Chapter 7, 9

CSE-214

Presented By:-

MD. IFTEKHARUL ISLAM SAKIB

Lecturer, CSE, BUET

AND, OR AND XOR

- Syntax

```
AND     destination, source
OR      destination, source
XOR     destination, source
```

- The restrictions of destination and source are the same as ADD or SUB instructions

AND, OR AND XOR

- Effect on flags
  - ✓ SF, ZF, PF reflect the result
  - ✓ AF is undefined
  - ✓ CF, OF = 0

AND, OR AND XOR

- Example
  - ✓ XOR AX, AX  Clearing a Register
  - ✓ OR CX, CX
CMP CX,0  Testing a Register for 0
  - ✓ AND AL, 0Fh  Converting ASCII Digit to a number

NOT Instruction

- Works on a single operand
- Performs one's complement operation on the destination
- Syntax

  NOT destination

TEST Instruction

- Performs similarly to the AND instruction
- Except doesn't write the output on the destination.
- Only sets or resets the flags
- Syntax

  TEST destination, source
- Usually used for flow control

TEST Instruction

- Example

  - ✓ TEST AL,1
    CMP CX,0

Testing a number is even or not

Shift / Rotate Instructions

- Has two possible formats:

  Opcode destination, 1

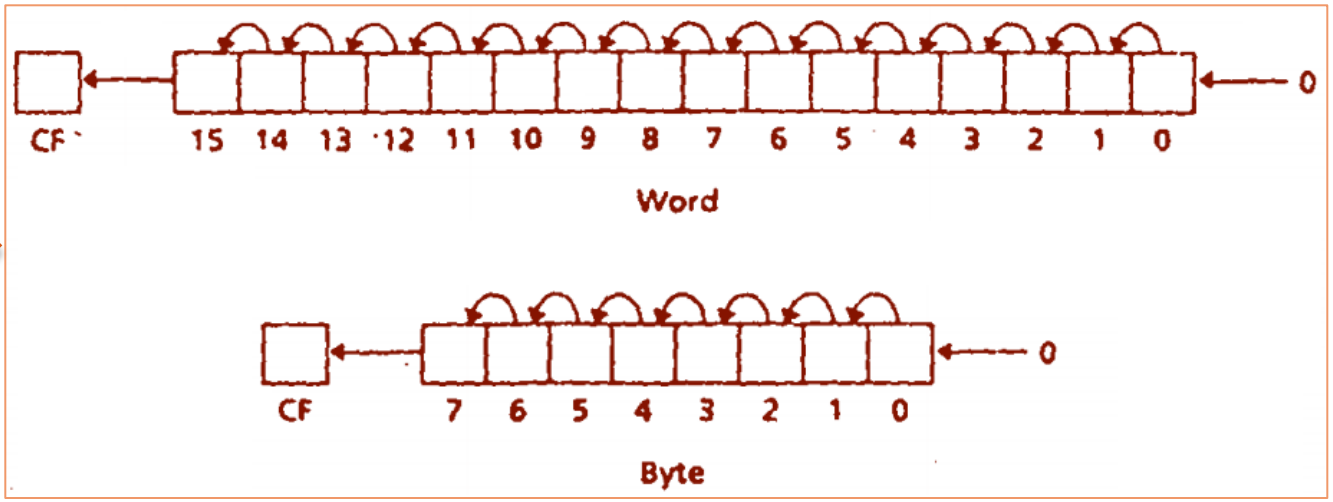
  Opcode destination, CL

- Effect on flags
  - ✓ SF, ZF, PF reflect the result
  - ✓ AF is undefined
  - ✓ CF value changes according to Shift / Rotate Type
  - ✓ OF =1 if result changes sign on last Shift / Rotation

SHL and SAL instructions

- Shifts the bits in the destination to the left
- The MSB is shifted into CF
- A 0 is shifted to LSB
- SAL and SHL generate the same machine code

