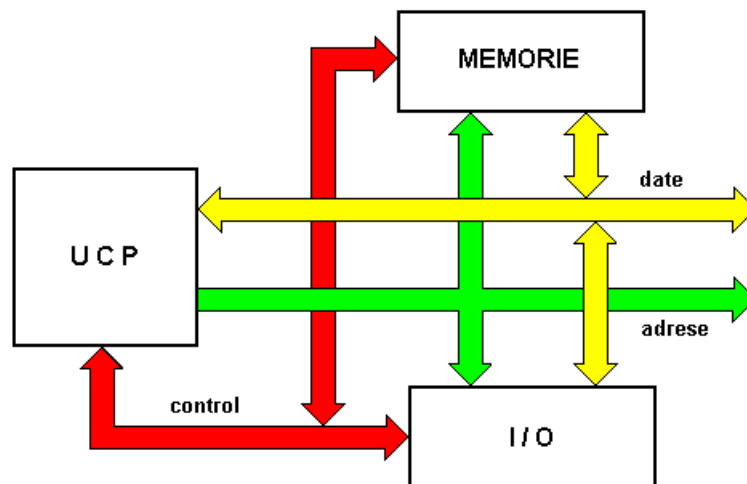


Microcalculatorul, structurat ca o mașină "VON NEUMANN", este un sistem programabil de prelucrarea informației care are două componente inseparabile și definitorii: **hardware și software**.



A. Componenta **hardware**; blocurile funcționale sunt:

1. UNITATEA CENTRALĂ DE PRELUCRARE (UCP); două funcții esențiale:

- prelucrarea datelor;
- controlul activității întregului microcalculator.

O *Unitate centrală de prelucrare informației*, având funcțiile enunțate mai sus, care coordonează un sistem structurat funcțional ca în figură și care, fizic, se prezintă sub forma unui singur cip se numește **MICROPROCESOR (μP)**.

2. MEMORIA este o secvență de locații pentru stocarea informației. Fiecare locație este definită prin două entități informaționale:

- **Conținutul**, reprezentat de o înșiruire de cifre binare **0** sau **1** ("**biți**");
 - numere
 - coduri etc.

Numărul de cifre binare conținute într-o locație depinde de modul în care microprocesorul organizează informația în memorie; mărimea unei locații va fi denumită **formatul memoriei**, exprimat în număr de biți (de regulă 8, 16, 32 sau 64 biți).

- **Adresa**, reprezentând numărul de ordine al locației, care permite identificarea sa în cadrul secvenței de locații (există o corespondență biunivocă între fiecare locație de memorie și adresa sa).

Noțiuni aferente:

- "**Harta memoriei**": totalitatea locațiilor de memorie pe care le poate adresa un microprocesor.
- "**Pagini**" și/sau "**segmente**": subdiviziuni *logice* ale hărții memoriei, ale căror dimensiuni, fixe sau dinamice, sunt specifice modului în care un microprocesor anume organizează memoria.

Structura fizică a memoriei unui microcalculator este formată din unul sau mai multe cipuri, cu capacități diverse; capacitatea totală de stocare a informației pe care o realizează fizic cipurile de memorie într-un microcalculator este definită ca "memorie internă". Aceasta nu acoperă, în mod necesar, harta memoriei aferentă microprocesorului respectiv.

Semnificația conținutului memoriei microcalculatorului → două zone:

- **Memoria de date** conține operanzi și/sau rezultate; fizic, această porțiune de memorie este de tip RAM (cu scriere/citire).
- **Memoria de program** care conține instrucțiuni; de regulă, (dar nu obligatoriu) această zonă este o memorie de tip ROM (memorie din care se poate doar citi).

Instrucțiunea: informația codificată (binar) prin care se impune microprocesorului desfășurarea unei acțiuni specifice. Observații:

- Fiecare instrucțiune este asociată în mod biunivoc cu un șir de cifre binare; deoarece acestea "codifică" instrucțiunile, vor fi denumite **coduri**.
- O instrucțiune reprezintă cea mai simplă acțiune, cu rezultat bine precizat, din activitatea unui microcalculator a cărui *unitate centrală de prelucrare a informației* este un microprocesor anume.
- Un microprocesor concret poate "recunoaște" și executa numai codurile corespunzătoare instrucțiunilor pentru care a fost construit; totalitatea instrucțiunilor pe care un microprocesor le poate recunoaște și executa alcătuiește **setul de instrucțiuni** al microprocesorului respectiv.
- Înșiruirea instrucțiunilor în memoria de program nu este haotică ci sub formă de **programe**, noțiune definită ca fiind o secvență de coduri de instrucțiuni organizate în mod logic și coerent după un anumit algoritm, astfel încât întregul microcalculator să execute o sarcină prestabilită. Noțiunea de "**sarcină**" (**task**) nu trebuie confundată cu cea de program: sarcina unui microcalculator corespunde unei alocări dinamice a resurselor hardware și software; există sarcini pentru a căror îndeplinire sunt necesare mai multe programe.

