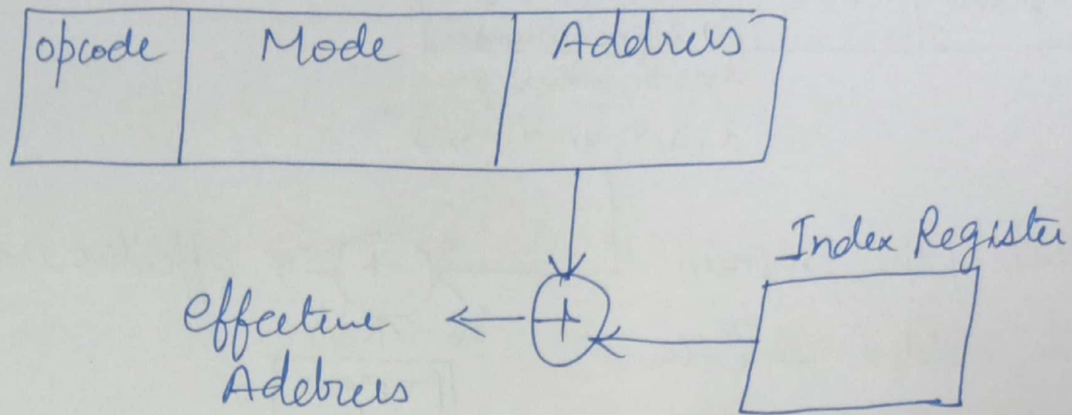
e.g.



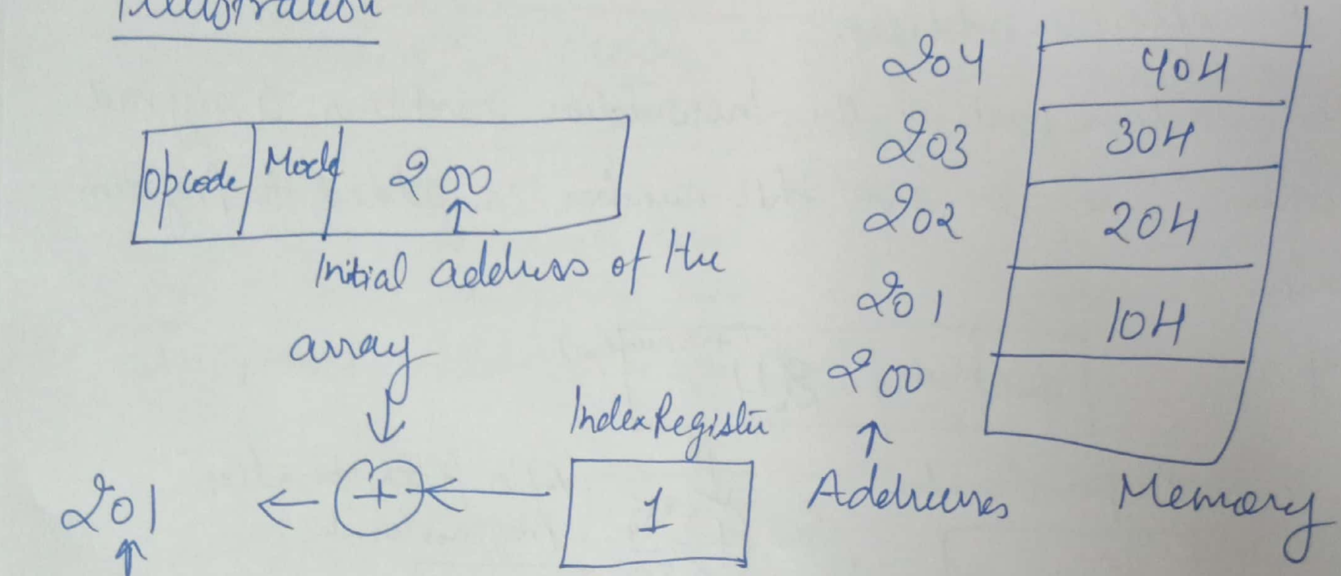
Instruction at the given address is fetched. Now the Program Counter will contain next address in seq. that will be added to Given signed number in Instruction

826 + 24 → 850 (effective address).

9. Indexed Addressing Mode



illustration

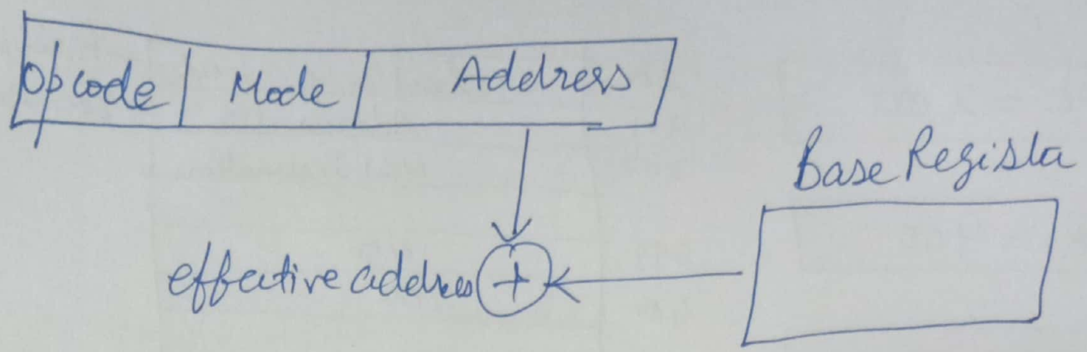


effcti Add where operand is present.

