7. Integer Arithmetic

Tuesday, October 8, 2024 12:03 PM

* Shift and Rotate Instructions

* Bit shifting - means to move bits right & left inside an operand.

SHL, SHR, SAL, SAR – shift bits left & right
ROL, ROR, RCL, RCR – rotate bits left & right
SHLD, SHRD – double precision shift left & Right

SHL shift left (add ϕ @ LSB)

SHL dest., count
reg, imm8
Mem, imm8
reg, CL
mem, CL

imm8
0-255

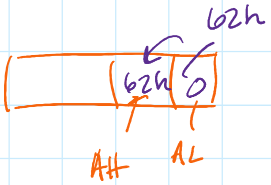
Example:

mov AL, 10110010b
SHL AL, 1 one space to left

MSB LSB
1 0 1 1.0 0 1 0

Carry Flag ← add trailing ϕ
1

0 1 1 0 0 1 0 0



* Bitwise Multiplication

* Every shift is equivalent to multiplying by 2.

$\text{number} \times 2^{n}$ → number of shifts

* Conditional jump (jc)

* jump if carry – jump if carry flag = 1
jump if carry flag is set.

SHR - shift Right (add Φ @ MSB)

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

$\underbrace{\text{imm8}}_{0-255}$ → shift right max (255)

Example:

```
MOV  AL, 10110010b
SHR  AL, 1
```

add leading Φ →

MSB 1 0 1 1 0 0 1 0 LSB → Carry Flag

0 1 0 1 1 0 0 1 → 0

* Bitwise Division
* Shift to the right results in dividing by 2.

number / 2^n → number of shifts

SAL - shift arithmetic left
→ identical to SHL

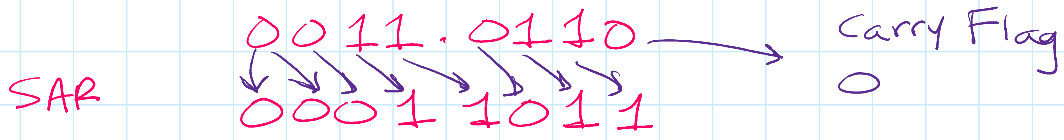
SAR - shift arithmetic right

* same as SHR, however, MSB is copied with sign bit (sign of value is preserved)

MSB 1011.0110 LSB → Carry Flag

SAR 1101.1011 → 0

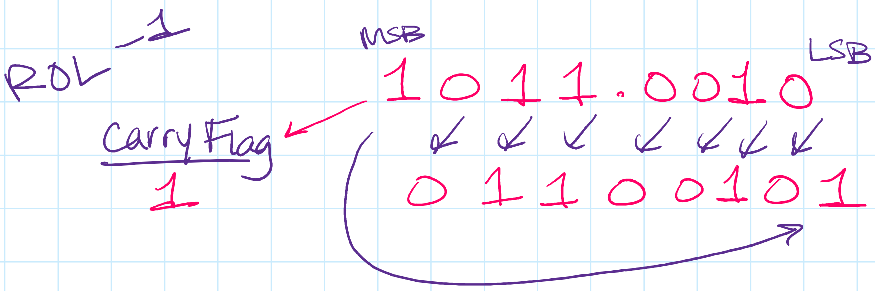
0011.0110 → Carry Flag



Signed Division

ROL — rotate left (encryption)

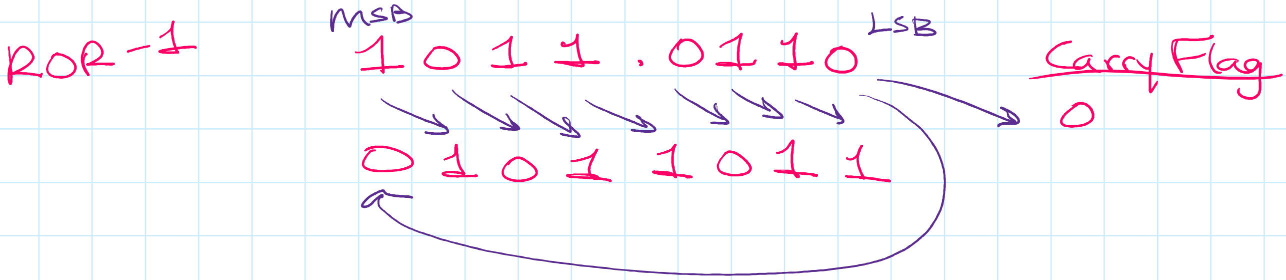
* Bitwise rotation preserves bits.
* Like shifting left, except MSB is rotated to LSB position.



```
MOV   AL, 62h
ROL   AL, 4          ; AL=26h
```

ROR — rotate right (encryption)

* Like shift right, except, LSB bit is copied to MSB position



Multiplication & Division

MUL – unsigned integer multiplication
IMUL – signed integer multiplication

MUL – unsigned multiplication (3 versions)

multiplier / multiplicand

1. 8-bit operand by AL register
2. 16-bit operand by AX register
3. 32-bit operand by EAX register

MUL reg/mem8
reg/mem16
reg/mem32

* when AX is multiplied by 16-bit operand, product is stored in combined DX & AX registers.

Example:

```
MOV AX, 2000h
MUL 0010h
```

→ multiply AX times 10h = 00020000h

$\underbrace{0002}_{2\text{ bytes}}\underbrace{0000}_{2\text{ bytes}}h$

DX	AX	CF
0002	0000	1

product is contained in DX:AX

* Carry Flag is set (1) if DX ≠ 0

Multiplicand	Multiplier	Product
AL	reg/mem8	AX
AX	reg/mem16	DX:AX

AL	reg/mem8	AX
AX	reg/mem16	DX:AX
EAX	reg/mem32	EDX:EAX

* Carry Flag is set if AH, DX, or EDX ≠ 0

* jc - jump if carry flag is set.

DIV - unsigned division (positive numbers)

→ Dividing positive integer by another one.

→ Operand is divisor

$$\frac{5 \rightarrow \text{Dividend}}{3 \rightarrow \text{divisor}} = \overset{\text{quotient}}{1} \; r \; \overset{\text{remainder}}{2}$$

```
           1 → quotient
divisor → 3 ⟌ 5 ← Dividend
            -3
             2 ← remainder
```

2 times bigger → Divisor

Dividend	Divisor	Quotient	Remainder
AX	reg/mem8	AL	AH
DX : AX	reg/mem16	AX	DX
EDX : EAX	reg/mem32	EAX	EDX

Example:

```
MOV  AX, 0083h      ; dividend
MOV  BL, 2          ; divisor
DIV  BL             ; AL=41h , AH=01h
```

$$\overset{AX}{\boxed{0083}} \;/\; \overset{BL}{\boxed{02}} \longrightarrow \overset{AX}{\boxed{01|41}}$$

$$\overset{AX}{\boxed{0083}} \Big/ \overset{BL}{\boxed{02}} \longrightarrow \overset{AX}{\underset{AH \qquad AL}{\boxed{01 \mid 41}}}$$