Lecture 5

8086 programming-Integer instructions and computations

- Subtraction subgroup of instruction set is similar to the addition subgroup.
- For subtraction the carry flag **CF** acts as borrow flag
- If borrow occur after subtraction then **CF = 1**.
- If NO borrow occur after subtraction then **CF = 0**.
- Subtraction subgroup content instruction shown in table below.

Mnemonic	Meaning	Format	Operation	Flags affected
SUB	Subtract	SUB D,S	(D)–(S)→(D) Borrow→(CF)	OF, SF, ZF, AF, PF, CF
SBB	Subtract with borrow	SBB D,S	(D)–(S)–(CF)→(D)	OF, SF, ZF, AF, PF, CF
DEC	Decrement by 1	DEC D	(D)–1→(D)	OF, SF, ZF, AF, PF
NEG	Negate	NEG D	0–(D)→(D) 1→(CF)	OF, SF, ZF, AF, PF, CF
DAS	Decimal adjust for subtraction	DAS		SF, ZF, AF, PF, CF OF undefined
AAS	AASCII adjust for subtraction	AAS		AF, CF, OF, SF, ZF, PF undefined

(a)

Destination	Source
Register	Register
Register	Memory
Memory	Register
Accumulator	Immediate
Register	Immediate
Memory	Immediate

(b)

Destination
Reg16
Reg 8
Memory

(c)

Destination
Register
Memory

(d)

(a) Subtraction arithmetic operations
(b) Allowed operands for SUB and SBB instructions
(c) Allowed operands for DEC instruction
(d) Allowed operands for NEG instruction

- **SBB** is primarily used for multiword subtract operations.
- Another instruction called **NEG** is available in the subtraction subgroup
- The **NEG** instruction evaluate the 2'complement of an operand

Example 12: What is the result of executing the following instruction sequence?

```
NEG BX
```


م.م اثير هادي عيسى الرماحي

Solution :

Multiplication and Division instructions:

Mnemonic	Meaning	Format	Operation	Flags affected
MUL	Multiply (unsigned)	MUL S	(AL).(S8)→(AX) (AX)·(S16)→(DX).(AX)	OF, CF, SF, ZF, AF, PF, undefined
DIV	Division (unsigned)	DIV S	(1) Q((AX)/(S8))→(AL) R((AX)/(S8))→(AH) (2) Q((DX, AX)/(S16))→(AX) R((DX,AX)/(S16))→(DX) If Q is FF_{16} in case (1) or $FFFF_{16}$ in case (2), then type 0 interrupt occurs.	OF, SF, ZF, PF, CF undefined
IMUL	Integer multiply (signed)	IMUL S	(AL)·(S8)→(AX) (AX)·(S16)→(DX),(AX)	OF, CF, SF, ZF, AF, PF undefined
IDIV	Integer divide (signed)	IDIV S	(1) Q((AX)/(S8))→(AX) R((AX)/(S8))→(AH) (2) Q((DX,AX)/(S16))→(AX) R((DX,AX)(S16))→(DX) If Q is positive and exceeds $7FFF_{16}$, or if Q is negative and becomes less than 8001_{16}, then type 0 interrupt occurs	OF, SF, ZF, AF, PF, CF undefined
AAM	Adjust AL for multiplication	AAM	Q((AL)/10))→AH R((AL)/10))→AL	SF, ZF, PF, OF, AF, CF undefined
AAD	Adjust AX for division	AAD	(AH) · 10 + AL → AL 00 → AH	SF, ZF, PF, OF, AF, CF undefined
CBW	Convert byte to word	CWD	(MSB of AL)→(All bits of AH)	None
CWD	Convert word to double word	CWD	(MSB of AX)→(All bits of DX)	None

(a)

Source
Reg 8
Reg 16
Mem 8
Mem 16

(b)

(a) Multiplication and division instructions (b) Allowed operands.

- **MUL** instruction used to multiply unsigned number in **AL** with an 8 bit operand (in register or memory) and store the result in **AX**
- **MUL** instruction used to multiply unsigned number in **AX** with an 16 bit operand (in register or memory) and store the result in **DX** and **AX**
- Note that the multiplication of two 8-bit number is 16-bit number
- Note that the multiplication of two 16-bit number is 32-bit number
- **IMUL** is similar to **MUL** but is used for signed numbers
- Note that the destination operand for instructions **MUL** and **IMUL** is **AL** or **AX**

م.م اثير هادي عيسى الرماحي

Example 13: What is the result of executing the following instruction?