SHL and SAL



SHL and SAL instructions

- Example
  - ✓ If DH=8AH, CL=3
    What is the result of
    SHL DH, CL

    DH=50H
    CF=0
    OF=0

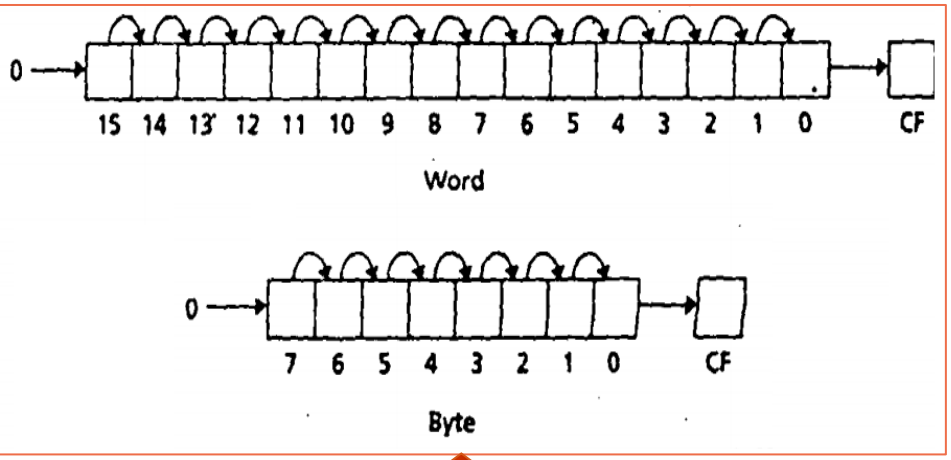
  - ✓ If DX=8AH, CX=3
    What is the result of
    SHL DX, CX

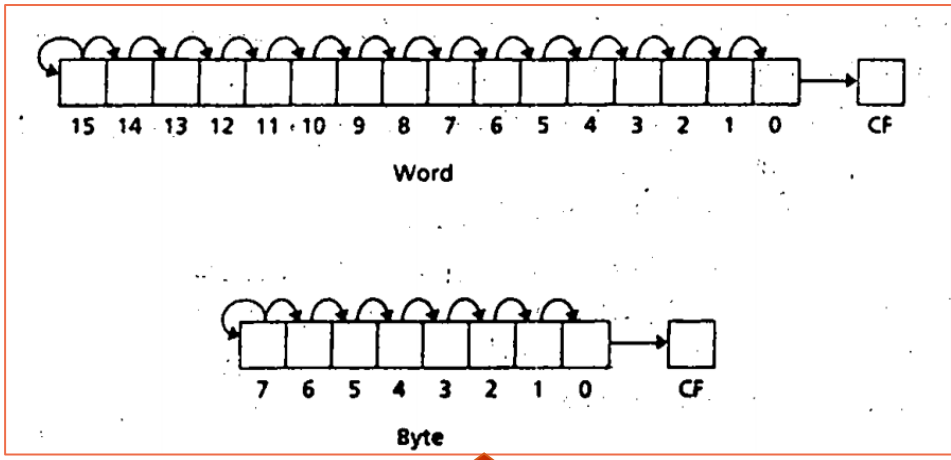
    DX=450H
    CF=0
    OF=0

SHR and SAR instructions

- Shifts the bits in the destination to the right
- The LSB is shifted into CF
- In case of SAR MSB retains its original value



SHR



SAR

SHR and SAR instructions

- SHR should be used for unsigned interpretation as it does not preserve sign
- SAR should be used for signed interpretation as it preserves sign

