Addressing Modes 8085

4

Instruction Set of Intel 8085

4.1 INTRODUCTION

An instruction is a command given to the computer to perform a specified operation on given data. The instruction set of a microprocessor is the collection of the instructions that the microprocessor is designed to execute. The instructions described in this chapter are of INTEL 8085. These instructions are of Intel Corporation. They cannot be used by other microprocessor manufacturers. The programmer can write a program in assembly language using these instructions. These instructions have been classified into the following groups :

1. Data Transfer Group
2. Arithmetic Group
3. Logical Group
4. Branch Control Group
5. I/O and Machine Control Group.

Data Transfer Group. Instructions which are used to transfer data from one register to another register, from memory to register or register to memory, come under this group. Examples are : MOV, MVI, LXI, LDA, STA etc. When an instruction of data transfer group is executed, data is transferred from the source to the destination without altering the contents of the source. For example, when MOV A, B is executed the content of the register B is copied into the register A, and the content of register B remains unaltered. Similarly, when LDA 2500 is executed the content of the memory location 2500 is loaded into the accumulator. But the content of the memory location 2500 remains unaltered.

Arithmetic Group. The instructions of this group perform arithmetic operations such as addition, subtraction; increment or decrement of the content of a register or memory. Examples are : ADD, SUB, INR, DAD etc.

Logical Group. The instructions under this group perform logical operation such as AND, OR, compare, rotate etc. Examples are: ANA, XRA, ORA, CMP, RAL etc.

Branch Control Group. This group includes the instructions for conditional and unconditional jump, subroutine call and return, and restart. Examples are: JMP, JC, JZ, CALL, CZ, RST etc.

I/O and Machine Control Group. This group includes the instructions for input/output ports, stack and machine control. Examples are : IN, OUT, PUSH, POP, HLT etc.

4.2 INSTRUCTION AND DATA FORMATS

Intel 8085 is an 8-bit microprocessor. It handles 8-bit data at a time. One byte consists of 8 bits. A memory location for Intel 8085 microprocessor is designed to accommodate 8-bit data. If 16-bit data are to be stored, they are stored in consecutive memory locations. The address of a memory location is of 16 bits *i.e.* 2 bytes.

The various techniques to specify data for instructions are :

(*i*) 8-bit or 16-bit data may be directly given in the instruction itself.

(*ii*) The address of the memory location, I/O port or I/O device, where data resides, may be given in the instruction itself.

(*iii*) In some instructions only one register is specified. The content of the specified register is one of the operands. It is understood that the other operand is in the accumulator.

(*iv*) Some instructions specify two registers. The contents of the registers are the required data.

(*v*) In some instructions data is implied. The most instructions of this type operate on the content of the accumulator.

Due to different ways of specifying data for instructions, the machine codes of all instructions are not of the same length.

There are three types of the Intel 8085 instructions as described below :

(1) Single-Byte Instruction
(2) Two-Byte Instruction
(3) Three-Byte Instruction

This has already been discussed in Chapter 3; see details.

4.3 ADDRESSING MODES

Each instruction requires certain data on which it has to operate. It has already been explained that there are various techniques to specify data for instructions. These techniques are called *addressing modes*. Intel 8085 uses the following addressing modes :

1. Direct addressing
2. Register addressing
3. Register indirect addressing
4. Immediate addressing.

4.3.1 Direct Addressing

In this mode of addressing the address of the operand (data) is given in the instruction itself.

Examples are :

(1) STA 2400 H — Store the content of the accumulator in the memory location 2400 H.
32, 00, 24 — The above instruction in the code form.

In this instruction 2400H is the memory address where data is to be stored. It is given in the instruction itself. The 2nd and 3rd bytes of the instruction specify the address of the memory location. Here, it is understood that the source of the data is accumulator.

(2) IN 02 — Read data from the port C.
DB, 02 — Instruction in the code form.

In this instruction 02 is the address of the port C of an I/O port from where the data is to be read. Here, it is implied that the destination is the accumulator. The 2nd byte of the instruction specifies the address of the port.

4.3.2 Register Addressing

In register addressing mode the operand is in one of the general purpose registers. The opcode specifies the address of the register(s) in addition to the operation to be performed. Examples are :

(1) MOV A, B — Move the content of register B to register A.
78 — The instruction in the code form.

