



STACK

- The stack is a group of memory location in the R/W memory that is used for temporary storage of binary information during execution of program.
 - The starting of memory location of the stack is define in the main program and space is reserved.
 - The stack is a LIFO (last in ,first out) data structure implimented in the RAM area and is used to store address and data when the microprocessor branches to the subroutine.
- 

STACK POINTER

- SP is a special purpose 16-bit register, used as a memory pointer.
 - The stack pointer will hold the address of the top location of the stack.
 - The Stack is defined by loading 16-bit address in the stack pointer.
 - Each time when data is loaded into stack, stack pointer gets decremented by 1.
 - Inversely, it is incremented by 1 when data is retrieved from stack.
- 

INSTRUCTIONS RELATED TO STACK

□ LXI SP, data 16-bit

- Using this instruction, we can load 16-bit immediate data/address on the Stack Pointer (SP).
- It occupies 3-Bytes in the memory.

Example.

Let us consider a sample instruction LXI SP, 4050H

Address	Hex Codes	Mnemonic	Comment
2000	31	LXI SP, 4050H	Initializing SP register with 16-bit data 4050H
2001	50		Low order Byte of the data
2002	40		High order Byte of the data

□ INX SP

- INX SP instruction is used to increment the SP contents by 1.
- This instruction occupies only 1-Byte in memory.
- Flags are not affected.

$$\begin{array}{c} \boxed{\mathbf{X}} \\ \mathbf{SP} \end{array} + 1 \rightarrow \begin{array}{c} \boxed{\mathbf{X+1}} \\ \mathbf{SP} \end{array}$$

□ DCX SP

- DCX SP instruction is used to decrement the SP contents by 1.
- This instruction occupies only 1-Byte in memory.
- No flag is affected.

$$\begin{array}{c} \boxed{\mathbf{X}} \\ \mathbf{SP} \end{array} + (-1) \rightarrow \begin{array}{c} \boxed{\mathbf{X - 1}} \\ \mathbf{SP} \end{array}$$

□ DAD SP

- DAD SP instruction is a special case of DAD rp instruction.
- In this instruction contents of HL and SP will get added and sum thus produced will get stored onto HL register pair.
- It occupies only 1-Byte in memory.
- Only carry flag is affected.

EXAMPLE-

We are considering that HL and SP registers are having initial contents as 4050H and 5050H. So after execution of DAD SP instruction, the addition result will be 90A0H which will get stored on the HL register pair.

Registers	Before	After
(HL)	5050H	90A0H
(SP)	4050H	4050H
(F)	Any values	Cy=0, No change in other flag bits

□ SPHL

- SPHL is an instruction with the help of which Stack Pointer will get initialized with the contents of register pair HL.
- It is an indirect way of initializing the stack pointer. It occupies only 1-Byte in memory, compared to the other instruction LXI SP instruction, which is 3-Bytes long used for initializing SP on the other hand.
- Due to this advantage, SPHL can be useful when SP is required to get initialized to a specific value a number of times in a program.

EXAMPLE-

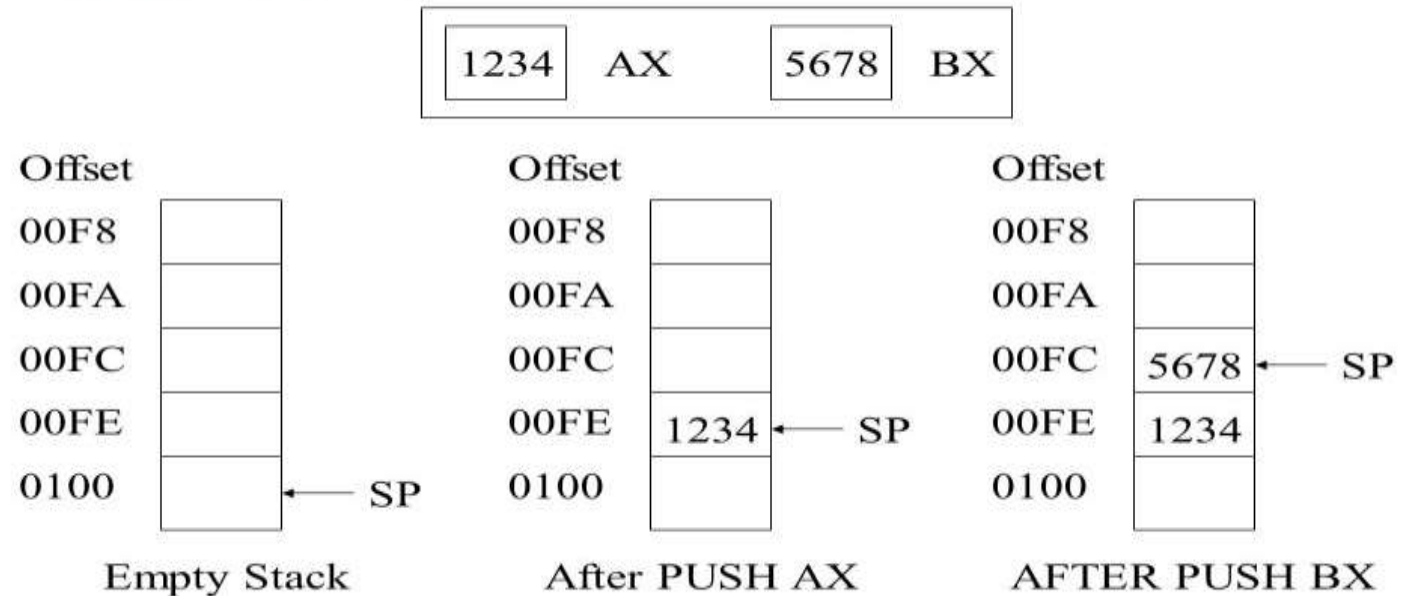
Registers	Before	After
(HL)	AABBH	AABBH
(SP)	CCDDH	AABBH

