Seminar 911 (s1 11:10-13:00)

Thursday, October 14, 2021 11:07 AM

Representation / Interpretation

mov AX, 2

- unsigned representation of numbers
  - we represent positive natural nr.
  - base 2

  unsigned int a = 1;

  ex. 17 on a byte: 0001 0001
- signed representation of numbers
  - we represent positive / negative nr.
  - for positive nr: base 2
  - for negative nr.: 2's complement

+

−

MSB = sign bit { 1 −, 0 + }

17 0001 0001

1's complement 1110 1110 −17

17 + (−17) 0001 0001 +
1110 1110
1111 1111

	unsigned	signed
000	0	0
001	1	1
010	2	2
011	3	3
100	4	−4
101	5	−3
110	6	−2
111	7	−1

−17 |−17| = 17 0001 0001

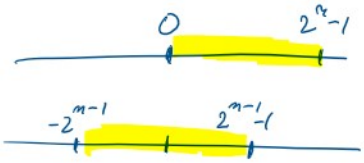
$(1110\ 1111)_2 = (-17)_{10}$

n = 3 [0, 7] unsigned rep.

[−4, 3] signed

n — unsigned $[0, 2^n - 1]$

signed $[-2^{n-1}, 2^{n-1} - 1]$



0001 0001

1110 1110 +
1
1110 1111

xxxx xxx1
1111 1111
0000 0000 −
0001 0001
1110 1111

17

A1

1110 1111

17
AL

Interpretation

AL = 1110 1111 $= (239)_{10}$ $[0, 2^8-1]$ $[0, 255]$

mul bl ; AX = AL · BL unsigned interpretation

imul bl ; AX = AL · BL signed interpretation

1 110 1111 -17

$(000\ 1\ 0001)_2 = (17)_{10}$

$AL = (0001\ 0011)_2 = (19)_{10}$

mov AL, -2
mov AL, 2

mul AL
imul AL

17 0001 0001 +
-17 1110 1111
0000 0000

CF 1 OF

On IA 32 related to signed / unsigned representation of nr.

1) → instr. which do not care about unsigned / signed repr. of nrs.
mov, add, sub
2) → instr. which interpret the operands as unsigned nr.: div, mul
3) → — // — // — // — signed nr.: idiv, imul, cbw cwd, cwde

EAX

AX
00 19 FF CO

AH AL

big endian little endian

-2 byte FE FE

-2

byte	FE	FE
word	FFFE	FE FF
dword	FFFF FFFE	FE FFFF FF

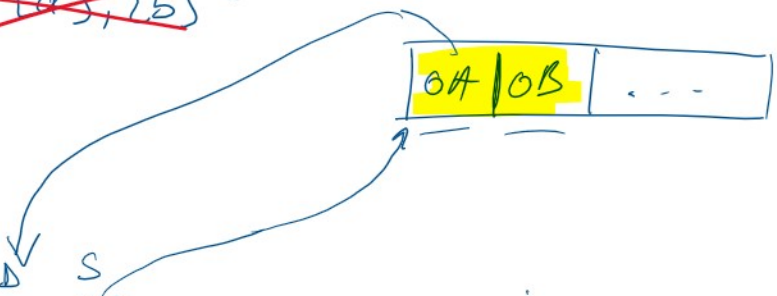
big endian 1 2 3 4

little endian 4 3 2 1

---

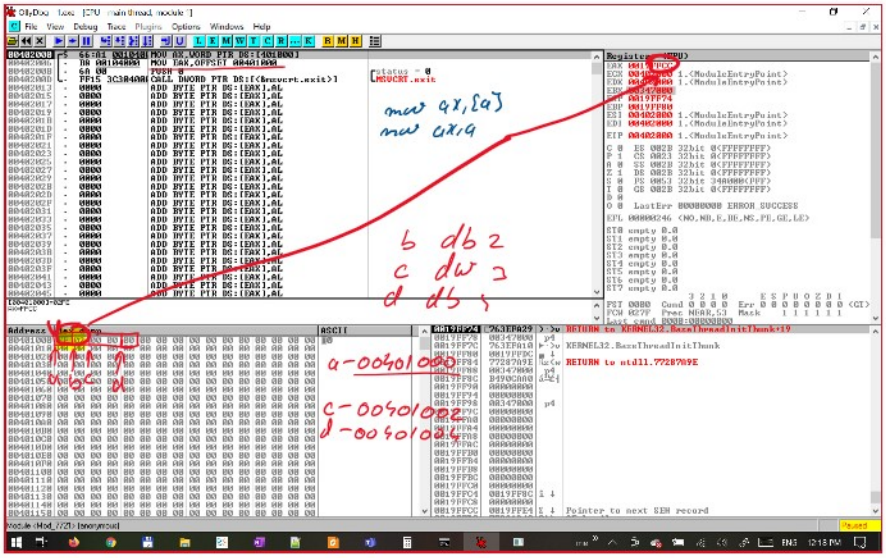
a db 10
b db 11
⋮
add eax, [a]
~~add [a], [b]~~