Address part of the instruction contains the intial address of the array (200 in this example). and Index register contains the diff. b/w initial address and address of the operand in memory (1 in this example)

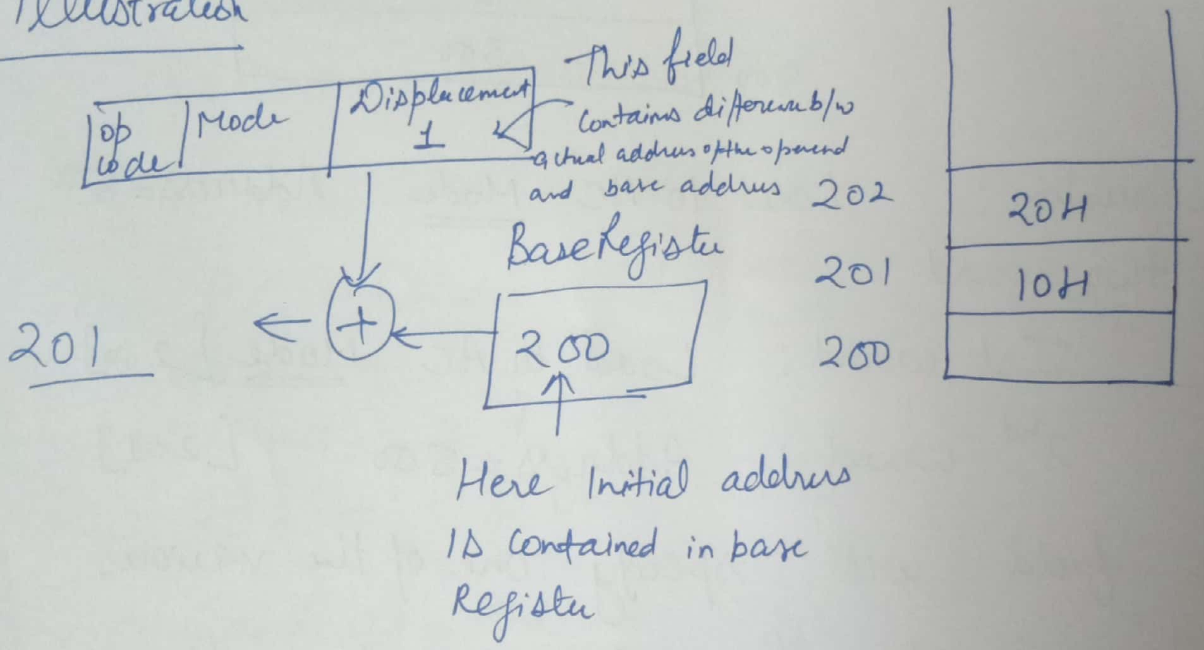
* Note: Index register can be Incremented to have access to consective opera

Base Register addressing Mode.



