Addressing Modes

→ The different ways of specifying the location of an operand in an instruction are called as addressing modes.

(1) **Implied** → the instruction contains the operand implicitly.
→ it is used to design zero address instructions.
Eg. CLC → reset carry flag to 0.
CMA → complement accumulator contents.

(2) **Stack** → the operand is contained at top of stack.
Eg. ADD → this pops out two symbols contained at top of stack, performs operations and stores result back to top of stack.

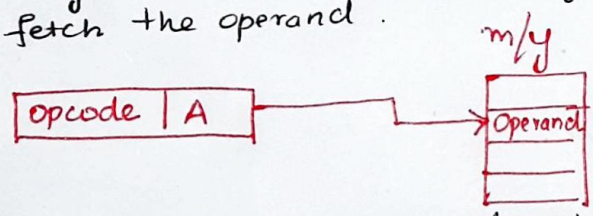
(3) **Immediate** → the operand is directly specified in the instruction itself.

Instruction

opcode	operand

Eg. ADD 10 → will add accumulator value by 10.
Mov R #20 → stores 20 in register R.

(4) **Direct** → the address field of instruction contains the effective address of operand.
→ only one reference to memory is required to fetch the operand.



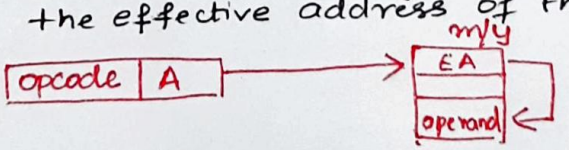
EA = A

* EA → effective address

Eg. ADD X → increment value stored in accumulator by value stored at m/y location X.

Acc ← Acc + [X]

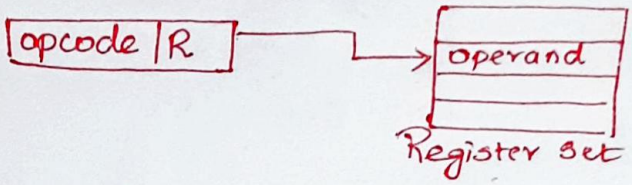
(5) **Indirect** → the address field of instruction specifies the address of memory location that contains the effective address of the operand.



EA = [A]

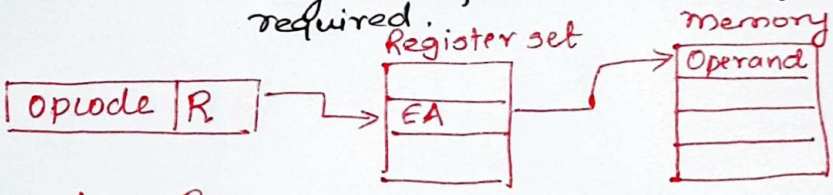
Eg. ADD X
Acc ← Acc + [[X]]

(6) **Register Direct** → the operand is contained in register set.
→ the address field of the instruction refers to a CPU register that contains the operand.
→ No reference to m/y is required to fetch the operand.



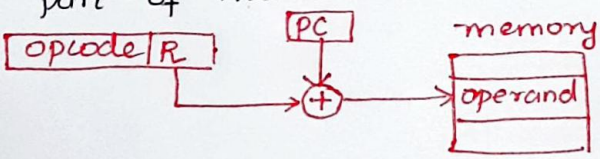
Eg. Add R
$Acc \leftarrow Acc + [R]$

(7) **Register Indirect** → the address field of instruction refers to CPU register that contains the effective address of the operand.
→ only one reference to memory is required.



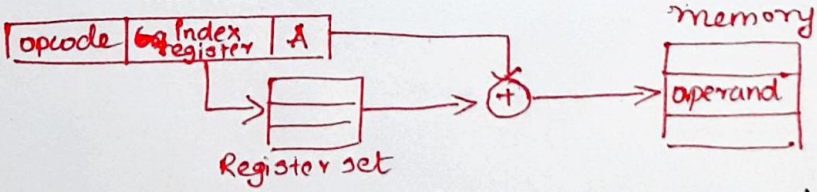
Eg. ADD R
$Acc \leftarrow Acc + [[R]]$

(8) **Relative** → effective address of operand is obtained by adding content of program counter with address part of instruction.



EA = PC contents + address part of instruction

(9) **Indexed** → effective address is obtained by adding content of index register with address part of instruction.



EA = content of index + address part of inst

(10) **Base** → EA = content of base register + address part of instruction

———x———x———x———.