Registers in the 8086 Microprocessor

- In computer architecture, a processor register is a quickly accessible location available to a computer's central processing unit (CPU).

- Registers usually consist of a small amount of fast storage, although some registers have specific hardware functions, and may be read-only or write-only. Registers are normally measured by the number of bits they can hold, for example, an "8-bit register", "32-bit register" or a "64-bit register".

- In the 8086 processor , the registers are categorized into mainly four types:

  1. **General Purpose Registers**
  2. **Segment Registers**
  3. **Pointers and Index Registers**
  4. **Flag or Status Register**

1) General Purpose Registers

✓ General Purpose Registers are a kind of registers which can store both data and addresses. All general registers of the Intel 8086 microprocessor can be used for arithmetic and logic operations and data movement.

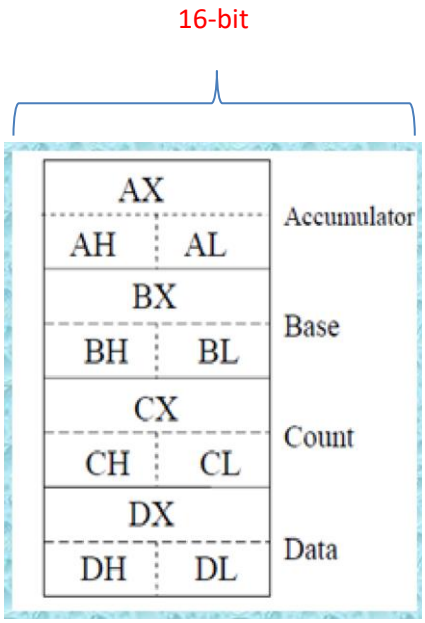
✓ There are 4 general-purpose registers of 16-bit length each.

✓ Each of them is further divided into two subparts of 8-bit length each: one high, which stores the higher-order bits and another low which stores the lower order bits

Higher order-8bit Lower order-8bit

i. AX = [AH:AL]
ii. BX = [BH:BL]
iii. CX = [CH:CL]
iv. DX = [DH:DL]

