6.1 Bit Manipulation Instructions:

These instructions are used to perform Bit wise operations.

LOGICAL INSTRUCTIONS	SHIFT INSTRUCTIONS	ROTATE INSTRUCTIONS
NOT AND OR XOR TEST	SHL / SAL SHR SAR	ROL ROR RCL RCR

When binary data are manipulated in a register or a memory location, the rightmost bit position is always numbered bit 0. Bit position numbers increase from bit 0 toward the left, to bit 7 for a byte, and to bit 15 for a word.

6.2 Logical Instructions:

Logic operations provide binary bit control in low-level software. The logic instructions allow bits to be set, cleared, or complemented. Low-level software appears in machine language or assembly language form and often controls the I/O devices in a system. All logic instructions affect the flag bits. Logic operations always clear the carry and overflow flags, while the other flags change to reflect the condition of the result.

1. The 8086 processor has instructions to perform bit by bit logic operation on the specified source and destination operands.
2. Uses any addressing mode except memory-to-memory and segment registers
3. These instructions perform their respective logic operations.
4. Below the format and the operand for these instructions.

Mnemonic	Meaning	Format	Operation	Flags Effected
NOT	Logical NOT	NOT D	$(\overline{D}) \rightarrow (D)$	None
AND	Logical AND	AND D,S	$(S) .D) \rightarrow (D)$	O, S, Z, P, C
OR	Logical Inclusive OR	OR D,S	$(S) + D) \rightarrow (D)$	O, S, Z, P, C
XOR	Logical Exclusive OR	XOR D,S	$(S) \oplus D) \rightarrow (D)$	O, S, Z, P, C

Logical AND: often executes in about a microsecond used to clear certain bits in the operand (masking), also used to perform logical multiplication. The AND instruction can replace discrete AND gates if the speed required is not too great, although this is normally reserved for embedded control applications.

```
  xxxx xxxx   Unknown number
• 0000 1111   Mask
-------------
  0000 xxxx   Result
```

The AND instruction uses any addressing mode except memory-to-memory and segment register addressing.

Example: Clear the high nibble of BL register

```
MOV BL,55H
AND BL, 0FH          ; (xxxxxxxxAND 0000 1111 = 0000 xxxx)
```

Example: Clear bit 5 of DH register

```
MOV DH,75H
AND DH, DFH          ; (xxxxxxxxAND1101 1111 = xx0xxxxx)
```

HW: Check the condition of the flag register in the above two examples.

Logical OR: Used to set certain bits, also used to performs logical addition. The OR instruction uses any of the addressing modes allowed to any other instruction except segment register addressing. In embedded controller applications, the OR instruction can also replace discrete OR gates.

```
x x x x  x x x x   Unknown number
+ 0 0 0 0  1 1 1 1   Mask
x x x x  1 1 1 1   Result
```

Example: Set the lower three bits of BL register

```
MOV BL,55H
OR BL, 07H           ; (xxxxxxxxOR 0000 0111 = xxxx x111)
```

Example: Set bit 7 of AX register

```
MOV AH,05H
OR AH, 80H           ; (xxxxxxxxOR1000 0000 = 1xxxxxxx)
```

HW: Check the condition of the flag register in the above two examples.

Logical Exclusive-OR (XOR): The Exclusive-OR instruction (XOR) sometimes called a comparator is differs from (OR) instruction. The XOR instruction uses any addressing mode except segment register addressing. Exclusive-OR can replace discrete logic circuitry in embedded applications.

- Used to invert certain bits (toggling bits) in register or memory. This instruction allows part of a number to be inverted or complemented. This is ideal for control system applications in which equipment must be turned on (1), turned off (0), and toggled from on to off or off to on.
- Used to clear a register to zero by XORed it with itself. The XOR instruction is often used to clear a register in place of a move immediate.

```
x x x x  x x x x   Unknown number
⊕0 0 0 0  1 1 1 1   Mask
x x x x  x̄ x̄ x̄ x̄   Result
```

Example: Invert bit 2 of DL register

```
MOV BL,55H
XOR BL, 04H          ; (xxxxxxxxOR0000 0100 = xxxxx x xx)
```

Example: Clear DX register

```
MOV DX,1A1AH
XOR DX, DX           ; (DX will be 0000H)
```

Example: Write an 8086-microprocessor program to clear bits 0 to 5, set bits 6 to 10 and complement bits 11 to 15 of word stored in a memory location with offset 4000H.

```
MOV DI, 4000H
AND [DI], FFC0H
OR [DI], 07C0H
XOR [DI], F800H
```

HW: Check the condition of the flag register in the above three examples.

