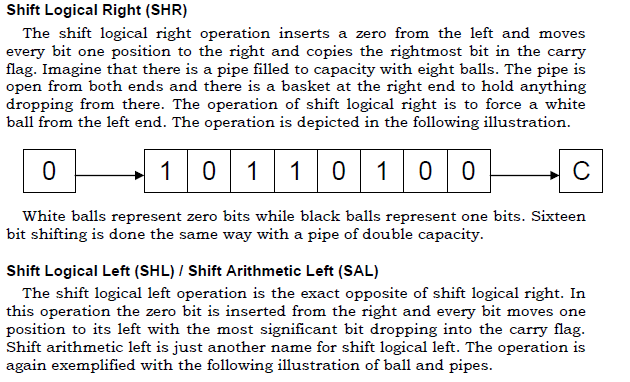
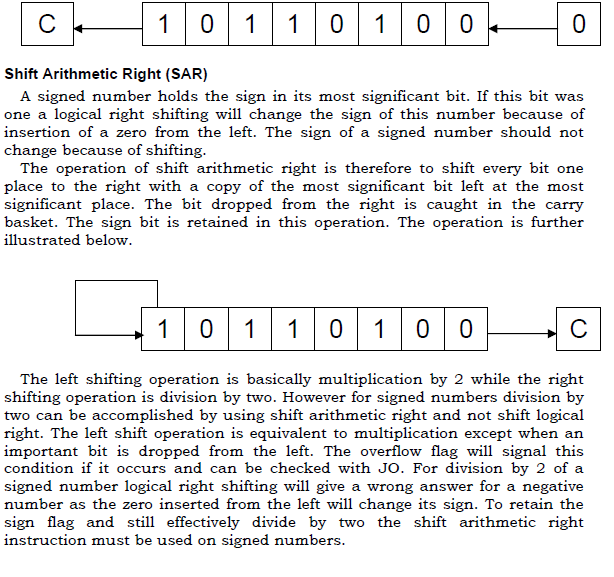
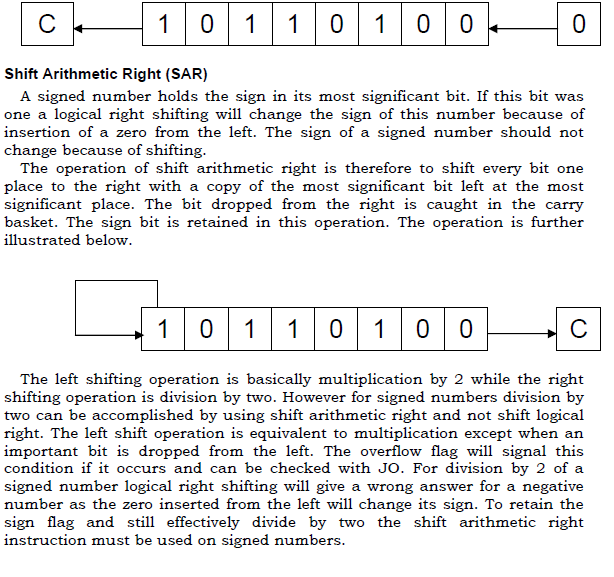
**Computer Organization and Assembly Language**

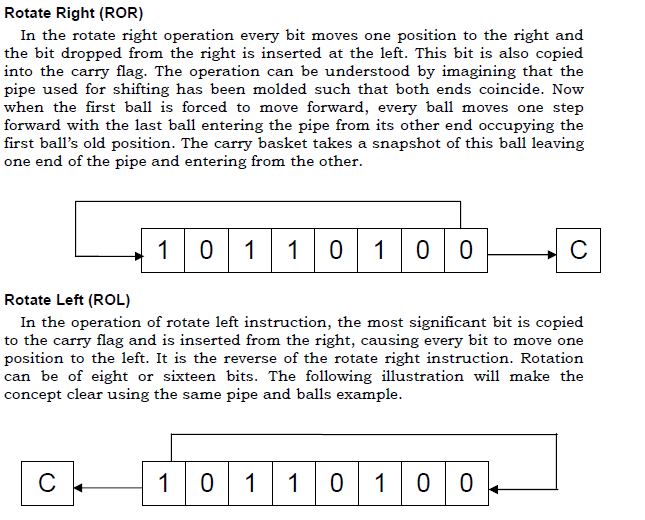
|  |  |
| --- | --- |
| **Lab 05** | |
| **Topic** | * Arithmetic & Logical instructions * Selective bit setting/clearing/complimenting * Shifting and Rotations variations * Extended addition and subtraction |

