UNIVERSITATEA TEHNICĂ A MOLDOVEI

ARHITECTURA CALCULATORELOR

Deprtamentul:
Informatică și Ingineria Sistemelor

conf. univ., dr. Victor ABABII

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Tematica disciplinei Arhitectura Calculatoarelor:

Tema 1. Introducere. Bazele fundamentale ale Arhitecturii Calculatoarelor;

Tema 2. Microprocesoare și Microcontrolere. Limbajul de programare Assembler;

Tema 3. Dispozitive pentru achiziția datelor;

Tema 4. Dispozitive pentru afișarea și imprimarea datelor;

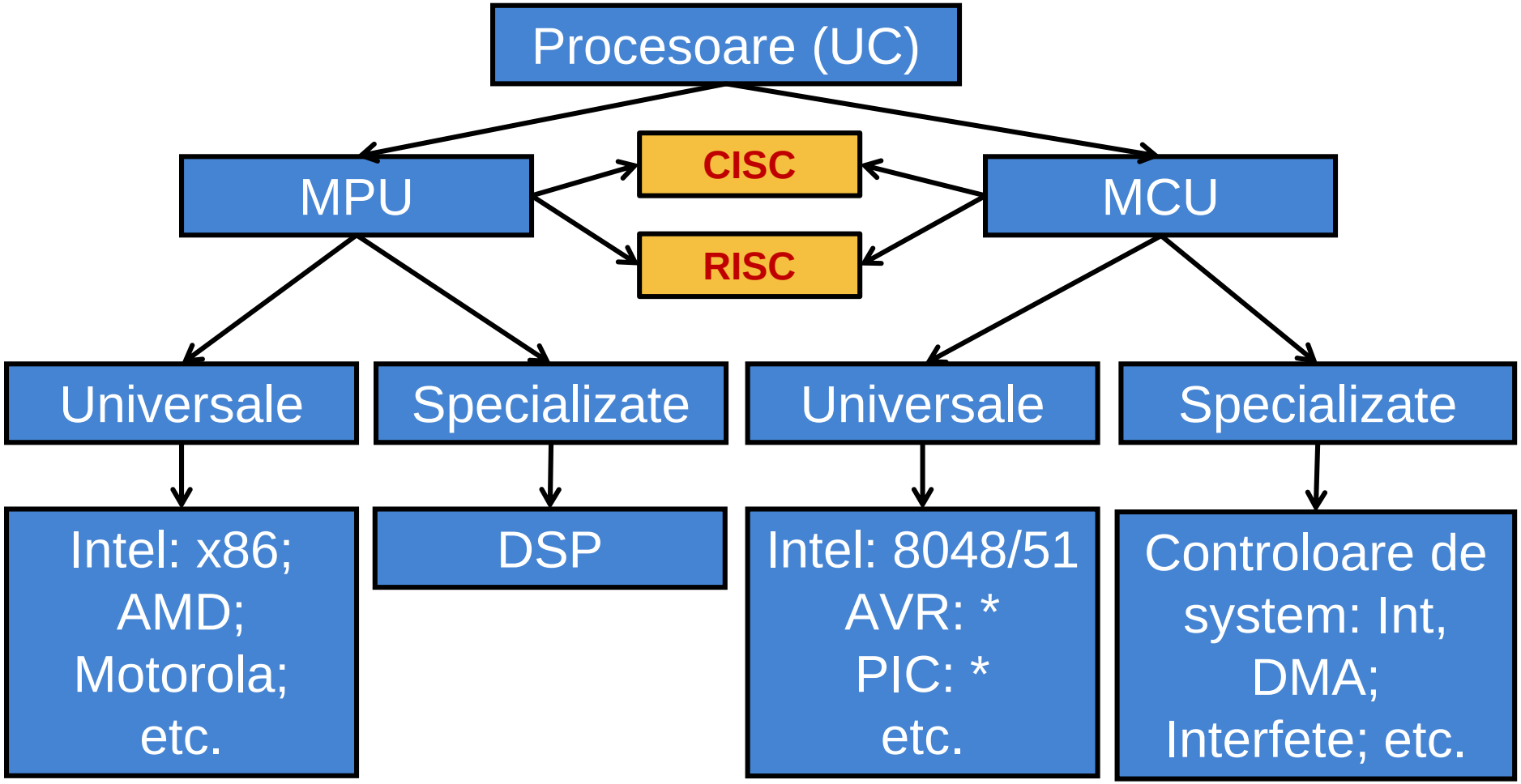
Tema 5. Dispozitive pentru stocarea datelor.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Tema 2. Microprocesoare și Microcontrolere. Limbajul de programare Assembler :

1. Clasificarea MPU și MCU.
2. MPU cu arhitectura x86.
3. MCU Intel, AVR, MicroChip.
4. Limbajul de programare Assembler. Setul de instrucțiuni x86.
5. Medii pentru dezvoltarea, compilarea și testarea produselor program.
6. Memoria internă a sistemelor de calcul.
7. Magistrala de sistem. Ciclul magistralei de sistem.
8. Proiectarea memoriei.
9. Controloare specializate: Int, DMA, Timet.

2.1. Clasificarea MPU și MCU.



UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.1. Clasificarea MPU și MCU.

Arhitectura și Setul de instrucțiuni: CISC, RISC;
Producator – tehnologie: TTL, I2L, MOS, n-MOS, P-MOS, CMOS;
Magistrala de Adresă: 16, 20, 24, 32, 64, 128;
Magistrala de Date: 8, 16, 32, 64, 128;
Magistrala A/D: multiplexată, non-multiplexată;
Magistrala de control: semnale fizice, comenzi de magistrală;
Magistrala de stare: semnale fizice, cuvinte de stare;
Memorie cache: Date / Program – depinde de Arhitectură;
Frecventa de ceas: 10 MHz – 3,8GHz;
Tensiunea de alimentare: Nucleu -> Periferie (1,5 – 12 V);
Putere consumata: < 120W
Tehnologia de producere, nm: 180, 130, 90, 65, 45, 32, 22, 14, 10.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.1. Clasificarea MPU și MCU.

CISC:

Procesor cu setul de instrucțiuni complex.

Obiectivul principal al proiectării CISC este de a reduce numărul de instrucțiuni dintr-un program. Prin urmare, aceeași sarcină poate fi obținută cu un număr mai mic de instrucțiuni utilizând CISC.

RISC:

Procesor cu setul de instrucțiuni redus.

Este proiectat pentru a reduce timpul de execuție prin simplificarea setului de instrucțiuni. Utilizează instrucțiuni extrem de optimizate.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

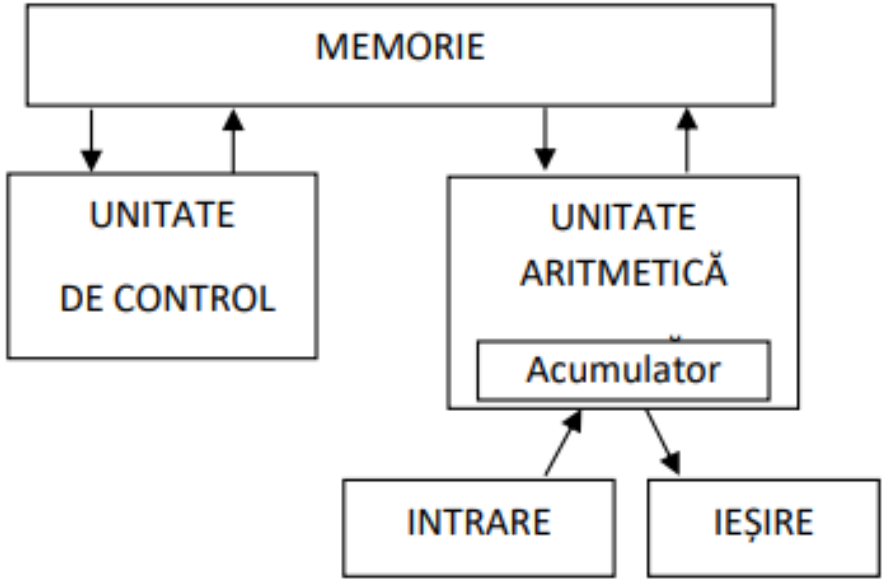
2.1. Clasificarea MPU și MCU.

Parametrul de comparație	RISC	CISC
Accent pe	SoftWare	HardWare
Numărul de instrucțiuni	Mic	Mare
Formatul instrucțiunilor	Fix (32 biți)	Variabil (16 – 64 biți)
Registre de uz general	32 - 192	8 – 24
Memorie Cache	Divizată: Data & Instructions	Unificată: Data & Instructions
Cicluri pe instrucțiune	Unic pentru toate instrucțiunile	Multiplu: 2 - 15

UNIVERSITATEA TEHNICĂ A MOLDOVEI

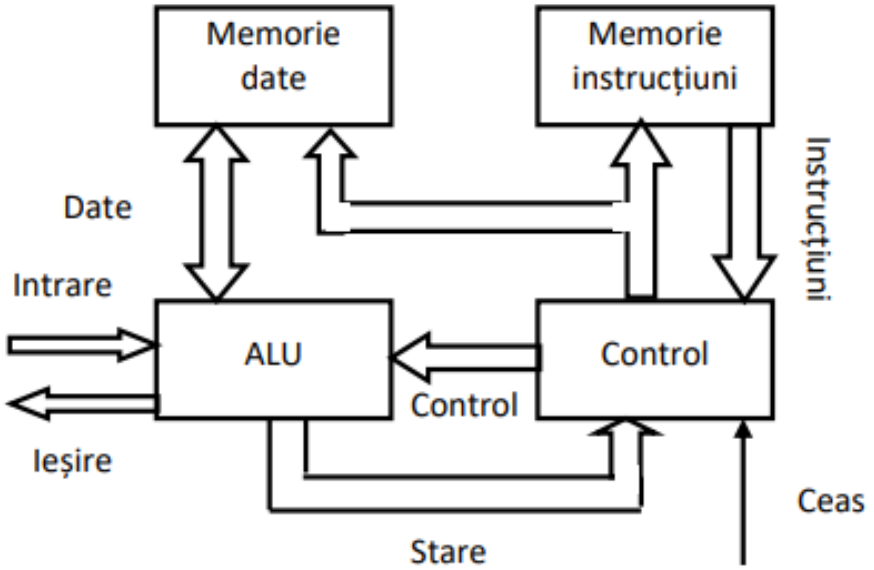
2.2. Arhitecturi clasice ale MPU/MCU.

Arhitectura von Neumann



Avantaje?

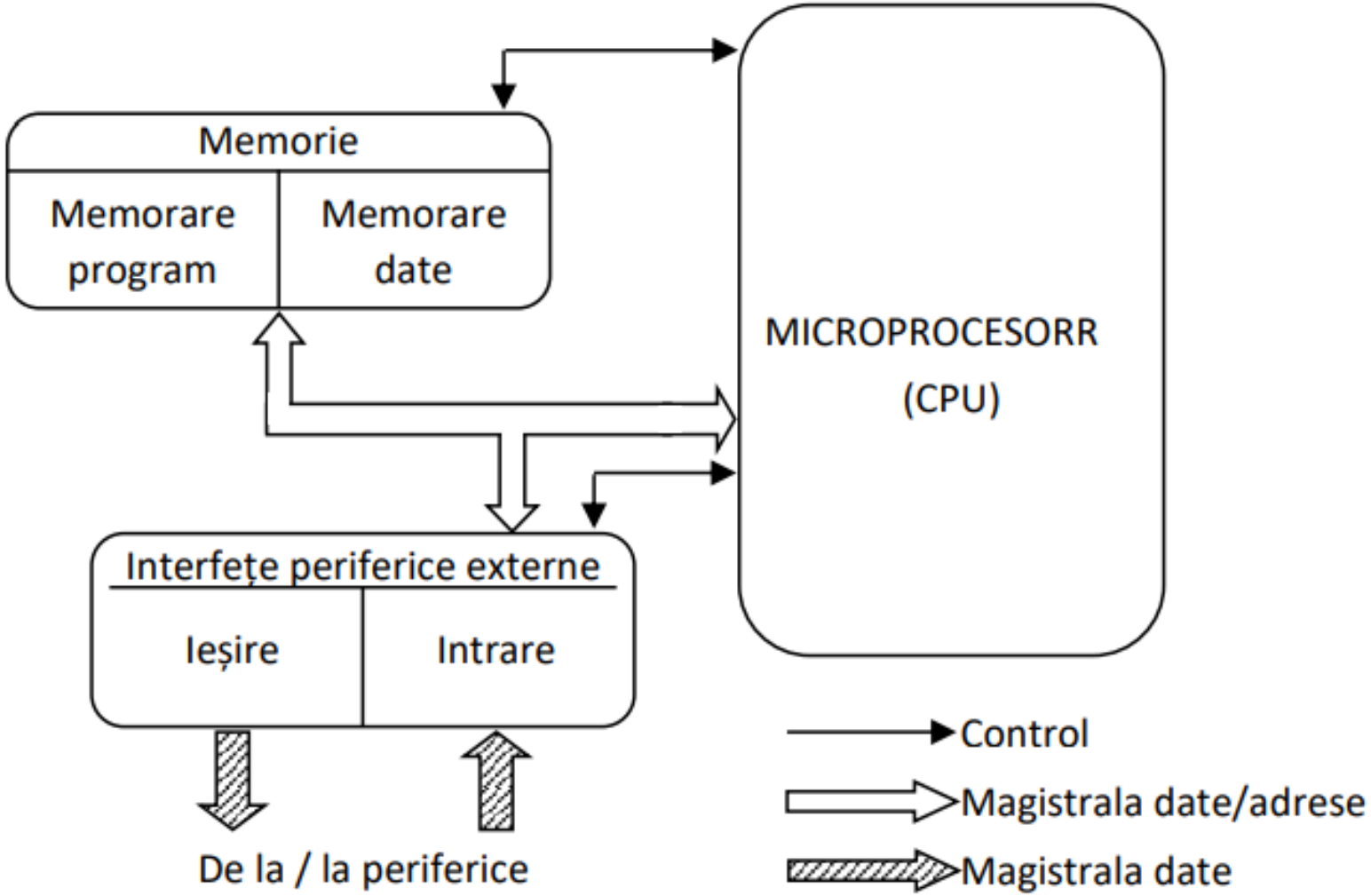
Arhitectura Harvard



Dezavantaje?