(2) ADD B	Add the content of register B to the content of register A.
80	The instruction in the code form.

In Example 1 the opcode for MOV A, B is 78H. Besides the operation to be performed the opcode also specifies source and destination registers. The opcode 78H can be written in binary form as 01111000. The first two bits, i.e. 01 are for MOV operation, the next three bits 111 are the binary code for register A, and the last three bits 000 are the binary code for register B.

In Example 2 the opcode for ADD B is 80H. In this instruction one of the operands is register B (its content is one of the data) which is indicated in the instruction itself. In this type of instruction (arithmetic group) it is understood that the other operand is in the accumulator. The opcode 80H in the binary form is 10000000. The first five bits, *i.e.* 10000 specify the operation to be performed, *i.e.* ADD. The last three bits 000 are the binary code for register B for 8085 microprocessor.

4.3.3 Register Indirect Addressing

In this mode of addressing the address of the operand is specified by a register pair. Examples are :

(1) LXI H, 2500 H	Load H-L pair with 2500 H.
MOV A, M	Move the content of the memory location, whose address is in H-L pair (*i.e.* 2500 H) to the accumulator
HLT	Halt.

In the above program the instruction MOV A, M is an example of register indirect addressing. For this instruction the operand is in the memory. The address of the memory is not directly given in the instruction. The address of the memory resides in H-L pair and this has already been specified by an earlier instruction in the program, *i.e.* LXI H, 2500 H.

(2) LXI H, 2500 H	Load the H-L pair with 2500 H.
ADD M	Add the content of the memory location, whose address is in H-L pair (*i.e.* 2500 H), to the content of the accumulator.
HLT	Halt.

In this program the instruction ADD M is an example of register indirect addressing.

4.3.4 Immediate Addressing

In immediate addressing mode the operand is specified within the instruction itself, examples are :

(1) MVI A, 05	Move 05 in register A.
3E, 05	The instruction in the code form.
(2) ADI 06	Add 06 to the content of the accumulator.
C6, 06	The instruction in the code form.

In these instructions the 2nd byte specifies data.

(3) LXI H, 2500 is an example of immediate addressing. 2500 is 16-bit data which is given in the instruction itself. It is to be loaded into H-L pair.

4.3.5 Implicit Addressing

There are certain instructions which operate on the content of the accumulator. Such instructions do not require the address of the operand. Examples are : CMA, RAL, RAR etc.

4.4 STATUS FLAGS

There is a set of five flip-flops which indicate status (conditions) arising after the execution of arithmetic and logic instructions. It has already been discussed in Section 3.1.3.

4.5 SYMBOLS AND ABBREVIATIONS

The symbols and abberviations which have been used while explaineing Intel 8085 instructions are as follows :

Symbol/Abbreviations	*Meaning*
addr	16-bit address of the memory location.
data	8-bit data.
data 16	16-bit data.
r, r_1, r_2	One of the registers A, B, C, D, E, H or L,
A, B, C, D, H, L	8-bit register
A	Accumulator
H-L	Register pair H-L
B-C	Register pair B-C
D-E	Register pair D-E
PSW	Program Status Word
M	Memory whose address is in H-L pair
H	Appearing at the end of a group of digits specifies hexadecimal, *e.g.* 2500H.
rp	One of the register pairs. The representation of a register pair is made as described below : B represents B-C pair, B is high order register and C low order register. D represents D-E pair, D is high order register and E is low order register. H represents H-L pair, H is high order register and L low order register. SP represents 16-bit stack pointer, SPH is high order 8 bits and SPL low order 8 bits of register SP.
rh	The high order register of a register pair.
rl	The low order register of a register pair.
PC	16-bit program counter, PCH is high order 8 bits and PCL low order 8 bits of register PC.
CS	Carry status.
[]	The contents of a register identified within bracket.
[[ ]]	The content of the memory location whose address is in the register pair identified within brackets.
∧	AND operation
∨	OR operation.
∀ or ⊕	EXCLUSIVE-OR
←	Move data in the direction of arrow.
⇔	Exchange contents.