Microprocessor & Interfacing
Lecture 9
8085 Addressing Modes

ECS DEPARTMENT
DRONACHARYA COLLEGE OF ENGINEERING

Contents

- Introduction
- Types of Addressing Modes

Introduction

- There is a method in which the instructions address the data to be operated. The method of specifying the data to be operated by the instruction is known as **addressing.**
- The way by which the microprocessor identifies the operands for a particular instruction is known as **Addressing mode.**

Types of Addressing Modes

- Immediate addressing mode
- Direct addressing mode
- Register addressing mode
- Register indirect addressing mode
- Implicit addressing mode

Immediate Addressing Mode:

- In this type of addressing mode the operand is specified within the instruction itself.
- For example Consider this instruction:

  ADI 34H
- This instruction adds the immediate data, 34H to the accumulator.
- 34H is the data here.
- H represents Hexadecimal value and the immediate value is added to the accumulator.
- In this case 34H is added to the accumulator. Suppose if accumulator has a value 8H and when this instruction is executed, 34H is added to the 8H and the result is stored in accumulator.
- In the above instruction the operand is specified within instruction itself.

Direct Addressing Mode:

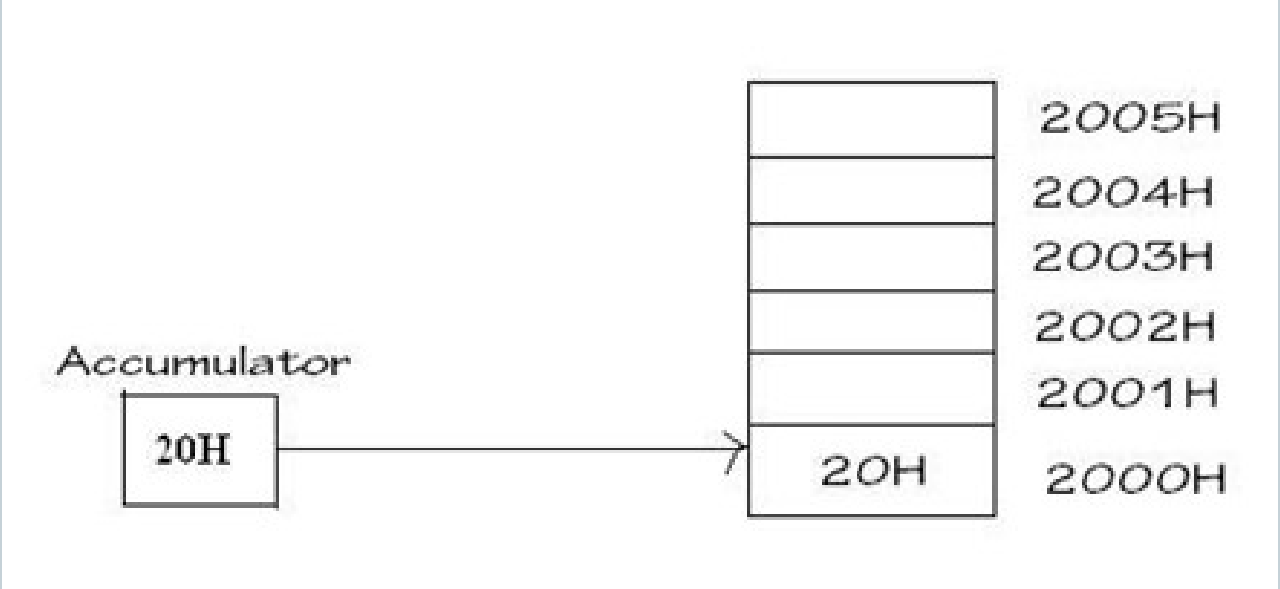
- In this mode of addressing, the address of the data (operand) is specified within the instruction.
- There is a subtle difference between the direct addressing modes and immediate addressing modes. In immediate addressing mode the data itself is specified within instruction, but in direct addressing mode the *address of the data is specified in the* instruction.
- **Example:**

OUT 10H
LDA 4100H
STA 2000H

- Consider the instruction STA 2000H

  When this instruction is executed, the contents of the accumulator are stored in the memory location specified. In the above example the contents of accumulator are stored in memory location 2000H.

Cont..



Register Addressing Mode:

- In this type of addressing mode the instruction specifies the name of the register in which the data is available and Opcode specifies the name (or) address of the register on which the operation would be performed.
- **Example:**

MOV A, B

- Here the Opcode is MOV. If the above instruction is executed, the contents of Register B are moved to the Register A, which is nothing but the accumulator.

Cont..

- Other examples:

ANA B

- On executing the above instruction the contents of Register B or logically ANDed with contents of register A (accumulator).

