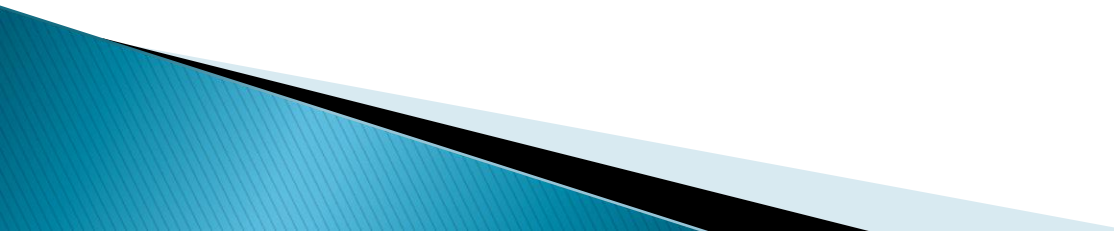


# Modalități de reprezentare a calculatoarelor

– *Curs7* –

## Subiecte abordate:

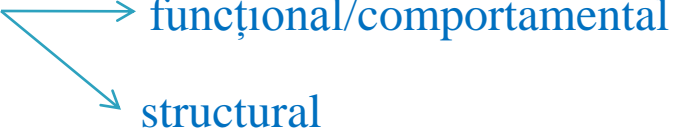
---

- Reprezentarea funcțional/comportamentală
  - Arhitectura calculatorului numeric
  - Reprezentarea structurală a unui calculator
  - Instrucțiunile calculatorului
- 

## REPREZENTAREA FUNCȚIONAL/COMPORTAMENTALĂ

---

Calculatoarele reprezintă sisteme complexe care pot fi examinate la diverse niveluri de abstractizare, în funcție de scopul urmărit.

- Un calculator poate fi examinat sub aspect: 
  - funcțional/comportamental
  - structural

Din punct de vedere funcțional/comportamental, un calculator se poate reprezenta prin tripletul  $\langle \mathbf{I}, \mathbf{E}, \mathbf{C} \rangle$  unde:

- $\mathbf{I}$  constituie mulțimea intrărilor,
- $\mathbf{E}$  corespunde mulțimii ieșirilor,
- $\mathbf{C} \subset \mathbf{I} \times \mathbf{E}$  reprezintă o submulțime a produselor carteziene între elementele mulțimii  $\mathbf{I}$  și elementele mulțimii  $\mathbf{E}$ .  $\mathbf{C}$  realizează aplicații din mulțimea intrărilor  $\mathbf{I}$ , în mulțimea ieșirilor  $\mathbf{E}$ .

Reprezentarea sub forma unei *ierarhii de niveluri imbricate*.



Un *nivel* este constituit din *mulțimea aplicațiilor asupra elementelor mulțimii de intrare* pentru nivelul dat, cât și asupra elementelor *mulțimilor de intrare și ieșire de la nivelul inferior*, imbricat. Aplicațiile de la un nivel dat pot constitui aplicații și pentru nivelul superior următor.

Nivelul aplicațiilor/funțiilor primitive;  
mașina de bază (realizată în hardware)  
capabilă să execute operații elementare.

Nivelul funcțiilor standard/predefinite;  
nivelul instrucțiunilor mașinii convenționale

Nivelul funcțiilor construite de utilizator, pe baza funcțiilor  
de la nivelul inferior. Acesta este nivelul aplicativ al calculatorului.

La nivelul mașinii convenționale se definește noțiunea de arhitectură a unui calculator numeric ( procesor) prin quadruplul  $A = \langle \mathbf{PI}, \mathbf{PE}, \mathbf{RG}, \mathbf{I} \rangle$  unde:

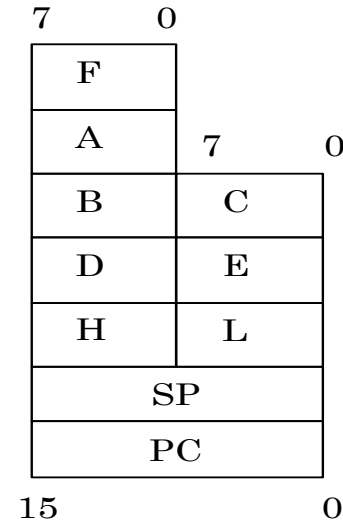
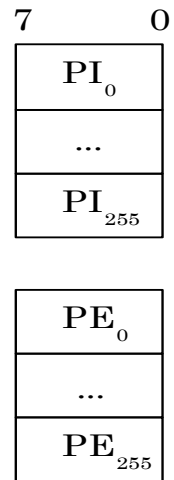
$\mathbf{PI}$  =  $\{\mathbf{PI}_0, \dots, \mathbf{PI}_i\}$  este mulțimea porturilor de intrare,

$\mathbf{PE}$  =  $\{\mathbf{PE}_0, \dots, \mathbf{PE}_j\}$  este mulțimea porturilor de ieșire,

$\mathbf{RG}$  =  $\{\mathbf{RG}_0, \dots, \mathbf{RG}_k\}$  este ansamblul registrelor generale din unitatea de execuție

$\mathbf{I}$  =  $\{\mathbf{I}_0, \dots, \mathbf{I}_l\}$  este setul instrucțiunilor calculatorului.

## Arhitectura Microprocesorului Intel 8080



### Registre cu caracter specializat:

- **A** este registrul acumulator principal, folosit în cele mai multe operații aritmetice și logice;
- **B, C, D, E, H, L** sunt registre de uz general, deși unele au și utilizări speciale; **H** și **L** sunt folosite adesea pentru formarea unor adrese de operanzi pe 16 biți;



- **F** este registrul indicatorilor de condiții, furnizați de către unitatea de execuție după fiecare operație (**CY** - transport în afara rangului de semn, **AC** – transport auxiliar între tetradele octetului rezultat, **S** - semnul rezultatului, **Z** - indicator de rezultat zero, **P** - indicator privind paritatea numărului de unități din rezultat).

7	6	5	4	3	2	1	0
S	Z	*	AC	*	P	*	CY

- **SP** este indicatorul de stivă, care asigură accesul la o structură de tip stivă organizată în memorie;
- **PC** contor program, pentru adresarea instrucțiunilor din memorie. Lista de instrucțiuni a microprocesorului 8080 conține 78 de instrucțiuni.

## Arhitectura microprocesorului 8086

	15	8	7	0	
AX:	AH		AL		Acumulator
BX:	BH		BL		Registru – bază
CX:	CH		CL		Registru – contor
DX:	DH		DL		Registru – date
	SP				Indicator - stivă
	BP				Indictor - bază
	SI				Indicator - sursă
	DI				Indicator - destinație

Indicatorii de condiții

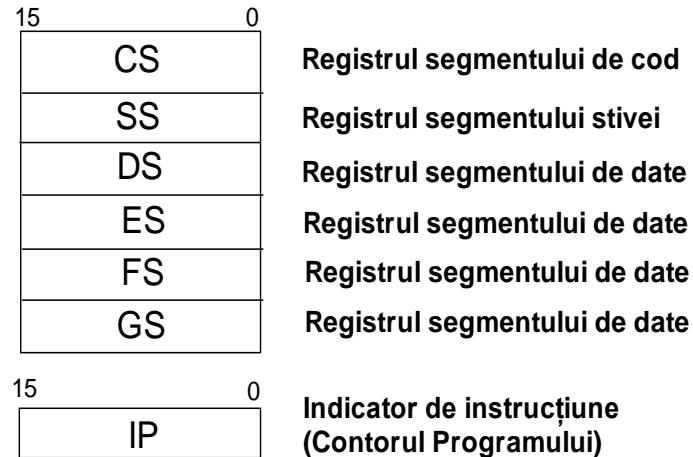
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*	*	*	*	O	D	I	T	S	Z	*	A	*	P	*	C

Semnificațiile indicatorilor de condiții sunt :

