BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

Lab Sheet

Introduction to Shift and Rotate Instructions

CSE 312

MICROPROCESSOR AND ASSEMBLY LANGUAGE LAB

Objective:

The objective of this lab is to

1) Learn different Shift and rotate instructions such as SHL, SHR, SAL, SAR, ROL, ROR, RCL and RCR

Introduction:

SHL Instruction

The SHL (shift left) instruction performs a logical left shift on the destination operand, filling the lowest bit with 0. The highest bit is moved to the Carry flag, and the bit that was in the Carry flag is lost:

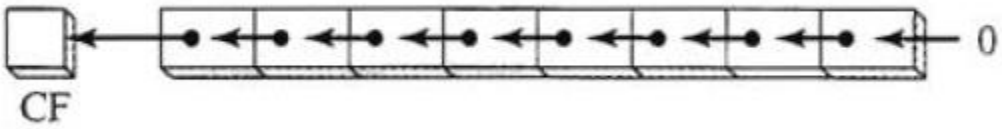


Figure: SHL instruction

Syntax:

The first operand is the destination , and the second is the shift count:

```
SHL destination, count
```

The following lists the types of operands permitted by this instruction:

```
SHL reg,imm8
SHL mem,imm8
SHL reg, CL
SHL mem,CL
```

Example:

In the following instructions, BL is shifted once to the left. The highest bit is copied into the Carry flag and the lowest bit position is cleared :

```
mov bl,8Fh     ; BL=10001111b
shl bl,l       ; BL=00011110b CF=1
```

SHR Instruction

The SHR (shift right) instruction performs a logical right shift on the destination operand, replacing the highest bit with a 0. The lowest bit is copied into the Carry flag, and the bit that was in the Carry flag is lost:

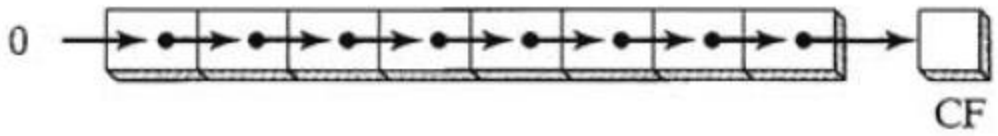


Figure: SHR instruction

Syntax:
SHR uses the same instruction formats as SHL.

SHR destination, count

The following lists the types of operands permitted by this instruction:

SHR reg,imm8
SHR mem,imm8
SHR reg, CL
SHR mem,CL

Example:
In the following example, the 0 from the lowest bit in AL is copied into the Carry flag, and the highest bit in AL is cleared:

```
mov al,0D0h   ; AL=11010000b
shr al, l     ; AL=01101000b CF=0
```

Logical Shift

- A ***Left Logical Shift*** of one position moves each bit to the left by one. The vacant least significant bit (LSB) is filled with zero and the most significant bit (MSB) is discarded.
- A ***Right Logical Shift*** of one position moves each bit to the right by one. The least significant bit is discarded and the vacant MSB is filled with zero.

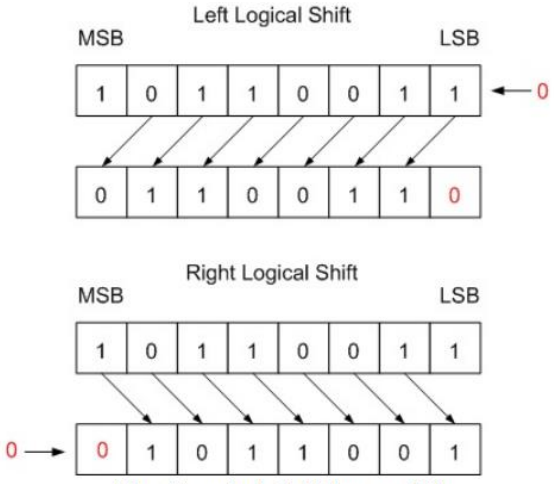


Fig. 1 Logical Shift by one bit

SAL Instruction

SAL (shift arithmetic left) is identical to the SHL instruction .

SAR Instruction

The SAR (shift arithmetic right) instruction performs a right arithmetic shift on its destination operand:

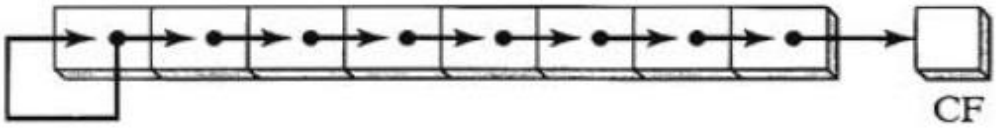


Figure: SAR instruction

Syntax:

The syntax and operands for SAL and SAR are identical to those for SHL and SHR. The shift may be repeated, based on the counter in the second operand:

SAR *destination, count*

Example:

The following example shows how SAR duplicates the sign bit. AL is negative before and after it is shifted to the right :

```
mov al,0F0h    ; AL=11110000b
sar al,1       ; AL=11111000b CF=0
```

Arithmetic Shift

- A ***Left Arithmetic Shift*** of one position moves each bit to the left by one. The vacant least significant bit (LSB) is filled with zero and the most significant bit (MSB) is discarded. It is identical to Left Logical Shift.
- A ***Right Arithmetic Shift*** of one position moves each bit to the right by one. The least significant bit is discarded and the vacant MSB is filled with the value of the previous (now shifted one position to the right) MSB.

