Assembly Language for Intel-Based Computers, 4th Edition

Kip R. Irvine

Chapter 7: Integer Arithmetic

Slides prepared by Kip R. Irvine

Revision date: 07/11/2002

- Chapter corrections Chapter corrections (Web) Assembly language sources (Web)

Lecture Overview

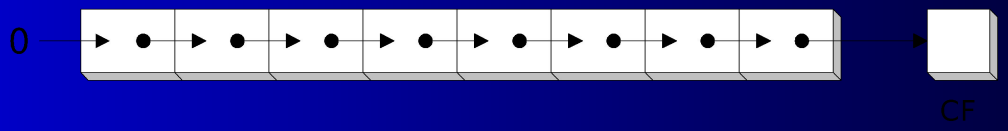
- Shift and Rotate Instructions
- Shift and Rotate Applications

Shift and Rotate Instructions

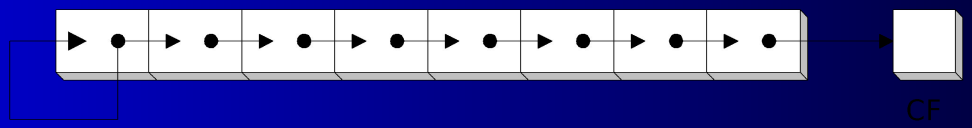
- Logical vs Arithmetic Shifts
- SHL Instruction
- SHR Instruction
- SAL and SAR Instructions
- ROL Instruction
- ROR Instruction
- RCL and RCR Instructions

Logical vs Arithmetic Shifts

- A logical shift fills the newly created bit position with zero:

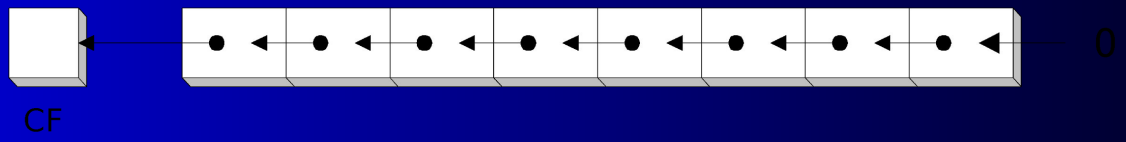


- An arithmetic shift fills the newly created bit position with a copy of the number's sign bit:



SHL Instruction

- The SHL (shift left) instruction performs a logical left shift on the destination operand, filling the lowest bit with 0.



Fast Multiplication

Shifting left 1 bit multiplies a number by 2

```
mov dl,5
shl dl,1
```

Before: 00000101 = 5

After: 00001010 = 10

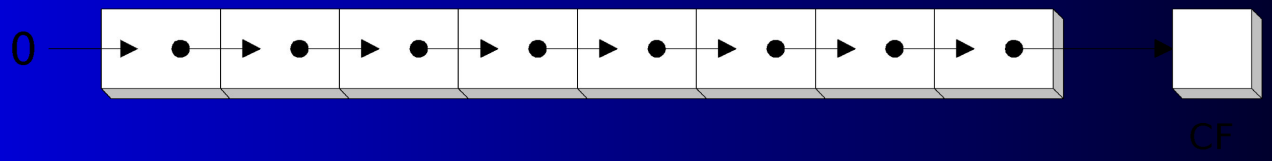
Shifting left *n* bits multiplies the operand by 2^n

For example, $5 * 2^2 = 20$

```
mov dl,5
shl dl,2   ; DL = 20
```

SHR Instruction

- The SHR (shift right) instruction performs a logical right shift on the destination operand. The highest bit position is filled with a zero.

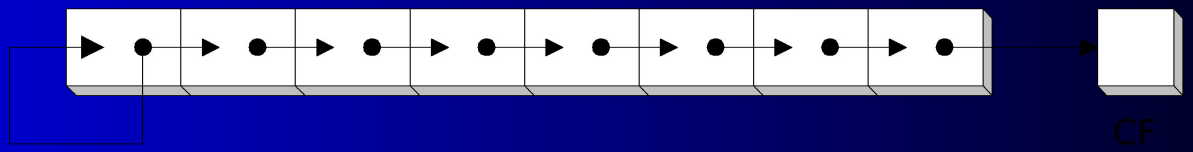


Shifting right *n* bits divides the operand by 2^n

```
mov dl,80
shr dl,1   ; DL = 40
shr dl,2   ; DL = 10
```

SAL and SAR Instructions

- SAL (shift arithmetic left) is identical to SHL.
- SAR (shift arithmetic right) performs a right arithmetic shift on the destination operand.



