**Lab Session 07**

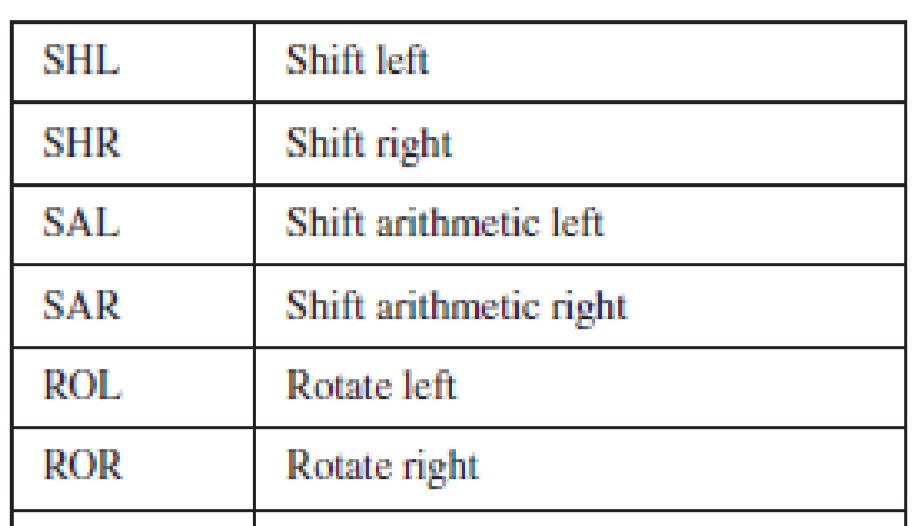
**Objective:**

* To understand the use of Shift and Rotate instructions.
* To be able to differentiate between Arithmetic shift and Logical shift.

**Theory:**

**Shift and Rotate Instructions**

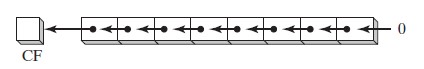
Along with bitwise instructions, shift instructions are among the most characteristic of assembly language. *Shifting* means to move bits right and left inside an operand. x86 processors provide a particularly rich set of instructions in this area, all affecting the Overflow and Carry flags.



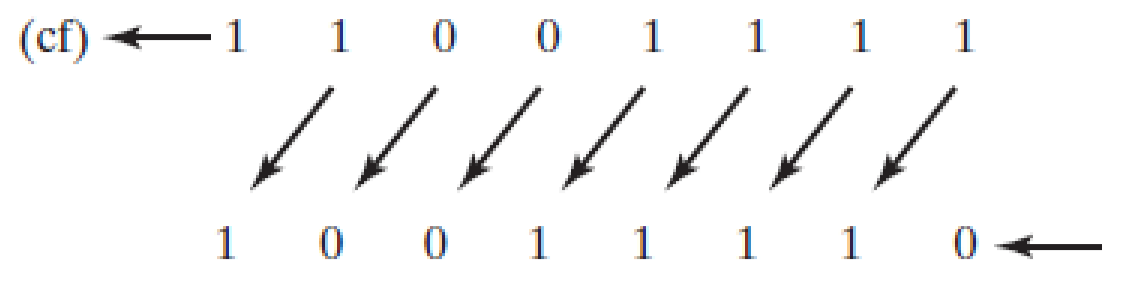
**Logical Shifts and Arithmetic Shifts**

**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 discarded:



If you shift 11001111 left by 1 bit, it becomes 10011110:



The first operand in SHL 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

x86 processors permit *imm8* to be any integer between 0 and 255. Alternatively, the CL register can contain a shift count. Formats shown here also apply to the SHR, SAL, SAR, ROR, ROL, RCR, and RCL instructions.

*Example*

In the following instructions, BL is shifted once to the left. The highest bit is copied into the Carry flag and the lowe+9st bit position is assigned zero:

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

# Multiple Shifts

When a value is shifted leftward multiple times, the Carry flag contains the last bit to be shifted out of the most significant bit (MSB). In the following example, bit 7 does not end up in the Carry flag because it is replaced by bit 6 (a zero):

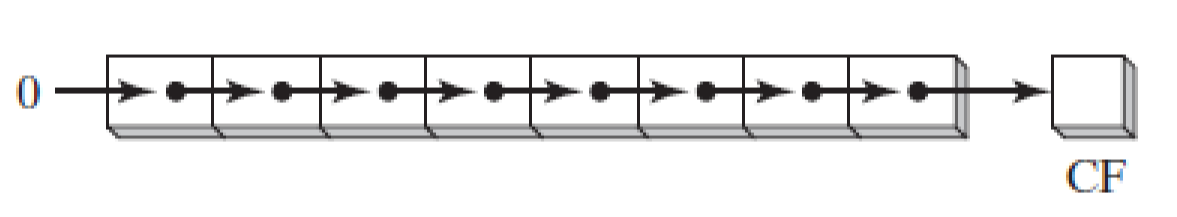
mov al,10000000b

shl al,2 ; CF = 0, AL = 00000000b

Similarly, when a value is shifted rightward multiple times, the Carry flag contains the last bit to be shifted out of the least significant bit (LSB).

