Instruction Set of 8085

Instruction and Data Formats

The various techniques to specify data for instructions are:

1. 8-bit or 16-bit data may be directly given in the instruction itself.
2. The address of the memory location, I/O port or I/O device, where data resides, may be given in the instruction itself.
3. In some instructions, only one register is specified. The content of the specified register is one of the operands.
4. Some instructions specify two registers. The contents of the registers are the required data.
5. 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. It may 1-byte, 2-byte or 3-byte instruction.

Addressing Modes

Each instruction requires some data on which it has to operate. There are different techniques to specify data for instructions. These techniques are called **addressing modes**. Intel 8085 uses the following addressing modes:

* **Direct Addressing**

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

**Example**

**STA 2400H:** It stores the content of the accumulator in the memory location 2400H.

**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.

* **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.

**Example:**

MOV A, B: Move the content of B register to register A.

**78:** The instruction in the code form.

In the above example, 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. 0 1 are for MOV operation, the next three bits 1 1 1 are the binary code for register A, and the last three bits 000 are the binary code for register B.

* **Register Indirect Addressing**

In Register Indirect mode of addressing, the address of the operand is specified by a register pair.

**Example**

* **LXI H, 2500 H** - Load H-L pair with 2500H.
* **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.

* **Immediate Addressing**

In this addressing mode, the operand is specified within the instruction itself.

**Example**

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.

* **Implicit Addressing**

There are certain instructions which operate on the content of the accumulator. Such instructions do not require the address of the operand.

**Example**

CMA, RAL, RAR, etc.

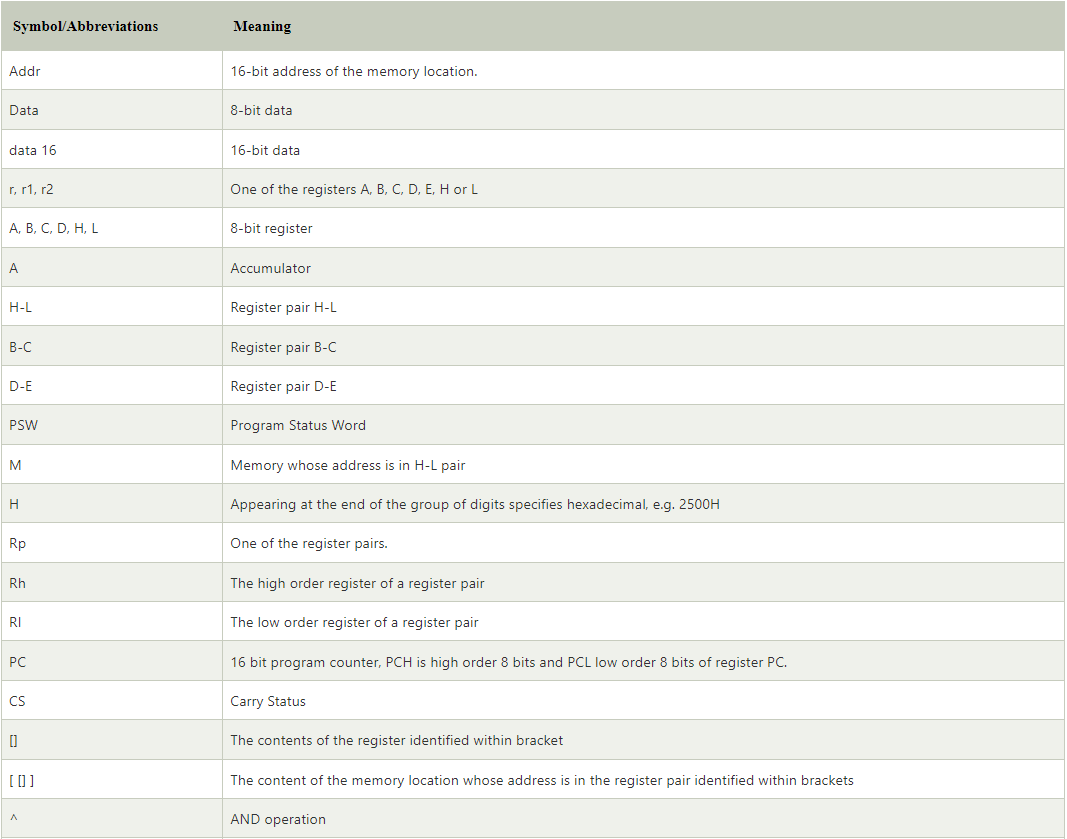
Status Flags

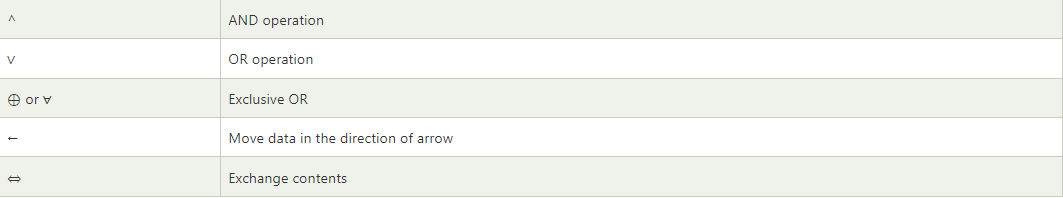
There is a set of five flip-flops which indicate status (condition) arising after the execution of arithmetic and logic instructions. These are:

* Carry Flag (CS)
* Parity Flag (P)
* Auxiliary Carry Flags (AC)
* Zero Flags (Z)
* Sign Flags (S)

Symbols and Abbreviations

The symbol and abbreviations which have been used while explaining Intel 8085 instructions are as follows:





## Intel 8085 Instructions

An **instruction** of a computer is a command given to the computer to perform a specified operation on given data. In microprocessor, the **instruction** set is the collection of the instructions that the microprocessor is designed to execute.

The programmer writes a program in assembly language using these instructions. These instructions have been classified into the following groups:

## Data Transfer Group

Instructions which are used to transfer the data from a register to another register from memory to register or register to memory come under this group.



## Arithmetic Group

The instructions of this group perform arithmetic operations such as addition, subtraction, increment or decrement of the content of a register or a memory





## Logical Group

The instructions in this group perform logical operation such as AND, OR, compare, rotate, etc.