Semnificația conținutului locațiilor de memorie este conferită de programator în concordanță cu funcțiile specifice realizate de microprocesor:

- *numere binare* atunci când ne referim la date (operanzi/rezultate);
- *coduri* când ne referim la instrucțiuni.

În schema bloc funcțională propusă, **memoria nu are nici un control asupra semnificației informației pe care o conține.**

3. DISPOZITIVELE DE INTRARE/IEȘIRE (I/O): circuitele prin care se realizează legătura între microcalculator și lumea exterioară. O unitate elementară de conversație cu exteriorul poartă numele de "**port de intrare/ieșire**".

Între porturi și locațiile din Memorie există niște similitudini:

- Porturile sunt în esență tot locații de memorare a informației, adresabile; informația care se folosește uzual aici este alcătuită din operanzi/rezultate (date).
- Există o "**hartă a porturilor**" care poate sau nu să facă parte din harta memoriei.

Singura deosebire esențială față de locațiile de memorie este *legătura fizică pe care porturile o asigură cu exteriorul*; pentru microprocesor, de multe ori, această legătură fizică este transparentă și ne semnificativă.

"**Magistrală**": un set de conexiuni fizice între blocuri prin care informația care circulă are o semnificație prestabilită. Sistemele la care ne referim au o magistrală unică, ce le caracterizează; din punct de vedere funcțional, există trei componente ale acestei magistrale:

1. *Magistrala de date*, bidirecțională, permite circulația datelor (operanți/rezultate), a instrucțiunilor și chiar a adreselor.

2. *Magistrala de adrese*, unidirecțională, permite microprocesorului să localizeze informația în Memorie sau în Dispozitivele de intrare/ieșire; deci pe această magistrală circulă numai adrese.

3. *Magistrala de control* permite circulația, bidirecțională, a semnalelor de comandă și control de la/microprocesor, în calitate sa de Unitate centrală.

B. Componenta *software*: o serie de programe organizate în moduri specifice.

Două categorii de software:

1. *Sistemul de operare*: totalitatea programelor care permit utilizatorului accesul complet la resursele sistemului (exemple: MS-DOS, UNIX etc.). Poate fi: rezident (permanent în memoria internă) sau încărcabil dintr-o memorie externă (operație denumită "bootstrap").

2. *Software-ul utilizatorului*, alcătuit din totalitatea programelor folosite pentru îndeplinirea unor sarcini specifice.

Caracteristicile arhitecturii "Von Neumann":

- Microprocesorul constituie Unitatea centrală de prelucrare a unui sistem având schema bloc funcțională din figură. El concentrează și funcția de prelucrare și pe cea de comandă.
- Toate celelalte componente ale sistemului nu au putere de decizie. Memoria nu controlează și nici nu e necesar să controleze semnificația informației pe care o deține și modul în care este organizată *logic*.
- Legătura dintre blocuri este asigurată de o magistrală unică cu trei componente funcționale; pe magistrala de date circulă toate tipurile de informații.
- Funcționarea sistemului se face pe baza unor programe alcătuite din secvențe de instrucțiuni. *Acestea sunt citite din memorie de către microprocesor, recunoscute și apoi executate.*

Arhitectură: **totalitatea atributelor sistemului (în cazul de față, microprocesorul) care sunt disponibile ("vizibile") utilizatorului (ca, de pildă: registrele, modurile de adresare, tipurile de transferuri de date, modul de organizare logică a memoriei, tehnicile de intrare/ieșire, setul de instrucțiuni etc).**

- *Microprocesor, microcalculator, minicalculator.*

Asemănarea: caracteristicile globale ale atributelor de arhitectură.

Deosebirile între ultimele două: resurse (memorie internă și externă, echipamente periferice) și performanțe (viteză de prelucrare, cost, număr de componente, gabarit).