Test and Bit Test Instructions: The TEST instruction performs the AND operation. The difference is that the AND instruction changes the destination operand, whereas the TEST instruction does not. A TEST only affects the condition of the flag register, which indicates the result of the test. The TEST instruction uses the same addressing modes as the AND instruction. The TEST instruction functions in the same manner as a CMP instruction. The difference is that the TEST instruction normally tests a single bit (or occasionally multiple bits), whereas the CMP instruction tests the entire byte, or word. The zero flag (Z) is a logic 1 (indicating a zero result) if the bit under test is a zero, and (indicating a nonzero result) if the bit under test is not zero. Usually the TEST instruction is followed by either the JZ (jump if zero) or JNZ (jump if not zero) instruction. The destination operand is normally tested against immediate data. The value of immediate data is 1 to test the rightmost bit position, 2 to test the next bit, 4 for the next, and so on.

Example:

```
MOV AL,1FH
TEST AL,01H          ;test right bit
JNZ RIGHT            ;if set
HLT
RIGHT: MOV AL,22H
```

Example:

```
MOV AX,01FFH
TEST AL,128          ;test left bit
JNZ LEFT             ;if set
HLT
LEFT: MOV AL,11H
```

NOT: Logical inversion, or the one's complement (NOT), can use any addressing mode except segment register addressing. The NOT instruction inverts all bits of a byte, or word.

Example:

```
MOV AL,55H
NOT AL
```

6.3 Shift instructions:

Shift instructions position or move numbers to the left or right within a register or memory location. They also perform simple arithmetic such as multiplication by powers of 2^{+n} (left shift) and division by powers of 2^{-n} (right shift). The microprocessor's instruction set contains four different shift instructions: Two are logical shifts and two are arithmetic shifts.

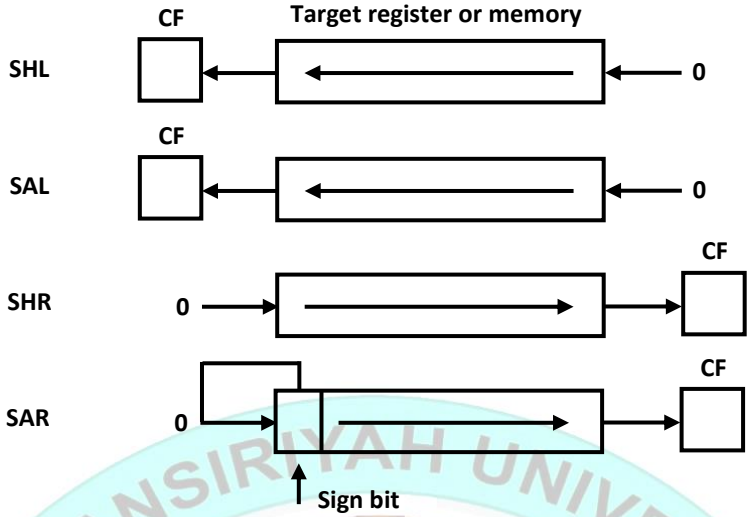
1. Shift instructions can perform two basic types of shift operations; the logical shift and the arithmetic shift. Also, each of these operations can be performed to the right or to the left.
2. Shift instructions are used to
   a. Align data
   b. Isolate bit of a byte of word so that it can be tested
   c. Perform simple multiply and divide computations
3. The source can specified in two ways
   a. Value of 1 : Shift by One bit
   b. 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***.

SHL, SHR, SAL, and, SAR instructions:

The operation of these instructions is described below:

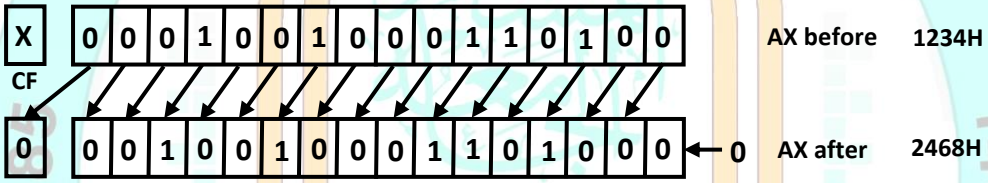
Mnem.	Meaning	Format	Operation	Flags Effected
SAL/SHL	Shift arithmetic left /shift logical left	SAL D, Count SHL D,Count	Shift the (D) left by the number of bit positions equal to Count and fill the vacated bits positions on the right with zeros	C, P, S, Z A undefined O undefined if count ≠1
SHR	shift logical right	SHR D,Count	Shift the (D) right by the number of bit positions equal to Count and fill the vacated bit positions on the left with zeros	C, P, S, Z A undefined O undefined if count ≠1
SAR	Shift arithmetic right	OR D,S	Shift the (D) right by the number of bit positions equal to Count and fill the vacated bit positions on the left with the original most significant bit	C, P, S, Z A undefined O undefined if count ≠1