SUB H

- If we execute the above instruction the contents of Register H will be subtracted from the contents of the accumulator.

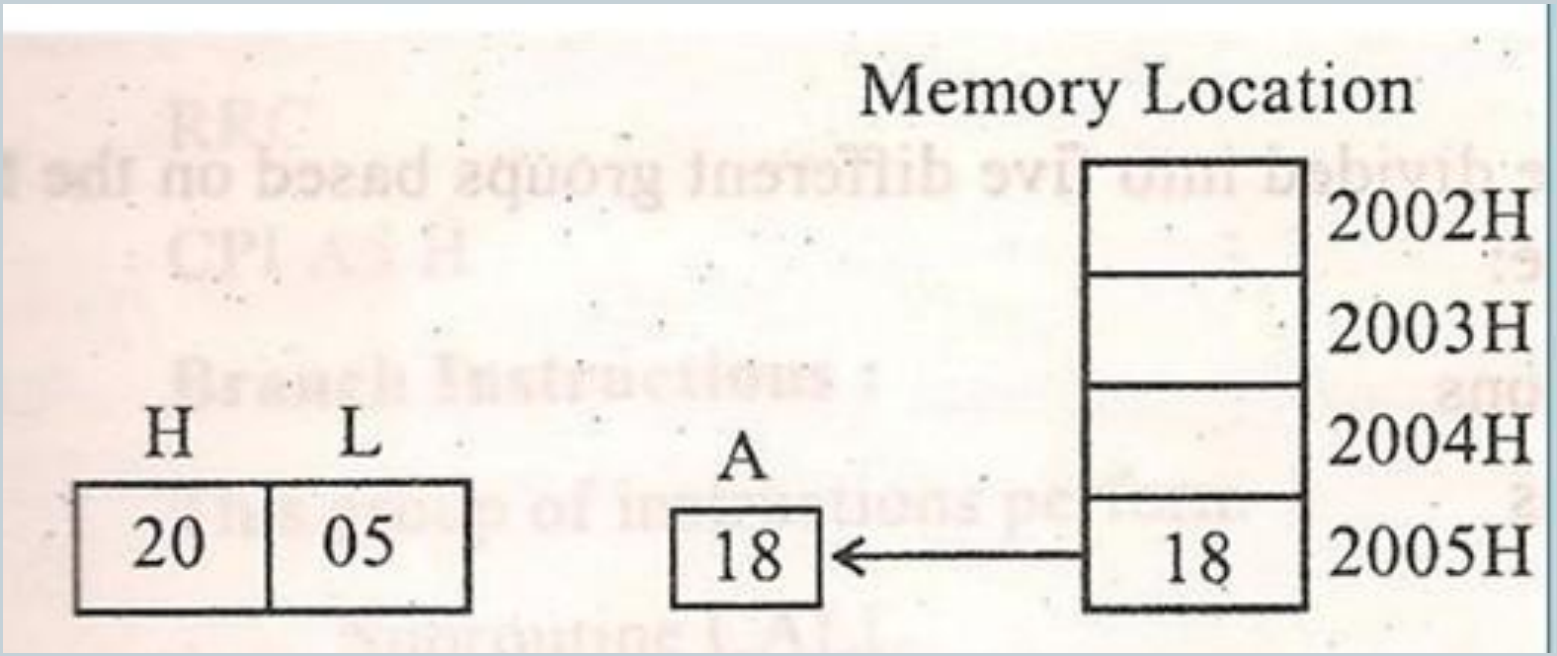
Register Indirect Addressing Mode:

- This is indirect way of addressing. In this mode the instruction specifies the name of the register in which the address of the data is available.
- Example:

```
MOV A, M
SUB M
DCR M
```

- Consider MOV A, M. This instruction will move the contents of memory location, whose address is in H-L register pair to the accumulator.
- M represents the address present in the H-L register pair. So when MOV A, M is executed, the contents of the address specified in H-L register pair are moved to accumulator.

Cont..



Implicit Addressing Mode:

- There are certain instructions in 8085 which does not require the address of the operand to perform the operation. They operate only upon the contents of accumulator.
- **Example:**

CMA

RAL

RAR

- CMA complements the contents of accumulator.
- If RAL is executed the contents of accumulator is rotated left one bit through carry.
- If RAR is executed the contents of accumulator is rotated right one bit through carry.

Cont..

Type	Instruction	Source	Destination
Direct	STA 2005 H	Accumulator	Memory 2005H
Register	MOV A, B	Register B	Accumulator
Register Indirect	MOV A, M	Memory [[HL]]	Accumulator
Immediate	MVI A, 18H	Data 18H	Accumulator

Scope of Research

- Here scope of research are design new addressing modes that should be easy to understand, can increase the processing speed and reduce the complexity.