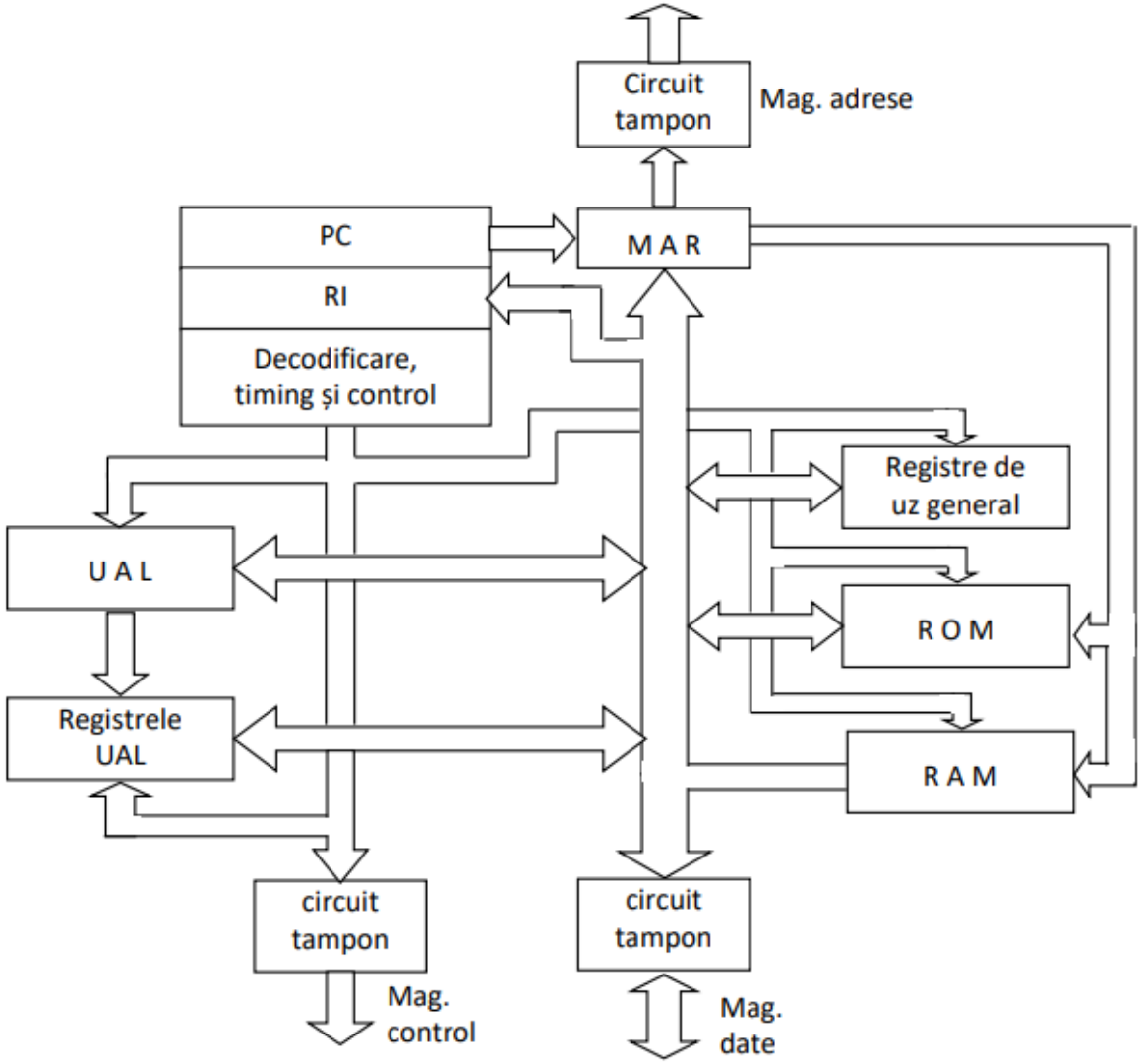
UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.2. Interconectarea MPU cu perifericele.



UNIVERSITATEA TEHNICĂ A MOLDOVEI

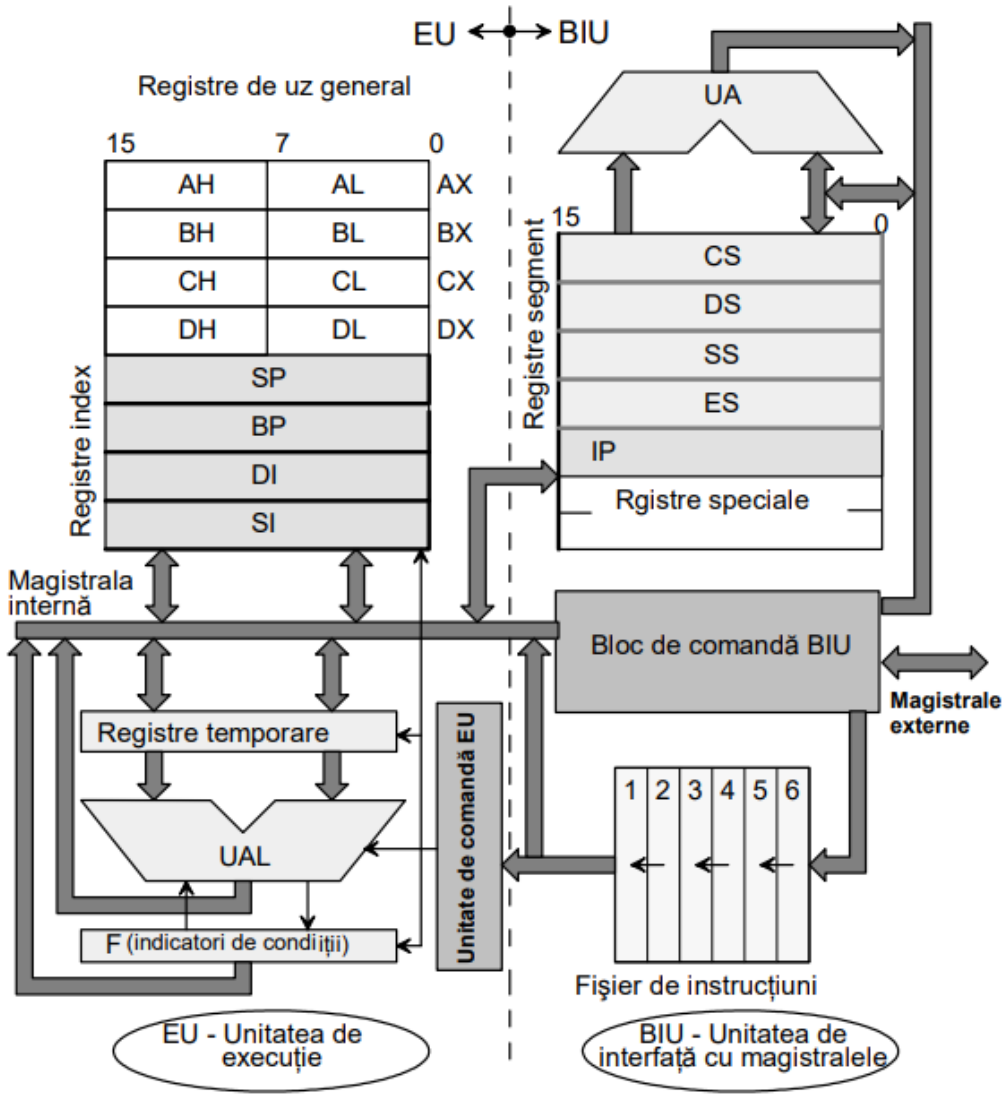
2.2. Structura classică a MPU.



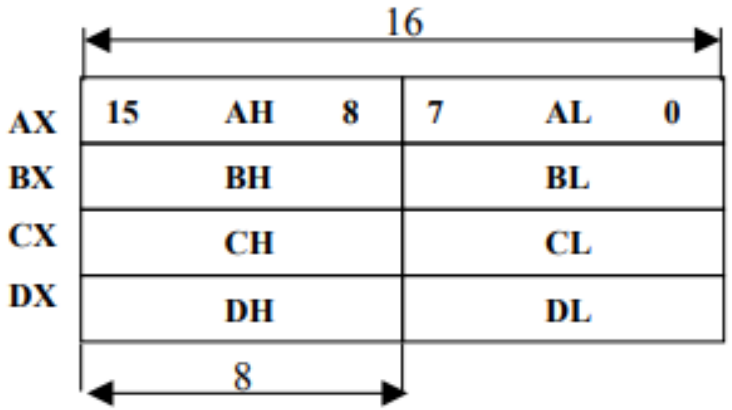
PC = IP – Program counter;
RI – Rg Instructiuni;
MAR – Rg Ardesa RAM;
UAL – Unitate logica – aritmetica.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.2. MPU cu arhitectura x86.



REGISTRU	OPERAȚII
AX	Înmulțiri, împărțiri și I/E pe cuvânt
AL	Înmulțiri, împărțiri și I/E pe octet, translatări, aritmetică zecimală
AH	Înmulțiri, împărțiri pe octet
BX	Translatări
CX	Operații cu șiruri de caractere, contor pentru operații repetate
CL	Deplasări și rotiri cu mai mult de o poziție
DX	Înmulțiri și împărțiri pe cuvânt, I/E cu adresare indirectă
SP	Operații cu stiva
SI, DI	Operații cu șiruri de caractere

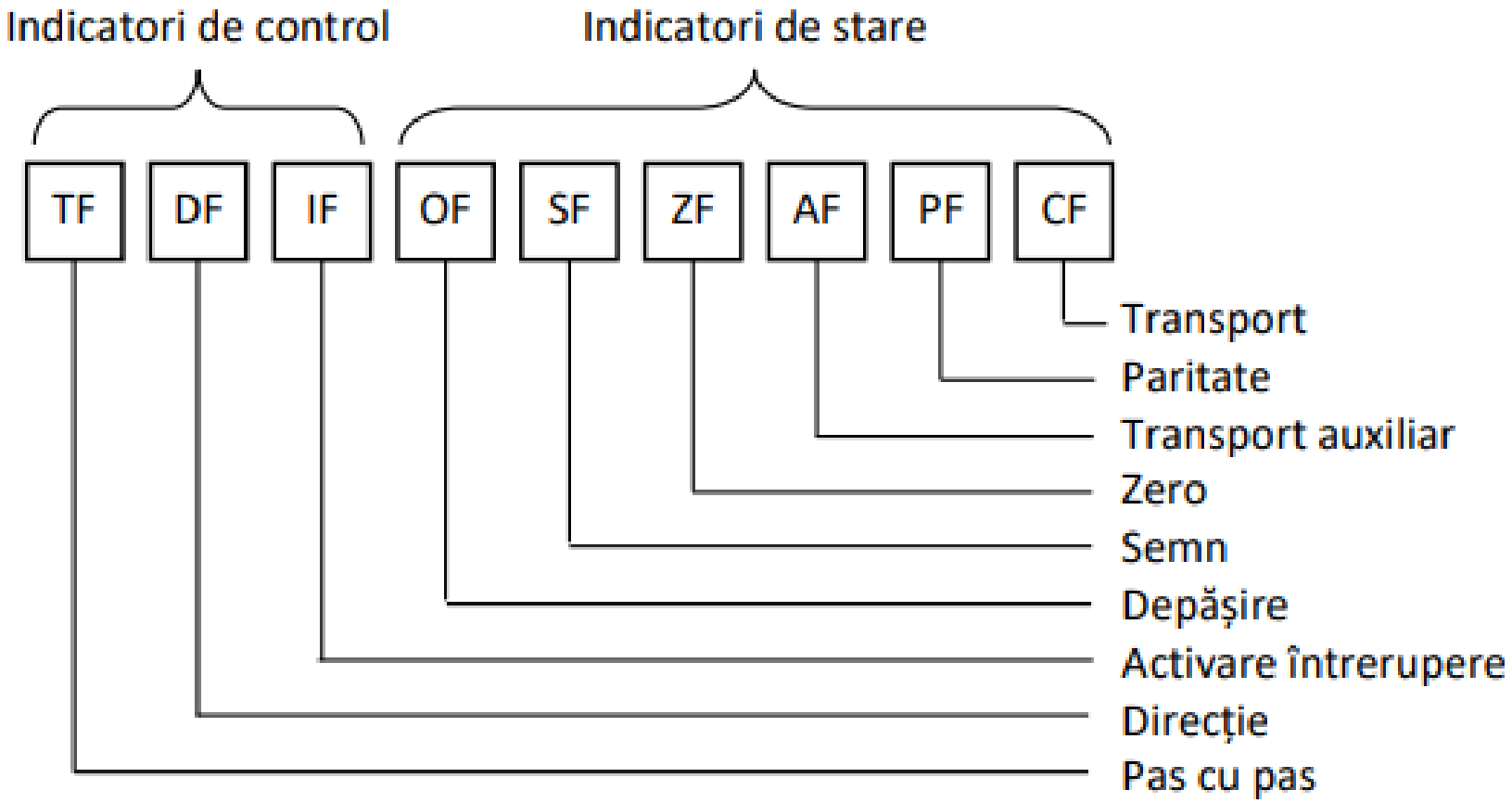


2.2. MPU cu arhitectura x86.

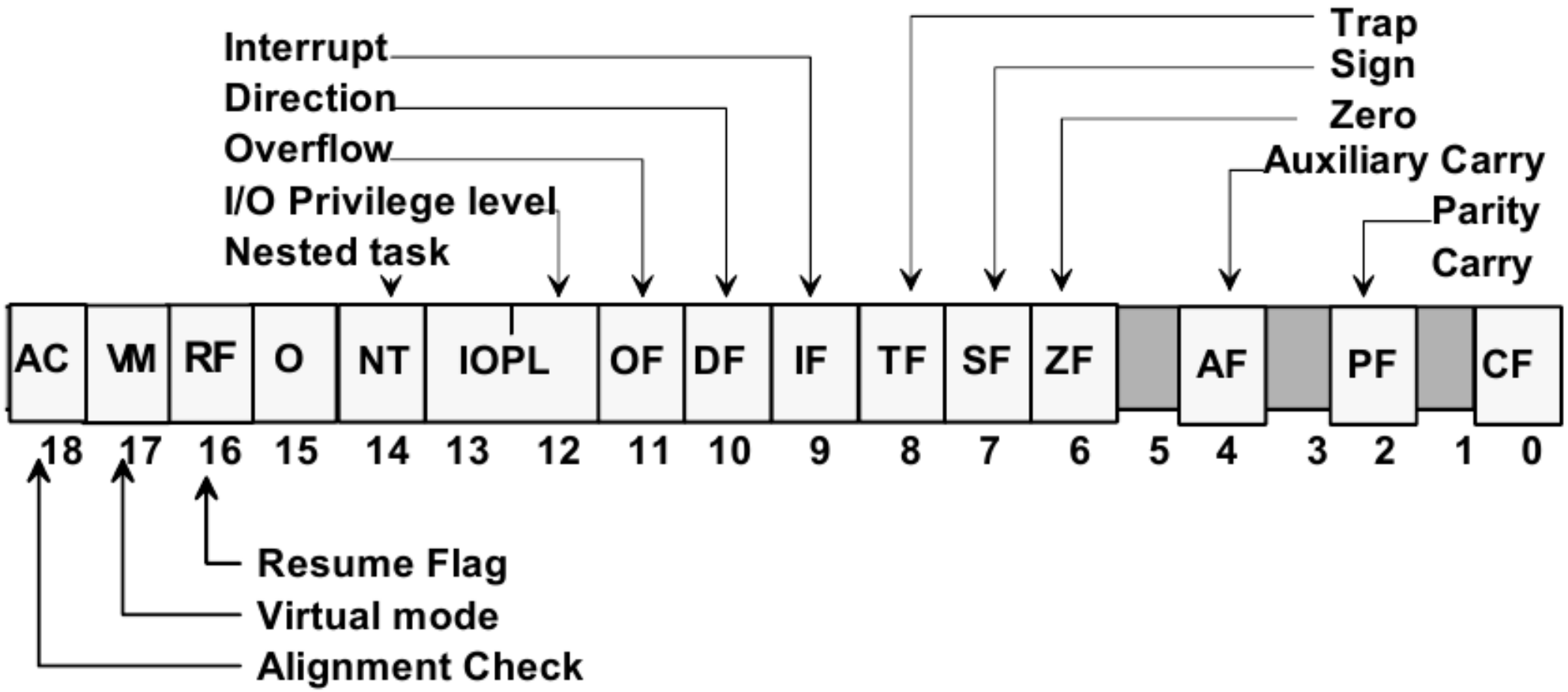
FX

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	OF	DF	IF	TF	SF	ZF	X	AF	X	PF	X	CF

