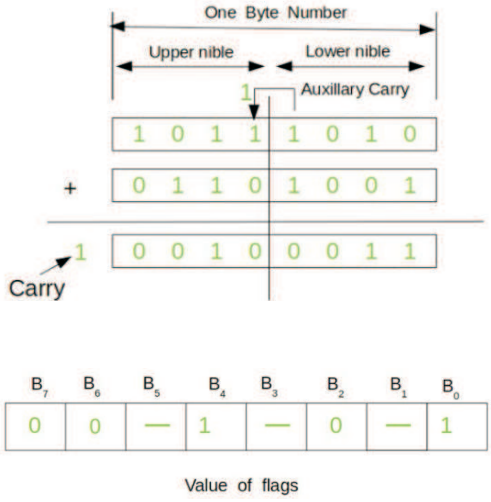
Memory Registers
There are two 16-bit registers used to hold memory addresses. The size of these registers is 16 bits because the memory addresses are 16 bits. They are:

Program Counter
This register is used to sequence the execution of the instructions. The function of the program counter is to point to the memory address from which the next byte is to be fetched. When a byte (machine code) is being fetched, the program counter is incremented by one to point to the next memory location.

Stack Pointer
It is used as a memory pointer. It points to a memory location in read/write memory, called the stack. It is always incremented/decremented by 2 during push and pop operation.

Example
Here two binary numbers are added. The result produced is stored in the accumulator. Now let's check what each bit means. Refer to the below explanation simultaneously to connect them with the example.



Value of flags

- Sign Flag (7th bit): It is reset (0), which means number stored in the accumulator is positive.
- Zero Flag (6th bit): It is reset (0), thus result of the operations performed in the ALU is non-zero.
- Auxiliary Carry Flag (4th bit): We can see that B3 generates a carry which is taken by B4, thus auxiliary carry flag gets set (1).
- Parity Flag (2nd bit): It is reset (0), it means that parity is odd. The accumulator holds odd number of 1's.
- Carry Flag (0th bit): It is set (1), output results in more than 8 bit.

Addressing Modes
The way of specifying data to be operated by an instruction is called addressing mode. 8085 microprocessor there are 5 types of addressing modes:

Immediate Addressing Mode
In immediate addressing mode the source operand is always data. If the data is 8-bit, then the instruction will be of 2 bytes, if the data is of 16-bit then the instruction will be of 3 bytes.

Example

- MVI B 45 (move the data 45H immediately to register B)
- LXI H 3050 (load the HL pair with the operand 3050H immediately)
- JMP address (jump to the operand address immediately)

Register Addressing Mode
In register addressing mode, the data to be operated is available inside the register(s) and register(s) is (are) operands. Therefore the operation is performed within various registers of the microprocessor.

Example

- MOV A, B (move the contents of register B to register A)
- ADD B (add contents of registers A and B and store the result in register A)
- INR A (increment the contents of register A by one)

Direct Addressing Mode
In direct addressing mode, the data to be operated is available inside a memory location and that memory location is directly specified as an operand. The operand is directly available in the instruction itself.

Example

- LDA 2050 (load the contents of memory location into accumulator A)
- LHLD address (load contents of 16-bit memory location into HL register pair)

Register Indirect Addressing Mode
In register indirect addressing mode, the data to be operated is available inside a memory location and that memory location is indirectly specified b a register pair.

Example

- MOV A, M (move the contents of the memory location pointed by the HL pair to the accumulator)
- LDAX B (move contains of BC register to the accumulator)
- LXIH 9570 (load immediate the HL pair with the address of the location 9570)

Implied/Implicit Addressing Mode
In implied/implicit addressing mode the operand is hidden and the data to be operated is available in the instruction itself.

Example

- CMA (finds and stores the l's complement of the contains of accumulator A in A)

- RRC (rotate accumulator A right by one bit)
- RLC (rotate accumulator A left by one bit)

Logical Instructions
Logical instructions are the instructions which perform basic logical operations such as AND, OR, etc. In 8085 microprocessor, the destination operand is always the accumulator. Here logical operation works on a bitwise level.