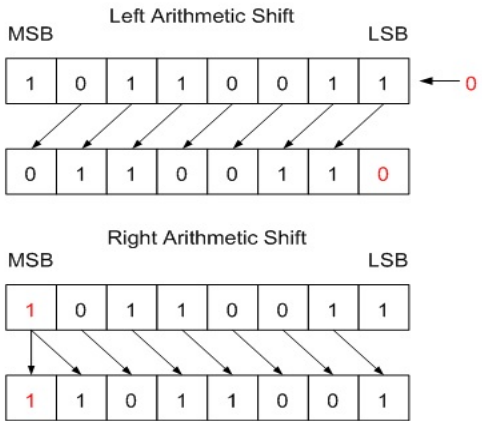


Fig. 1 Left and Right Arithmetic Shift by One Bit

ROL Instruction

The ROL (rotate left) instruction shifts each bit to the left. Also, the highest bit is copied both into the Carry flag and into the lowest bit. The instruction format is the same as for the SHL instruction:

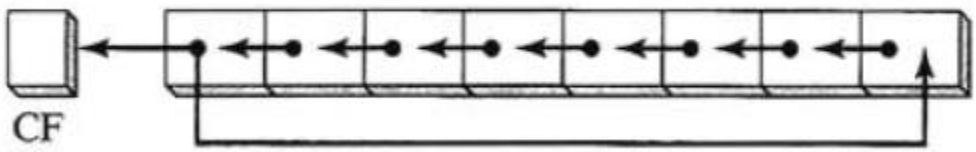


Figure: ROL instruction

Example:

Bit rotation differs from bit shifting in that the former does not lose any bits. A bit that is rotated off one end of a number appears again at the other end. In the following example, the high bit is copied into both the Carry flag and bit position 0:

```
mov al,40h    ; AL=01000000b
rol al,l      ; AL=10000000b    CF=0
rol al,l      ; AL=00000001b    CF=1
rol al,l      ; AL=00000010b    CF=0
```

ROR Instruction

The ROR (rotate right) instruction shifts each bit to the right. Also, the lowest bit is copied into the Carry flag and into the highest bit at the same time. The instruction format is the same as for SHL:

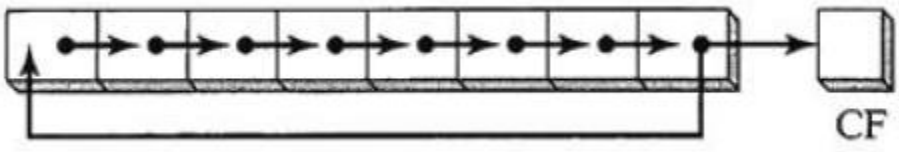


Figure: ROR instruction

Example:

In the following example, the lowest bit is copied into the Carry flag and into the highest bit of the result:

```
mov al,0lh    ; AL=00000001b
ror al,l      ; AL=10000000b CF=1
ror al,l      ; AL=01000000b CF=0
```

RCL Instruction

The RCL (rotate carry left) instruction shifts each bit to the left, copies the Carry flag to the least significant bit (LSB), and copies the most significant bit (MSB) into the Carry flag:

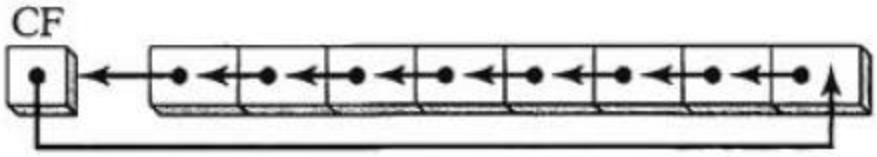


Figure: RCL instruction

Example:

If you think of the Carry flag as just an extra bit added to the high end of the number, then RCL becomes a simple rotate left operation .

