# LECTURE 10

1. PUSHF

2) POP EAX

- after 1), 2) the actual config of the flag register is in EAX

-how to put back the register:

-PUSH EAX

-POPF

!!!!!!this is how we change the value of a flag/whole register

KNOW VERY WELL FOR SATURDAY/SESIUNE: CF, OF, SF, ZF

**Type conversion instructions (destructive)**

-CWD/CWDE backwards compatibility

**MOVZX d,s**

-zero extension (equiv to mov ah,0/mov dx,0)

**MOVSX d,s**

-sign extension

-d must always be a register

-larger than the source

-destination operand cannot be a memory operand (destructive conv=there is an assignment=>d must be a L-VALUE)

Ex:

Movsx ah, [v] -we cannot fit the instruction in any way (syntax error)

Movzx ax, [v] -ok (movzx ax, byte ptr[v])

Movsx eax, [v] -syntax error (operation size not specified!)

Both possibilities are valid: -convert byte [v] to eax

-convert word [v] to eax

-same to type operators (the assembler cannot choose for us)

a)Movzx eax, -1 -syntax error

b)Movsx eax, v

-syntax error

-address is a doubleword

-1)mov eax, v

-2)mov ax, v-correct but comes with a warning

-if v can be fit in a word and then it can be converted in a doubleword BUT THE SYNTAX SAYS THAT THE SOURCE MUST BE REGISTER/MEMORY OPERAND -here it is a constant!!

-if we have instruction a op1 op2 (operands can be constants, register or memory address [])

a),b) similar!!

Why do we have imul and idiv but not idd or isub?

-addition/subtraction work the same

**Branching,jumps,loops**

**JMP**

-flags-offer the possibility to put conditions in conditional jumps

-the operand of a jump can be a register

Ex: 1)mov eax, etich

Jmp eax ;register operand take the contents of the register and interpret them as a 32bit offset

-if we would have had jmp [eax]-transform the register in memory address operand

-we would go to the offset 00402008(for ex) and take the content from this address (which we don’t want-we will not find what we find)

Etich: ….

2)why do we write jmp[salt]?

Segment data

Salt.. -the offsets of the var of data segment are 00401…h

Lets assume the offset of salt=00401000h

Dest 00402008h -the dest of the jump must be 00402008h (for ex)

If we don’t put [] it goes to 00401000h

**Cmp/test**

-they are not destructive instructions

-cmp performs fictious d-s (it actually modifies the flags)

-why fictious subtraction?

-we want to know how is d to s => how is d-s<0

-who does the comparison?

-no one of these does compare!!

-they only do fictious subtraction

-the comparison is done at conditional jumps!!

UNCONDITIONAL JUMP

-why do we use a call??

-to return (come back)

-if we perform a call we have to also perform a ret (but it is not mandatory)

- CALL salveaza adresa de revenire pe stiva si face un jump operand!!

-why do we use ret??

-the operand is implicit

-if we don’t have a call and perform ret it will take sth from the stack and perform a jump

-ret -important to multi-module programming!!

Ret[n]- the n=if the n appears-we command the processor to free the stack from n bytes

**Loop**

-JECXZ it was intended to be ,,a guard” when entering the loop

Dinnou

Mov eax,89

…

Jmp maideparte

Resd1000h

Maideparte:

Mov ebx, 17

Loop dinnou= Dec ecx

Jnz dinnou-so as not to have a syntax error

**String constants**

* ‘x’-ascii code size=1
* “x”-size=2 (string)-‘x’|’\0’

-