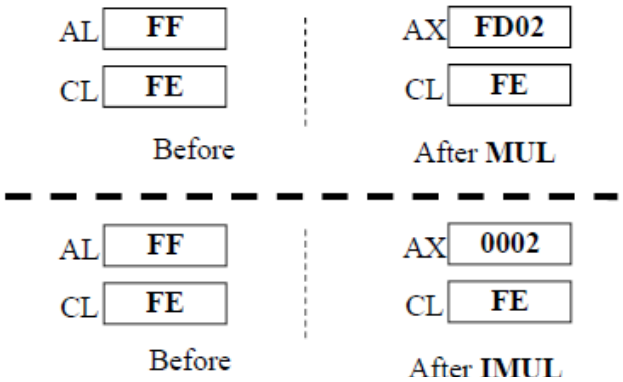
```
MUL CL
```

What is the result of executing the following instruction?

```
IMUL CL
```

Assume that AL contains FFH (the 2'complement of the number 1), CL contain FEH (the 2'complement of the number 2).

Solution :



Ex1:Assume that each instruction starts from these values:

AL = 85H, BL = 35H, AH = 0H

1. MUL BL =AL . BL = 85H * 35H = 1B89H →AX = 1B89H
2. IMUL BL =AL . BL= 2'SAL * BL= 2'S(85H) * 35H

   =7BH * 35H = 1977H→2's comp→E689H →AX.
3. DIV BL $= \frac{AX}{BL} = \frac{0085H}{35H} =$

AH (remainder)	AL (quotient)
1B	02

4. IDIV BL $= \frac{AX}{BL} = \frac{0085H}{35H} =$

AH (remainder)	AL (quotient)
1B	02

Example: Assume that each instruction starts from these values:

AL = F3H , BL = 91H , AH = 00H

1. MUL BL = AL * BL = F3H * 91H = 89A3H →AX = 89A3H
2. IMUL BL =AL * BL =2'SAL *2'SBL= 2'S(F3H) *2'S(91H)
   = 0DH * 6FH = 05A3H →AX.
3. IDIV BL $= \frac{AX}{BL} = \frac{00F3H}{2'(91H)} = \frac{00F3H}{6FH}$ = 2 quotient and 15H remainder:

AH (remainder)	AL (quotient)
1B	02

→ , but $\frac{\text{Positive}}{\text{negative}}$ = negative , so

AH (remainder)	AL (quotient)
1B	2'comp(02)

→

AH (remainder)	AL (quotient)
1B	FE

م.م اثير هادي عيسى الرماحي

4. DIV BL $\frac{AX}{BL} = \frac{00F3H}{91H} = 01$

AH (remainder)	AL (quotient)
62	01

Example: Assume that each instruction starts from these values:
AX= F000H, BX= 9015H, DX= 0000H

1. MUL BX = F000H * 9015H =

DX	AX
8713	**B000**

2. IMUL BX =2'S(F000H) *2'S(9015H) = 1000 * 6FEB =

DX	AX
06FE	**B000**

3. DIV BL $\frac{AX}{BL} = \frac{F000H}{15H}$ = 0B6DH → more than FFH → Divide Error
4. IDIV BL $\frac{AX}{BL} = \frac{2'(F000H)}{15H} = \frac{1000H}{15H}$ =C3H → more than 7FH → Divide Error

Example : Assume that each instruction starts from these values:
AX= 1250H, BL= 90H

1. IDIV BL $\frac{AX}{BL} = \frac{1250H}{90H} = \frac{positive}{negative} = \frac{positive}{2'negative} = \frac{1250}{2'(90H)} = \frac{1250H}{70H}$
= 29H quotient and 60H remainder

But 29H(positive) →2'S(29H)= D7H →

AH (Remainder)	AL (quotient)
60H	**D7H**

2. DIV $\frac{AX}{BL} = \frac{1250H}{90H}$ =20H →

AH (Remainder)	AL (quotient)
50H	**20H**

To divide an 8-bit dividend by and 8-bit divisor by extending the sign bit of Al to fill all bits of AH. This can be done automatically by executing the Instruction (CBW). In a similar way 16-bit dividend in AX can be divided by 16-bit divisor. In this case the sign bit in AX is extended to fill all bits of DX. The instruction CWD perform this operation automatically.
Note that CBW extend 8-bit in AL to 16-bit in AX while the value in AX will Be equivalent to the value in AL. Similarly, CWD convert the value in AX to 32-bit In (DX,AX) without changing the original value.