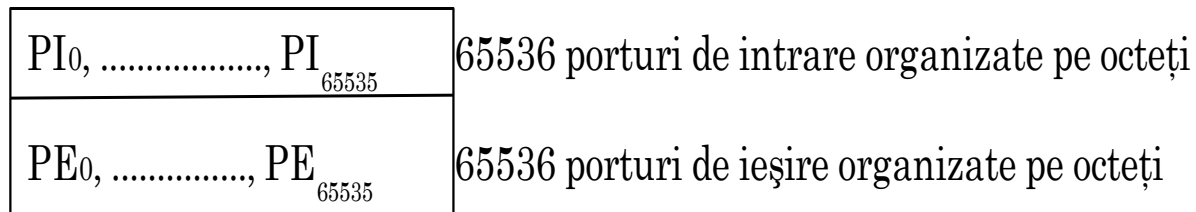
- **O** - depășire aritmetică;
- **D** - direcția la explorarea șirurilor;
- **I** - activare/dezactivare întreruperi;
- **T** - capcană pentru lucrul pas cu pas;
- **S** – semn;
- **Z** – zero;
- **A** - transport auxiliar;
- **P** – paritate;
- **C** - transport în afara rangului de semn.



## Arhitectura microprocesorului 8086: registrele segmentelor și indicatorul instrucțiunii.

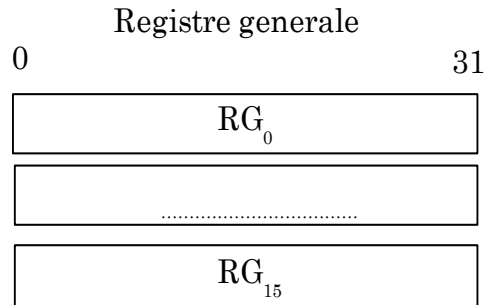


Microprocesorul 8086 mai posedă câte două tablouri de porturi de intrare/ieșire, a câte 65536 octeți fiecare :

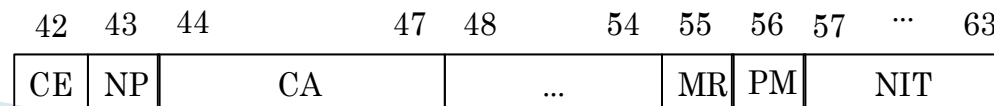
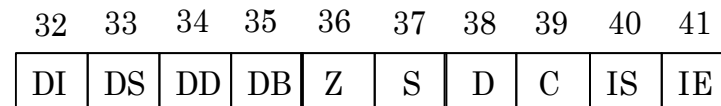
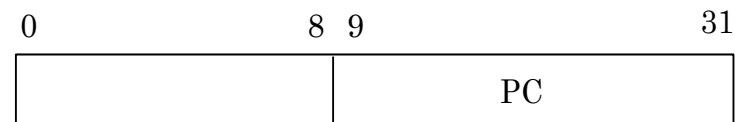


## Unitatea centrală a calculatorului FELIX 5000

- 16 registre generale  $RG_0, \dots, RG_{15}$ , de câte 32 de biți,;
- un cuvânt dublu de stare program (*PSW Program Status Word*), care stochează atât contorul programului, cât și o serie de indicatori de condiții.



Cuvântul de stare program (PSW)



## Câmpurile PSW

**PC** - contor program;

### **Măști de depășire:**

**DI, DS** - depășire inferioară/superioară în virgulă mobilă:  $-64 < \mathbf{E} < 64$ , ( $\mathbf{E}$  = exponent );

**DD** - depășire zecimală;

**DB** - depășire binară:  $-1 < \mathbf{n} < 1$ , ( $\mathbf{n}$  = mantisa ).

Dacă masca este unu atunci derutarea este interzisă.

### **Indicatorii de condiții ai rezultatului:**

**Z** = 1 - rezultat nul;

**S** = 1 - rezultat  $< 0$ ;

**D** = 1 - depășire;

**C** = 1 - transport.

### **Măști de întrerupere:**

**IS** - mască întreruperi de I/E;

**IE** - mască întreruperi externe;

**CE** - mască întreruperi contor nul.

Dacă masca este unu, întreruperea este inhibată și rămâne în așteptare.



