CSE 331L / EEE 332L
Microprocessor Interfacing & Embedded System
Section: 6 & 7, Summer 2021
Arithmetic and Logic Instructions

Logic Instructions: AND, OR, XOR, NOT

AND - Logical AND between all bits of two operands.

1 AND 1 = 1
1 AND 0 = 0
0 AND 1 = 0
0 AND 0 = 0

OR - Logical OR between all bits of two operands.

1 OR 1 = 1
1 OR 0 = 1
0 OR 1 = 1
0 OR 0 = 0

XOR - Logical XOR (exclusive OR) between all bits of two operands.

1 XOR 1 = 0
1 XOR 0 = 1
0 XOR 1 = 1
0 XOR 0 = 0

TEST: The same as **AND** but **for flags only**.

Format: **TEST destination, souce** Example: **TEST AL, 1** ****TEST and AND** are similar, the only difference is TEST doesn't write the result of the operation on destination.	AL: 1001 1011 (155) 0000 0001 --- 0000 0001 => not zero => odd AL: 1001 1010 (154) 0000 0001 --- 0000 0000 => zero => even

Example: Read a character and check if the input contains an even number. If it is even, print 'e' otherwise do nothing.

```
MOV AH, 1
INT 21H
XOR AH, AH  ;set ah to 0
MOV BL, 2

DIV BL      ;AX/BL: quotient in al,
            ;remainder in ah

CMP AH, 0   ;check if remainder is 0
            ;means even number
JE PRINT_E  ;goto label

JMP EXIT    ;if not even, goto exit

PRINT_E:
MOV AH, 2
MOV DL, 'E'
INT 21H     ;print 'E'

EXIT:
MOV AH, 4CH
INT 21H
```

```
MOV AH, 1
INT 21H

TEST AL, 1
;checks if LSB is zero
JZ PRINT_E

JMP EXIT
;if not even, goto exit

PRINT_E:
MOV AH, 2
MOV DL, 'E'
INT 21H

EXIT:
MOV AH, 4CH
INT 21H
```

Arithmetic Instructions: ADD, SUB, INC, DEC, MUL, IMUL, DIV, IDIV

Instruction	**Algorithm (= is assignment)**
MUL (unsigned multiplication)	MUL Source (register/memory loc) **Algorithm (byte):** AX = AL x Source **Algorithm (word):** DX:AX = AX x Source (register/memory loc)
IMUL (signed multiplication)	IMUL Source (register/memory loc) **Algorithm (byte):** AX = AL x Source **Algorithm (word):** DX:AX = AX x Source
DIV (unsigned multiplication)	DIV divisor (register/memory loc) **Algorithm (byte):** AL (quotient) = AX / divisor AH (remainder) = AX % divisor **Algorithm (word):** AX (quotient) = (DX:AX) / divisor DX (remainder) = (DX:AX) % divisor
IDIV (signed multiplication)	IDIV divisor (register/memory loc) **Algorithm (byte):** AL (quotient) = AX / divisor AH (remainder) = AX % divisor **Algorithm (word):** AX (quotient) = (DX:AX) / divisor DX (remainder) = (DX:AX) % divisor

Example: Factorial of 5

```
01 include emu8086.inc
02
03 .model small
04 .stack 100h
05 .data
06     n db 5
07 .code
08
09     mov ax, @data
10     mov ds, ax
11
12     mov cl, n
13     mov al, 1
14
15     fact:
16     mul cl
17
18     dec cl
19     cmp cl, 1
20     jne fact
21
22     call print_num
23
24     define_print_num
25     define_print_num_uns
26
```

Example: division

```
.model small
.stack 100h
.data
    n db 4
.code
    mov ax, @data
    mov ds, ax

    mov ax, 25

    div n

    mov dl, al  ;quotient in dl
    mov dh, ah  ;remainder in dh

    mov ah, 2
    add dl, 30h
    int 21h     ;display quotient

    mov dl, 20h
    int 21h     ;print space

    mov dl, dh
    add dl, 30h
    int 21h     ;display remainder

    mov ah, 4ch
    int 21h
```