SHR and SAR instructions

- Example
  - ✓ If AL=-15, CL=1
    What is the result of
    SHR AL, CL

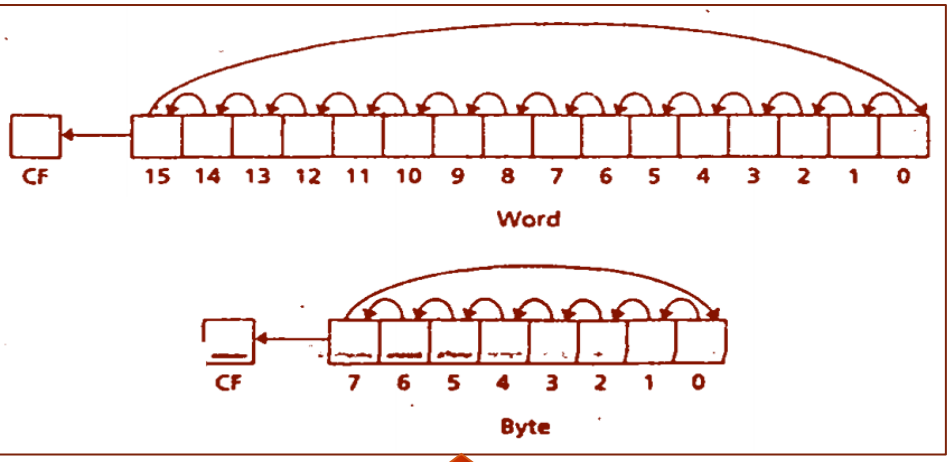
AL=120H
CF=1
OF=1

  - ✓ If AL=-15, CL=1
    What is the result of
    SAR AL, CL

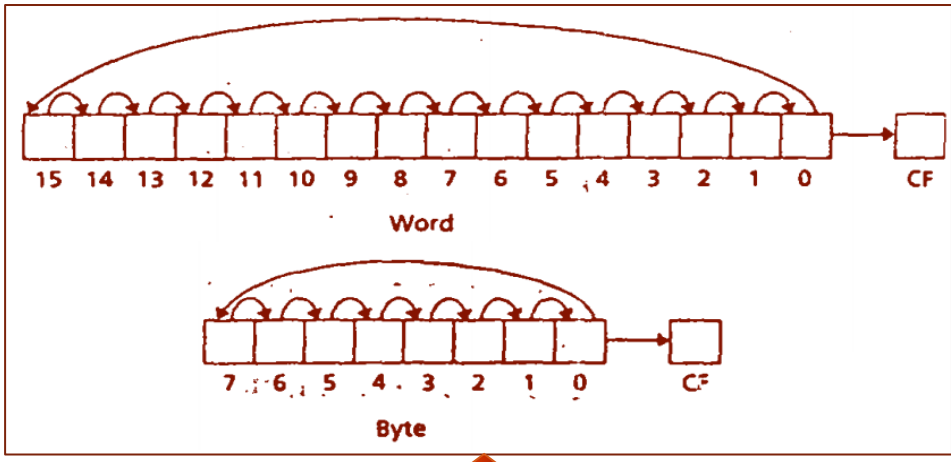
AL=-8
CF=1
OF=1

ROL and ROR

- ROL and ROR shifts bits in destination to the left and right respectively
- For ROL MSB is shifted into the rightmost bit and CF
- FOR ROR rightmost bit is shifted into MSB and CF