Example: Let AX=1234H what is the value of AX after execution of next instruction SHL AX,1

Solution:

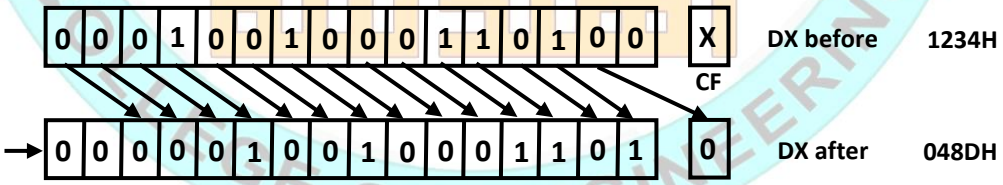
```
MOV AX, 1234H
SHL AX, 1
```



Example:

```
MOV DX, 1234H
MOV CL, 2H
SHR DX, CL
```

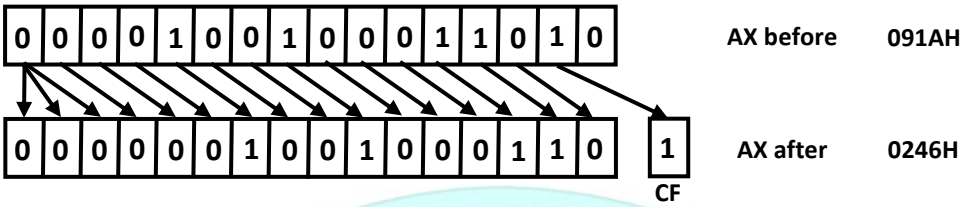
Solution:



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

Solution:

```
MOV AX, 091AH
MOV CL, 2H
SAR AX, CL
```



This operation is equivalent to division by powers of 2 ($091AH \div 2^2 = 0246H$) as long as the bits shifted out of the LSB are zeros.

Example: Multiply AX by 10 using shift instructions.

Solution:

```
MOV AX, 10H          Result: AX=A0H
SHL AX, 1                    BX=20H
MOV BX, AX                   CX=02H
MOV CL, 2
SHL AX, CL
ADD AX, BX
```

Example: Assume DL contains signed number; divide it by 4 using shift instruction?

Solution:

```
MOV DL, 80H          Result: 80H÷4=20H
MOV CL , 2                   DL=E0H
SAR DL , CL
```

Example: Write A Program to multiply the contents of AX by (10.5) using shift instructions.

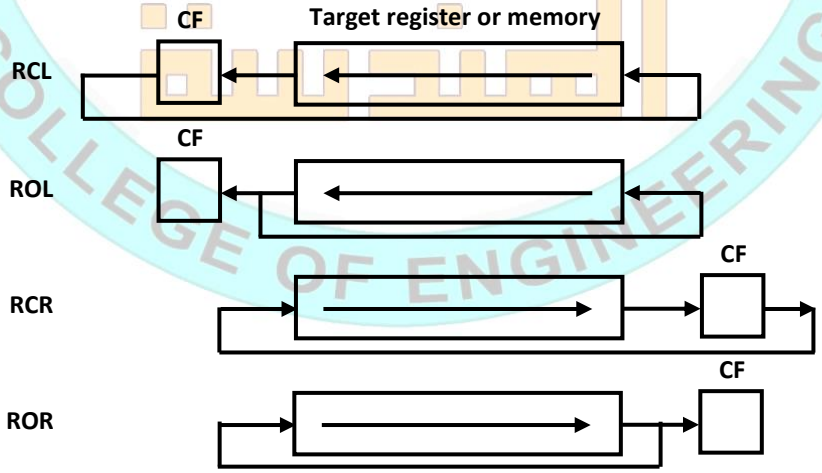
```
MOV AX, 04H
SHR AX,1        ;AX/2=2
MOV DX, AX      ;DX=2
SHL AX,1        ;AX*2=4
SHL AX,1        ;AX*2=8
MOV BX, AX      ;BX=8
MOV CL,2H       ;CX=2
SHL AX, CL      ;AX*4=32=20H
ADD AX, BX      ;AX=40=28H
ADD AX,DX       ;AX=42=2AH
HLT
```