### **Nivelul programului:**

**NP** = 0 - unitatea centrală operează în modul privilegiat (poate executa toate instrucțiunile);

**NP** = 1 - unitate centrală operează în modul normal (încercarea de a executa instrucțiunile de sistem generează o derutare).

### **Cheia de acces:**

**CA** = cheia de acces la paginile de memorie, de câte 2Ko, protejate prin chei de protecție.

### **Masca de rotunjire:**

**MR** = 0 - se rotunjește rezultatul operației în virgulă mobilă;

**MR** = 1 - rezultatul nu se rotunjește.

### **Paritate memorie:**

**PM** = 0 - eroarea de paritate la memorie este tratată prin derutare;

**PM** = 1 - eroarea se tratează prin întrerupere.

### **Nivelul de întrerupere asociat programului:**

**NIT**- definește nivelul de prioritate al programului în curs de execuție (pot exista cel mult 128 niveluri de prioritate).

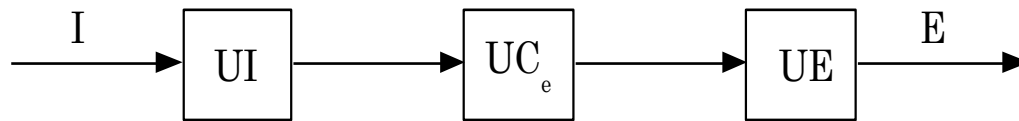
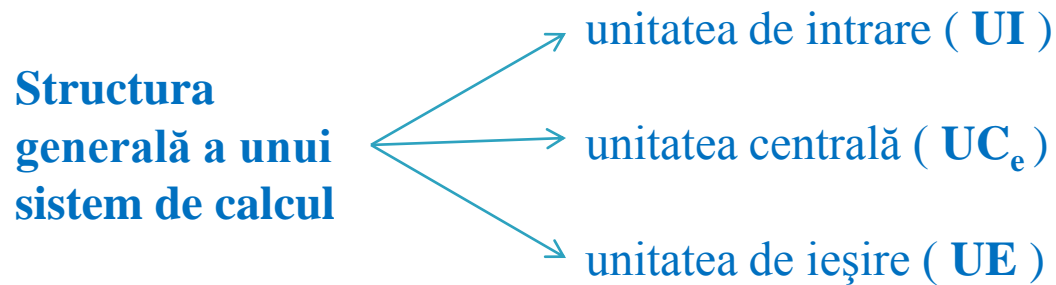


**Calculatorul FELIX 5000** are implementate 102 instrucțiuni, din 128 instrucțiuni posibile.

## REPREZENTAREA STRUCTURALĂ A UNUI CALCULATOR

În acest caz se pleacă de la ideea că un calculator reprezintă un *sistem*, constituit din *componente primitive* (primitive funcționale) *interconectate* într-o manieră dată, pentru a putea executa operații specifice, de prelucrare a informației.

*Componentele primitive se caracterizează prin*  *debit de transfer*  
*capacitate de stocare a informațiilor.*