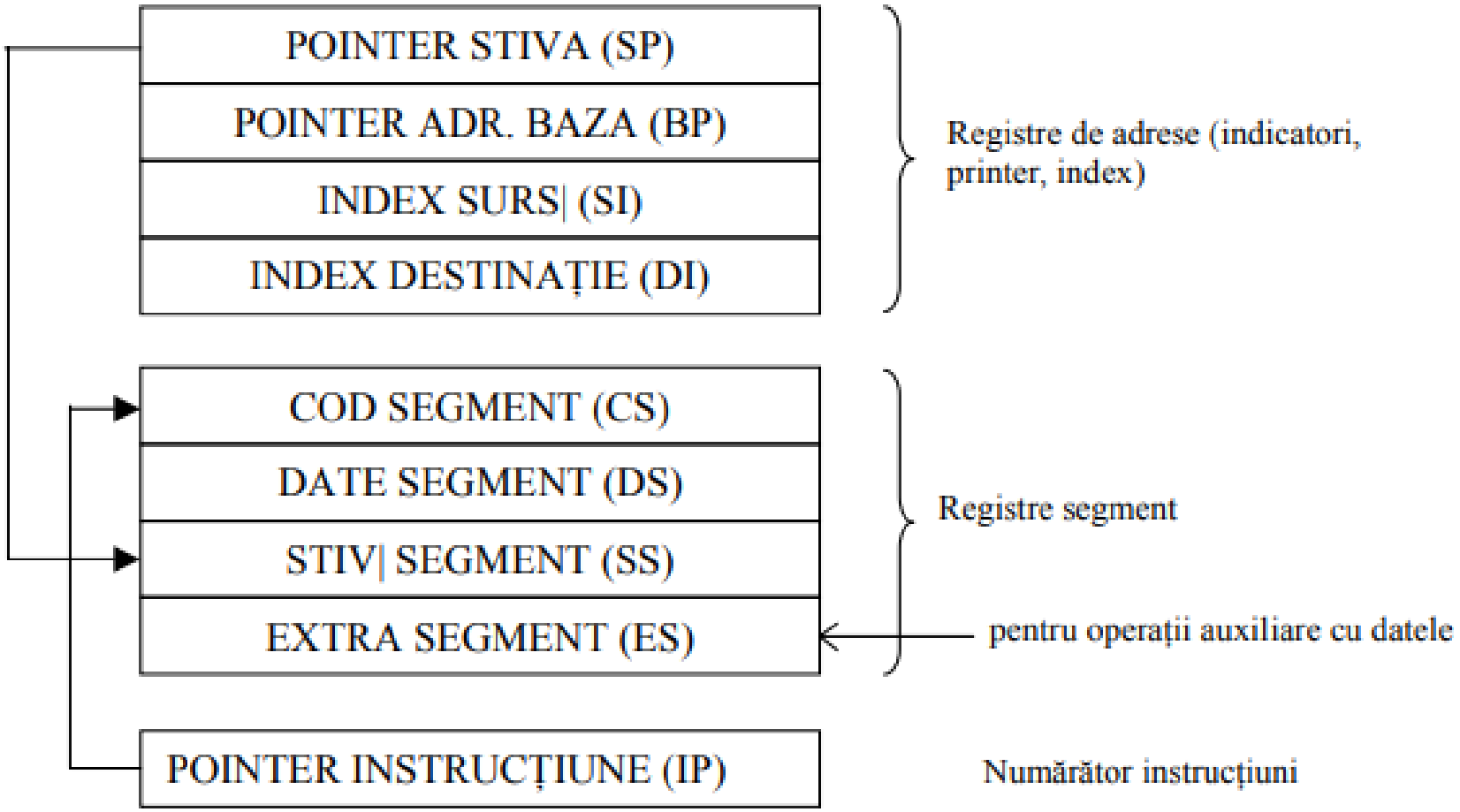
- Rg Fanioane (condiții) I8086



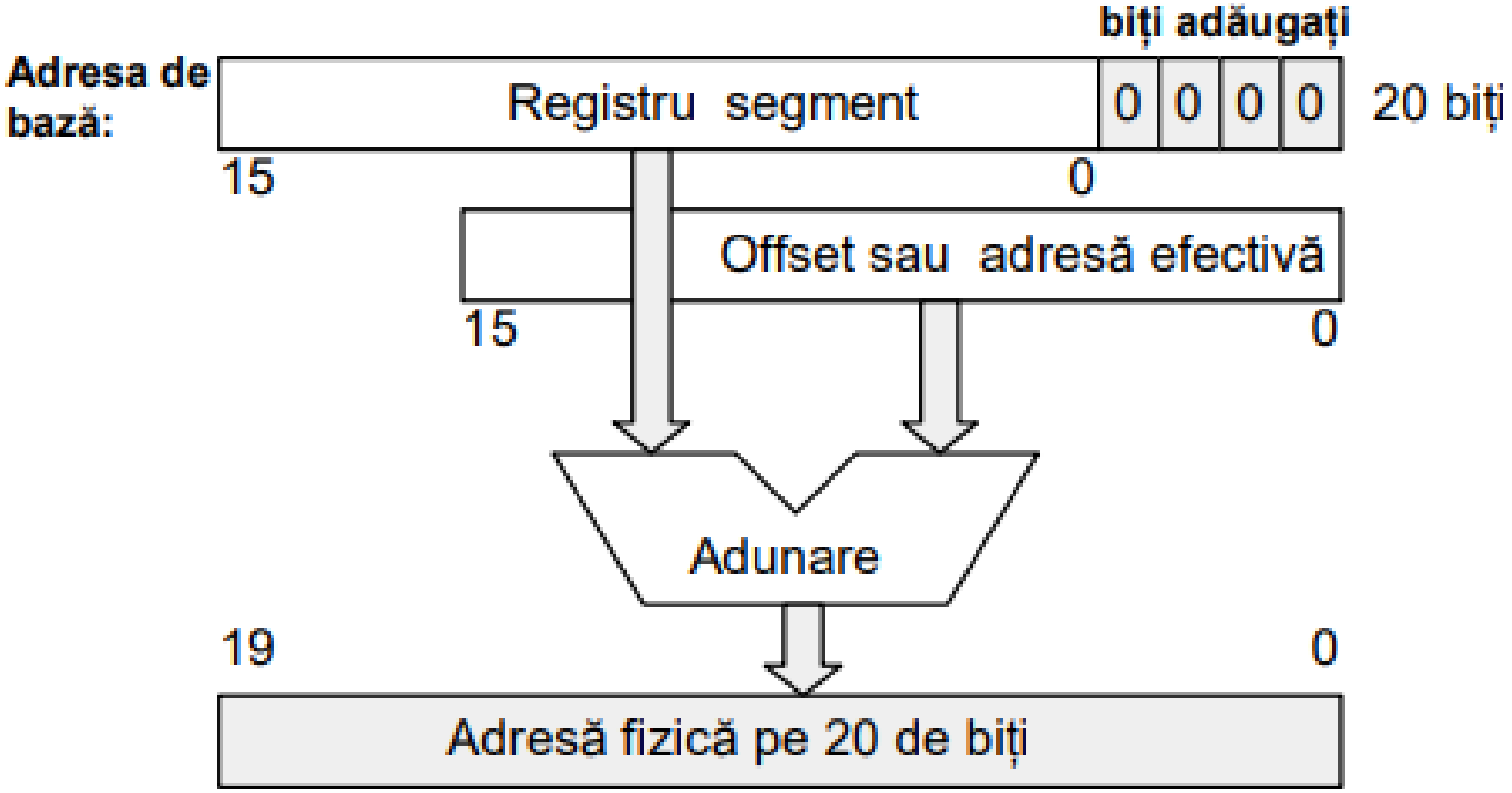
Rg Fanioane (condiții) I80486



Registre de adrese

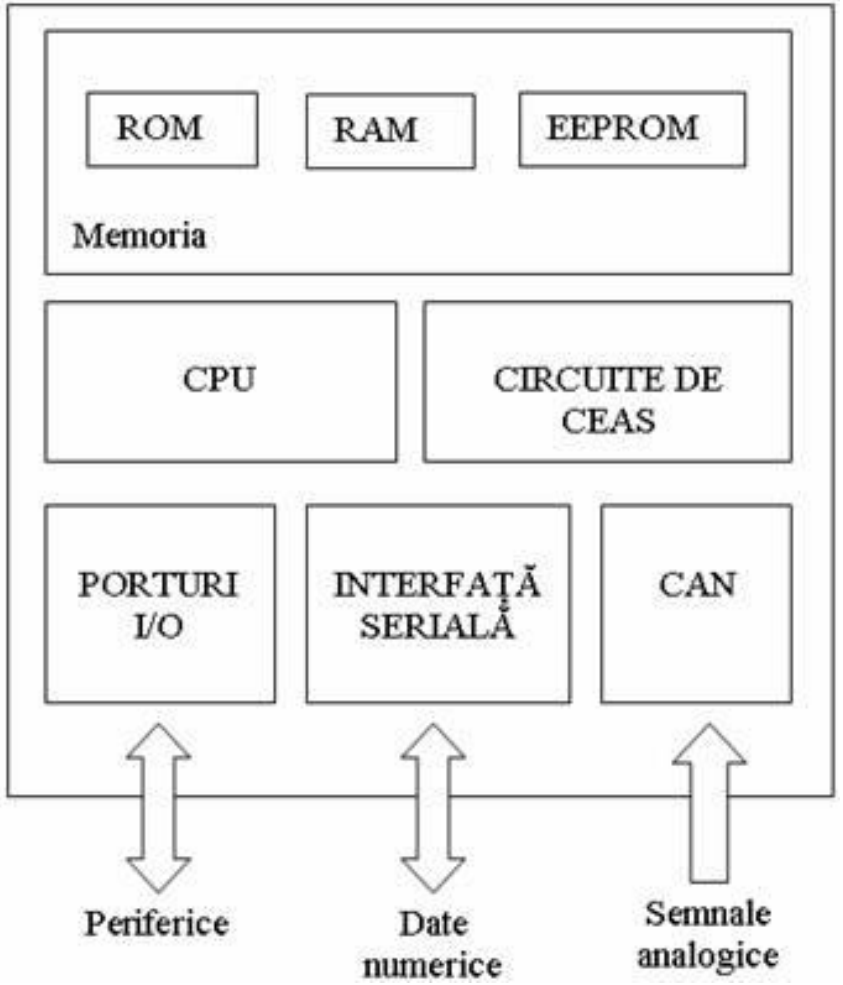


Adresa fizică RAM



2.3. MCU: Intel, AVR, MicroChip.

Structura generală a unui microcontroler:



Ce oferă un MCU?

MCU - este un calculator pe chip, proiectat și optimizat pentru a realiza funcții de control pentru anumite dispozitive electronice.

MCU – UCP + memorii + I/O + Timer - incorporate pe un chip.

MCU - Limitări în ceea ce privește memoriile și posibilitățile de interfațare.

MCU - se integrează ideal în aplicații în care spațiul ocupat, consumul de putere și prețul pe unitate sunt aspecte critice, în raport cu puterea de calcul.

MCU - Costuri reduse de implementare.

MCU - Sisteme de dezvoltare accesibile, la costuri rezonabile.

2.3. MCU: Intel, AVR, MicroChip.

Producători de MCU:

Microchip:

- Peste 10 miliarde de exemplare vandute până la moment.
- Economice și cu multe periferice: PIC 12, PIC14, PIC16, PIC18, PIC24, PIC32.
- Arhitectura RISC.

Motorola:

- Are sute de instrucțiuni, Arhitectura CISC.
- Exemple: 68HC05, 68HC08, 68HC11.

Intel: Producători diferiți: AMD, Atmel, Philips, Dallas/MAXIM Semiconductor.

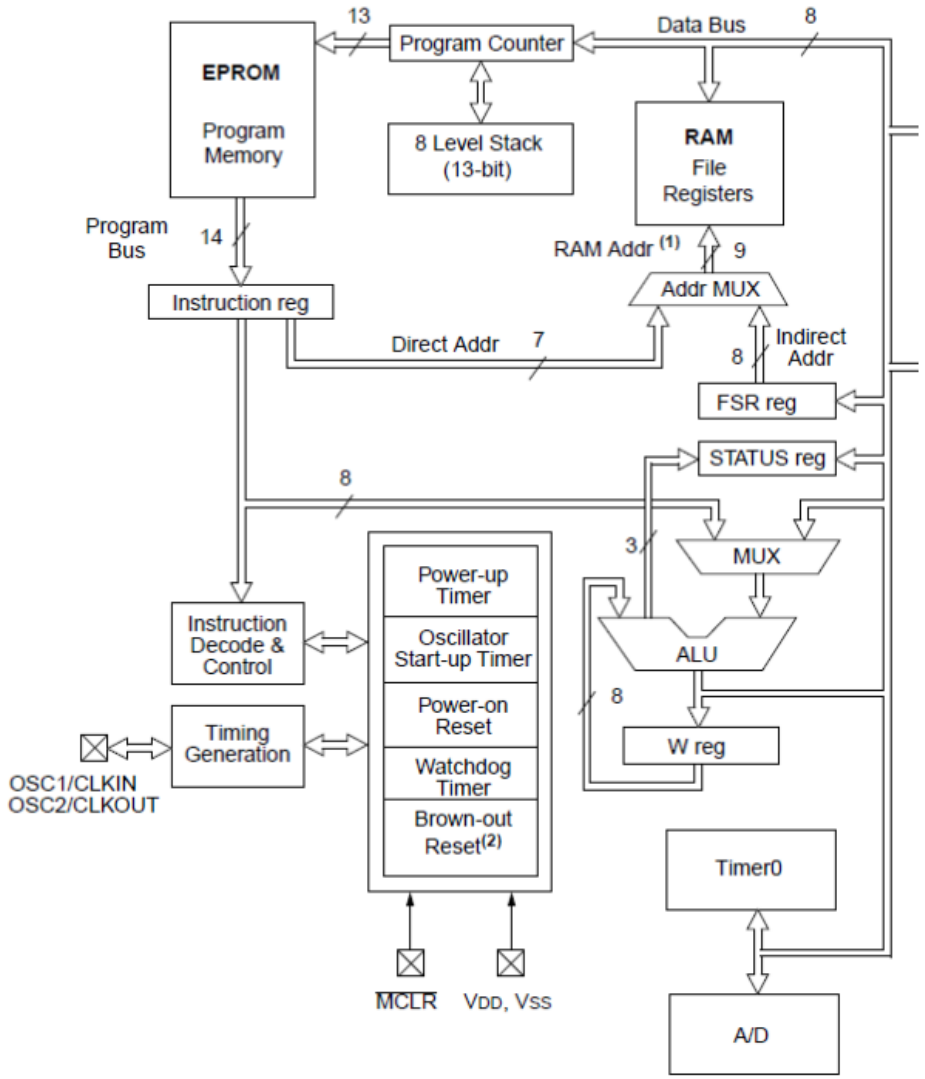
- Are sute de instructiuni, Arhitectura CISC.
- Exemple: 8051, 8052.