→ all operands must have the same size/type
↳ at least one of the operands must be a general purpose reg. or a constant and if it is a constant, the constant can not appear as a destination operand



mov ax, [a] ; AX= 0B 0A
mov eax, a ; EAX= ?

[a] a



init.

MUL , DIV , IMUL , DIV

n bits + n bits → CF (0/1) → n bits

n bits · n bits → 2n bits

2n bits / n bits → n bits + n bits

MUL source

if source - byte AX = AL · source

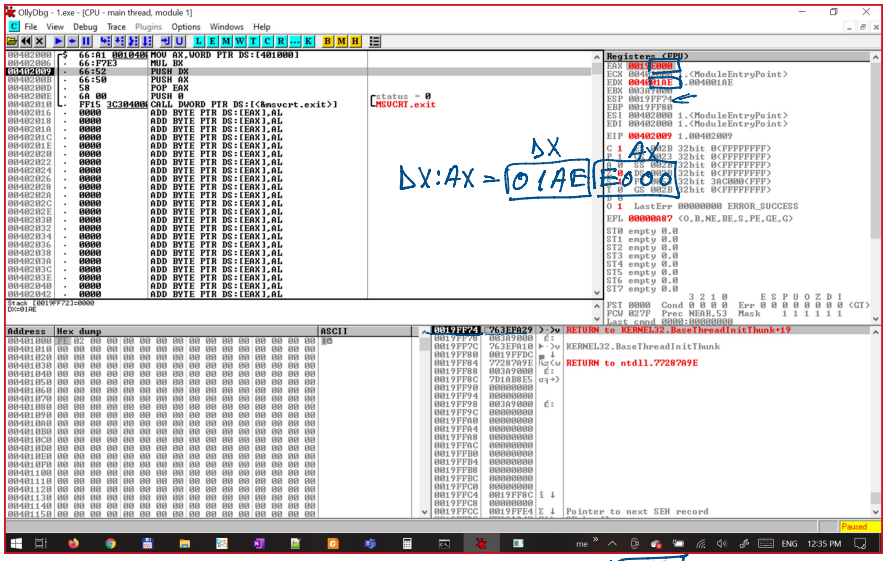
word DX:AX = AX · source

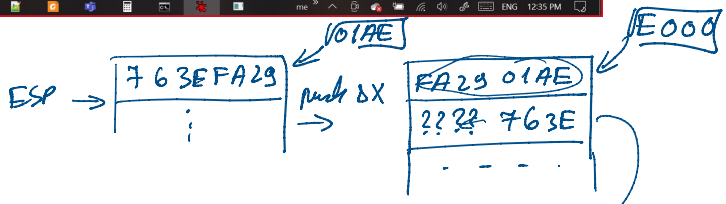