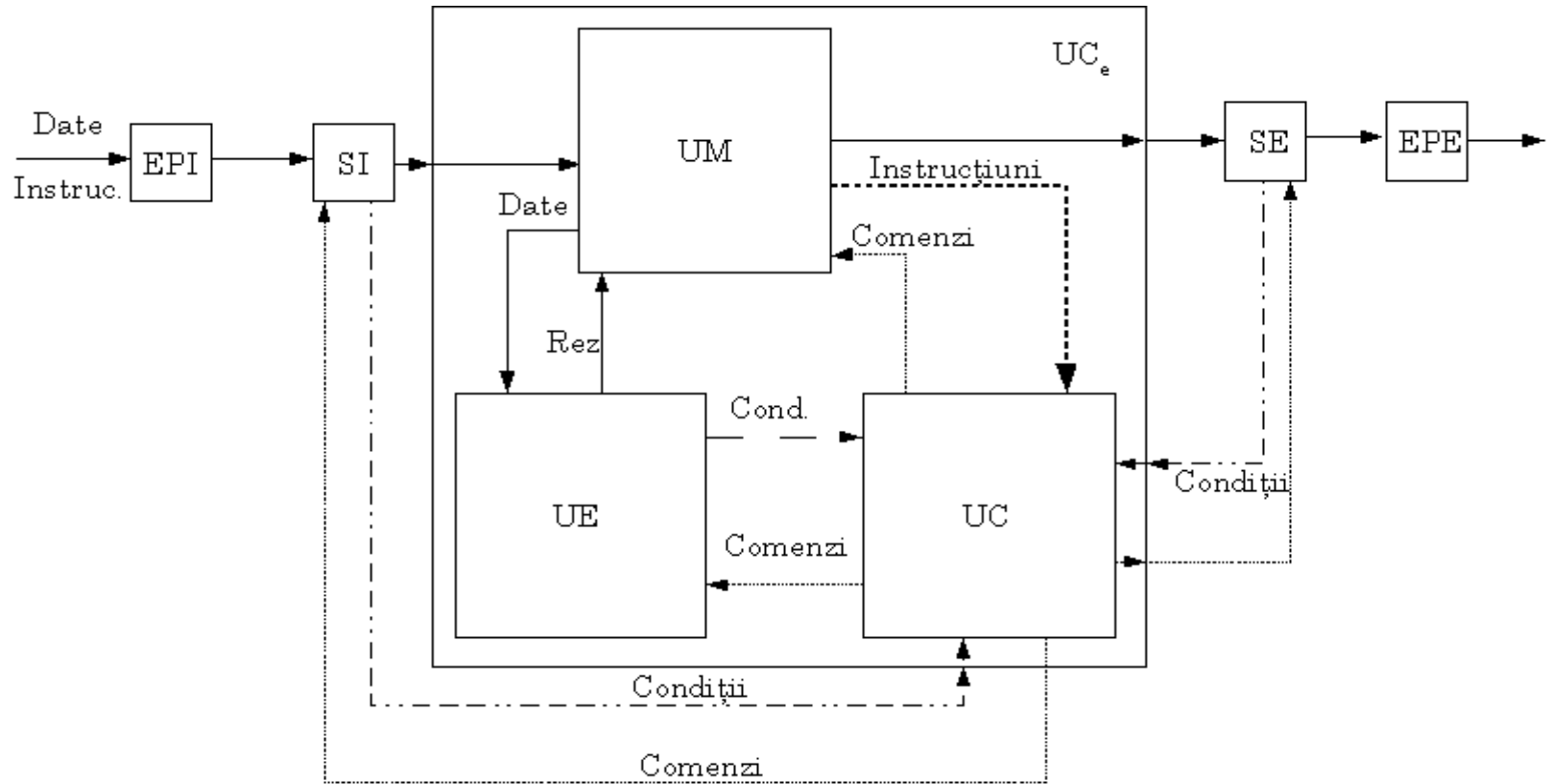
❖ *Unitățile de intrare și de ieșire* asigură legătura sistemului cu echipamentele periferice primare (traductoarele/elementele de execuție), care preiau informația din mediul extern și o furnizează în sistem sau care execută diferite acțiuni asupra mediului extern, ca urmare a interpretării informației prelucrate de calculator.

❖ *Unitatea centrală* asigură stocarea programului, a datelor și realizează prelucrarea automată a acestora pe baza interpretării programului dat.

**Rafinarea structurii  
sistemului de calcul în  
zonele UI și UE**

- subsistemul de intrare (**SI**)
- subsistemul de ieșire (**SE**)
- echipamentele periferice de intrare (**EPI**)
- echipamentele periferice de ieșire (**EPE**).

