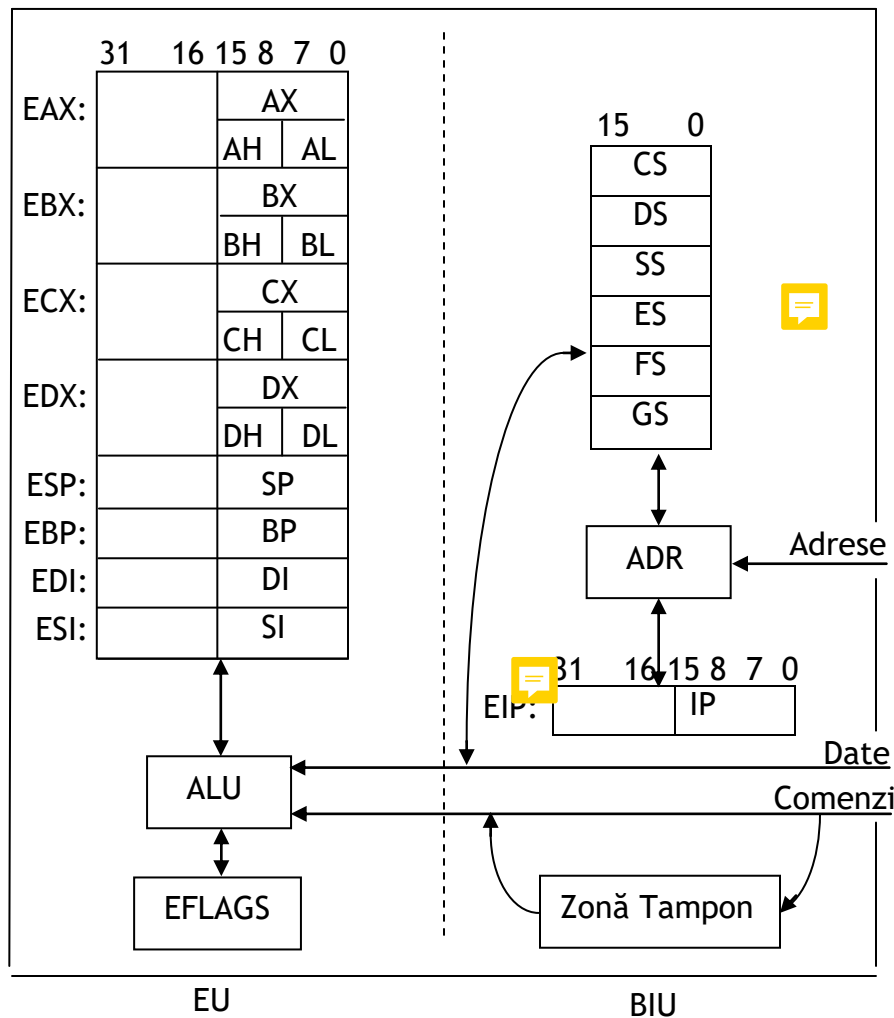


2.6. ARHITECTURA MICROPROCESOARELOR x86 (IA-32)

2.6.1. Structura microprocesorului

Microprocesorul x86 este format din două componente principale:

- **EU** (*Executive Unit*) - execută instr. mașină prin intermediul componentei **ALU** (*Aritmetic and Logic Unit*).
- **BIU** (*Bus Interface Unit*) - pregătește execuția fiecărei instrucțiuni mașină. Citește o instrucțiune din memorie, o decodifică și calculează adresa din memorie a unui eventual operand. Configurația rezultată este depusă într-o zonă tampon cu dimensiunea de 15 octeți, de unde va fi preluată de **EU**.



EU si **BIU** lucrează în paralel - în timp ce **EU** execută instrucțiunea curentă, **BIU** pregătește instrucțiunea următoare. Cele două acțiuni sunt sincronizate - cea care termină prima așteaptă după cealaltă.

2.6.2. Registrii generali EU

Registrul **EAX** este *registrul acumulator*. El este folosit de către majoritatea instrucțiunilor ca unul dintre operanzi.

Registrul **EBX** - *registru general*

Registrul **ECX** - *registru de numărare (registru contor)* pt instr care au nevoie de indicații numerice.

Registrul **EDX** - *registru de date*. Împreună cu EAX se folosește în calculele ale căror rezultate depășesc un dublucuvânt (32 biți).

"Word size" refers to the number of bits processed by a computer's CPU in one go (these days, typically 32 bits or 64 bits). Data bus size, instruction size, address size are usually multiples of the word size.

Just to confuse matters, for backwards compatibility, Microsoft Windows API defines a WORD as being A DATA TYPE on 16 bits, a DWORD being A DATA TYPE on 32 bits and a QWORD as 64 bits, regardless of the processor.

Registrii **ESP** și **EBP** sunt regiștri destinați lucrului cu *stiva*. O stivă se definește ca fiind o zonă de memorie în care se pot depune succesiv valori, extragerea lor ulterioară făcându-se în ordinea inversă depunerii.

