***ASC – LAB1 (13.10.2023)***

curs 60%

laborator 40%

- test lab 3 -> 10%

- tema cu deadline in ianuarie/decembrie -> 40% (minim 5)

- test lab 50% (minim 5)

**AVANTAJE**

* debugging low level
* putem face inginerie inversa
* optimizararea programelor
* vom intelege cum functioneaza input/output-ul intr-un sistem de operare

**DEZAVANTAJE**

* scrie mai mult cod
* un program in Assembly e mai greu de inteles
* debugging mai dificil

x86, sintaxa AT&T

A aparut pe Intel 8086 -> AMD

* CISC = Complex Instruction Set Computer
* RISC = Reduced Instruction Set Computer: ARM, RISC-V, MIPS

1. **REGISTRII**

Procesorul poate realiza operatii folosing registrii. Valorile acestora sunt preluate de regula din memorie.

1. **INSTRUCTIUNI**

Procesorul executa instructiuni in ordine.

1. **ACCESAREA MEMORIEI**
2. **STIVA**

Zona de memorie de tip LIFO.

1. **FLAG-U**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **16 biti** | **8 biti** | **8 biti** |
| **eax** | AX: | AH | AL |
| **ebx** | BX: | BH | BL |
| **ecx** | CX: | CH | CL |
| **edx** | DX: | DH | DL |
| **Source Index** | ESI | | |
| **Destination Index** | EDI | | |
| **Base Pointer** | EBP | | |
| **Stack Pointer** | ESP | | |
|  | | | |

***byte*** *–* folosit pt. numere intregi (1 byte)

***single*** – folosit pt. a salva numere fractionare (4 bytes)

***word*** – numere intregi (2 bytes)

***long*** – numere intregi (4 bytes)

***quad*** – numere intregi pe 64 de biti (8 bytes)

***ascii*** – siruri de caractere fara *caracter terminal*

***asciz*** – siruri de caractere cu *caracter terminal*

***space*** – zona de memorie de n bytes

*exemplu:*

x: .long 5

y: .single 5.12

z: .asciz “Hello, World!”

w: .space 400

mov $1, %eax <- eax=1

mov $0, %ebx <- ebx=0

exit(0) <- codul apelului este 1

in ebx punem argumentul apelului de sistem, in cazul nostru este 0

int $0x80 <- fa un apel de sistem

write(fd, text, lungime\_text)

mov $4, %eax

mov $1, %ebx

mov $text, %ecx

mov $20, %edx

int $0x80