In the following example, the CLC instruction clears the Carry flag. The first RCL instruction moves the high bit of BL into the Carry flag, and shifts all other bits to the left. The second RCL instruction moves the Carry flag into the lowest bit position, and shifts all other bits to the left:

```
clc          ; CF=0
mov bl,88h   ; BL=10001000b
rcl bl,l     ; BL=00010000b CF=1
rcl bl,l     ; BL=00100001b CF=0
```

RCR Instruction

The RCR (rotate carry right) instruction shifts each bit to the right , copies the Carry flag into the most significant bit, and copies the least significant bit into the Carry flag:

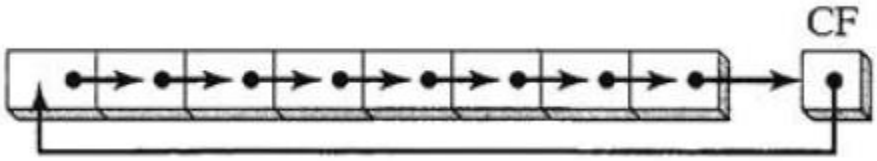


Figure: RCR instruction

Example:

As in the case of RCL, it helps to visualize the integer in this figure as a 9-bit value, with the Carry flag to the right of the least significant bit.

In the following example**, STC (set carry flag)** sets the Carry flag before rotating the Carry flag into the MSB,and rotating the LSB into the Carry flag:

```
stc          ; CF=1
mov ah,10h   ; AH=00010000b      CF=1
rcr ah,l     ; AH=10001000b      CF=0
```

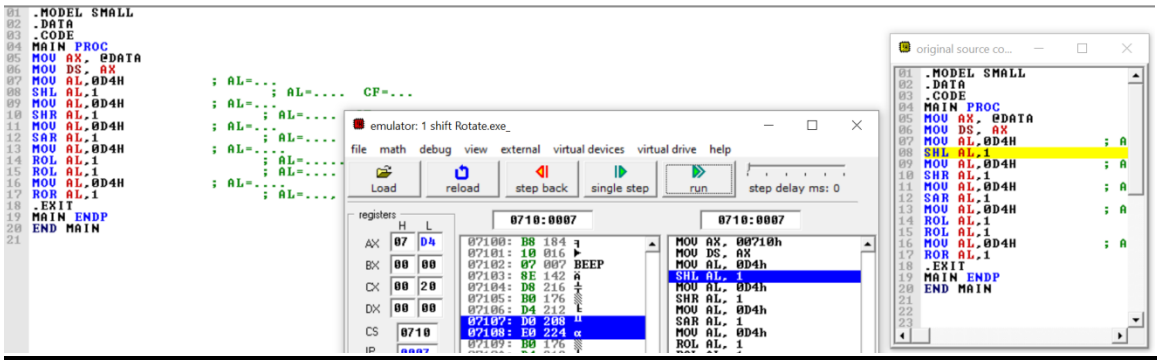
; **Name**: ……..
; **Student ID**: …….
; **Date**: …….
; *****Important***: Submit this exercise to the teacher.

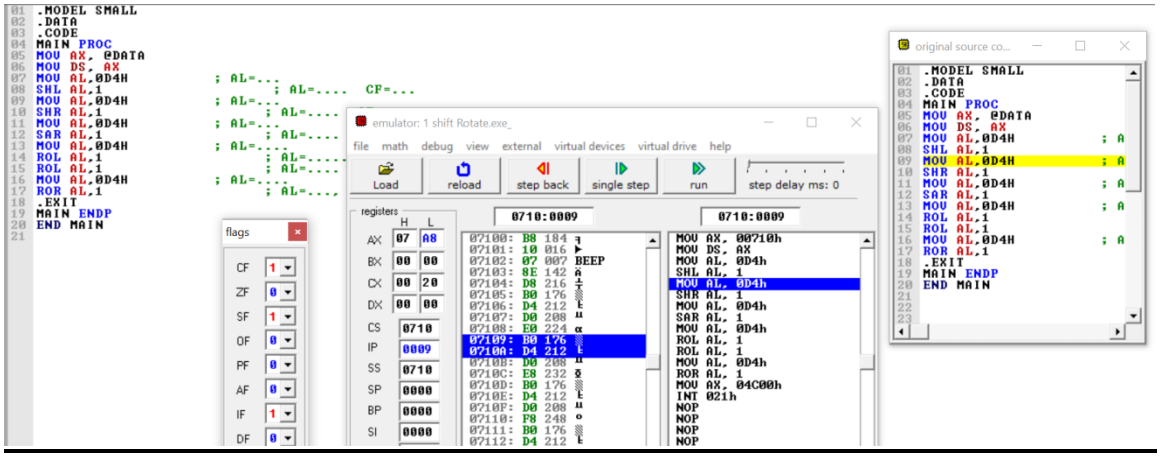
Exercise1:

Execute the following program and write the values of the registers and flags in the comment field**……**: [0.5 point]

```
TITLE SHIFT.ASM
.MODEL SMALL
.DATA
.CODE
MAIN PROC
MOV AX, @DATA
MOV DS, AX
MOV AL,0D4H       ; AL=……
SHL AL,1          ; AL=……. CF=…….
MOV AL,0D4H       ; AL=……
SHR AL,1          ; AL=……. CF=…….
MOV AL,0D4H       ; AL=……
SAR AL,1          ; AL=……. CF=…….
MOV AL,0D4H       ; AL=…….
ROL AL,1          ; AL=……., CF=……
ROL AL,1          ; AL=……., CF=……
MOV AL,0D4H       ; AL=…….
ROR AL,1          ; AL=……., CF=……
.EXIT
MAIN ENDP
END MAIN
```

Show your calculation here





Procedure of debugging

Step1: Open **Command Prompt** and go to the directory (type **cd masm32** and then type **cd bin**). Then open editor (type **edit**) and type the assembly program. Save the file as **SHIFT. ASM** (**File>Save**) and exit (**File>Exit**).

Step 2: Assemble and link the assembly program (type **ml SHIFT.ASM and link SHIFT.OBJ respectively**). If no error then go to Step 3. If there is error, then open editor (type **edit**) and open the file (**File>open> SHIFT.ASM**) and correct the errors.

Step 3: Debug the program by typing **debug SHIFT.EXE**). Open Register window. Single step the program (Press **R to view the registers' status and then press T for tracing the program**). Write the values of the registers and flags in the comment fields. Show your calculation.

Step 4: Show your calculation.

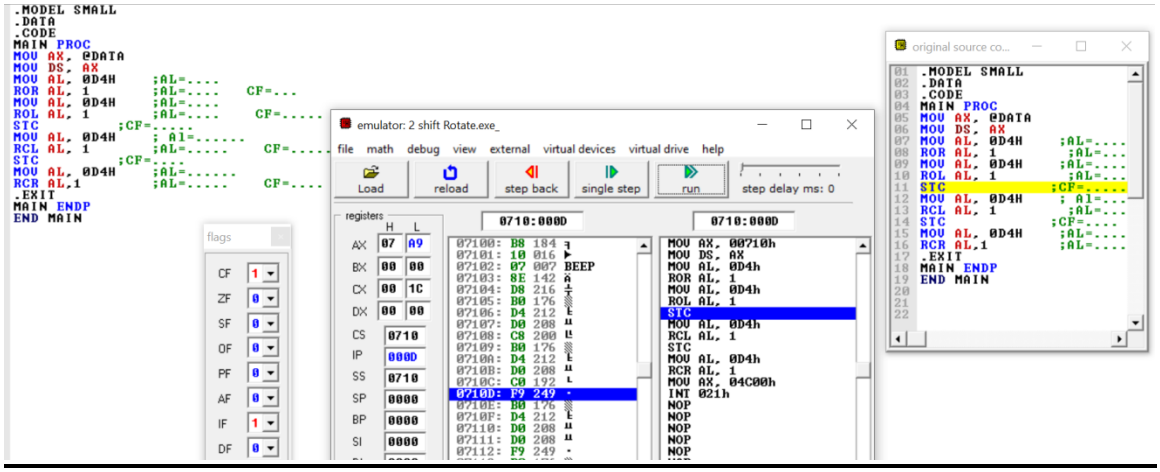
Exercise2:

Execute the following program and write the values of the registers and flags in the comment field**……**: [0.5 point]

Problem 2:

```
TITLE SHIFT2.ASM
.MODEL SMALL
.DATA
.CODE
MAIN PROC
MOV AX, @DATA
MOV DS, AX
MOV AL, 0D4H      ;AL=…….
ROR AL, 1         ;AL=…….. CF=…….
MOV AL, 0D4H      ;AL=……
ROL AL, 1         ;AL=……  CF=…….
STC               ;CF=…….
MOV AL, 0D4H      ; Al=……..
RCL AL, 1         ;AL=……. CF=…..
STC               ;CF=……
MOV AL, 0D4H      ;AL=……..
RCR AL,1          ;AL=……. CF=……
.EXIT
MAIN ENDP
END MAIN
```

Show your calculation here



Procedure of debugging

Step1: Open **Command Prompt** and go to the directory (type **cd masm32** and then type **cd bin**). Then open editor (type **edit**) and type the assembly program. Save the file as **SHIFT. ASM** (**File>Save**) and exit (**File>Exit**).

Step 2: Assemble and link the assembly program (type **ml SHIFT.ASM and link SHIFT.OBJ respectively**). If no error then go to Step 3. If there is error, then open editor (type **edit**) and open the file (**File>open> SHIFT.ASM**) and correct the errors.

Step 3: Debug the program by typing **debug SHIFT.EXE**). Open Register window. Single step the program (Press **R to view the registers' status and then press T for tracing the program**). Write the values of the registers and flags in the comment fields. Show your calculation.

Step 4: Show your calculation.