Understanding Stack Pointer with CALL and RET

Stack Pointer & CALL/RET Explained

What is the Stack Pointer (SP)?

- The Stack is a memory area used to store temporary data like return addresses.
- The Stack Pointer (SP) always points to the top of the stack.
- In 8086, the stack grows downward (from high memory to low memory).
- Stack works in LIFO: Last In, First Out.

What Happens During CALL?

Example:

```
0100:  CALL MyFunc
0103:  MOV AX, BX   ; next instruction
```

Step-by-step:

1. Before CALL:
   - SP = 0FFEh, IP = 0100h

2. CALL MyFunc:
   - CPU stores 0103h onto the stack.
   - SP becomes 0FFCh.
   - Memory[0FFCh] = 0103h (return address)
   - IP is updated to address of MyFunc

3. RET:
   - CPU pops 0103h from the stack into IP.
   - SP becomes 0FFEh.
   - Program continues from 0103h.

Understanding Stack Pointer with CALL and RET

Summary Table:

Action	SP	Stack Content	IP
Before	0FFEh	-	0100h
After CALL	0FFCh	[0FFCh] = 0103h	MyFunc
After RET	0FFEh	-	0103h