- Definiția microprocesorului ca Unitate centrală de prelucrare; am presupus implicit că sistemul din care face parte este un micro(mini)calculator, deci *un sistem de calcul*. Putem extinde însă noțiunea și asupra **sistemelor de comandă și control** (de tip "**controler**"), ceea ce mărește aria de cuprindere a noțiunii de microprocesor.
- Noțiunea de "*logică programată*". Sistemele cu logică programată nu înseamnă, în mod automat, sisteme cu microprocesor. Microprocesorul poate constitui una dintre modalitățile de proiectare a sistemelor cu logică programată. Nu se va face confuzia "sistem cu logică programată cu microprocesor" ≠ "sistem microprogramat".
- Clasificări ale noțiunii de microprocesor:
 - a) După lățimea magistralei de date: microprocesoare pe 8, 16, 32 sau pe 64 de biți.
 - b) După tipul de sarcini eficient realizabile:
 - microprocesoare de uz general (μ PUG), nespecializate;
 - microprocesoare specializate, ca de pildă:
 - procesoare de intrare/ieșire, pentru conversații complexe între microcalculator și lumea exterioară; exemplu: Intel 8089;
 - coprocesoare aritmetice, specializate pentru funcții aritmetice de utilitate generală (exponențiale, trigonometrice etc); exemplu: Intel 80387;
 - procesoare digitale de semnal, specializate pentru algoritmi specifici prelucrării semnalelor (FFT, produse de corelație, filtre digitale, calcul matriceal etc.); exemplu: Texas Instruments TMS 320.
 - c) După principiile de bază ale arhitecturii care guvernează funcționarea:
 - **procesoare cu set complex de instrucțiuni (CISC)** numite microprocesoare "standard" sau simplu "microprocesoare";
 - **procesoare cu set redus de instrucțiuni (RISC).**

- **bit** (prescurtat **b**) pentru o cifră binară **0** sau **1**;
- **nibble** (prescurtat **n**) pentru o înșiruire de 4 biți;
- **byte** sau **octet** (prescurtat **B**) pentru o înșiruire de 8 biți;
- **cuvânt** sau **word** (prescurtat **w**) pentru o înșiruire de 2 octeți;
- **cuvânt dublu** sau **double word** (prescurtat **dw**) pentru o înșiruire de 4 octeți;
- prefixele:
 - **k** pentru $2^{10} \approx 10^3$;
 - **M** pentru $2^{20} \approx 10^6$;
 - **G** pentru $2^{30} \approx 10^9$;
 - **T** pentru $2^{40} \approx 10^{12}$.

a) Reprezentarea programelor

Fiecare instrucțiune este reprezentată în memorie de un *cod binar*. **Formatul instrucțiunilor**, adică totalitatea cifrelor binare necesare pentru codificare, are, de regulă, drept cuantă de informație, octetul. Pentru fiecare instrucțiune există un număr prestabilit de octeți cu care e codificată (de pildă, pentru Intel 8086, este între 1 și 6 octeți).

b) Reprezentarea numerelor

1) *Reprezentarea întregilor fără semn în "binar natural"*: este reprezentarea uzuală, "naturală" a numerelor binare.

$$\text{număr binar cu 8 cifre} = \sum_{i=0}^7 b_i 2^i \quad \text{cu } b_i \in \{0, 1\}$$

$$\begin{aligned} \text{număr binar cu 8 cifre} &= b_7b_6b_5b_4b_3b_2b_1b_0 \\ b_0 &: \text{lsb} \\ b_7 &: \text{msb} \end{aligned}$$

2) *Reprezentarea întregilor cu semn în "binar natural"*:

Semnul numărului este reprezentat de **msb** cu următoarea convenție:

msb = 0 semnifică număr **pozitiv**;

msb = 1 semnifică număr **negativ**.

Pentru un număr fără semn cu 8 biți, plaja numerelor reprezentabile acoperă 256 de poziții, între **0** și **255**, în zecimal.

Pentru un număr cu semn, plaja numerelor reprezentabile acoperă tot 256 de poziții, dar în intervalul **-128 ÷ +127**, presupunând **0** număr pozitiv.

Convenții de reprezentare:

Tipul reprezentării	+ 5	- 5
"mărime și semn"	0000101	1000101
"complement față de 1"	0000101	11111010
"complement față de 2"	0000101	11111011

Regulile de reprezentare în aceste trei convenții:

- Numerele pozitive se reprezintă identic.
- În "mărime și semn", numerele negative diferă de cele pozitive numai prin bitul de semn.
- În "complement față de 1", mărimea numărului negativ se obține din reprezentarea precedentă prin *complementare* bit cu bit; convenția pentru bitul de semn se păstrează.
- În "complement față de 2", mărimea numărului negativ se obține din reprezentarea precedentă prin *adunarea* unei cifre binare **1** la **lsb**.