Registrul **ESP** (*Stack Pointer*) punctează spre elementul ultim introdus în stivă (elementul din *vârful stivei*).

Registrul **EBP** (*Base pointer*) punctează spre primul element introdus în stivă (indică *baza stivei*).

Regiștrii **EDI** și **ESI** sunt *regiștrii de index* utilizați de obicei pentru accesarea elementelor din șiruri de octeți sau de cuvinte. Denumirile lor (*Destination Index* și *Source Index*) precum și rolurile lor vor fi clarificate în cap. 4.

Fiecare dintre regiștrii EAX, EBX, ECX, EDX, ESP, EBP, EDI, ESI au capacitatea de 32 biți. Fiecare dintre ei poate fi privit în același timp ca fiind format prin concatenarea (alipirea) a doi (sub)regiștri de câte 16 biți. Subregistrul superior, care conține cei mai semnificativi 16 biți ai registrului de 32 biți din care face parte, nu are denumire și nu este disponibil separat. Subregistrul inferior poate însă fi accesat individual, având astfel regiștrii de 16 biți **AX, BX, CX, DX, SP, BP, DI, SI**. Dintre aceștia, regiștrii AX, BX, CX, și DX sunt fiecare la rândul lor, formați din câte doi alți subregiștri a câte 8 biți. Există astfel regiștrii **AH, BH, CH, DH**, conținând cei 8 biți superiori (partea HIGH a regiștrilor AX, BX, CX și DX), respectiv **AL, BL, CL, DL**, conținând cei 8 biți inferiori (partea LOW).

2.6.3. Flagurile

Un *flag* este un indicator reprezentat pe un bit. O configurație a *registrului de flaguri* indică un rezumat sintetic a execuției fiecărei instrucțiuni. Pentru x86 registrul EFLAGS (the *status* register) are 32 biți dintre care sunt folosiți uzual numai 9.

| 31 | 30 | ... | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|-----|----|----|----|----|----|----|----|---|----|---|----|---|----|
| x | x | ... | x | OF | DF | IF | TF | SF | ZF | x | AF | x | PF | x | CF |

CF (*Carry Flag*) este flagul de transport. Are valoarea 1 în cazul în care în cadrul ultimei operații efectuate (UOE) s-a efectuat transport în afara domeniului de reprezentare a rezultatului și valoarea 0 în caz contrar.

PF (*Parity Flag*) - Valoarea lui se stabilește a.î. împreună cu numărul de biți 1 din octetul cel mai puțin semnificativ al reprezentării rezultatului UOE să rezulte un număr impar de cifre 1.

AF (*Auxiliary Flag*) indică valoarea transportului de la bitul 3 la bitul 4 al rezultatului UOE. De exemplu, în adunarea de mai sus transportul este 0.

ZF (*Zero Flag*) primește valoarea 1 dacă rezultatul UOE este egal cu zero și valoarea 0 la rezultat diferit de zero.

SF (*Sign Flag*) primește valoarea 1 dacă rezultatul UOE este un număr strict negativ și valoarea 0 în caz contrar.

TF (*Trap Flag*) este un flag de depanare. Dacă are valoarea 1, atunci mașina se oprește după fiecare instrucțiune.

IF (*Interrupt Flag*) este flag de întrerupere. Detalii în cap.5 din manual.

DF (*Direction Flag*) - pt operare asupra șirurilor de octeți sau de cuvinte. Dacă are valoarea 0, atunci deplasarea în șir se face de la început spre sfârșit, iar dacă are valoarea 1 este vorba de deplasări de la sfârșit spre început.

OF (*Overflow Flag*) este flag pentru depășire **CU SEMN**. Dacă rezultatul ultimei instrucțiuni în interpretarea CU SEMN a operanzilor nu a încăput în spațiul rezervat operanzilor (intervalul de reprezentare admisibil), atunci acest flag va avea valoarea 1, altfel va avea valoarea 0.

Categorii de flag-uri

Flag-urile se pot împărți în două categorii :

- a). cu efect anterior generat de către Ultima Operație Efectuată (UOE) : CF, PF, AF, ZF, SF și OF
- b). cu efect ulterior setării lor de către programator pentru influențarea modului de operare al instrucțiunilor care urmează : CF, TF, DF și IF.

Instrucțiuni specifice de setare a valorilor unor flag-uri

Având în vedere categoria b) este normal ca limbajul de asamblare să ne pună la dispoziție instrucțiuni specifice de setare a valorii flag-urilor care vor avea un efect ulterior. Aceste instrucțiuni sunt în număr de 7:

CLC – efectul fiind $CF=0$,

STC – efectul fiind $CF=1$,

CMC – complementarea valorii din CF

CLD – având ca efect $DF=0$,

STD – având ca efect $DF=1$

CLI – având ca efect $IF=0$, //nu pot fi activate în mod protejat

STI – având ca efect $IF=1$

Având în vedere riscul major de setare accidentală a valorii din TF precum și rolul său absolut special în dezvoltarea de depanatoare, NU există disponibile instrucțiuni de acces direct la valoarea din TF !!