Following is the table showing the list of logical instructions:

Opcode	Operand	Destination	Example
ANA	R	A = A AND R	ANA B
ANA	M	A = A AND Mc	ANA 2050
ANI	8-bit data	A = A AND 8-bit data	ANI 50
ORA	R	A = A OR R	ORA B
ORA	M	A = A OR Mc	ORA 2050
ORI	8-bit data	A = A OR 8-bit data	ORI 50
XRA	R	A = A XOR R	XRA B
XRA	M	A = A XOR Mc	XRA 2050
XRI	8-bit data	A = A XOR 8-bit data	XRI 50
CMA	none	A = 1's compliment of A	CMA
CMP	R	Compares R with A and triggers the flag register	CMP B
CMP	M	Compares Mc with A and triggers the flag register	CMP 2050
CPI	8-bit data	Compares 8-bit data with A and triggers the flag register	CPI 50
RRC	none	Rotate accumulator right without carry	RRC
RLC	none	Rotate accumulator left without carry	RLC
RAR	none	Rotate accumulator right with carry	RAR
RAL	none	Rotate accumulator left with carry	RAR
CMC	none	Compliments the carry flag	CMC
STC	none	Sets the carry flag	STC

In the table,
R stands for register
M stands for memory
Mc stands for memory contents

ROTATE Instructions
ROTATE is a logical operation of 8085 microprocessor. It is a 1 byte instruction. This instruction does not require any operand after the opcode. It operates the content of accumulator and the result is also stored in the accumulator. The Rotate instruction is used to rotating the bits of accumulator.

Types of ROTATE Instruction

There are 4 categories of the ROTATE instruction

- Rotate accumulator left (RLC)
- Rotate accumulator left through carry (RAL)
- Rotate accumulator right (RRC)
- Rotate accumulator right through carry (RAR)

Among these four instructions; two are for rotating left and two are for rotating right. All of them are explained briefly in the following sections:

Rotate accumulator left (RLC)
In this instruction, each bit is shifted to the adjacent left position. Bit D7 becomes DO. Carry flag CY is modified according to the bit D7.

Example
A = D7 D6 D5 D4 D3 D2 D1 D0
//before the instruction A = 10101010; CY=0
//after 1st RLC A = 01010101; CY=1
//after 2nd RLC A - 10101010; CY=0

Rotate accumulator left through carry (RAL)
In this instruction, each bit is shifted to the adjacent left position. Bit D7 becomes the carry bit and the carry bit is shifted into DO. Carry flag CY is modified according to the bit D7.

Example
A = D7 D6 D5 D4 D3 D2 D1 DO
//before the instruction A - 10101010; CY=0
//after 1st RAL A = 01010100; CY=1
//after 2nd RAL A = 10101001; CY=0

Rotate accumulator right (RRC)
In this instruction, each bit is shifted to the adjacent right position. Bit D0 becomes D7. Carry flag CY is modified according to the bit D0.

Example
A = D7 D6 D5 D4 D3 D2 D1 D0
//before the instruction A = 10000001; CY=0
//after 1st RRC A = 11000000; CY=1
//after 2nd RRC A = 01100000; CY=0

Rotate accumulator right through carry (RAR)
In this instruction, each bit is shifted to the adjacent right position. Bit D0 becomes the carry bit and the carry bit is shifted into D7. Carry flag CY is modified according to the bit D0.

Example A - D7 D6 D5 D4 D3 D2 D1 D0
//before the instruction A - 10000001; CY=0
//after 1st RAR A = 01000000; CY=1
//after 2nd RAR A = 10100000; CY=0

Application of ROTATE Instructions
The ROTATE instructions are primarily used in arithmetic multiply and divide operations and for serial data transfer.

Example
If A is 0000 1000 = 08H

By rotating 08H right: A = 0000 0100 = 04H This is equivalent to dividing by 2.
By rotating 08H left: A = 0001 0000 = 10H This is equivalent to multiplying by 2.

However, these procedures are invalid when logic 1 is rotated left from D7 to D0 or vice versa. For example, if 80H is rotated left it becomes 01H.