"Fanioane"

- "Transportul" care apare între rangul unui număr binar și cel imediat superior în operațiile aritmetice (la scădere, îl vom numi "împrumut"): **C** (de la "*carry*").
- "Depășirea": **O** (de la "*overflow*"). După cum numărul are semn sau nu, se poate scrie că:

$$O = C_{msb} \nearrow \text{ SAU } C_{msb-1} \nearrow msb$$

Extinderea numerelor cu semn reprezentate în complement față de 2, de la 8 la 16 biți.

	Reprezentare cu 8 biți	Reprezentare cu 16 biți
+ 1	00000001	0000000000000001
- 1	11111111	1111111111111111

Regulile de "extindere a numerelor cu semn, în complement față de 2":

- Bitul de semn rămâne pe poziția cea mai semnificativă.
- Partea care reprezintă mărimea numărului va ocupa pozițiile cele mai puțin semnificative ale numărului extins.
- Restul pozițiilor din numărul extins se completează cu cifre binare identice cu cea care reprezintă semnul (**0** pentru numere pozitive și **1** pentru numere negative).

3) Reprezentarea întregilor în "zecimal codificat binar" (ZCB): se reprezintă fiecare cifră zecimală separat, în binar natural, cu un *nibble*.

Microprocesoarele folosesc două tipuri de reprezentări ZCB:

- Reprezentarea "ZCB împachetat" în care fiecare octet din memorie cuprinde câte două cifre zecimale, una pe *nibble*-ul mai puțin semnificativ și cealaltă pe *nibble*-ul superior. Plaja de numere zecimale acoperită de o reprezentare cu 8 biți se micșorează de la 256 la 100 de numere: **0 ÷ 99**.

- Reprezentarea "ZCB neîmpachetat" în care fiecare octet cuprinde o singură cifră zecimală pe *nibble*-ul mai puțin semnificativ. Restul cifrelor binare se completează cu **0**.

4) Reprezentarea numerelor cu zecimale "cu virgulă fixă": se folosește principiul de a alocă un număr fix, prestabilit, de cifre binare pentru a reprezenta partea întreagă și respectiv partea zecimală a unui număr.

Se poate folosi fie reprezentarea în binar natural fie în ZCB. Pentru partea întreagă se folosește regula de reprezentare a numerelor întregi cu semn, iar pentru partea zecimală regula de reprezentare a întregilor fără semn. (Apar: "*trunchierea*" sau "*rotunjirea*" numărului).

Modul de reprezentare folosește următoarele convenții:

- Se rezervă un șir de biți cu care se exprimă numărul total de cifre ale numărului care urmează să fie reprezentat.
- Se rezervă, apoi, un șir de biți în care se înscrie numărul de zecimale cu care se va reprezenta numărul.
- Urmează reprezentarea propriu-zisă a numărului înșiruiind reprezentările pentru partea întreagă și cea zecimală fără o altă delimitare explicită între ele.

Un exemplu în ZCB împachetat:

0000 0100 0010 0000 1001 0110 0001 0101

numărul reprezentat este **+96.15** .

5) Reprezentarea numerelor cu zecimale "cu virgulă mobilă"; reprezentare normalizată.
Două entități informaționale: "*mantisa*" **M** și "*exponentul*" **EXP**:

$$\text{număr binar} = M * 2^{EXP}$$

$$2^{-1} \leq M < 2^0 .$$

Un exemplu:

b₃₁.....b₂₄b₂₃.....b₀ ,

în care: - **b₃₁ ÷ b₂₄** reprezintă exponentul, având semnul în poziția **b₃₁**.

- **b₂₃ ÷ b₀** reprezintă mantisa cu semnul la **b₂₃**.

Plaja numerelor reprezentabile în acest fel: **M * 2^{±128}**

c) Reprezentarea datelor alfanumerice

Vom înțelege prin "date alfanumerice" sau "caractere" oricare dintre semnele care pot fi tipărite de la tastatura unui calculator.

"*Seturi de caractere*": grupuri minime de simboluri considerate suficiente pentru a asigura o editare cât mai completă a unui text.

Pentru fiecare caracter se va folosi o reprezentare binară, *un cod*, cu care caracterul (dintr-un set prestabilit) este în relație biunivocă.