# Nivelul UCe



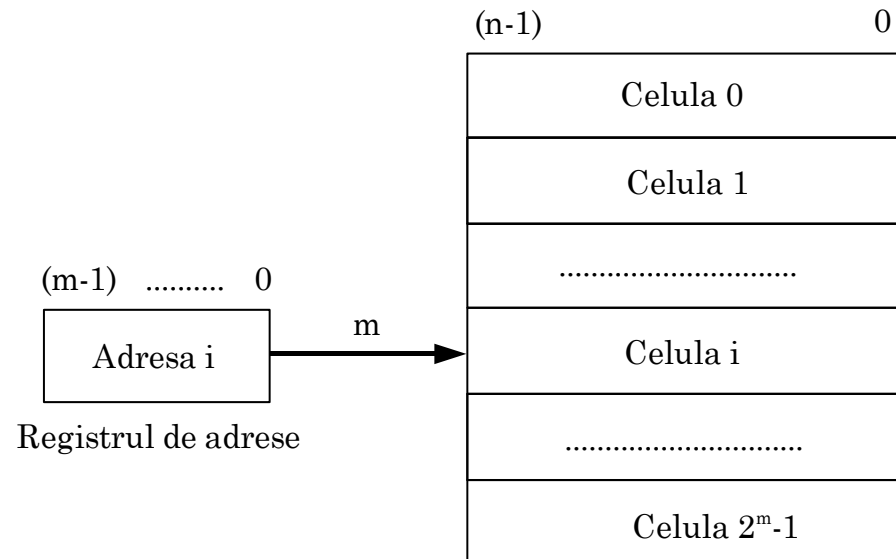
- date/adrese
- - - - - instrucțiuni
- ..... comenzi
- · - · - condiții/indicatori de stare/stări

**Unități  
funcționale**

- unitatea de memorie (UM)
- unitatea de execuție (UE)
- unitatea de comandă (UC)

## Unitatea de memorie

- are funcția de stocare a datelor inițiale, a programului, a rezultatelor intermediare și finale;
- în sistemele moderne ea poate opera autonom, atât cu **SI/SE**, cât și cu procesorul (ansamblul unitate de comandă - unitate de execuție );
- are o organizare liniară, constând în celule de stocare a informației, al căror conținut poate fi manipulat prin specificarea adresei celulei date.
- adresa ia valori cuprinse între **0** și  $2^m - 1$ , unde **m** este numărul de ranguri binare ale registrului de adrese.





## Unitatea de execuție

- ★ • asigură, sub controlul *unității de comandă*, o succesiune de operații aritmetice și logice asupra datelor preluate din unitatea de memorie sau din memoria locală-propră (implementată sub forma unor registre generale -RG-), rezultatele fiind returnate în unitatea de memorie sau în registrele generale.
- ★ • după fiecare operație **UE** actualizează starea unor *indicatori de condiții*, care reflectă caracteristicile rezultatului curent ( $< 0$ ,  $> 0$ ,  $= 0$ , paritate, transport, depășire etc.).

## Unitatea de comandă

- ★ • prelucrează fluxul de instrucțiuni, care constituie programul.
- ★ • furnizează semnale de comandă pentru celelalte unități, coordonând funcționarea lor în conformitate cu cerințele programului.