3. Logical & Shift Instructions:

- Logical instructions: The 8086 processor has instructions to perform bit by bit logic operation on the specified source and destination operands.
- Uses any addressing mode except **memory-to-memory** and **segment registers**

م.م اثير هادي عيسى الرماحي

Mnemonic	Meaning	Format	Operation	Flags affected
AND	Logical AND	AND D, S	(S) . (D) → (D)	OF, SF, ZF, PF, CF, AF undefined
OR	Logical Inclusive —OR	OR D, S	(S) + (D) → (D)	OF, SF, ZF, PF, CF, AF undefined
XOR	Logical Exclusive —OR	XOR D, S	(S) ⊕ (D) → (D)	OF, SF, ZF, PF, CF, AF undefined
NOT	Logical NOT	NOT D	($\overline{D}$) → (D)	None

(a)

Destination	Source
Register	Register
Register	Memory
Memory	Register
Register	Immediate
Memory	Immediate
Accumulator	Immediate

(b)

Destination
Register
Memory

(c)

(a) Logic instructions
(b) Allowed operands for the AND, OR and XOR instructions
(c) Allowed operands for NOT instruction.

AND

- used to clear certain bits in the operand(masking)

Example Clear the high nibble of BL register
AND BL, 0FH (xxxxxxxx AND 0000 1111 = 0000 xxxx)
Example Clear bit 5 of DH register
AND DH, DFH (xxxxxxxx AND 1101 1111 = xx0x xxxx)

OR

- Used to set certain bits

Example Set the lower three bits of BL register
OR BL, 07H (xxxxxxxx **OR** 0000 0111 = xxxx x111)
Example Set bit 7 of AX register
OR AH, 80H (xxxxxxxx **AND** 1000 0000 = 1xxx xxxx)

XOR

- Used to invert certain bits (toggling bits)
- Used to clear a register by XORed it with itself

Example Invert bit 2 of DL register
XOR BL, 04H (xxxxxxxx OR 0000 0100 = xxxx x xx)
Example Clear DX register
XOR DX, DX (DX will be 0000H)
Example

XOR AX , DL	not valid	size don't match
OR AX,DX	valid	
NOT CX , DX	not valid	Not instruction has one operand
AND WORD PTR [BX + DI + 5H]	valid	
AND WORD PTR [BX + DI] , DS	not valid	source must not be segment register

م.م اثير هادي عيسى الرماحي

4. Shift instruction

- The four shift instructions of the 8086 can perform two basic types of shift operations: the logical shift, the arithmetic shift
- Shift instructions are used to
  o Align data
  o Isolate bit of a byte of word so that it can be tested
  o Perform simple multiply and divide computations
- The source can specified in two ways
  Value of 1 : Shift by One bit
  Value of CL register : Shift by the value of CL register

Note that the amount of shift specified in the source operand can be defined explicitly if it is **one bit** or should be stored in CL if **more than 1.**

Mnemonic	Meaning	Format	Operation	Flags affected
SAL/SHL	Shift arithmetic left/shift logic left	SAL/SHL D, Count	Shift the (D) left by the number of bit positions equal to Count and fill the vacated bit positions on the right with zeros.	CF, PF, SF, ZF, OF AF undefined OF undefined if count ≠ 1
SHR	Shift logical right	SHR D, Count	Shift the (D) right by the number of bit positions equal to Count and fill the vacated bits positions on the left with zeros.	CF, PF, SF, ZF, OF AF undefined OF undefined if count ≠ 1
SAR	Shift arithmetic right	SAR D, Count	Shift the (D) right by the number of bit positions equal to Count and fill the vacated bits positions on the left with original most significant bit.	OF, SF, ZF, CF, PF AF, undefined

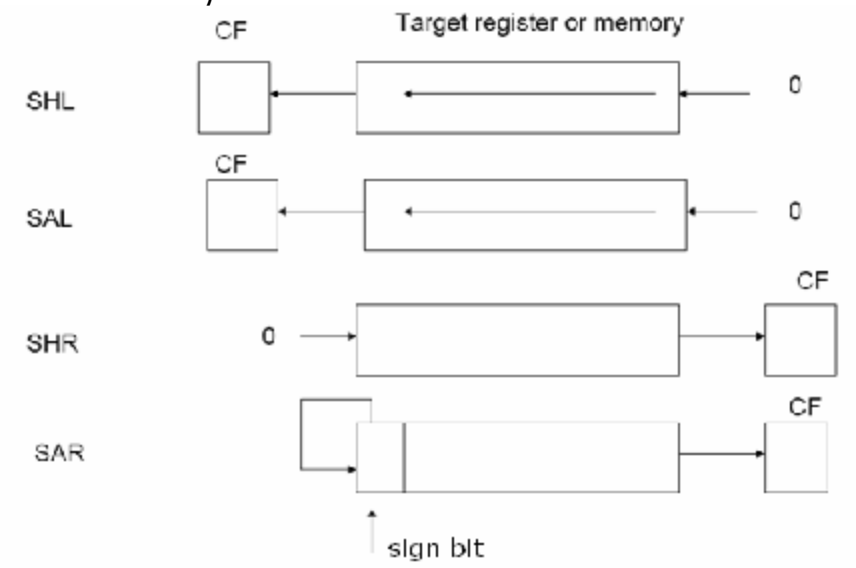
(a)