Standardul "**ASCII**" (de la "**American Standard Code for Information Interchange**") cu care se codifică următorul set de caractere:

- 26 de litere mari ale alfabetului latin;
- 26 de litere minuscule corespunzătoare;
- 10 simboluri numerice: **0 ÷ 9**;
- 20 de simboluri speciale adiționale: +, -, (,), [,], {, }, *, #, \$ etc.

Codificare cu 7 biți; **msb** : "*bit de paritate*". Convenția folosită este următoarea:

msb = 0 dacă codul are un număr *par* de cifre binare **1**;

msb = 1 dacă codul are un număr *impar* de cifre binare **1**;

De exemplu: **A = 01000001**;
B = 01000010;
C = 11000011 etc.

Bitul de paritate → fanion dedicat (**P**).

Reprezentarea externă

Reprezentarea externă se referă la modul în care informația prelucrată de un microcalculator apare utilizatorului (programatorului).

a) Pentru **codurile instrucțiunilor** se vor folosi abrevierile sugestive pe care, de regulă, fabricantul le impune și pe care limbajul de asamblare le folosește ca atare: "*mnemonice*".

b) Pentru **numere** se utilizează mai multe tipuri de reprezentări:

- Reprezentarea *binară*, imagine fidelă a conținutului locațiilor de stocare a informațiilor.
- Reprezentarea *octală*, care transformă numerele binare în baza de numerație 8.
- Reprezentarea *hexazecimală* : un simbol reprezentând o cifră în baza de numerație 16 înlocuiește 4 cifre binare. Caracterele folosite sunt cifrele zecimale **0 ÷ 9** și literele **A ÷ F**. Vom folosi convenția de a utiliza litera **H** ca sufix pentru numerele reprezentate în hexazecimal (de pildă **1199H**).

c) Pentru **caractere** se vor folosi chiar simbolurile cu care ele sunt individualizate. Programele utilitare folosite pentru examinarea conținutului locațiilor de stocare a informațiilor fac conversia **ASCII** → **simbol caracter** dacă programatorul stabilește că semnificația informației vizate impune această conversie.

Convenții pentru notații

1. Neterminali:

r	un registru oarecare;
r8	un registru de 8 biți;
r16	un registru de 16 biți;
r_i , r_j	registre individualizate, diferite;
mem	o locație de memorie oarecare (sau mai multe locații succesive);
mem8	o locație de memorie de un octet;
mem16	o locație de memorie de 16 biți (pot fi două locații succesive dacă formatul este octetul);
mem32	o locație de memorie de 32 de biți (pot fi patru locații succesive dacă formatul este octetul);
mem_i	o locație de memorie individualizată (în scopul de a o deosebi de alte locații de memorie);
adr	o adresă oarecare;
adr8	o adresă pe 8 biți;
adr16	o adresă pe 16 biți;
adr24	o adresă pe 24 de biți;
adr_i	o adresă individualizată (în scopul de a o deosebi de alte adrese);
(r)	conținutul unui registru oarecare;

(ri, rj)	conținutul a două registre concatenate;
(r)l	conținutul jumătății inferioare (mai puțin semnificativă) a unui registru;
(r)h	conținutul jumătății superioare (mai semnificativă) a unui registru;
((r))	conținutul unei locații de memorie a cărei adresă se află într-un registru (adresare indirectă);
(mem)	conținutul unei locații de memorie oarecare;
adr_l	jumătatea inferioară a unei adrese;
adr_h	jumătatea superioară a unei adrese;
data	un operand oarecare;
data8	un operand de 8 biți;
data16	un operand de 16 biți;
disp8	un deplasament pe 8 biți;
disp16	un deplasament pe 16 biți;
port	un port de intrare/ieșire oarecare;

2. Terminali:

R1, R2, A, AX, BP, A6, Dn, An	nume de registre;
(R1)	conținutul registrului R1 ;
(R1, R2)	conținutul perechii de registre R1 și R2 ;
((R1))	conținutul locației de memorie a cărei adresă se află în registrul R1 ;
MEM, MEM1	nume de locații de memorie;
ADR, ADR_n	nume de adrese;

3. Operatori:

←	atribuire;
↑	concatenare;
not	complementare (negație);
∇	operația logică SAU;
&	operația logică }I;
⊕	operația logică SAU EXCLUSIV;
+	adunare;
-	scădere;
*	înmulțire;
DIV	împărțire între numere întregi;
MOD	restul împărțirii între numere întregi;

4. Alte simboluri:

[]	încadrează elemente de sintaxă opționale;
 	delimitează elemente de sintaxă alternative.