Atmel:

- Economice si cu multe periferice.
- Arhitectura AVR RISC: ATtiny, ATmega, Xmega.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.3. MCU: Intel, AVR, MicroChip.

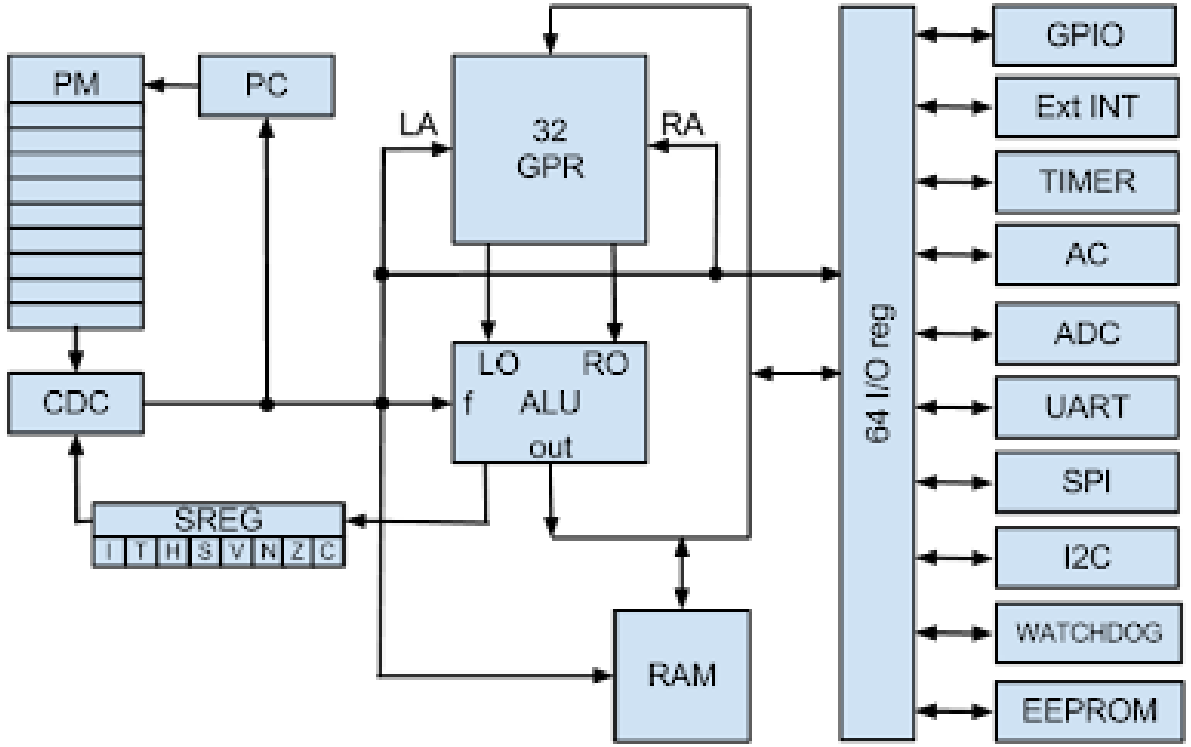


MCU PIC:

- **Porturi digitale I/O – 6 – 100;**
- **Instrucțiuni – 35, RISC;**
- **Controlor Int;**
- **Timere;**
- **EEPROM pentru date ~ 1KB;**
- **Flash – pentru Programe ~ 32KB;**
- **SRAM Data ~ 1KB;**
- **Porturi standard: USB, SPI, I2C, UART, LCD, CAD, Ethernet, PWM;**
- **Stiva integrată;**
- **Procesor – 8 / 32b.**

UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.3. MCU: Intel, AVR, MicroChip.

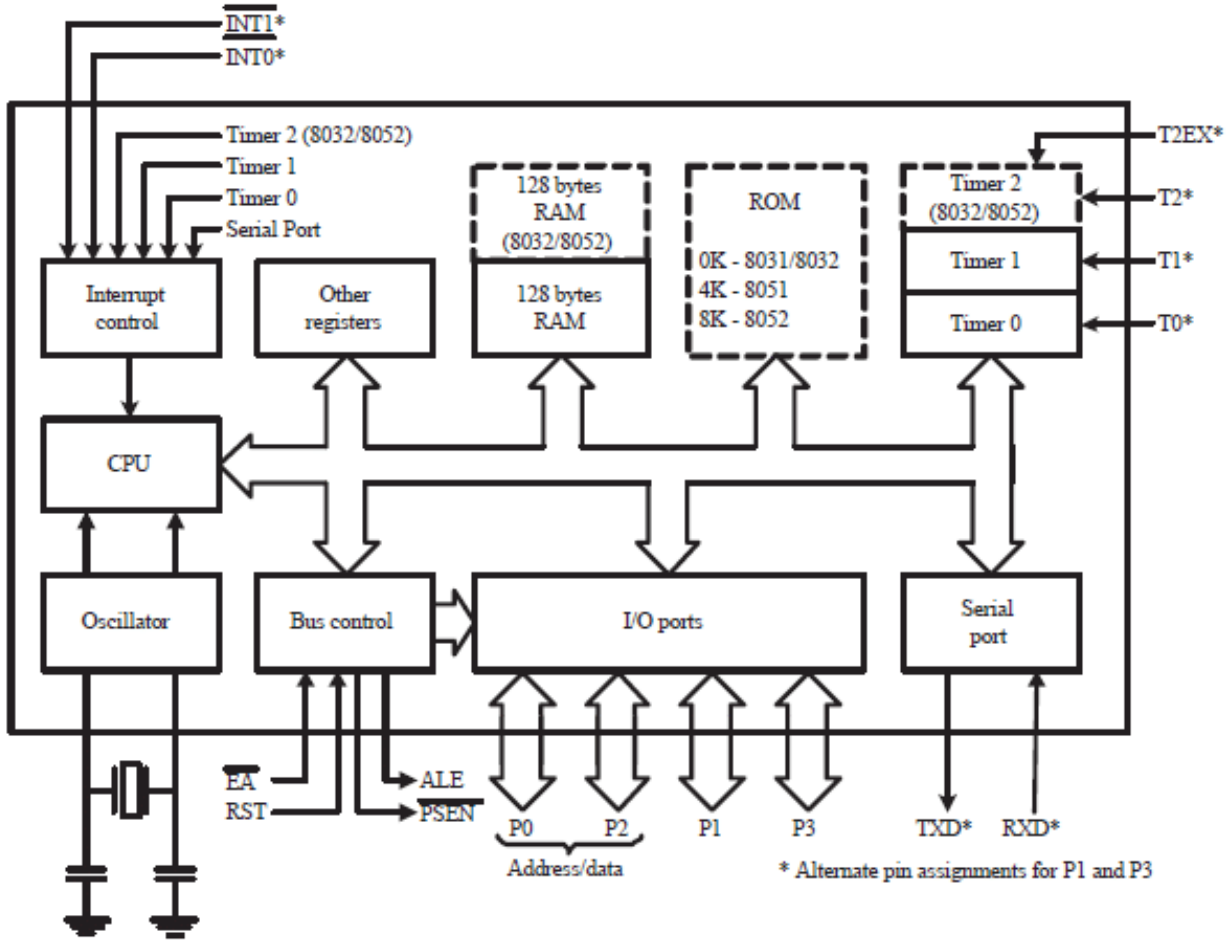


MCU AVR:

- **Porturi digitale I/O – 6 – 48;**
- **Instrucțiuni ~ 128, RISC;**
- **Controlor Int;**
- **Timere;**
- **EEPROM pentru date ~ 1KB;**
- **Flash pentru Programe 4KB – 256KB;**
- **SRAM Data – 1 – 16KB**
- **Porturi standard: USB, SPI, I2C, UART, LCD, CAD, Ethernet, PWM;**
- **Stiva integrată;**
- **Procesor – 8 / 32b.**

UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.3. MCU: Intel, AVR, MicroChip.



MCU AVR:

- **Porturi digitale I/O 32;**
- **Instrucțiuni ~ 256, CISC;**
- **Controlor Int;**
- **Timere;**
- **EEPROM pentru Programe 8KB ~ 64KB;**
- **RAM Data – 128 – 256B**
- **Porturi standard: UART (pentru MCU special este prevăzut și CAD);**
- **Stiva integrată;**
- **Procesor – 8b.**

Registrele microprocesoarelor x86-64

Fiecare program la execuţie obţine anumite resurse ale microprocesorului. Aceste resurse (registre) sunt necesare pentru executarea şi păstrarea în memorie a instrucţiunilor şi datelor programului, a informaţiei despre starea curentă a programului şi a microprocesorului.

Microprocesoarele pe 32 biţi funcţionează în diferite moduri, ce determină mecanismele de protecţie şi de adresare a memoriei: modul real 8086 (pe 16 biţi), modul virtual 8086 (V8086), modul protejat pe 32 biţi (inclusiv protejat pe 16 biţi). Modul de funcţionare a microprocesorului este impus de sistemul de operare (SO) în conformitate cu modul definit de aplicaţii (task-uri).

În microprocesoarele pe 64 biţi au fost introduse noi moduri de funcţionare:

Modul pe 64 biţi (64-bit mode) – acest mod susţine adresarea virtuală pe 64 biţi şi extensiile registrelor pe 64 biţi. În acest mod este folosit numai *modelul plat de memorie* (un segment comun pentru cod, date şi stivă).

Modul de compatibilitate (compatibility mode) permite SO să execute aplicaţii pe 32 şi 16 biţi. Pentru aplicaţii microprocesorul reprezintă un microprocesor pe 32 biţi cu toate atributele modului protejat, cu mecanismele de segmentare şi paginare.

2.4. Limbajul de programare Assembler. Setul de instrucțiuni x86.

Compatibilitatea regiţtrilor x86/64

Registre de uz general

63	31	15	0
RAX	eax	ah ax	al
RBX	ebx	bh bx	bl
RCX	ecx	ch cx	cl
RDX	edx	dh dx	dl
RBP	ebp	bp	
RSI	esi	si	
RDI	edi	di	
RSP	esp	sp	
R8			
R9			
R10			
R11			
R12			
R13			
R14			
R15			

63	31	15	0	
RFLAGS	eflags	flags		Registrul de fanioane
RIP	eip	ip		Registrul idicator de instrucţiuni

- CS - registru de segment de cod (code segment);
- DS - registru de segment de date (data segment);
- SS - registru de segment de stivă (stack segment);
- ES - registru de segment de date suplimentar (extra segment);

- AX - registru acumulator;
- BX - registru de bază general;
- CX - registru contor;
- DX - registru de date;
- BP - registru de bază pentru stivă (base pointer);
- SP - registru indicator de stivă (stack pointer);
- SI - registru index sursă;
- DI - registru index destinaţie.

Definirea datelor

Byte (octet).

Acest tip de date ocupa 8 biți, adică un octet (byte). Informaţia dintr-un octet poate fi: un *întreg fără semn* cuprins intre 0 si 225, *un întreg cu semn* cuprins intre –128 si 127, sau un *caracter* ASCII.

Definirea datelor de tip byte se face cu ajutorul directivelor BYTE şi SBYTE:

```
value1   BYTE    'A' ; character ASCII
value3   BYTE    255 ;  byte fără semn
value4   SBYTE   −128 ;  byte cu semn
value6   BYTE    ?  ; byte nedefinit
```

Definirea datelor de tip byte se face şi cu ajutorul directivei DB (*Define Byte*):

```
alfa    DB  65, 72h, 75o, 11011b, 11h+22h, 0ach
```

Definirea şirurilor de caractere:

```
salutare1   BYTE   "Good afternoon",0
greeting2   BYTE   'Good night',0
```

Utilizarea operatorului DUP (duplicate):

```
BYTE   20    DUP(0) ; 20 bytes, toate încărcate  cu zero
BYTE   20    DUP(?) ; 20 bytes, nedefiniţi
BYTE    4    DUP("STACK") ; 20 bytes: "STACKSTACKSTACKSTACK"
```

Definirea datelor

WORD (cuvânt).

Un cuvânt ocupa doi octeți (16 biți) si poate fi reprezentat intr-un registru de 16 biți sau in 2 octeți consecutivi de memorie.

Generarea datelor de tip cuvânt se poate face folosind directivele de tip WORD şi SWORD:

```
word1   WORD     65535 ;  întreg pe 16 biți fără semn
word2   SWORD   -32768 ; întreg pe 16 biți cu semn
word3   WORD ?           ; neiniţializat
```

Generarea datelor de tip cuvânt se poate face şi cu directiva de tip DW (*Define Word*):

```
beta DW     4567h,   0bc4ah,   1110111011b,   2476o
```

Double WORD (dublu cuvânt)

Un dublu cuvânt ocupa 2 cuvinte sau 4 octeți (32 biți) si poate fi reprezentat in memorie pe 4 octeți consecutivi, într-o pereche de registre de 16 biți sau într-un registru de 32 biți (la procesoarele de 32 biți).

Generarea datelor de tip dublu cuvânt se poate face folosind directivele DWORD şi SDWORD:

```
val1   DWORD    12345678h ; fără semn
val2   SDWORD   –21474836 ; cu semn
```

Generarea datelor de tip dublu cuvânt se face şi cu directiva DD (*Define Double Word*):

```
val1   DD   12345678h ; fără semn
```

Definirea datelor

QUAD – WORD (8 octeți)

Tipul Quad – word (QWORD) ocupa 8 octeți și este reprezentat in memorie pe 64 biți sau într-o pereche de registre de 32 biți (în cazul procesoarelor de 32 biți), sau într-un registru pe 64 biţi.

Informaţia stocata intr-un qword poate fi: un întreg cu sau fără semn pe 64 biți, sau un număr real în dublă precizie.

Generarea unor date de tip qword se face cu ajutorul directivei QWORD:

quad1 QWORD 1234567812345678h

Generarea unor date de tip qword se face şi cu directiva DQ (*Define Quad – word*):

quad1 DQ 1234567812345678h

Ten Bytes

Valorile Ten – byte (tbyte) ocupă 10 octeți consecutivi de memorie, sau unul din registrele coprocesorului matematic.

