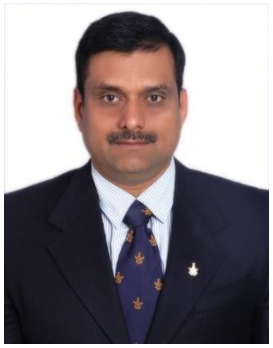
Microprocessors and Interfaces: 2021-22
Lecture 12
8086 Arithmetic Instructions : Part-1

By Dr. Sanjay Vidhyadharan



2. Arithmetic Instructions

- The arithmetic instructions include :
  - addition, subtraction, multiplication, division,
  - comparison, increment, and decrement.

Addition

- Addition (ADD) appears in many forms in the microprocessor.
- A second form of addition, called **add-with-carry**, is introduced with the ADC instruction.
- The only types of addition *not* allowed are memory-to-memory and segment register.
  - segment registers can only be moved, pushed, or popped
- Increment instruction (INC) is a special type of addition that adds 1 to a number.

Addition

ADD Destination, Source

(Source) + (Destination) ⟶ (Destination)

Source may be an immediate number, a register or a memory location specified by any one of the addressing methods.

Destination may be a register or a memory location specified by any one of the addressing methods

Both source and destination in an instruction cannot be memory locations

Source and Destination must be of same size

All Flags Affected.

Addition

MOV CL, 73H

MOV BL, 4FH

ADD CL, BL

Result in CL = C2H 11000010

CF = 0, PF = 0, AF = 1, ZF = 0, SF = 1, OF = 1

Addition

Add data placed in Memory 8000 with that of data placed in 8001 and 8002 and store result in 8003.

```
MOV AL, [8000]
ADD AL, [8001]
ADD AL, [8002]
MOV     [8003], AL
```

Addition with carry

ADC DESTINATION, SOURCE

(SOURCE) + (DESTINATION) + CF ⟶ (DESTINATION)