Ex: Mov AL, 65
تعني خزن القيمة 65 في المسجل
AL
65=01000001



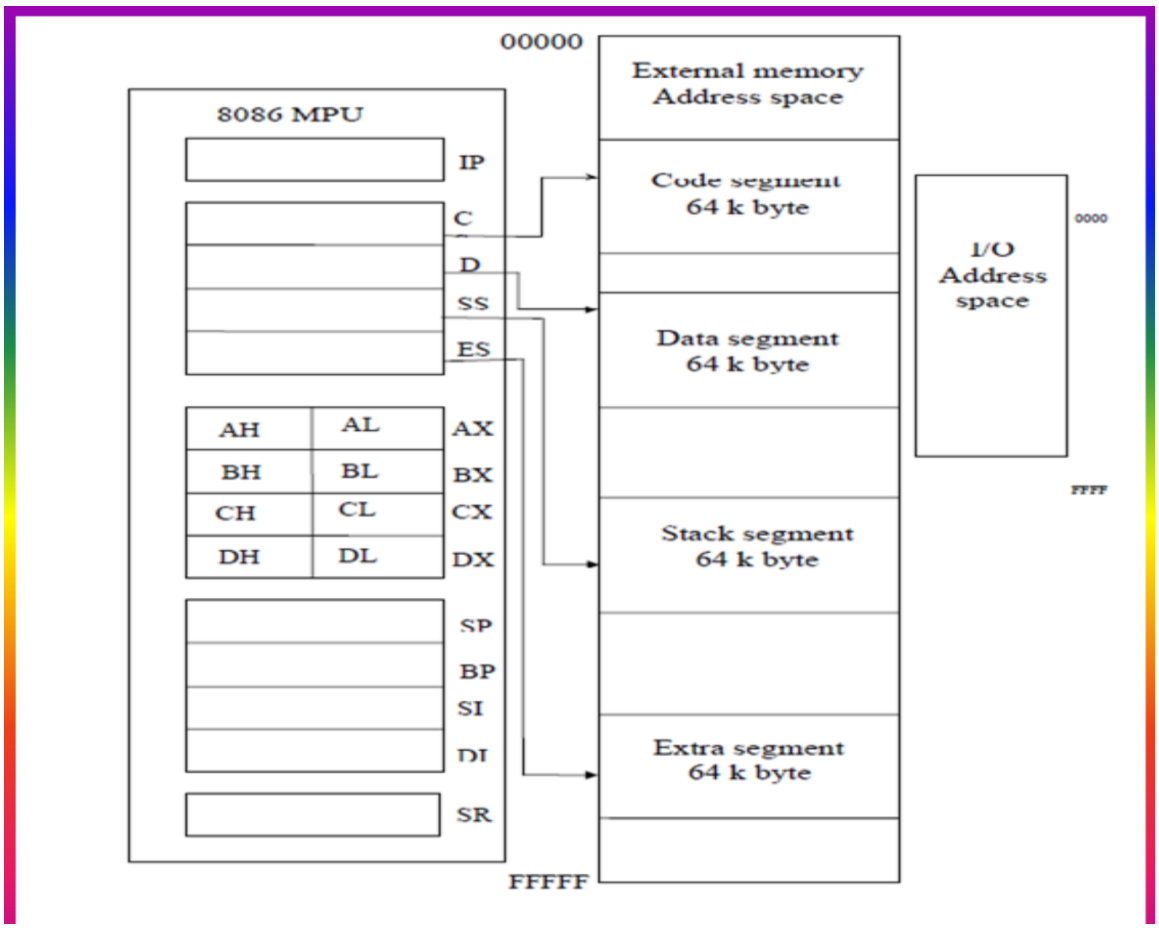


Fig. (2.4): Software Model of the 8086 microprocessor.

Table : General Purpose registers	
Register	**Operation**
AX	**Accumulator register:** It gets used in arithmetic, logic and data transfer instructions, I/O port access.
BX	**Base register:** BX register usually contains a data pointer used for based. It is used to hold the address of a procedure or variable. The BX register can also perform arithmetic and data movement .
CX	**Count register:** used in Loop, shift/rotate instructions and as a counter in string manipulation
DX	**Data regist**er: used as a port number in I/O operations. It is also used in multiplication and division.

2) Segment Registers

There are 4 segment registers in **8086 Microprocessor** and each of them is of 16 bit. The code and instructions are stored inside these different segments. Used in the stack.

1. **Code Segment (CS) Register:**
   Containing address of all executable instructions (code) in a program. The user cannot modify the content of these registers. Only the microprocessor's compiler can do this.
2. **Data Segment (DS) Register:**
   Containing address of base location for variables. The user can modify the content of the data segment.
3. **Stack Segment (SS) Registers:**
   The SS is used to store the information about the memory segment. The operations of the SS are mainly Push and Pop.
4. **Extra Segment (ES) Register:**
   By default, the control of the compiler remains in the DS where the user can add and modify the instructions. If there is less space in that segment, then ES is used. ES is also used for copying purpose.