Destination	Count
Register	1
Register	CL
Memory	1
Memory	CL

(b)

a) Shift instructions, (b) Allowed operands

- The SHL and SAL are identical: they shift the operand to left and fill the vacated bits to the right with **zeros**.
- The SHR instruction shifts the operand to right and fill the vacated bits to the left with **zeros**.
- The SAR instruction shifts the operand to right and fill the vacated bits to the left with the value of MSB (this operation used to shift the signed numbers)

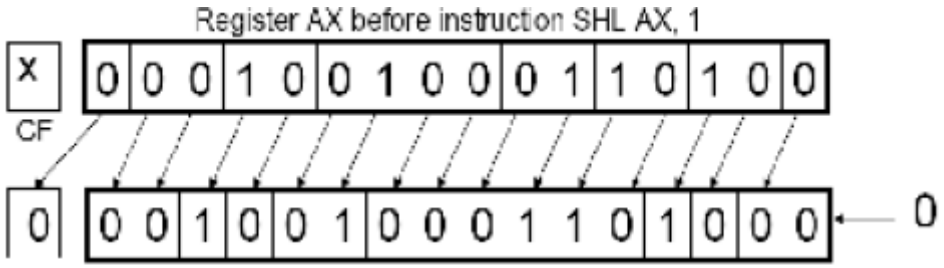


م.م اثير هادي عيسى الرماحي

Example let AX=1234H what is the value of AX after execution of next instruction

SHL AX,1

Solution: causes the 16-bit register to be shifted 1-bit position to the left where the
vacated LSB is filled with zero and the bit shifted out of the MSB is saved in CF



AX Before

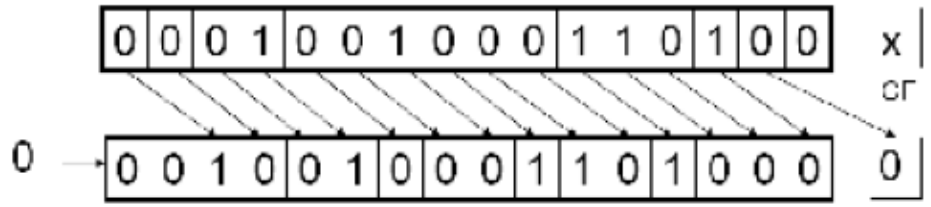
AX After

Example:

MOV CL, 2H

SHR DX, CL

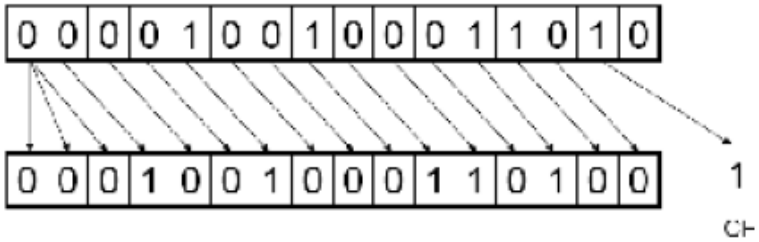
The two MSBs are filled with zeros and the LSB is thrown away while the second LSB is saved in CF.



DX Before

DX After

Example: Assume CL= 2 and AX= 091AH. Determine the new contents of AX And CF after the instruction SAR AX, CL is executed.



AX Before

AX After

- This operation is equivalent to division by powers of 2 as long as the bits shifted out of the LSB are zeros.

Example: Multiply AX by 10 using shift instructions:

Solution:

```
SHL AX, 1
MOV BX, AX
MOV CL,2
SHL AX,CL
ADD AX, BX
```


م.م اثير هادي عيسى الرماحي

Example: What is the result of SAR CL, 1 , if CL initially contains B6H?
Solution: DBH
Example: What is the result of SHL AL, CL , if AL contains 75H and CL contains 3?
Solution: A8H
Example: Assume DL contains signed number; divide it by 4 using shift instruction?
Solution: MOV CL , 2
SAR DL , CL

م.م اثير هادي عيسى الرماحي