Suri Vidyasagar College
Department of Computer Science
Sem-4 Study Material
Paper: Computer System Architecture

TOPIC: Addressing Modes

Addressing Modes-

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

1. Implied / Implicit Addressing Mode
2. Stack addressing mode
3. Immediate Addressing Mode
4. Direct Addressing Mode
5. Indirect Addressing Mode
6. Register Direct Addressing Mode
7. Register Indirect Addressing Mode
8. Relative Addressing Mode
9. Indexed Addressing Mode
10. Base Register Addressing Mode
11. Auto-Increment Addressing Mode
12. Auto-Decrement Addressing Mode

Implied Addressing Mode-In this addressing mode, the definition of the instruction itself specify the operands implicitly. It is also called as implicit addressing mode.

Examples-

- The instruction "Complement Accumulator" is an implied mode instruction (CMA).
- In a stack organized computer, Zero Address Instructions are implied mode instructions. (since operands are always implied to be present on the top of the stack)

Stack Addressing Mode-In this addressing mode, the operand is contained at the top of the stack.

Example- ADD
This instruction simply pops out two symbols contained at the top of the stack.
The addition of those two operands is performed.
The result so obtained after addition is pushed again at the top of the stack.

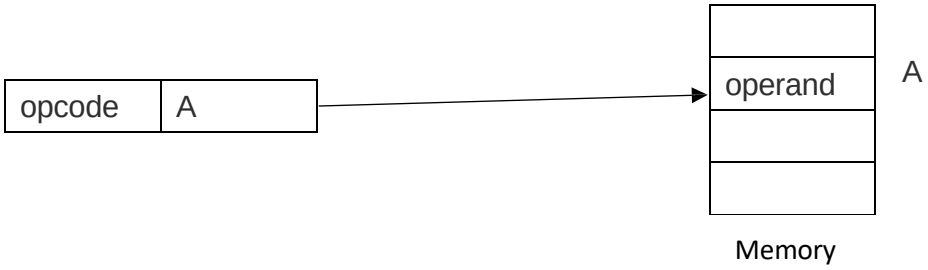
Immediate Addressing Mode- In this addressing mode, the operand is specified in the instruction explicitly. Instead of address field, an operand field is present that contains the operand.

Examples-

ADD 10 will increment the value stored in the accumulator by 10.

MOV R #20 initializes register R to a constant value 20.

Direct Addressing Mode- In this addressing mode, the address field of the instruction contains the effective address of the operand. Only one reference to memory is required to fetch the operand. It is also called as absolute addressing mode.

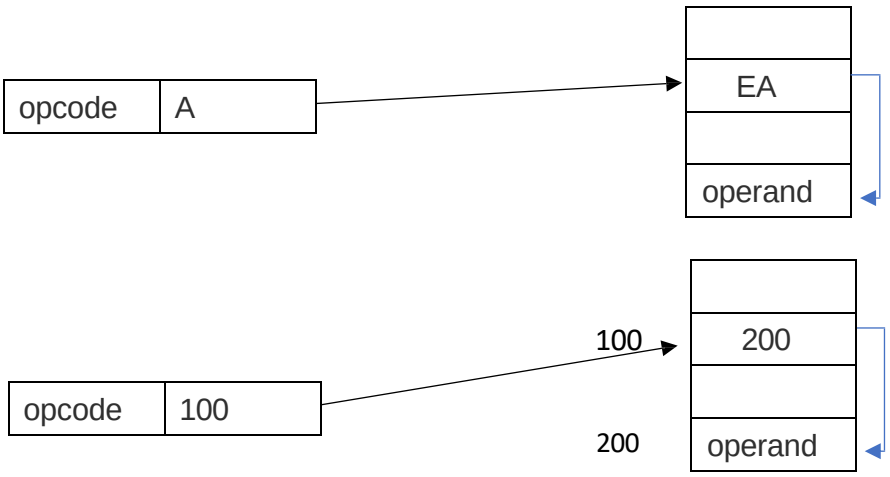


Example-

ADD X will increment the value stored in the accumulator by the value stored at memory location X.

AC ← AC + [X]

Indirect Addressing Mode- In this addressing mode, the address field of the instruction specifies the address of memory location that contains the effective address of the operand. Two references to memory are required to fetch the operand.



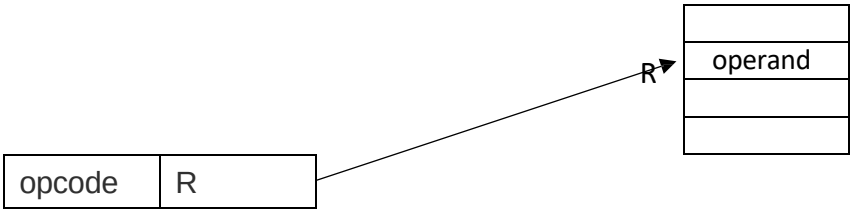
Example-

ADD X will increment the value stored in the accumulator by the value stored at memory location specified by X.

AC ← AC + [[X]]

Register Direct Addressing Mode- In this addressing mode, the operand is contained in a register set. The address field of the instruction refers to a CPU register that contains the operand.

No reference to memory is required to fetch the operand.



Register set

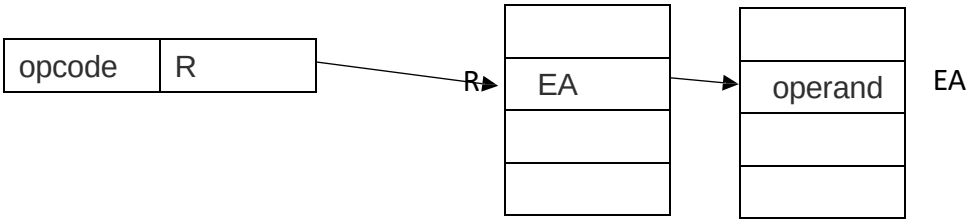
Example-

ADD R will increment the value stored in the accumulator by the content of register R.

AC ← AC + [R]

- This addressing mode is similar to direct addressing mode.
- The only difference is address field of the instruction refers to a CPU register instead of main memory.

Register Indirect Addressing Mode- In this addressing mode, the address field of the instruction refers to a CPU register that contains the effective address of the operand. Only one reference to memory is required to fetch the operand.



Register Set Memory

Example-

ADD R will increment the value stored in the accumulator by the content of memory location specified in register R.

AC ← AC + [[R]]

- This addressing mode is similar to indirect addressing mode.
- The only difference is address field of the instruction refers to a CPU register.

Base register addressing mode: Base register addressing mode is used to implement inter segment transfer of control. In this mode effective address is obtained by adding base register value to address field value.

EA= Base register + Address field value.

PC= Base register + Relative value.

Note:

PC relative and based register both addressing modes are suitable for program relocation at runtime.

Based register addressing mode is best suitable to write position independent codes.