6.4 Rotate Instructions:

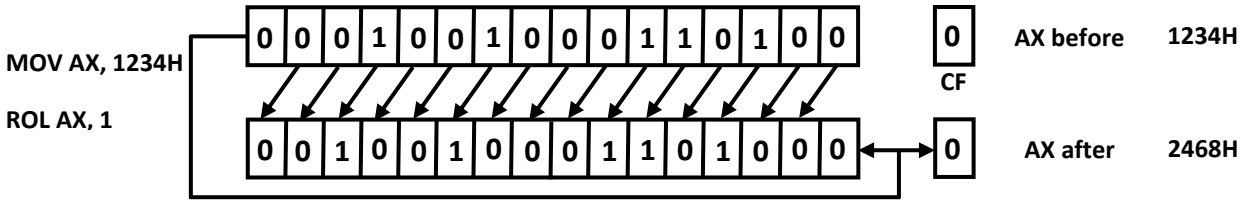
Numbers rotate through a register or memory location, through the C flag (carry), or through a register or memory location only. With either type of rotate instruction, the programmer can select either a left or a right rotate. Addressing modes used with rotate are the same as those used with shifts. A rotate count can be immediate or located in register CL.

1. They have the ability to rotate the contents of either an internal register or a storage location in memory.
2. Also, the rotation that takes place can be from 1 to 255 bit positions to the left or to the right.
3. Moreover, in the case of a multi-bit rotate, the number of bit positions to be rotated is specified by the value in CL.
4. The operation of these instructions is described in figure below.

Mnem.	Meaning	Format	Operation	Flags Effected
ROL	Rotate left	ROL D, Count	Rotate the (D) left by the number of bit positions equal to Count.Each bit shifted out from the leftmost bit goes back into the rightmost bit position.	C O undefined if count ≠1
ROR	Rotate right	RORD,Count	Rotate the (D) right by the number of bit positions equal to Count. Each bit shifted out from the rightmost bit goes back into the leftmost bit position.	C, O undefined if count ≠1
RCL	Rotate left through carry	RCLD,Count	Same as ROL except carry is attached to (D) for rotation.	C , O undefined if count ≠1
RCR	Rotate right through carry	RCRD,Count	Same as ROR except carry is attached to (D) for rotation.	C , O undefined if count ≠1



Example: Assume AX = 1234H, what is the result of executing the instruction
ROL AX, 1
Solution:

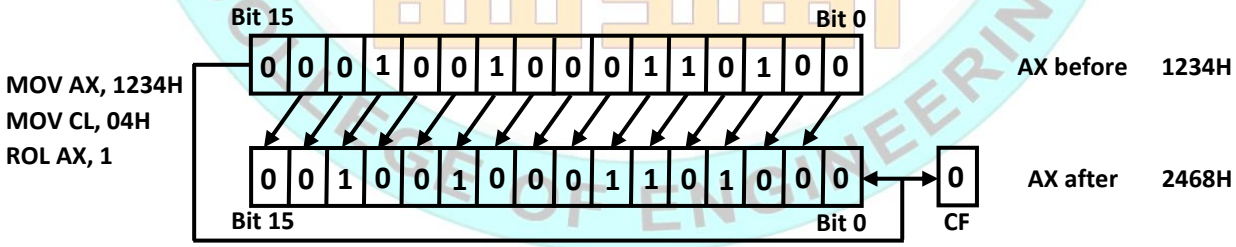


Example: Find the addition result of the two hexadecimal digits packed in DL.
Solution:

```
MOV DL, 12H          ; DL=12H
MOV CL, 04H          ; CL=04H
MOV BL, DL           ; BL=12H
ROR DL, CL           ; DL=21H
AND BL, 0FH          ; BL=02H
AND DL, 0FH          ; DL=01H
ADD DL, BL           ; DL=03H
```

Test instruction: is similar to the AND instruction. The difference is that the AND instruction change the destination operand, while the TEST instruction does not. A TEST only affects the condition of the flag register, which indicates the result of the test. The TEST instruction uses the same addressing modes as AND instruction. (TEST DL,0FH)

Example: If (CL) =04H and AX=1234A. Determine the new contents of AX and the carry flag after executing the instructions:
a) ROL AX, 1
b) ROR AX, CL
Solution:
a)



b)