Useful for multibyte addition of data

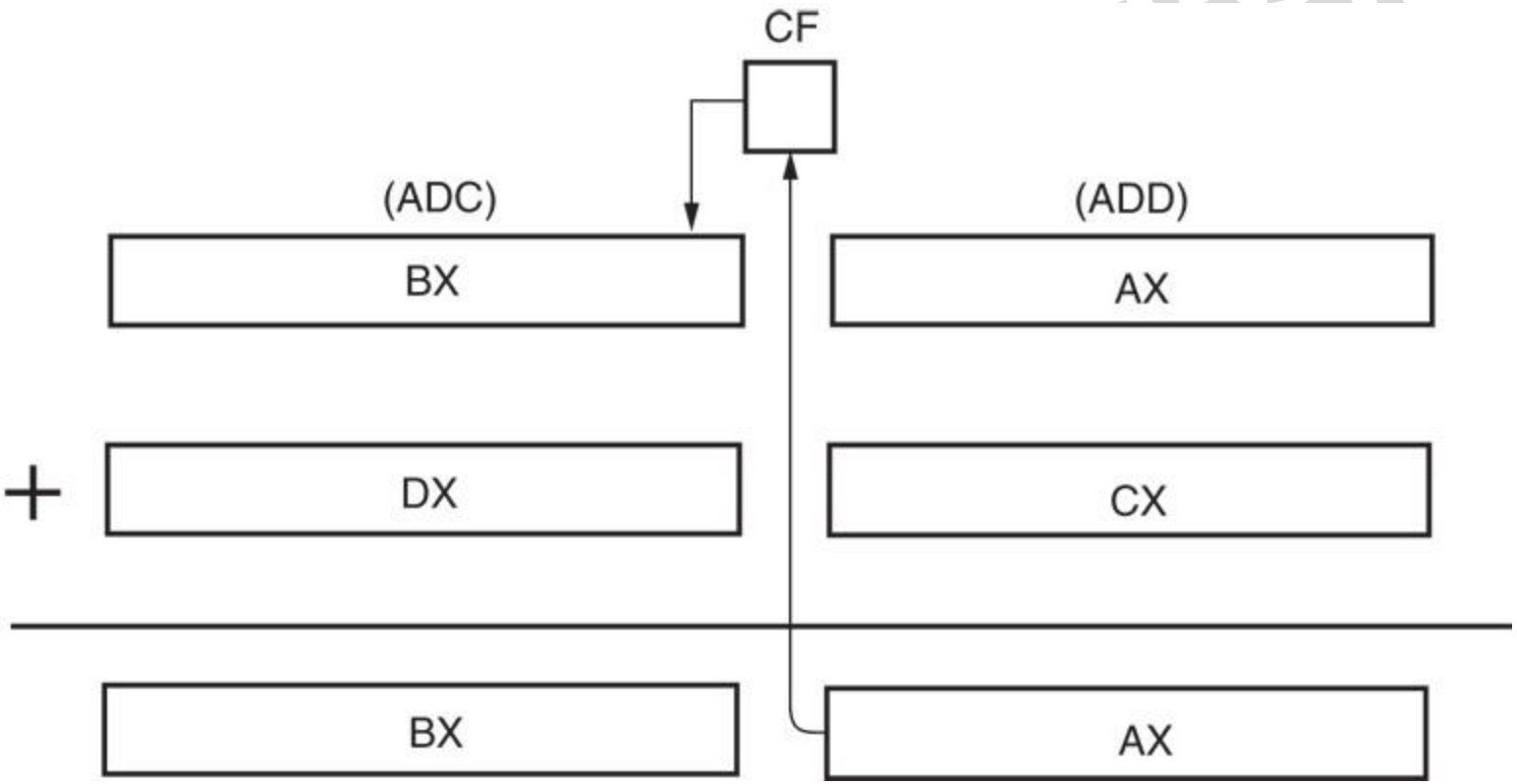
Ex: ADD AX, CX

ADC BX, DX

32 bit addition.

Addition

Addition-with-carry showing how the carry flag (C) links the two 16-bit additions into one 32-bit addition.



Add 2 –6 byte nos. stored in location 20000H and 21000H. Store the result starting from location 21000H

```
        MOV   AX,   2000H
        MOV   DS,   AX
        MOV   SI,   0000H
        MOV   DI,   1000H
        MOV   CL,   06H
        MOV   BL,   00
        CLC
X1:     MOV   AL,   [SI]
        ADC   [DI],  AL
        INC   SI
        INC   DI
        DEC   CL
        JNZ   X1
        JNC   X2
        INC   BL
X2:     MOV   [DI],  BL
```