An arithmetic shift preserves the number's sign.

```
mov dl,-80
sar dl,1   ; DL = -40
sar dl,2   ; DL = -10
```

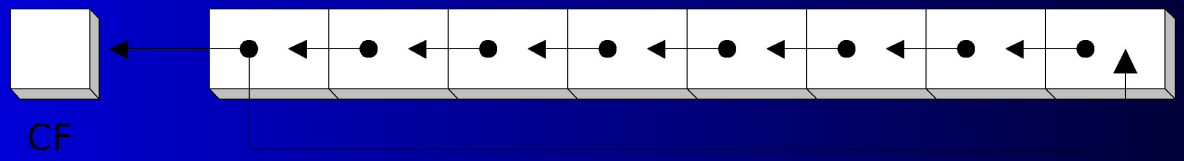
Your turn . . .

Indicate the hexadecimal value of AL after each shift:

```
mov al,6Bh
shr al,1  a.                    35h
shl al,3  b.                    A8h
mov al,8Ch
sar al,1  c.                    C6h
sar al,3  d.                    F8h
```

ROL Instruction

- ROL (rotate) shifts each bit to the left
- The highest bit is copied into both the Carry flag and into the lowest bit
- No bits are lost

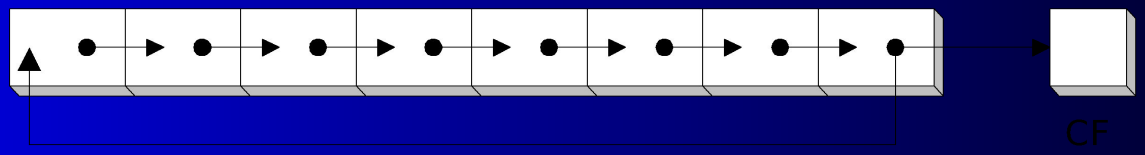



```
mov al,11110000b
rol al,1   ; AL = 11100001b

mov dl,3Fh
rol dl,4   ; DL = F3h
```

ROR Instruction

- ROR (rotate right) shifts each bit to the right
- The lowest bit is copied into both the Carry flag and into the highest bit
- No bits are lost




```
mov al,11110000b
ror al,1   ; AL = 01111000b

mov dl,3Fh
ror dl,4   ; DL = F3h
```

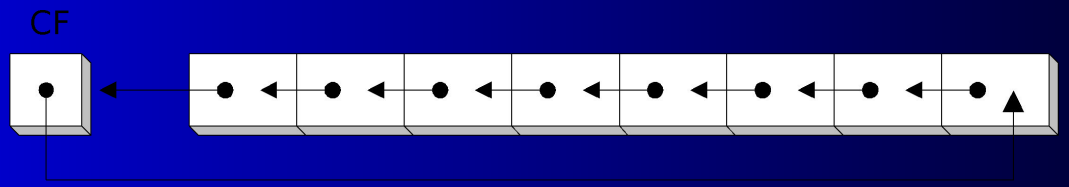
Your turn . . .

Indicate the hexadecimal value of AL after each rotation:

```
mov al,6Bh
ror al,1   a.                    B5h
rol al,3   b.                    ADh
```

RCL Instruction

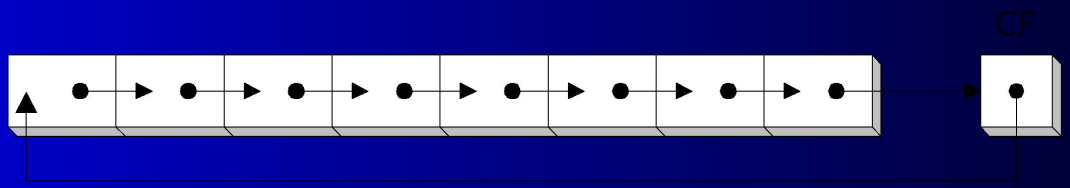
- RCL (rotate carry left) shifts each bit to the left
- Copies the Carry flag to the least significant bit
- Copies the most significant bit to the Carry flag




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

RCR Instruction

- RCR (rotate carry right) shifts each bit to the right
- Copies the Carry flag to the most significant bit
- Copies the least significant bit to the Carry flag




```
mov bl, FFh
add bl, 1 ; CF = 1
mov ah,10h   ; CF,AH = 00010000,1
rcr ah,1  ; CF,AH = 10001000,0
```

Your turn . . .

Indicate the hexadecimal value of AL after each rotation:

```
mov bl, FFh
add bl, 1
mov al,6Bh
rcr al,1   a.          B5h
rcl al,3   b.          AEh
```

The End