**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 previously in the Carry flag is lost:



SHR uses the same instruction formats as SHL. 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 filled with a zero:

mov al,0D0h ; AL = 11010000b

shr al,1 ; AL = 01101000b, CF = 0

# Multiple Shifts

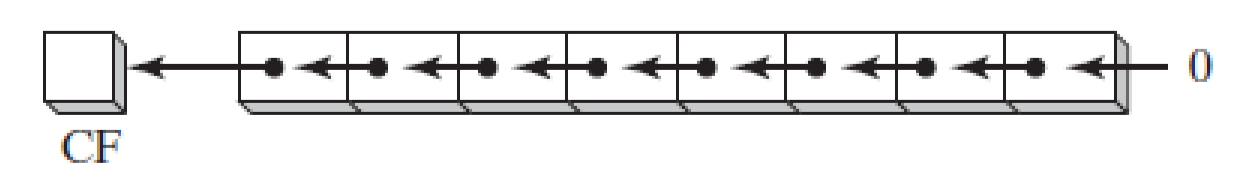
In a multiple shift operation, the last bit to be shifted out of position 0 (the LSB) ends up in the Carry flag:

mov al,00000010b

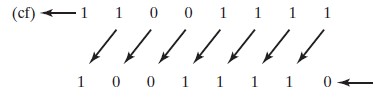
shr al,2 ; AL = 00000000b, CF = 1

**SAL and SAR Instructions**

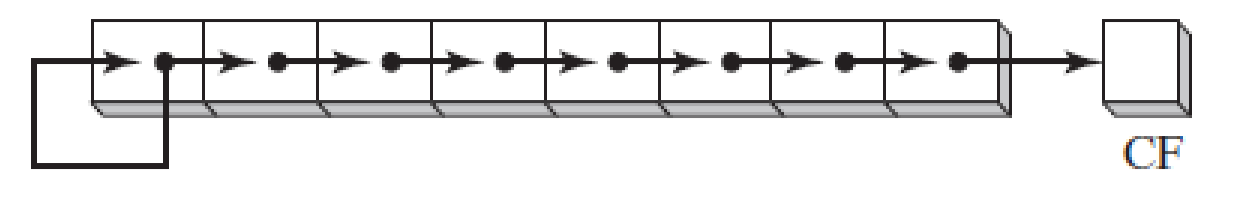
The SAL (shift arithmetic left) instruction works the same as the SHL instruction. For each shift count, SAL shifts each bit in the destination operand to the next highest bit position. The lowest bit is assigned 0. The highest bit is moved to the Carry flag, and the bit that was in the Carry flag is discarded:



If you shift binary 11001111 to the left by one bit, it becomes 10011110:



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

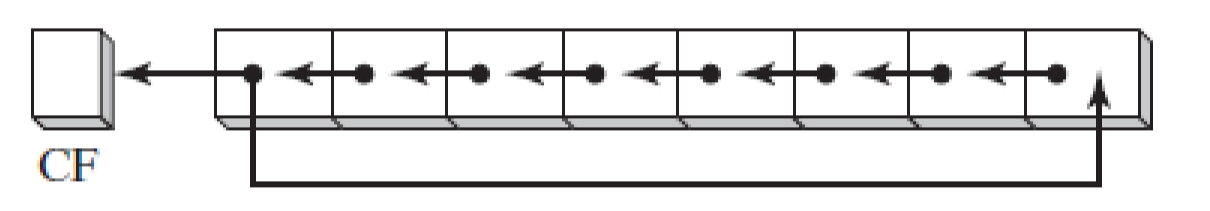


The operands for SAL and SAR are identical to those for SHL and SHR. The shift may be repeated, based on the counter in the second operand:

SAR *destination,count*

**ROL Instruction**

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



Bit rotation does not lose bits. A bit rotated off one end of a number appears again at the other end. Note in the following example how the high bit is copied into both the Carry flag and bit position 0:

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

# Multiple Rotations

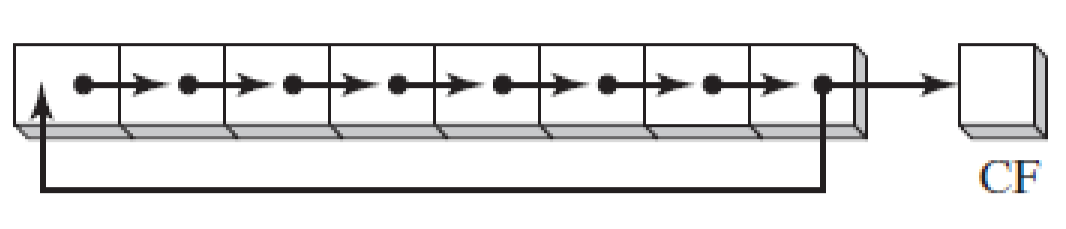
When using a rotation count greater than 1, the Carry flag contains the last bit rotated out of the MSB position:

mov al,00100000b

rol al,3 ; CF = 1, AL = 00000001b

**ROR Instruction**

The ROR (rotate right) instruction shifts each bit to the right and copies the lowest bit into the Carry flag and the highest bit position. The instruction format is the same as for SHL:



In the following examples, note how the lowest bit is copied into both the Carry flag and the highest bit position of the result: mov al,01h ; AL = 00000001b ror al,1 ; AL = 10000000b, CF = 1 ror al,1 ; AL = 01000000b, CF = 0

# Multiple Rotations

When using a rotation count greater than 1, the Carry flag contains the last bit rotated out of the LSB position:

mov al,00000100b

ror al,3 ; AL = 10000000b, CF = 1

Procedure:

Start *Emu8086* by selecting its icon.

Write the following codes in the text editor

Fill the observation tables accordingly

Program 01:

Shift operand1 Left. The number of shifts is set by operand2.

Algorithm:

Shift all bits left, the bit that goes off is set to CF.

Zero bit is inserted to the right-most position.

org 100h

MOV AL, 11100000b

SHL AL, 1

RET

|  |  |
| --- | --- |
| AL |  |
| CF |  |

Program 02:

Shift operand1 Right. The number of shifts is set by operand2.

Algorithm:

Shift all bits right, the bit that goes off is set to CF.

Zero bit is inserted to the left-most position.

Org 100h

MOV AL, 00000111b

SHR AL, 1

RET

|  |  |
| --- | --- |
| AL |  |
| CF |  |

Program 03:

Shift Arithmetic operand1 Left. The number of shifts is set by operand2.

Algorithm:

Shift all bits left, the bit that goes off is set to CF.

Zero bit is inserted to the right-most position.

Org 100h

MOV AL, 0E0h

SAL AL, 1

RET

|  |  |
| --- | --- |
| AL |  |
| CF |  |

Program 04:

Shift Arithmetic operand1 Right. The number of shifts is set by operand2.

Algorithm:

Shift all bits right, the bit that goes off is set to CF.

The sign bit that is inserted to the left-most position has the same value as before shift.

Org 100h

MOV AL, 0E0h

SAR AL, 1

MOV BL, 4Ch

SAR BL, 1

RET

|  |  |
| --- | --- |
| AL |  |
| CF |  |

Program 05:

Rotate operand1 left. The number of rotates is set by operand2.

Algorithm:

shift all bits left, the bit that goes off is set to CF and the same bit is inserted to the right-most position.

Org 100h

MOV AL, 1Ch

ROL AL, 1

RET

|  |  |
| --- | --- |
| AL |  |
| CF |  |

Program 06:

Rotate operand1 right. The number of rotates is set by operand2.

Algorithm:

shift all bits right, the bit that goes off is set to CF and the same bit is inserted to the left-most position.

Org 100h

MOV AL, 1Ch

ROR AL, 1

RET

|  |  |
| --- | --- |
| AL |  |
| CF |  |

Exercise:

1. Which instruction shifts each bit in an operand to the left and copies the highest bit into both the Carry flag and the lowest bit position?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Which instruction shifts each bit to the right, copies the lowest bit into the Carry flag, and copies the Carry flag into the highest bit position?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. In the following code sequence, show the value of AL after each shift or rotate instruction has executed:

Org 100h

mov al,0D4h

shr al,1 ; a. AL=\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

mov al,0D4h

sar al,1 ; b. AL=\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

mov al,0D4h

sal al,4 ; c. AL=\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

mov al,0D4h

rol al,1 ; d. AL=\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_