Subtraction

SUB DESINATION, SOURCE

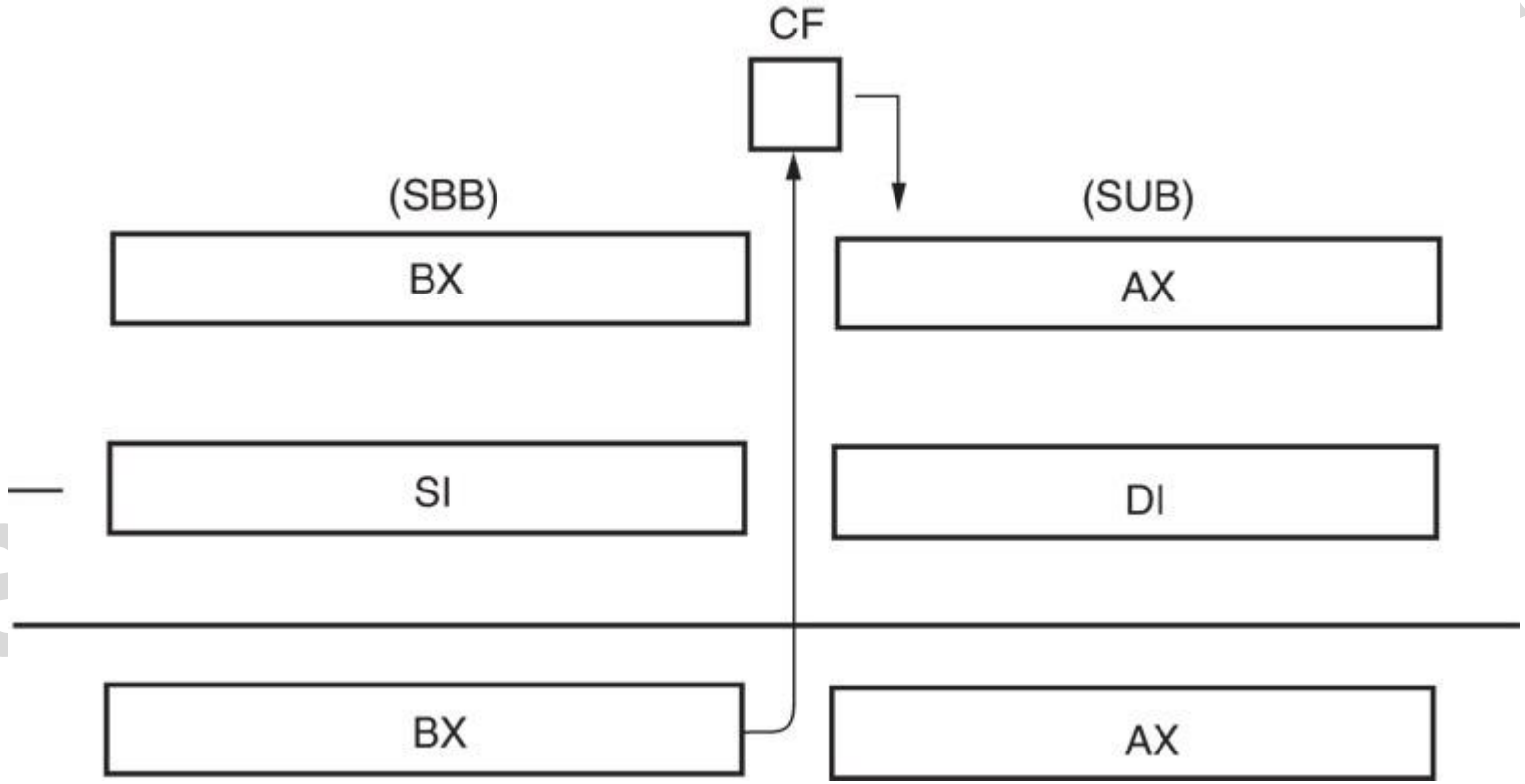
(DESTINATION) – (SOURCE) ⟶ (DESTINATION)

SBB DESTINATION, SOURCE

(DESTINATION) - (SOURCE) – CF ⟶ (DESTINATION)

Subtraction

Subtraction-with-borrow showing how the carry flag (C) propagates the borrow.



Subtraction

```
                                    2'compl.
 -1                   -1            -------->  1111 1111
44H 62H---> 0100 0100  0110 0010--------->  0100 0100  0110 0010
                                    2'compl.
-13H F1H--->-0001 0011 -1111 0001-------->+1110 1101 +0000 1111
 -------     ---------  ---------           ---------  ---------
 30H 71H---> 0011 0000  1111 0001           0011 0000  0111 0001(30H 71H)
```

C-Carry Flag This flag is set when there is a carry out of MSB in case of addition or a borrow in case of subtraction. For example, when two numbers are added, a carry may be generated out of the most significant bit position. The carry flag, in this case, will be set to '1'. In case, no carry is generated, it will be '0'. Some other instructions also affect or use this flag and will be discussed later in this text.

Compare Instruction

- ✓ Compare instruction is a subtraction that changes only the flag bits.
- ✓ CMP Destination, Source

	CF	ZF	SF
Equal	0	1	0
dest > source	0	0	0
dest < source	1	0	1

Ex: CMP CL, [BX]
CMP AX, 2000H
CMP [DI], CH

Compare Instruction

- Destination operand never changes
- Useful for checking the contents of a register or a memory location against another value.
- A CMP is normally followed by a conditional jump instruction, which tests the condition of the flag bits.

Multiplication

- Performed on bytes, words
  - can be signed (IMUL) or unsigned integer (MUL)
- Product after a multiplication always a double-width product.
  - two 8-bit numbers multiplied generate a 16-bit product; two 16-bit numbers generate a 32-bit;

8-Bit Multiplication

- With 8-bit multiplication, the multiplicand is always in the AL register, signed or unsigned.
  - multiplier can be any 8-bit register or memory location

- Immediate multiplication is not allowed unless the special signed immediate multiplication (in 80186) instruction appears in a program.

- The multiplication instruction contains one operand because it always multiplies the operand times the contents of register AL.