dword EDX:EAX = EAX · source

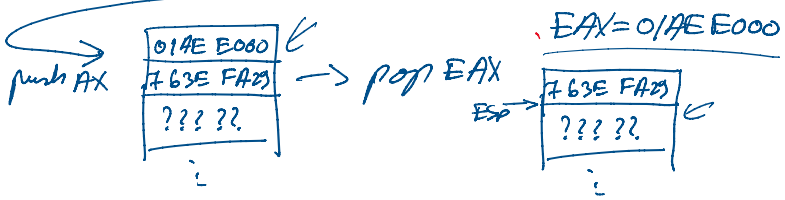
~~MUL 2~~

MUL word 2

MUL BX ; DX:AX = AX · BX



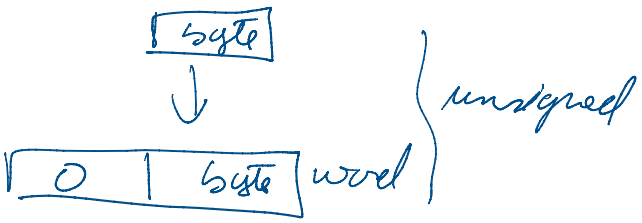


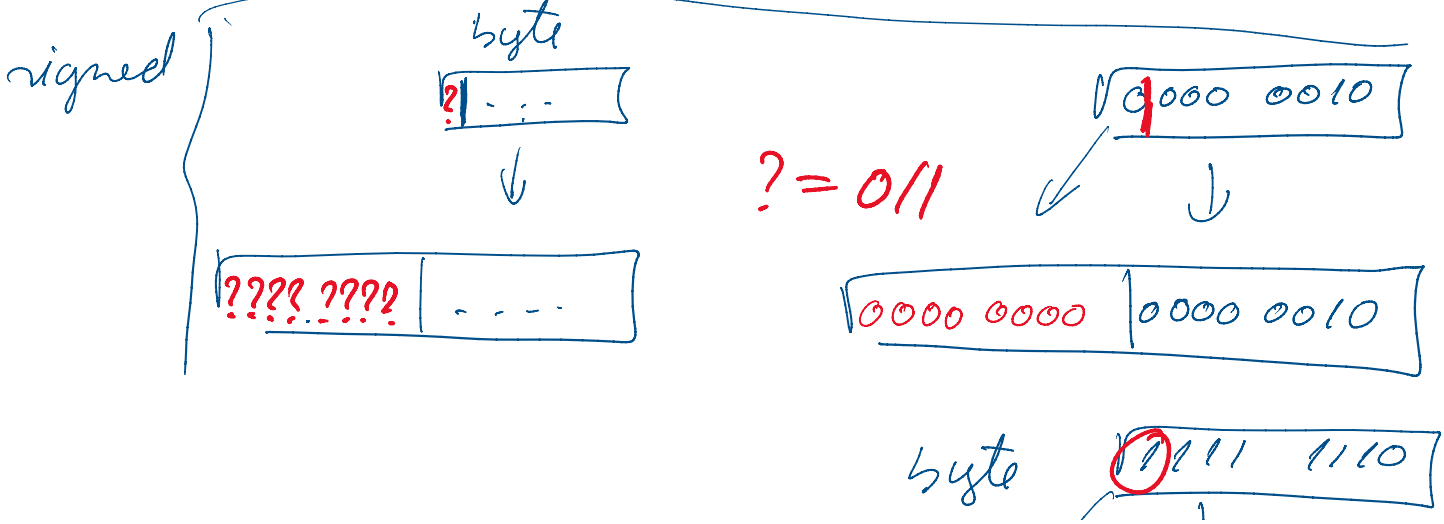


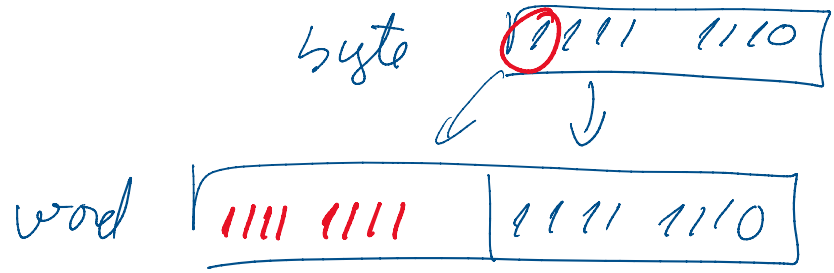
CBW , CWD , CWDE

mov al , 2
cbw

unsigned repr./interpr.








```
mov al, -2
cbw ;  AX = -2
```

CBW syntax
AX ← AL effect

CWDE

CWD syntax
DX:AX ← AX effect

```
DIV  BL  ; BL = 0
```

```
mov ax = -2   ; AX = (FFFE)16 → (65534)10
mov bl = 1      BL = 1
→ div bl ; AL =
           AH =
```