## La nivel PMS:

**P** - procesorul

**M** - memoria

**S** - comutatorul

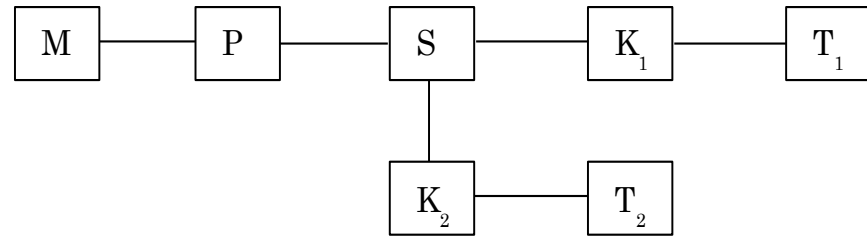
**D** - operatorul asupra datelor

**K** - operatorul de comandă

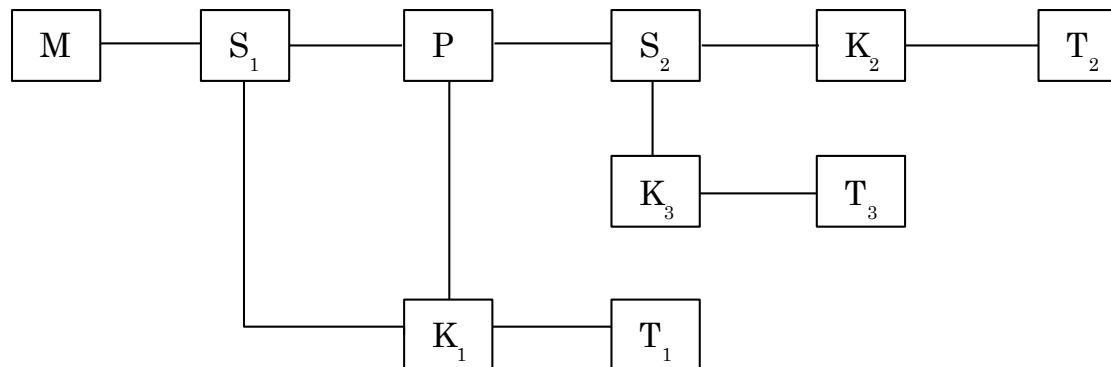
**L** - legătura

**T** - terminalul/traductorul

Sistem de calcul în care terminalele (echipamentele de I/E) transferă datele cu memoria prin intermediul procesorului.



Sistem de calcul în care unele terminale (echipamentele de I/E) transferă date cu memoria direct (T<sub>1</sub>) sau prin intermediul procesorului (T<sub>2</sub>, T<sub>3</sub>).



## INSTRUCȚIUNILE CALCULATORULUI

---

★ reprezintă o colecție de informații privind operațiile care se pot efectua într-un calculator dat.

**Categorii:**

- instrucțiuni *operaționale și de transfer* al informațiilor, inclusiv instrucțiunile de I/E;
- instrucțiuni *cu caracter de decizie*, care modifică secvența de execuție a programului în mod condiționat, de o serie de indicatori, sau în mod necondiționat.

**Câmpuri:**

- *codul de operație/funcția*, care specifică operația/acțiunea elementară evocată de instrucțiune: aritmetică, logică, transfer de date, transfer al comenzii etc.;
- *adresa operandului/instrucțiunii*, care specifică locația de memorie cu care se face transferul de date sau de la care se citește o instrucțiune.

## Formate simple de instrucțiuni:

Instrucțiunea cu o adresă

COP	ADRESA
-----	--------

Instrucțiunea cu două adrese

COP	ADRESA 1	ADRESA 2
-----	----------	----------

Instrucțiunea cu trei adrese

COP	ADRESA 1	ADRESA 2	ADRESA 3
-----	----------	----------	----------

Câmpul de adresă din instrucțiune poate avea mai multe semnificații:



- conține chiar operandul, în cazul *operandului imediat*;
- reprezintă adresa unui operand din memorie, în situația *adresării directe*;
- reprezintă adresa unei celule de memorie în care se află adresa unui operand, în cazul *adresării indirecte*.

În unele situații, la conținutul câmpului de adresă din instrucțiune se adună conținuturile unor *registre speciale*:

→ *registrele bază*, la *adresarea bazată*

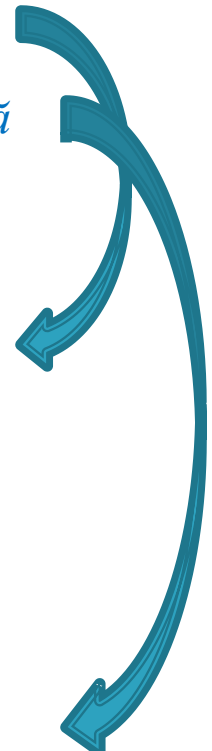
→ *registrele index*, la *adresarea indexată*

regitrul bază conține adresa de start (baza) a structurii, iar adresa din instrucțiune reprezintă o deplasare în structura dată;

regitrul index asigură adunarea unei cantități variabile ( incrementabile/ decrementabile - eventual automat )



*adresa efectivă*



Instrucțiunile, care afectează execuția programului, trebuie să specifice:

- ★ în câmpul codului de operație condițiile testate,
- ★ în câmpul de adresă deplasarea necesară calculului adresei efective.