CS 64K	4000H
DS 64K	3000H
SS 64K	2000H
ES 64K	1000H

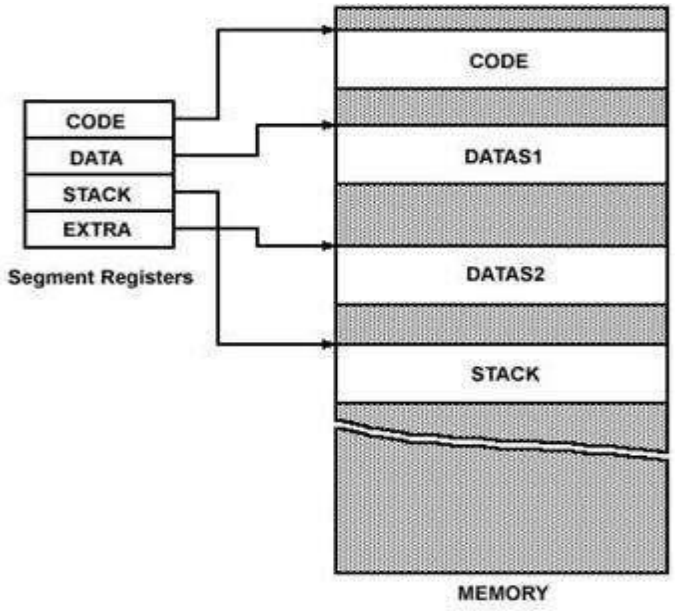


Fig. (2.5): Segment Registers.

3) Pointers and Index Registers

These registers contain the offset of data and instructions. The term offset refers to the distance of a variable, label, or instruction from its base segment. The pointers will always store some address or memory location. In **8086 Microprocessor**, they usually store the offset through which the actual address is calculated.

1. **Base Pointer (BP):**
   The Base pointer stores the base address of the memory. Also, it acts as an offset for Stack Segment (SS).
2. **Stack Pointer (SP):**
   The Stack Pointer points at the current top value of the Stack. Like the BP, it also acts as an offset to the Stack Segment (SS).
3. **Source Index (SI):** It stores the offset address of the source in string manipulation instructions.
4. **Destination Index (DI):** It stores the offset address of the Destination in string manipulation instructions.

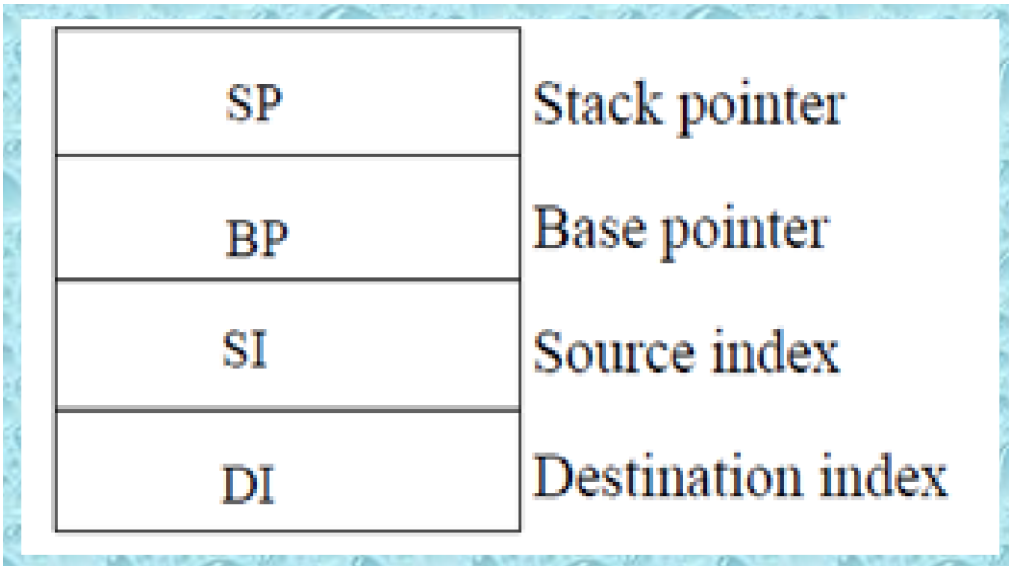


Fig. (2.6): Pointer and index registers

Instruction Pointer (IP):

The instruction pointer (IP or program counter) usually stores the address of the next instruction that is to be executed. The instruction pointer and the code segment register (CS) combine to form the complete address of the next instruction.

