Address Register

Why they did not implement addition and (-) with the result on the same size of the operands and not in the same way as the multiplication and division?

Why the designer of the processor provided 2 different flags for overflows

Because addition is only 1 and it gives in binary became 2 possible different interpretation

How can you tell the processor to perform an addition in the unsigned interpretation (binary)

You can't! always have 2 different interpretation

Why we don't have iadd and isub?

Because the interpretation would be the same as sub and add?

Please present isub, iadd?

iadd, isub do not exist!

- flags that shows us what happened in the last operation:
  → CF, OF, PF, ZF, SF, AF
- → flags that have to be set by the programmer to influence the execution of some instructions
  CF, IF, DF, TF

DF
- CLD (DF=0)
- STD (DF=1)

7 instructions

CF
- CLC (CF=0)
- STC (CF=1)
- CMC ($CF = \overline{CF}$)

Nr.	Starting Address (Base)	Limit
3	0752ABh	16KB
8	7CD90Bh	04GB
		4GB

$2^{16} = 2^{6} \cdot \underbrace{2^{10}}_{1KB} = 64KB$

Segment registers

Be. they express that these registers...

to 32 bit value

segment restricted

Segment selectors

Yes

Why the majority of registers were extended to 32 bits but not extended registers?

A: There is no need for
16 bits, they were initially
the starting address the starting
address of that segment but is
32 bits programming they are not allowed
to hold their starting address.
16 bits are enough...
32 bytes
4 GB
0 .... 4 GB
flat memory model
- from a physical pov:
from a logical pov:
bc. are separat parts

Nu se termină pasta?

Like I told you

We talk about segmentation
4 types seg
-> code segment
-> data seg
-> stack seg.
-> extra seg.

6 seg registers: CS V
DS V
SS V
ES V
FS -
GS -
corresponding EIP

What does cs seg contains?
-> contains the extra code seg.
CS -> seg selectors carries pointing to the ...
EIP -> instruction pointer
What is the offset?
or
Which
CS:EIP
-> darning the execution
shows the address of
the currently executing
programm 8:08
Yes