*adresa efectivă se poate obține direct, indirect, bazat, indexat sau ca urmare a unor operații combinate*

**Formatul  
instrucțiunii  
calculatorului  
FELIX 5000**

0	1	34	7	8	9	15	16	31
I	B	Q	X	F	D			

- **I** poate lua valoarea 0 în cazul adresării directe și 1, în cazul adresării indirecte;
- **B** ia valori între 0 și 7, specificând pentru valori mai mari decât 0, adresarea bazată cu unul din registrele generale **RG<sub>9</sub> - RG<sub>15</sub>**, iar pentru 0 adresarea nebazată;
- **Q** specifică: registrul general, care conține unul din operanzi, o constanta sau un vector logic sau adresa de origine și lungimea unui operand, în cazul manipulării șirurilor, și registrul general **RG<sub>0</sub> - RG<sub>15</sub>** folosit la indexare;
- **X** ia valoarea 0, în cazul adresării neindexate și valoarea 1, în cazul adresării indexate;
- **D** specifică deplasarea, folosită pentru calculul adresei efective, prin adunarea cu registrul bază.

## Modurile de adresare întâlnite la FELIX 5000

---

- *Adresare directă:*  $\mathbf{I} = \mathbf{0}; \mathbf{X} = \mathbf{0}; \mathbf{B} \neq \mathbf{0}.$

$\mathbf{A}_{\text{ef}} = (\mathbf{B} + \mathbf{8}) + \mathbf{D}$ ; unde  $(\mathbf{B} + \mathbf{8})$  specifică registrul general utilizat ca bază.

- *Adresarea indirectă:*  $\mathbf{I} = \mathbf{1}; \mathbf{X} = \mathbf{0}; \mathbf{B} \neq \mathbf{0}.$

Prima adresă efectivă,  $\mathbf{A}_{\text{ef}}$ , calculată va reprezenta adresa unui cuvânt din memorie în care se găsește o informație, care este tratată, din punctul de vedere al calculului unei noi adrese efective, ca o instrucțiune. Astfel, la adresa efectivă:

$\mathbf{A}_{\text{ef1}} = (\mathbf{B} + \mathbf{8}) + \mathbf{D}$  se va găsi un cuvânt care va avea câmpurile:  $\mathbf{I}_1, \mathbf{B}_1, \mathbf{Q}_1, \mathbf{X}_1, \mathbf{F}_1, \mathbf{D}_1$ . Dacă  $\mathbf{I}_1 = \mathbf{1}$ , atunci se va calcula noua adresă efectivă:

$\mathbf{A}_{\text{ef2}} = (\mathbf{B}_1 + \mathbf{8}) + \mathbf{D}_1$ , de la care se va citi un nou cuvânt, procesul fiind continuat pe maximum 5 niveluri de adresare indirectă, până când  $\mathbf{I}_i = \mathbf{0}$ .

- *Adresarea indexată directă:*  $\mathbf{I} = \mathbf{0}; \mathbf{X} = \mathbf{1}.$

$\mathbf{A}_{\text{ef}} = (\mathbf{Q}) + (\mathbf{B} + \mathbf{8}) + \mathbf{D}$ , unde  $(\mathbf{Q})$  specifică **RG** folosit ca registru index.

- *Adresarea indexată indirectă:*  $\mathbf{I} = \mathbf{1}; \mathbf{X} = \mathbf{1}.$

$\mathbf{A}_{\text{ef1}} = (\mathbf{B} + \mathbf{8}) + \mathbf{D},$

$\mathbf{A}_{\text{ef}(k+1)} = (\mathbf{B}_k + \mathbf{8}) + \mathbf{D}_k,$

$\mathbf{A}_{\text{ef}} = (\mathbf{Q}) + \mathbf{A}_{\text{ef}(k+1)} = (\mathbf{Q}) + (\mathbf{B}_k + \mathbf{8}) + \mathbf{D}_k.$