4) Flag or Status Register

- ✓ These flags tell about the status of the processor after any arithmetic or logical operation. It is done by setting the individual bits called flags. There are two kinds of FLAGS; Status FLAG and Control FLAG. Status FLAG reflect the result of an operation executed by the processor. The control FLAG enables or disables certain operations of the processor.
- ✓ The Flag or Status register is a 16-bit register which contains 9 flags, and the remaining 7 bits are idle (خاملة او عاطلة) in this register.

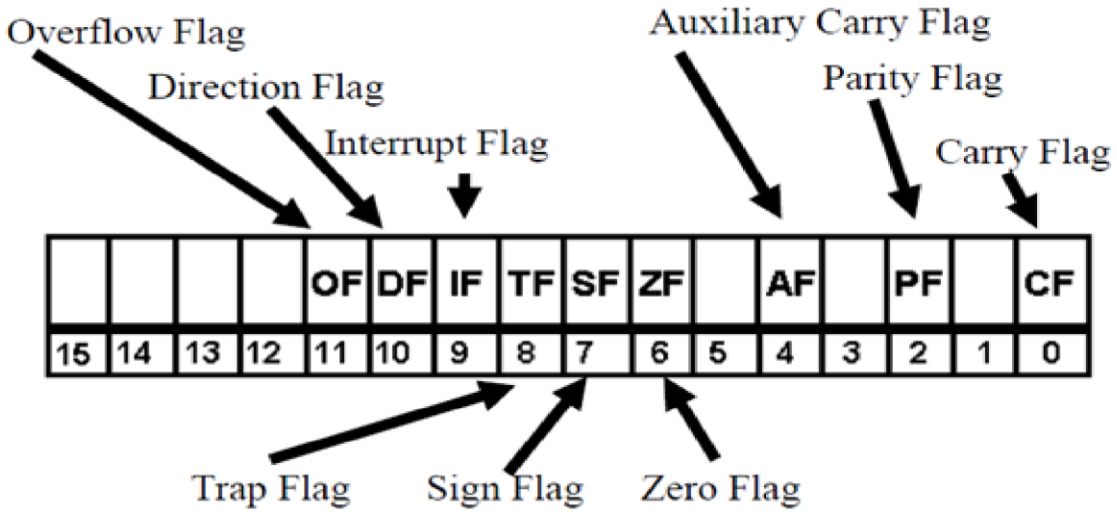


Fig. (2.7): Flag register

Status Flag Registers:

1. Overflow Flag (OF):

Overflow Flag is set to **1** when there is a **signed overflow**. For example, when you add bytes **100 + 50** (result is not in range -128…127).

2. Sign Flag (SF):

Sign Flag is set to **1** when result is **negative**. When result is **positive** it is set to **0**. This flag takes the value of the most significant bit.

3. Zero Flag (ZF):

Zero Flag (ZF) is set to **1** when result is **zero**. For non-zero result this flag is set to **0**.

4. Auxiliary Flag (AF):

Auxiliary Flag is set to **1** when there is an **unsigned overflow** for low nibble (4 bits).

5. Parity Flag (PF):

Parity Flag is set to **1** when there is even number of one bits in result, and to **0** when there is odd number of one bits.

6. Carry Flag (CF);

Carry Flag is set to **1** when there is an **unsigned overflow**. For example when you add bytes **255 + 1** (result is not in range 0…255). When there is no overflow this flag is set to **0**.

Control Flag Registers:

1. Direction Flag (DF):

Direction Flag is used by some instructions to process data chains, when this flag is set to **0** – the processing is done forward, when this flag is set to **1**the processing is done backward.

2. Interrupt Enable Flag (IF):

interrupt : قطع أو أعتراض

When Interrupt Enable Flag is set to **1** CPU reacts to interrupts from external devices.

3. Trap Flag (TF):

Trap Flag is used for on-chip debugging. if this flag is set, the processor enters the single step execution mode