Generarea unor date de tip tbyte se face cu directiva TBYTE, de ex. valoarea zecimală -1234:

intVal TBYTE 80000000000000001234h

Generarea datelor de tip tbyte se face şi cu directiva DT (*Define Ten Bytes*):

intVal DT 80000000000000001234h

Definirea datelor

Definirea datelor în virgulă mobilă

Definirea datelor în virgulă mobilă se face cu directivele:

REAL4 – defineşte variabile în virgulă mobilă, în simpla precizie pe 32 biţi;

REAL8 – defineşte variabile în virgulă mobilă, în dubla precizie pe 64 biţi;

REAL10 – defineşte variabile în virgulă mobilă, cu precizie extinsă pe 80 biţi.

```
rVal1   REAL4   -1.2
rVal2   REAL8   3.2E-260
rVal3   REAL10   4.6E+4096
ShortArray   REAL4 20 DUP(0.0)
```

Definirea datelor în virgulă mobilă se face şi cu directivele:

```
rVal1   DD   -1.2
rVal2   DQ   3.2E-260
rVal3   DT   4.6E+4096
```

Gama valorilor definite pot fi:

REAL4 - 1.18×10^{-38} până 3.40×10^{38}

REAL8 - 2.23×10^{-308} până 1.79×10^{308}

REAL10 - 3.37×10^{-4932} până 1.18×10^{4932}

2.4. Limbajul de programare Assembler. Setul de instrucțiuni x86.

Setul de instrucțiuni compatibil x86

Setul de instrucţiuni este grupat în 6 clase:

instrucţiuni de transfer, care deplasează date între memorie sau porturi de intrare/ieşire şi registrele microprocesorului, fără a executa nici un fel de prelucrare a datelor;

instrucţiuni aritmetice şi logice, care prelucrează date în format numeric;

instrucţiuni pentru şiruri, specifice operaţiilor cu date alfanumerice;

instrucţiuni pentru controlul programului, care în esenţă se reduc la salturi şl la apeluri de proceduri;

instrucţiuni specifice **întreruperilor hard şi soft**;

instrucţiuni pentru controlul procesorului.

Fiecare categorie de instrucţiuni este însoțită de specificarea explicită a flag-urilor (indicatorilor de condiţie) care sunt modificaţi în urma execuţiei.

Structura generală a instrucţiunilor x86 este următoarea:

[eticheta:] mnemonic [operanzi] [; comentariu]

Instrucţiunile pot conţine zero, unu, doi sau trei operanzi. Omitem câmpurile etichetă şi comentariu:

mnemonic

mnemonic **[*destinatie*]**

mnemonic **[*destinatie*],[*sursa*]**

mnemonic **[*destinatie*],[*sursa-1*],[*sursa-2*]**

Setul de instrucțiuni compatibil x86

Exemplu de program:

```
.data ; Segmentul de Date
val1 WORD 1000h
arrayB BYTE 10h,20h,30h,40h,50h
arrayW WORD 100h,200h,300h
arrayD DWORD 10000h,20000h
.code ; Segmenatul de cod
main PROC
mov bx, 0A69Bh
movzx eax, bx ; EAX = 0000A69Bh
movzx cx, bl ; CX = 009Bh
mov bx, 0A69Bh
movsx eax, bx ; EAX = FFFFA69Bh
mov ax, val1 ; AX = 1000h
mov val1, ax ; val1 = 2000h
mov al, arrayB ; AL = 10h
mov al, [arrayB+1] ; AL = 20h
mov ax, arrayW ; AX = 100h
mov ax, [arrayW+2] ; AX = 200h
mov eax, arrayD ; EAX = 10000h
mov eax, [arrayD+4] ; EAX = 20000h
mov eax, [arrayD+4] ; EAX = 20000h
```

Setul de instrucțiuni poate fi consultat în sursele de informare suplimentare. Modul de utilizare a acestora va fi studiat pe parcurs în procesul efectuării sarcinelor test.

2.5. Medii pentru dezvoltarea, compilarea și testarea produselor program.

Resursele Internet oferă o mulțime de medii pentru dezvoltarea, compilarea și testarea funcțională a produselor program elaborate în limbajul Assembler.

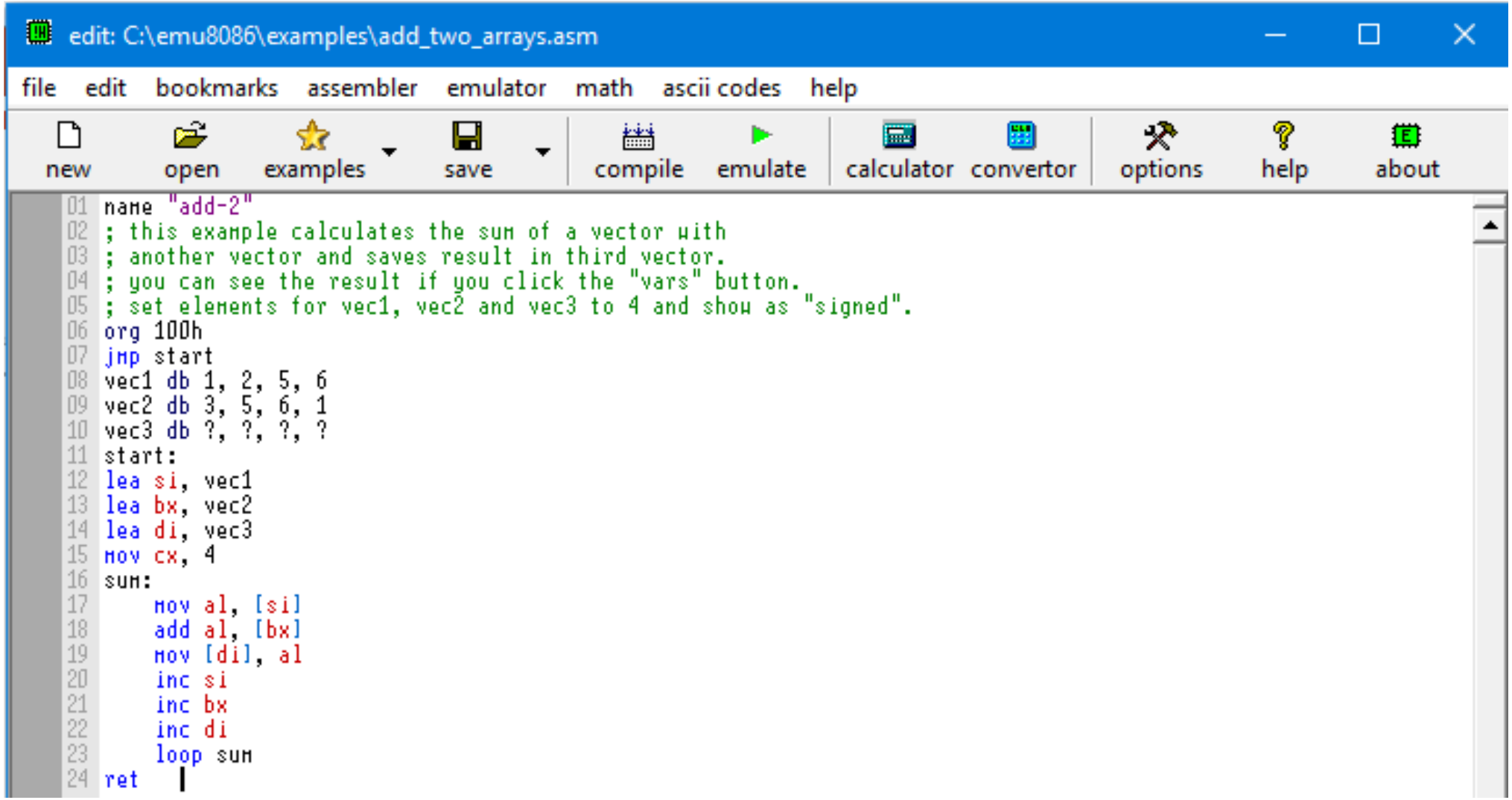
Majoritatea limbajelor de programare compilate oferă posibilitatea de a integra în structura s-a segmente de cod în limbajul Assembler ceea ce simplifică exențial accesul programatorului la resursele fizice ale calculatoarului. Modul de integrare este specificat de fiecare limbaj de programare în parte.

Pentru testarea exemlelor oferite în cadrul cursului va fi utilizat mediul EMU8086 care poate fi încarcat din resursele Internet:

- https://emu8086.ru.malavida.com/
- http://av-assembler.ru/asm/afd/start-emu8086.htm
- http://softbuka.ru/soft/view-EMU8086.html
- https://emu8086-microprocessor-emulator.softonic.ru/
- https://emu8086-microprocessor-emulator.en.softonic.com/download

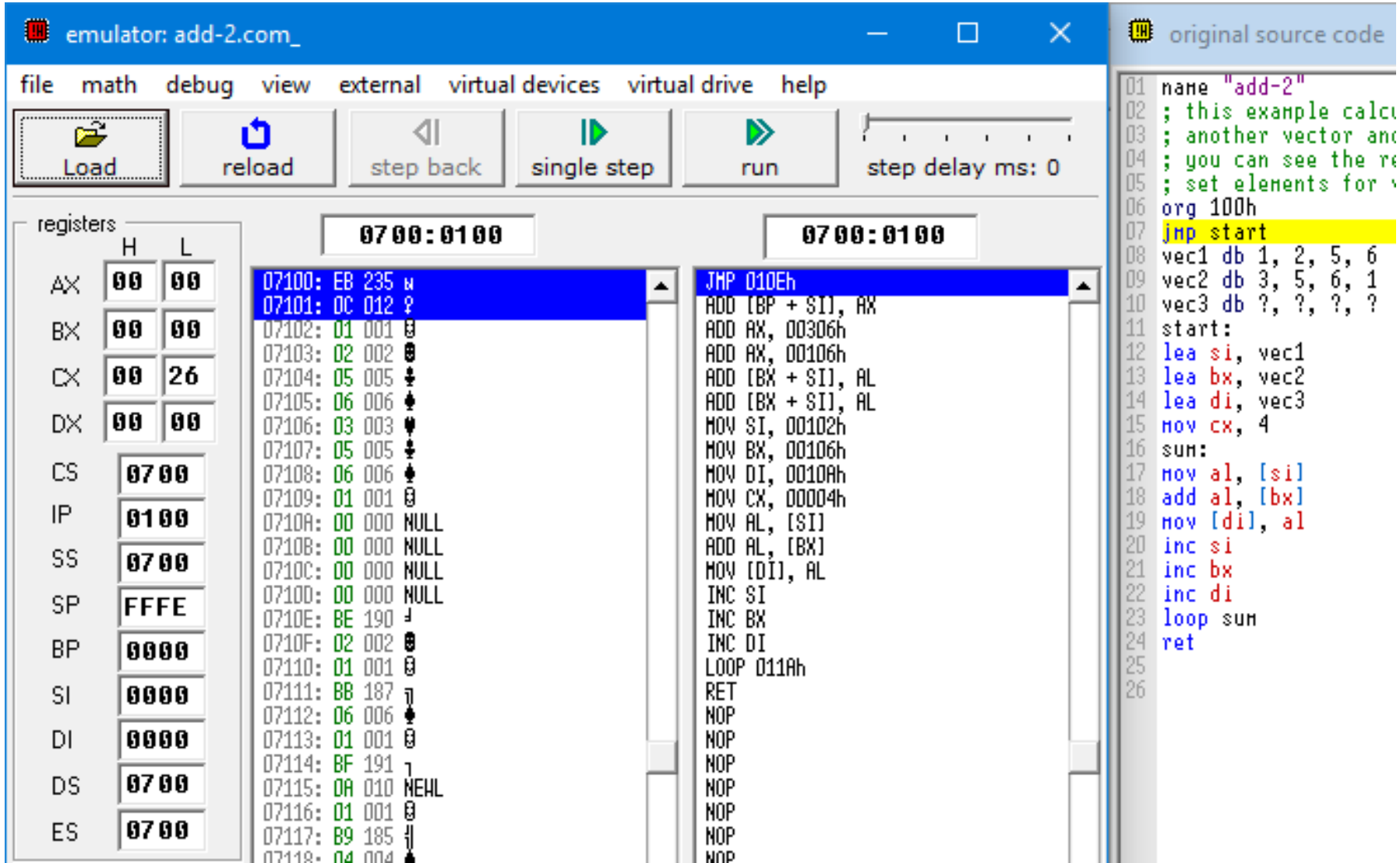
UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.5. Medii pentru dezvoltarea, compilarea și testarea produselor program.



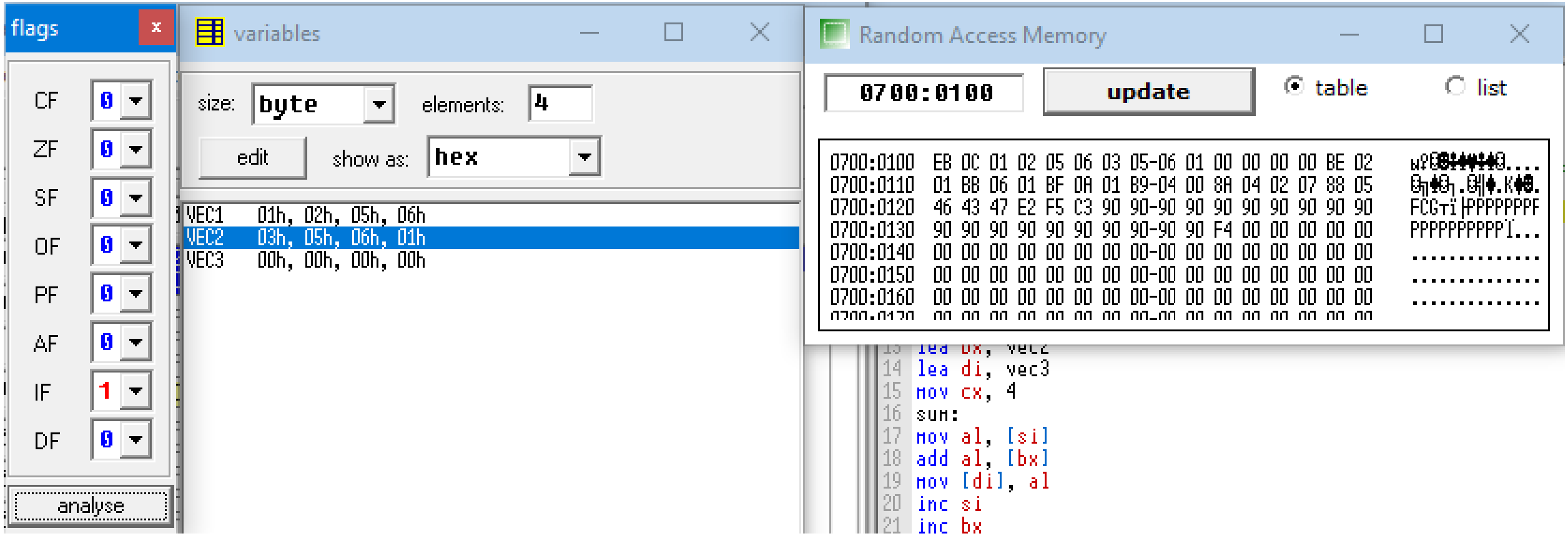
UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.5. Medii pentru dezvoltarea, compilarea și testarea produselor program.