16-Bit Multiplication

- Word multiplication is very similar to byte multiplication.
- AX contains the multiplicand instead of AL.
  - 32-bit product appears in DX–AX instead of AX
- The DX register always contains the most significant 16 bits of the product; AX contains the least significant 16 bits.
- As with 8-bit multiplication, the choice of the multiplier is up to the programmer.

Division

- Occurs on 8- or 16-bit numbers.
  - signed (IDIV) or unsigned (DIV) integers
- Dividend is always a double-width dividend, divided by the operand.
- There is no immediate division instruction available to any microprocessor.

Division

- A division can result in two types of errors:
  - attempt to divide by zero
  - other is a divide overflow, which occurs when a small number divides into a large number
- In either case, the microprocessor generates an interrupt if a divide error occurs.
- In most systems, a divide error interrupt displays an error message on the video screen.

8-Bit Division

- Uses AX to store the dividend divided by the contents of any 8-bit register or memory location.
- Quotient moves into AL after the division with AH containing a whole number remainder.
  - quotient is positive or negative; remainder always assumes sign of the dividend; always an integer

8-Bit Division

- Numbers usually 8 bits wide in 8-bit division .
  - the dividend must be converted to a 16-bit wide number in AX ; accomplished differently for signed and unsigned numbers

- **CBW/CWD** (convert byte to word) instruction performs this sign conversion.

16-Bit Division

- Sixteen-bit division is similar to 8-bit division
  - instead of dividing into AX, the 16-bit number is divided into DX–AX, a 32-bit dividend

- As with 8-bit division, numbers must often be converted to the proper form for the dividend.
  - if a 16-bit unsigned number is placed in AX, DX must be cleared to zero

- In the 80386 and above, the number is zero-extended by using the **MOVZX** instruction.

Increment

- The INC instruction adds 1 to any register or memory location, except a segment register.
- The size of the data must be described by using the BYTE PTR, WORD PTR directives.
- The assembler program cannot determine if the INC [DI] instruction is a byte-, word-sized increment.

Increment

- INC Destination

  Ex: INC CX ; Add 1 to the contents of CX.

  EX: INC BYTEPTR [BX] ; Increments the byte pointed to by the contents of BX.

INC Destination

Destination –Register or memory location (specified in 24 diff ways)

•(AF, OF, PF, SF, ZF affected, CF not affected)

•INC BL

•INC BX

•INC EDX

MOD / R/M	Memory Mode (EA Calculation)			Register Mode	
	00	01	10	W=0	W=1
000	(BX)+(SI)	(BX)+(SI)+d8	(BX)+(SI)+d16	AL	AX
001	(BX) + (DI)	(BX)+(DI)+d8	(BX)+(DI)+d16	CL	CX
010	(BP)+(SI)	(BP)+(SI)+d8	(BP)+(SI)+d16	DL	DX
011	(BP)+(DI)	(BP)+(DI)+d8	(BP)+(DI)+d16	BL	BX
100	(SI)	(SI) + d8	(SI) + d16	AH	SP
101	(DI)	(DI) + d8	(DI) + d16	CH	BP
110	d16	(BP) + d8	(BP) + d16	DH	SI
111	(BX)	(BX) + d8	(BX) + d16	BH	DI

Add 2 –6 byte nos. stored in location 20000H and 21000H. Store the result starting from location 21000H

```
        MOV   AX,   2000H
        MOV   DS,   AX
        MOV   SI,   0000H
        MOV   DI,   1000H
        MOV   CL,   06H
        MOV   BL,   00
        CLC
X1:     MOV   AL,   [SI]
        ADC   [DI],  AL
        INC   SI
        INC   DI
        DEC   CL
        JNZ   X1
        JNC   X2
        INC   BL
X2:     MOV   [DI],  BL
```

DEC Destination

Destination –Register or memory location (specified in 24 diff ways)

•(AF, OF, PF, SF, ZF affected, CF not affected)

•DEC BL

•DEC BX

INC/DEC the contents of a Memory location

- Specify the data size in memory

use directive

- BYTE PTR, WORD PTR, DWORD PTR
- INC WORD PTR [BX]
- INC BYTE PTR[BX]
- BX-1000_H DS-2000_H

Consider

21000	FF
21001	00

After execution of INC WORD PTR [BX]

21000	00
21001	01

After execution of INC BYTE PTR [BX]

21000	00
21001	00

Thankyou