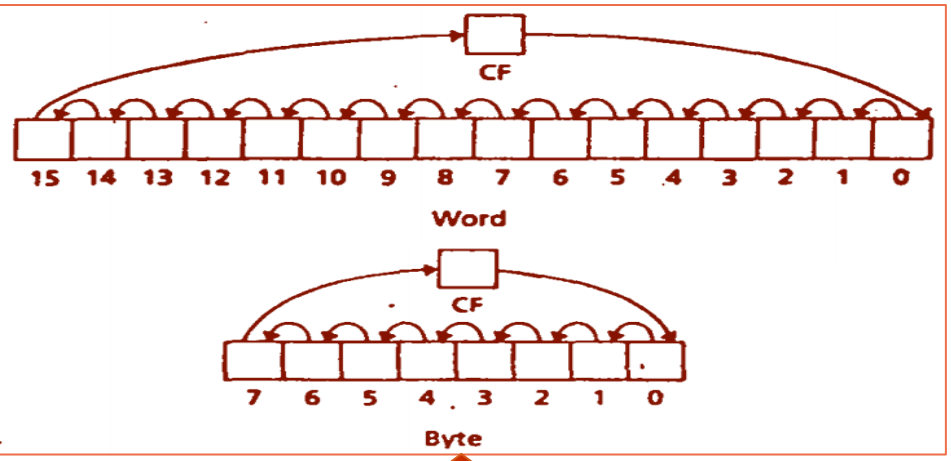
ROL



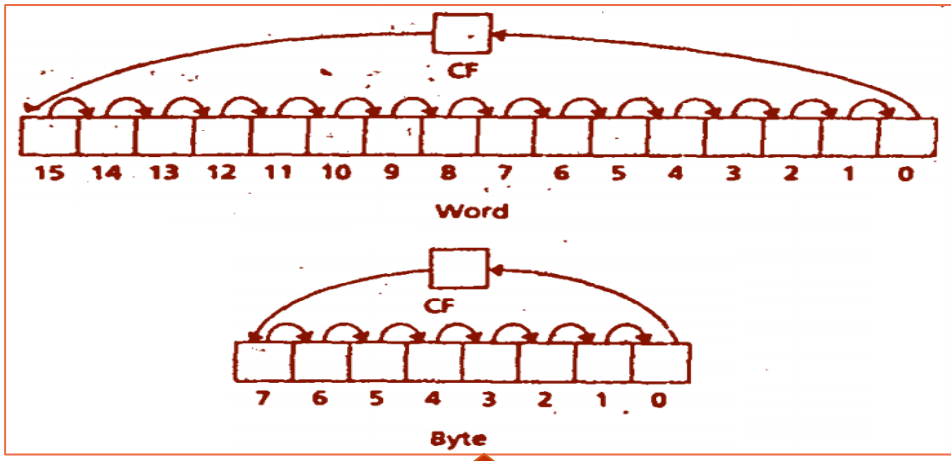
ROR

RCL and RCR instructions

- Works similarly to ROL and ROR respectively
- For RCL, CF is shifted to LSB and MSB is shifted to CF
- FOR RCR, CF is shifted to MSB and LSB shifted to CF



RCL



RCR

Multiplication instructions

- imul *source*
  - Signed multiply
- mul *source*
  - Unsigned multiply
- **Byte and Word Multiplication** (A X B)
  - If two **bytes** are multiplied, the result is a 16-bit **word**
    - A: **source**
    - B: **al**
    - product: **ax**
  - If two **words** are multiplied, the result is a 32-bit ***doubleword***
    - A: **source**
    - B: **ax**
    - Product (ms 16 bits): **dx**
    - Product (ls 16 bits): **ax**

Multiplication instructions

- ***source*** can be a register or memory location (not a constant)
- Byte form
  - AX=AL**source*
- Word form
  - DX:AX=AX**source*

•If **ax** contains **0001h** and **bx** contains **FFFFh**

- **•mul bx; dx = 0000h ax = FFFFh**
- **•imul bx ; dx = FFFFh ax = FFFFh (-1)**

Multiplication instructions

- Effect on flags
  - ✓ SF, ZF, AF, and PF Undefined
  - CF/OF
    - MUL
      - 0: if upper half result 0
      - 1: Otherwise
    - IMUL
      - 0: if upper half is sign extension of lower half.
      - 1: Otherwise

More Examples

•AX=FFFFh,BX=FF
FFh

Instruction	Hex Product	DX	AX	CF/OF
MUL BX	FFFE0001 (4294836225)	FFFE (!zero)	0001	1
IMUL BX	1	0000	0001	0

•AX=80h,BX=FFh

Instruction	Hex Product	AH	AL	CF/OF
MUL BX	7F80 (128)	7F(!zero)	80	1
IMUL BX	0080	00 (no sign extension)	80	1

Division instructions

- **cbw**
  - convert byte to word
- **cwd**
  - convert word to doubleword
- **div *source***
  - unsigned divide
- **idiv *source***
  - signed divide

Byte and Word Division (A/B)

- When division is performed
  - two results: the quotient and the remainder
  - Quotient and remainder are same **size** as the divisor
- For the byte form,
  - Divisor, B: **source** ;Dividend , A: **ax**
  - Quotient :**al** ;Remainder: **ah**
- For the word form,
  - Divisor, B: **source** ;Dividend , A: **dx:ax**
  - Quotient :**ax** ; Remainder: **dx**

An Example

```
;If dx = 0000h, ax = 0005h, and bx = FFFEh (-
  2)
div bx;   ax = 0000h    dx = 0005h
idiv bx;  ax = FFFEh    dx = 0001h
```

Divide Overflow

- It is possible that the **quotient** will be too big to fit in the specified destination (**al** or **ax**)
- if the **divisor** is much smaller than the **dividend**
- the program terminates and the system displays the message "**Divide Overflow**"

Sign Extension of the Dividend

- Word division
  - The dividend is in **dx:ax** even if the actual dividend will fit in **ax**
  - For **div**, **dx** should be cleared
  - For **idiv**, **dx** should be made the sign extension of **ax** using **cwd**

e.g. -1250/7

```
MOV AX,-1250
CWD ; sign extend
MOV BX,7
IDIV BX
```

Sign Extension of the Dividend

- Byte division
  - The dividend is in **ax** even if the actual dividend will fit in **al**
  - For **div**, **ah** should be cleared
  - For **idiv**, **ah** should be made the sign extension of **al** using **cbw**

THANK YOU