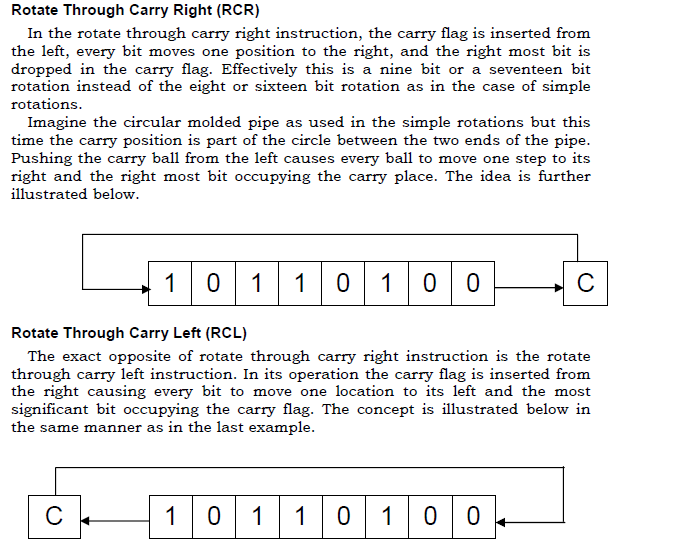
***Part 1***











**Example 1:**

**Let the binary of a number (0XABCD) is 10101011** **11001101.**

1. **Set the fourth bit.**

mov ax,0xABCD

or ax,0000000000010000b

mov ax,0x4c00

int 21h

1. **Clear the L.S.B.**

mov ax,0xABCD

and ax,1111111111111110b

mov ax,0x4c00

int 21h

1. **Invert the M.S.B.**

mov ax,0xABCD

xor ax,1000000000000000b

mov ax,0x4c00

int 21h

***Note: logical operations are bitwise operations.***

**Example 2:**

Multiply the number by 8 using shift operator.

Let the number is 7.

**mov al,7**

**shl al,3**

**Example 3:**

Rotate right 4 times the value in register bx.

Let BX=0xEFCD

**mov bx,0xEFCD**

**ror bx,4**

**Example 4(Extended addition)**

MOV AX, Num1 ;loads two bytes into AX register, AX=FFFF

MOV BX, [Num1+2] ;loads Next two bytes into AX register, AX=0001

ADD AX, Num2 ; adds into AX; AX=AX+0002;

ADC BX, [Num2+2]; Add with carry instruction

MOV SUM,AX ; Move the lower bits into Sum variable

MOV [SUM+2],BX ; Move the higher bits into Sum variable higher bits

mov ax,0x4c00

int 21h

Num1: dd 0x0001FFFF

Num2: dd 0x00010002

SUM: dd 0

|  |  |
| --- | --- |
| a = 10, b = 20, c = 5 , sum=0;    if (a < =b)  {  // L1  if (a < =c)  {  // L2  if (b>c)  {  // L3  sum = a + b + c;  }  else  {  // L4  sum = a - b - c;  }  }  else  {  // L5  sum = a + b - c;  }  }  else  {  // L6  sum = a - b + c;  } | [org 0x100]  mov al,[a]  mov bl,[b]  mov cl,[c]  cmp al,bl  jng l1  jg l6  l1:  cmp al,cl  jng l2  jg l5  l2:  cmp bl,cl  jg l3  jng l4  l3:  mov [sum],al  add [sum],bl  add [sum],cl  jmp exit  l4:  mov [sum],al  sub [sum],bl  sub [sum],cl  jmp exit  l5:  mov [sum],al  add [sum],bl  sub [sum],cl  jmp exit  l6:  mov [sum],al  sub [sum],bl  add [sum],cl  exit:  mov ax,0x4c00  int 21h  a: db 10  b: db 20  c: db 5  sum: db 0 |