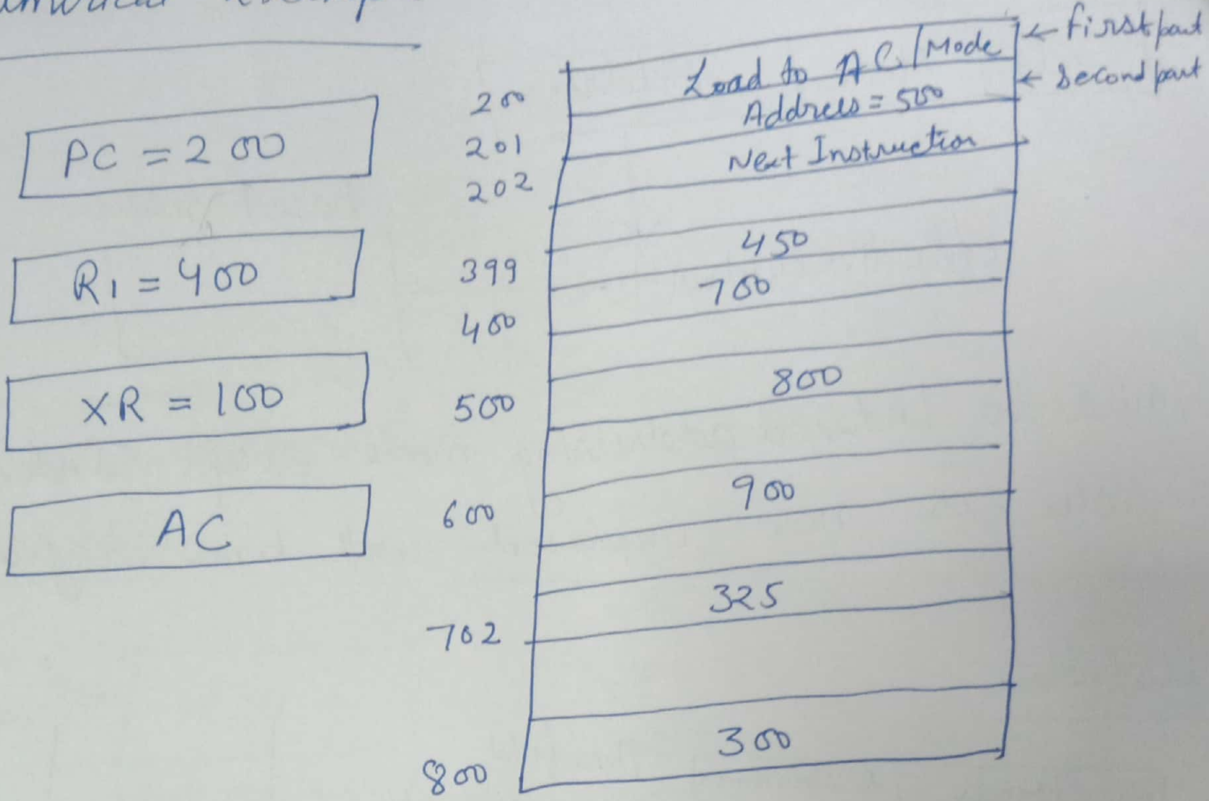
Similar to Indexed addressing mode except Index register is now replaced with base register.

Illustration



Op code
Mode
Displacement
1
This field contains difference b/w actual address of the operand and base address
Base Register
200
201
+
Here Initial address is contained in base Register
202
201
200
20H
10H

Numerical Example



Instruction : Load to AC Mode Address = 500

Its two word

Ist word : Load to AC Mode [200]

2nd word : Address = 500 [201]

Mode field will specify one of the various addressing modes. for every mode, effective address will be calculated in different manner. However, ~~after every~~ irrespective of the addressing mode given operand must be loaded into the AC.

1. Direct Address mode.

Effective Address(EA) = 500

Operand at 500 = 800

2. Immediate mode

EA = 201

Operand at ~~201~~ = 500

3. Indirect mode.

Given add = 500

Eff. Add = 800

operand at 800 = 300

4. In relative mode

EA = PC + Given add.

(after fetch of Ist Inst.)

202 + 500 = 702

operand at 702 = 325

5. In Index mode

EA = 500 + XR = 600 [XR=100]

operand at 600 = 900

6. Register mode

operand in R_1 = 400

7. Register Indirect mode.

EA in R_1 = 400

operand at 400 = 700

Types of Interrupt

Interrupt ?

Break in the normal executation of a program.

Types ?

1. External 2. Internal 3. Software

External → Comes from Input-Output (I/O) Devices. like

→ An I/O device requests transfer of data

→ An I/O device finished transfer of data

→ An time Out error of an event [e.g. loading of a web page in longer time]

→ Power failure [sudden shut down].

↳ Power failure Interrupt may have its service routine that transfer the complete state of the CPU in to a nondestructive memory (like Hard disks) in the few ms before Power failure.

Internal Interrupt (also known as traps)

Arise from

→ illegal or errorneous (wrong) use of an instruction or data.
e.g. invalid operation code, Stack overflow

→ premature termination of the instruction execution [means execution of Instruction is not completed].

- The Service program that process Internal Interrupt determines the corrective measure to be taken.

Difference b/w Internal & External ?

Internal	External
* Initiated by some exceptional condition Caused by the program itself.	* Initiated by external event.
* Synchronous with the program.	* Asynchronous.

Scanned with CamScanner

if program is rerun, internal Interrupt will occur in the same phase each time.	it depend on external condition that are indepedent of the program being executed at the time

Common

Both are initiated from signals that occur in the hardware of the CPU.

Software Interrupt.

* Initiated by executing an Instruction.
* Unlike a subroutine call, Software Interrupt is a special Call Instruction that behaves like an interrupt.
* Used by programmer to specify interrupt at any desired point in the program

Why these are required ?

~~for~~ There are certain complex task in the computer that are assigned to supervisor mode only rather than user mode.

e.g. a Complex input or output transfer

#) To do this

Initially user will write the code in the user mode. This code/program will contain the special call instruction (or Soft. Interrupt).

- This instruction causes the SI that stores the old CPU state and work on the new program Status Word (PSW) that took the control to the supervisor mode.