□ PUSH Rp

- PUSH Rp instruction stores contents of register pair Rp by pushing it into two locations above the top of the stack. Rp stands for one of the following register pairs.
- Rp = BC, DE, HL, or A-PSW
- As Rp can have any of the four values, there are four opcodes for this type of instruction. It occupies only 1-Byte in memory.

EXAMPLE-

The PUSH Instruction

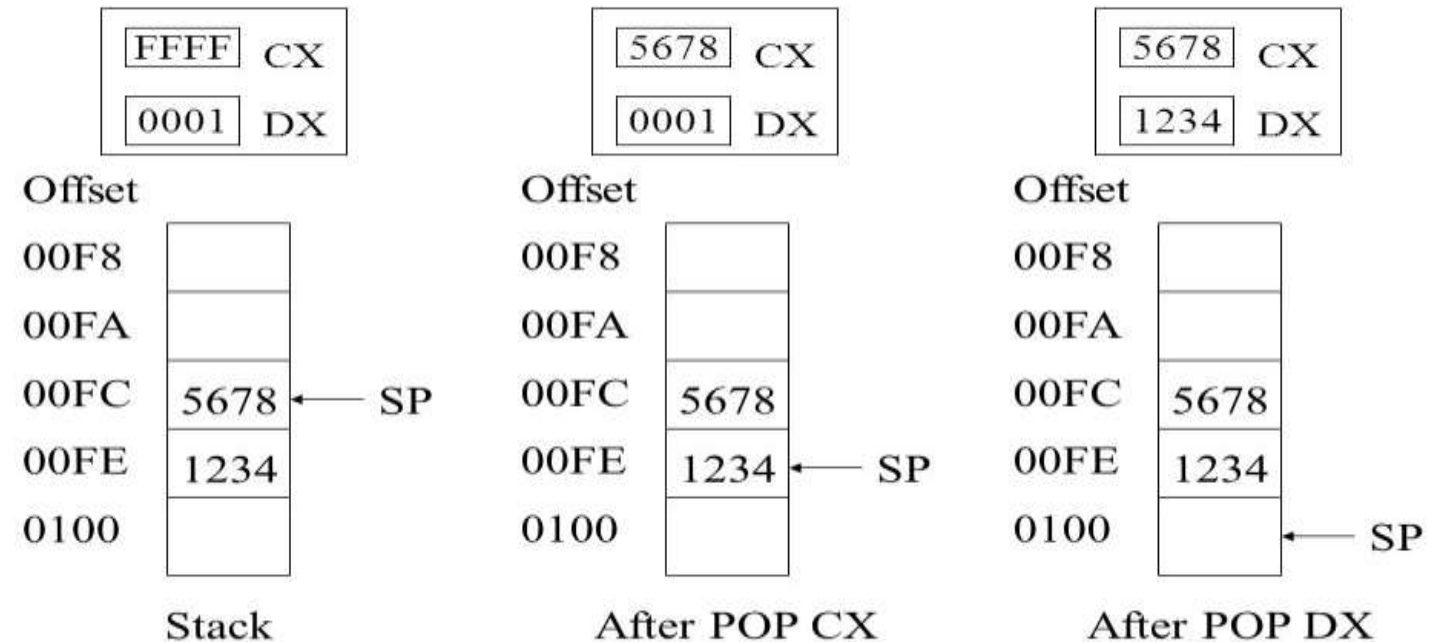


□ POP Rp

- Instruction set, with the mnemonic **POP**, we can pop out 2-Bytes from the top of the stack through Rp i.e. register pair e.g. BC, DE, HL or AF.
- Here AF is a register pair formed with Flag and Accumulator registers and also known as PSW (Processor Status Word). In PSW, Accumulator is the MS Byte, and Flags register is the LS Byte.
- It takes 6 T state

EXAMPLE -

The POP Instruction



□ XTHL

- Instruction set, XTHL is a mnemonic that stands for “exchange Top of stack with HL”.
- This instruction exchanges the contents of the top two locations of the stack with the contents of register pair HL.
- Here it is not an exchange between SP with HL . It occupies only 1-Byte in memory.
- $(2 \text{ read} + 2 \text{ write}) * 3 = 12 + 4 \text{ opcode fetch} = 16 \text{ T}$