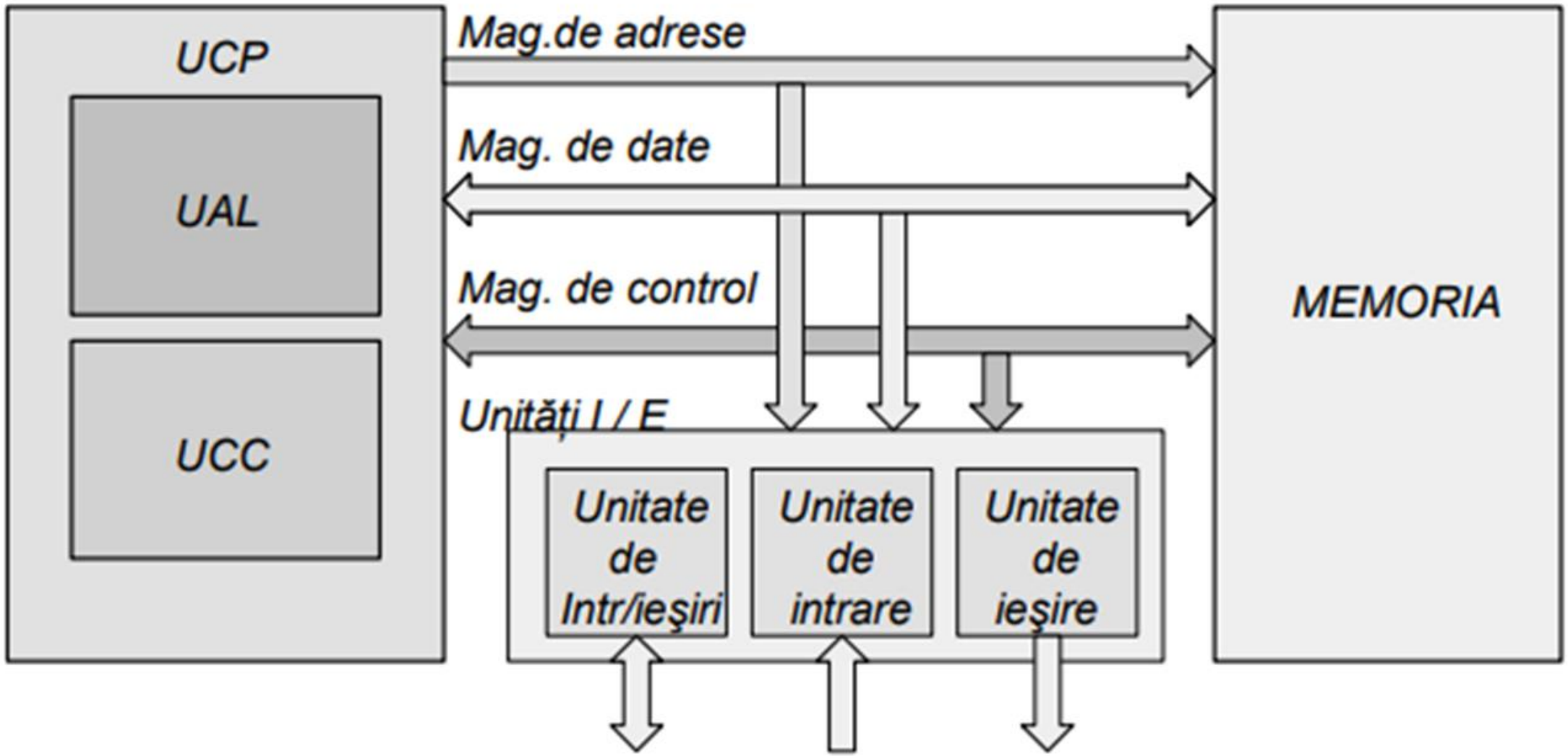
UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.5. Medii pentru dezvoltarea, compilarea și testarea produselor program.

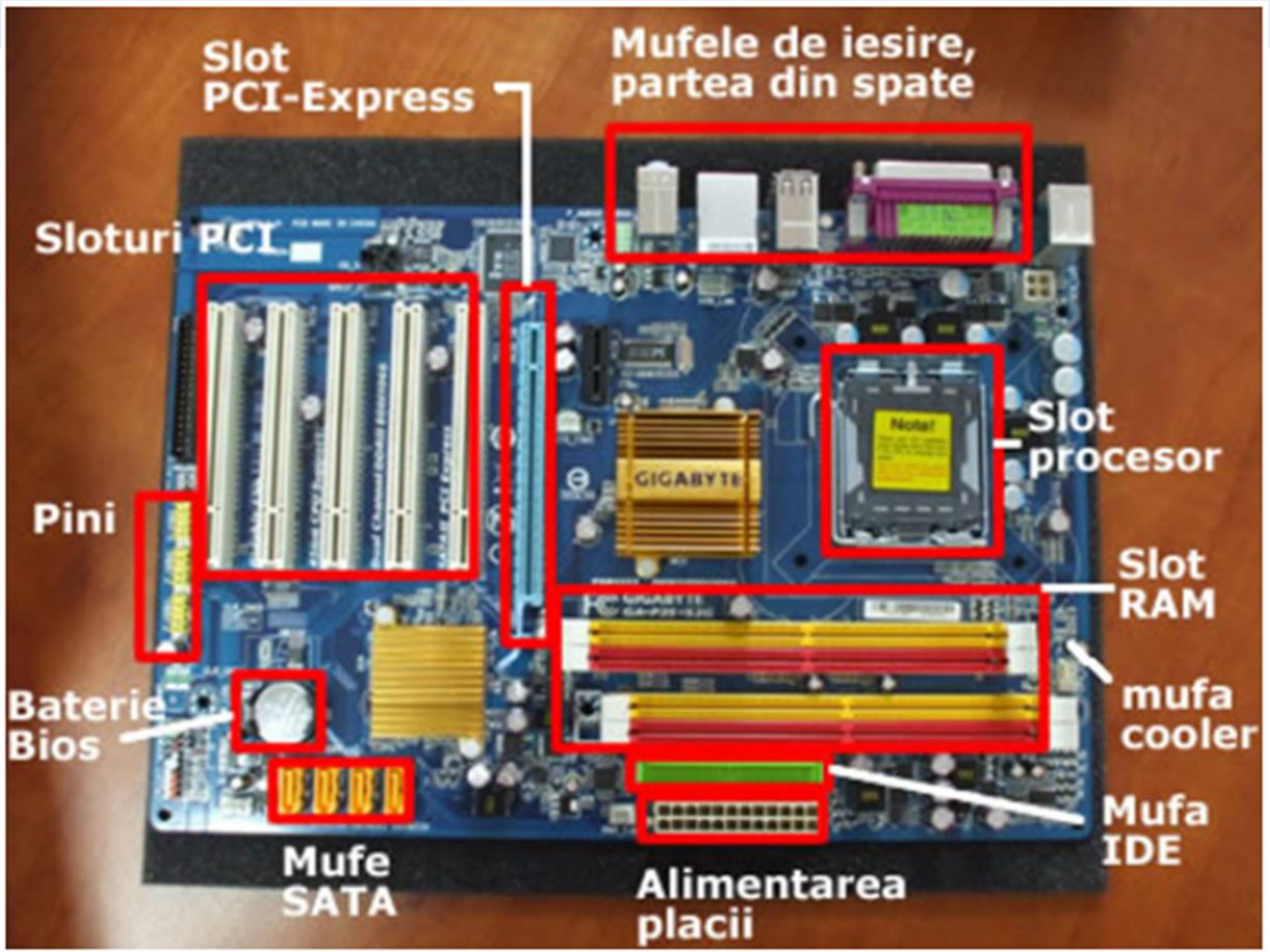


UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.6. Magistrala de sistem.



2.6. Magistrala de sistem.



2.6. Magistrala de sistem.

Caracteristicile MS:

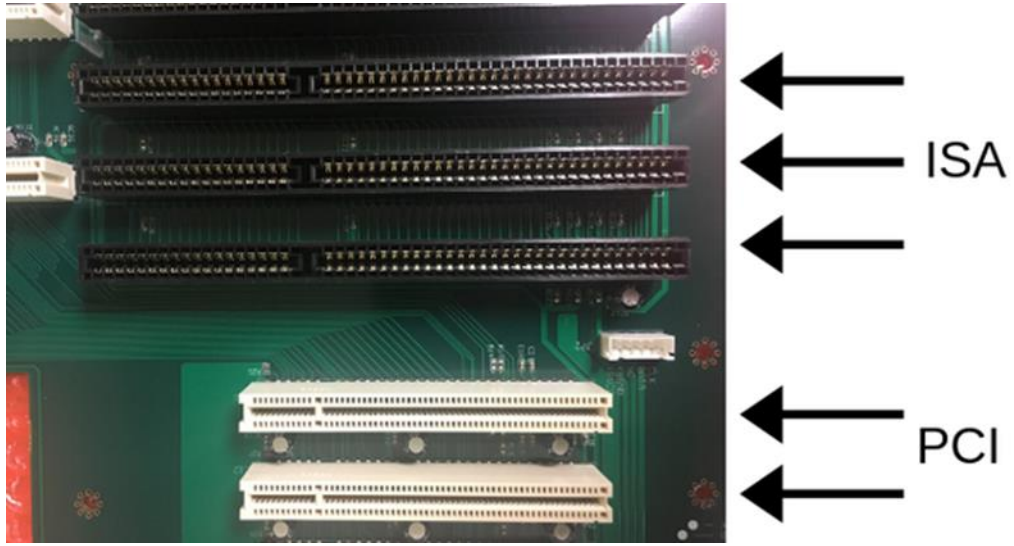
- **Magistrala de Adresa: 16, 20, 24, 32, 64, 128;**
- **Magistrala de Date: 8, 16, 32, 64, 128;**
- **Magistrala A/D: multiplexata, non-multiplexata;**
- **Magistrala de control: semnale fizice, comenzi de magistrala;**
- **Magistrala de stare: semnale fizice, cuvinte de stare.**

2.6. Magistrala de sistem.

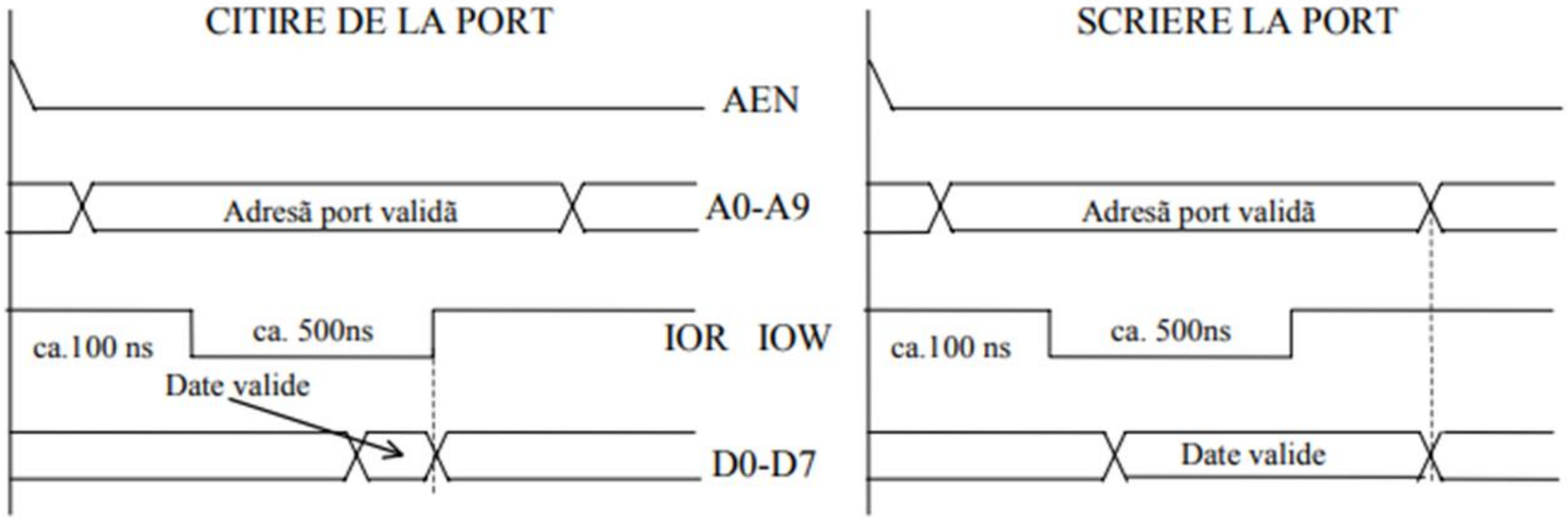
Standarde ale MS:

- **ISA – Industrial Standard Architecture – Date=8/16b, Adrese=20/24b;**
- **EISA – Extended ISA – A/D=16/32b;**
- **PS/2 - Micro Channel – A/D=16/32b;**
- **PCI – Peripheral Component Interconnect – A/D=32/64b;**
- **Express PCI – A/D=32/64b;**
- **VL-Bus – A/D=32/64b.**

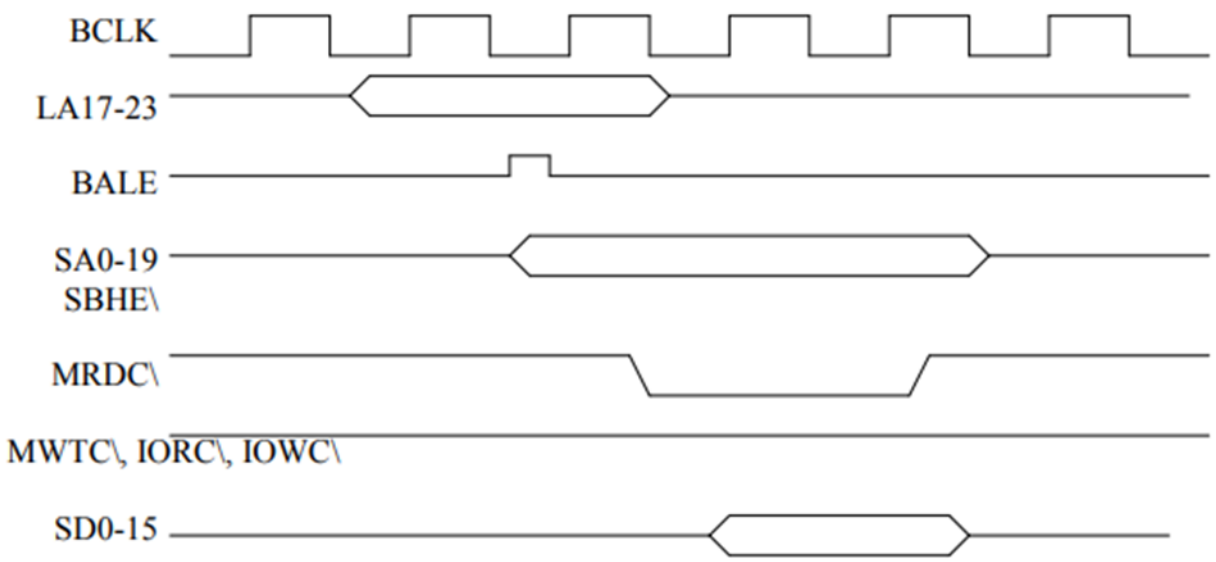
Standarde ale MS ISA și PCI:

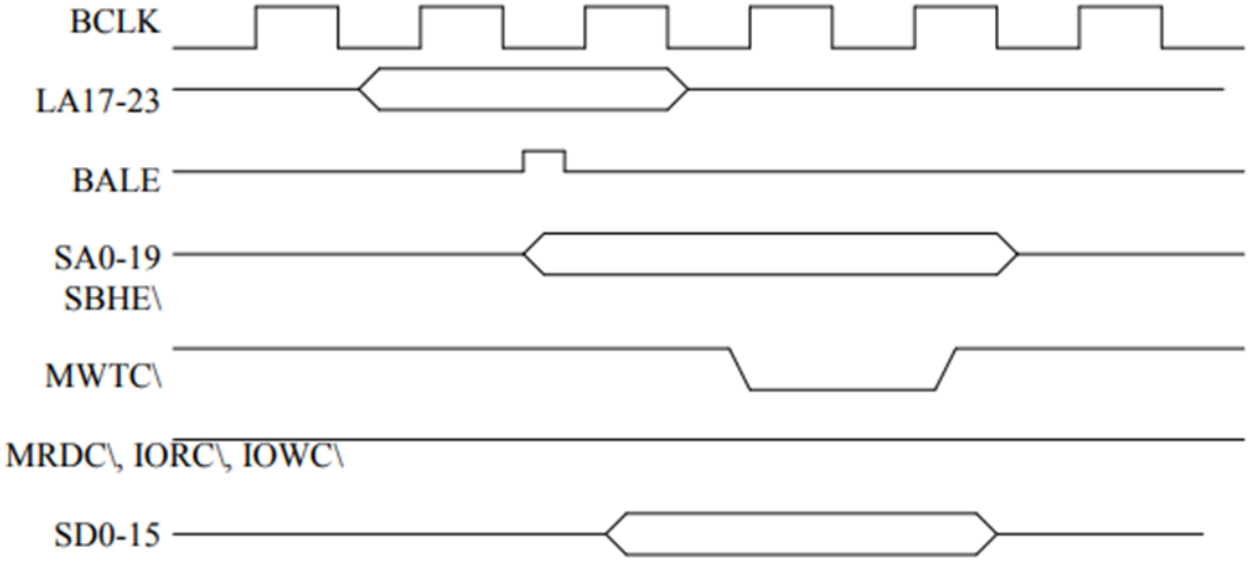


Ciclul MS ISA. Adresare la porturile I/O:

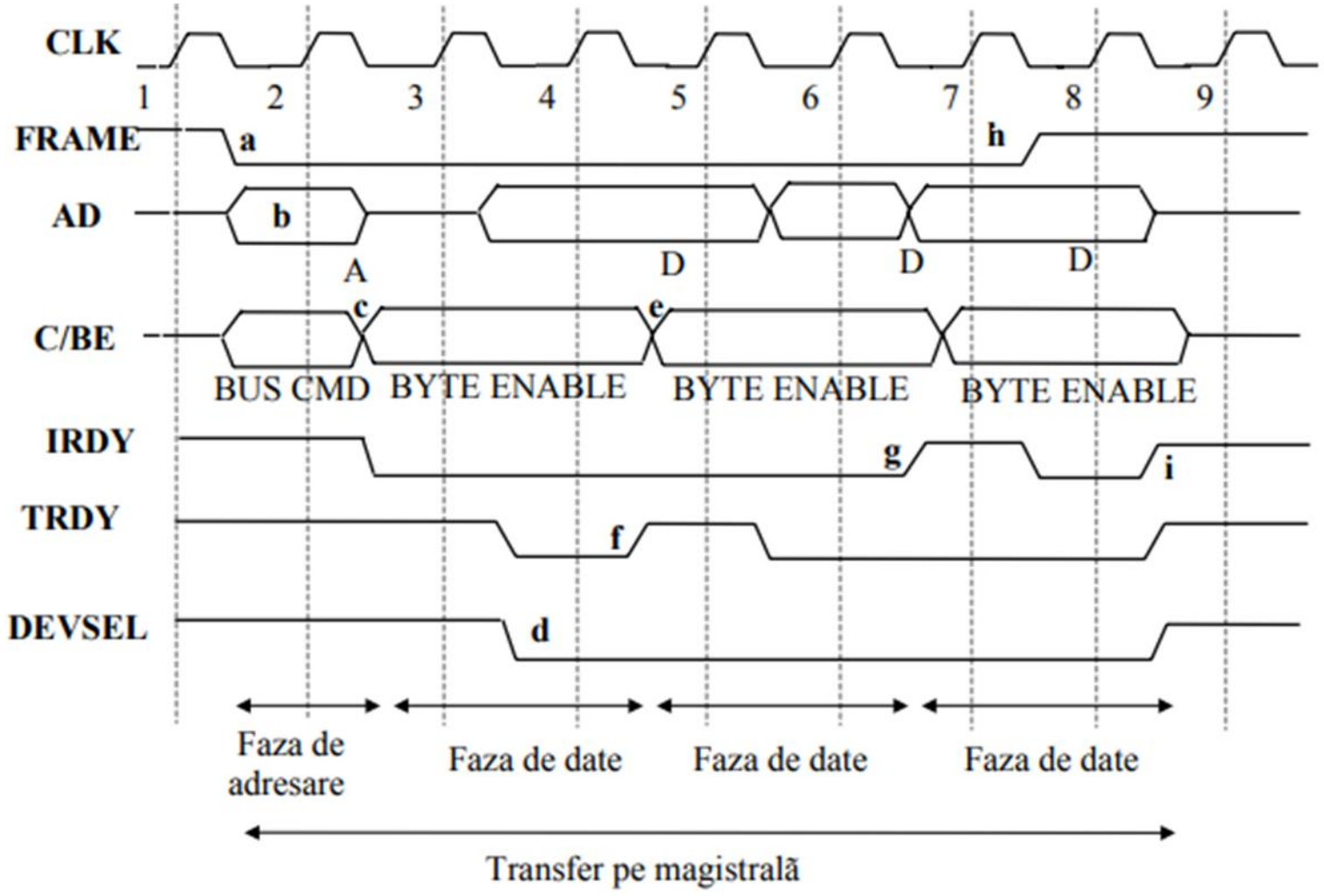


Ciclul MS ISA. Operații cu Memoria:

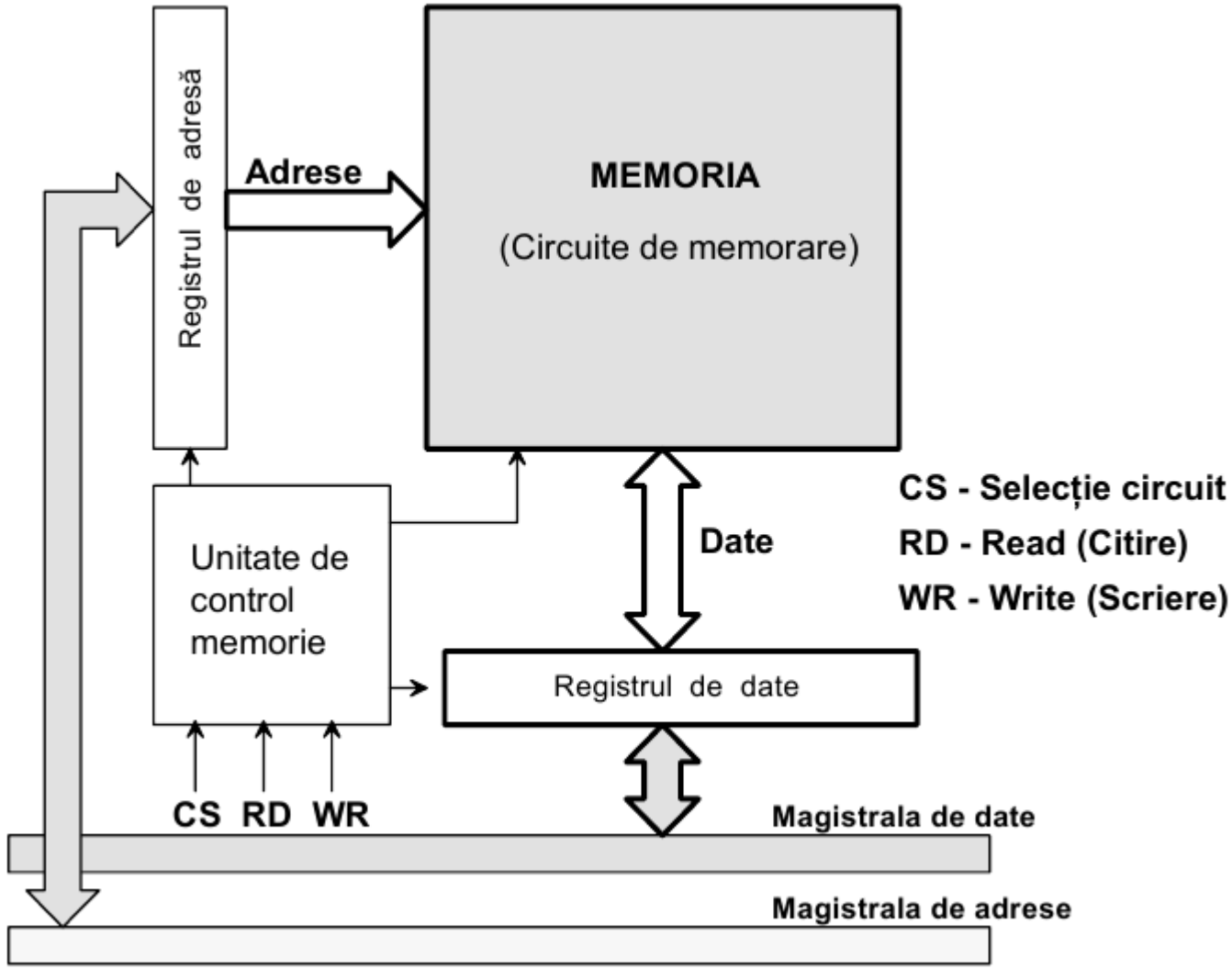




Ciclul MS PCI. Regim flux de date:

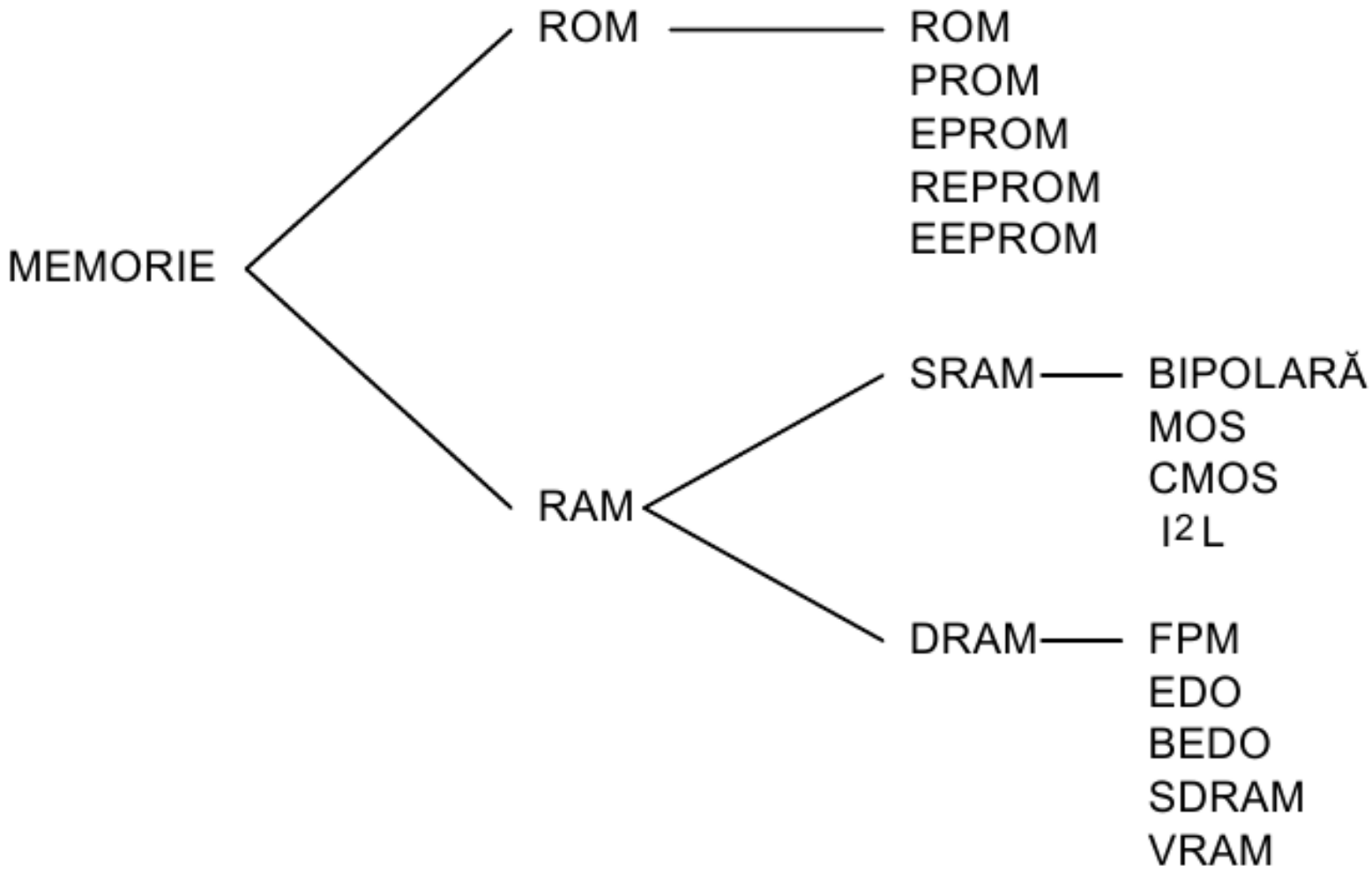


2.7. Memoria internă a sistemelor de calcul.

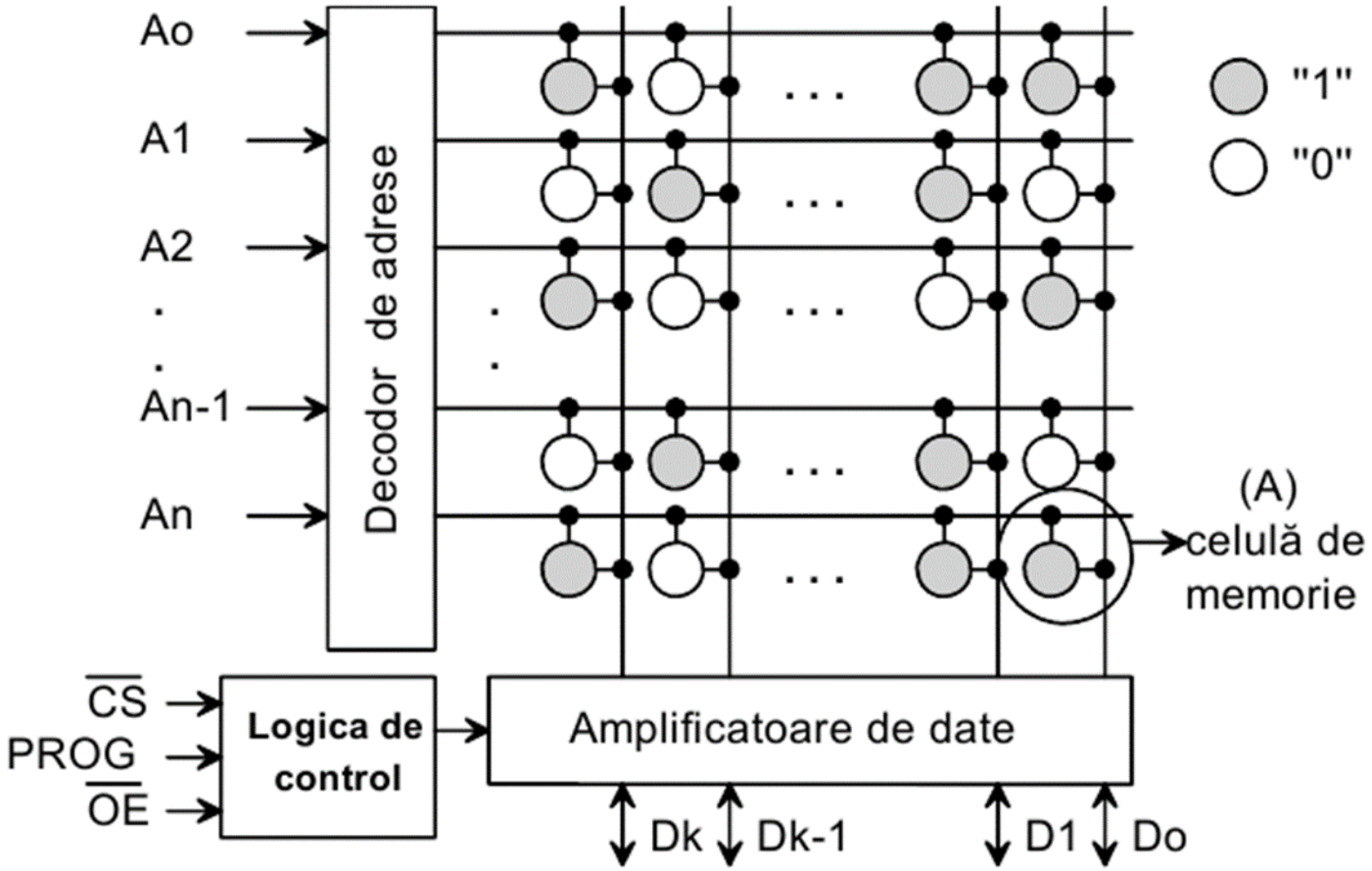


UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.7. Memoria internă a sistemelor de calcul.

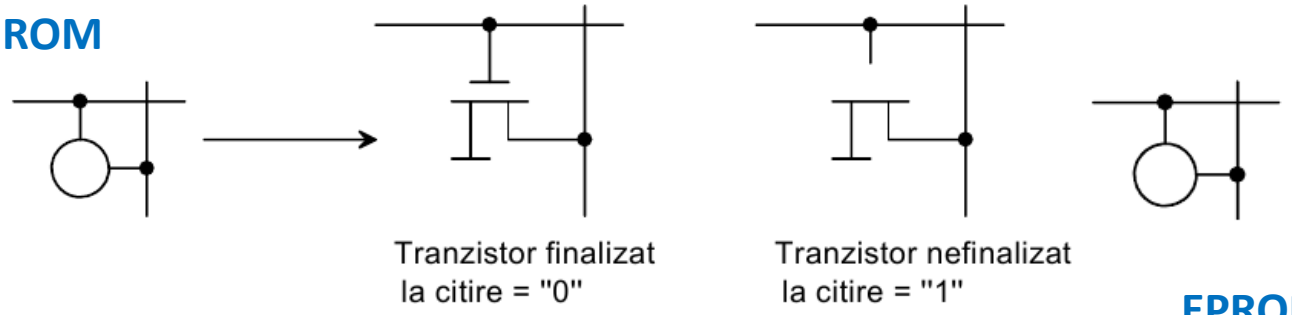


Memoria xROM (Read Only Memory):

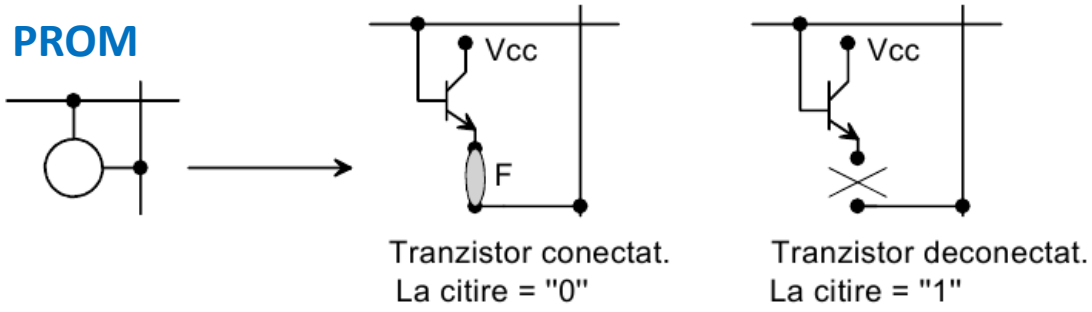


Realizarea tehnologică xROM:

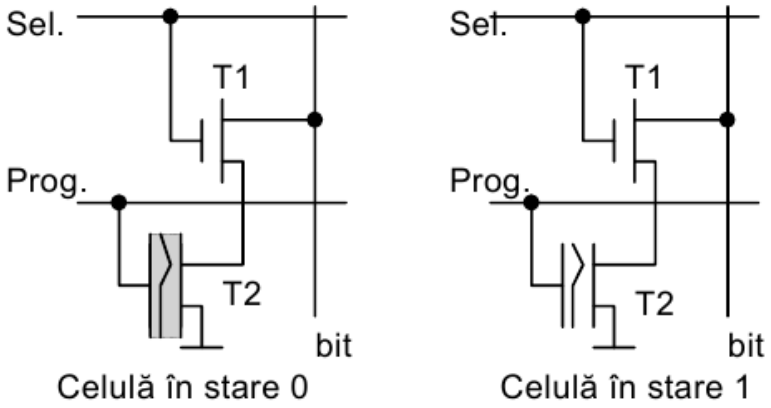
ROM



PROM

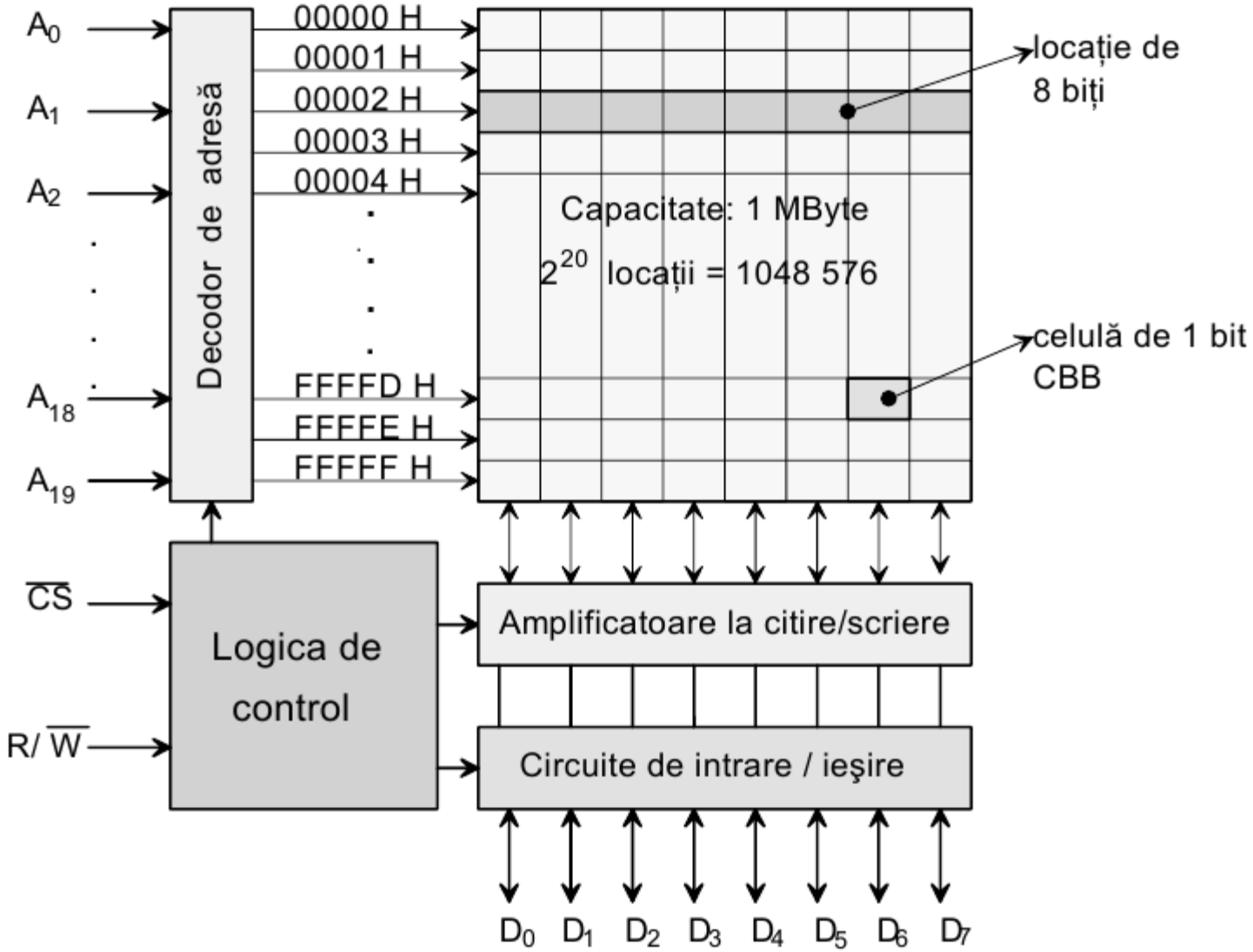


EPROM/ EEPROM/ Flash



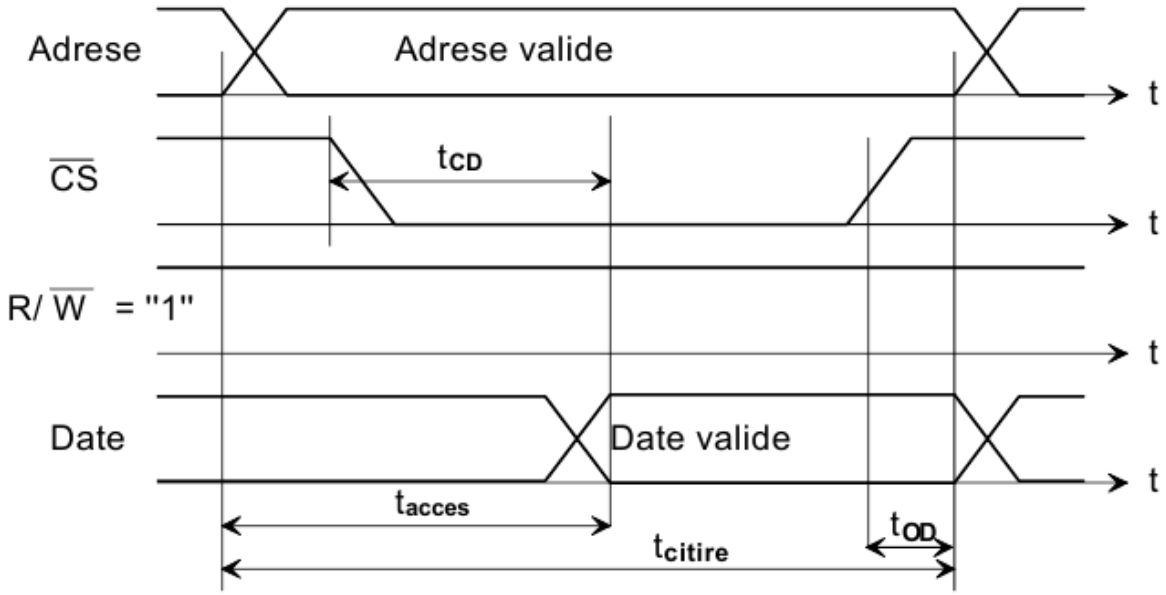
2.7. Memoria internă a sistemelor de calcul.

Realizarea tehnologică SRAM:

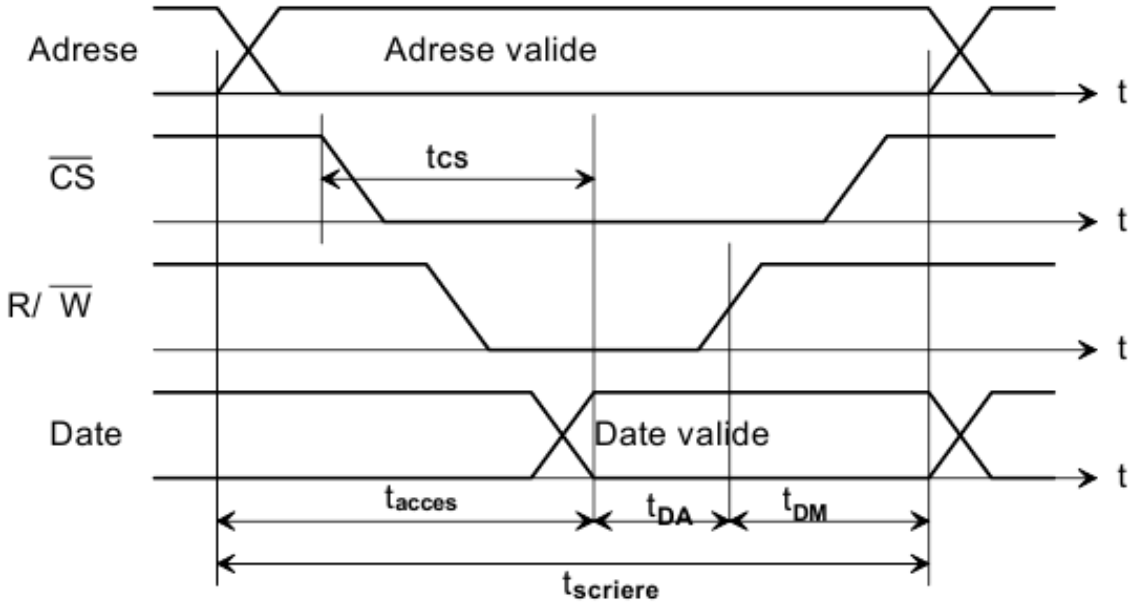


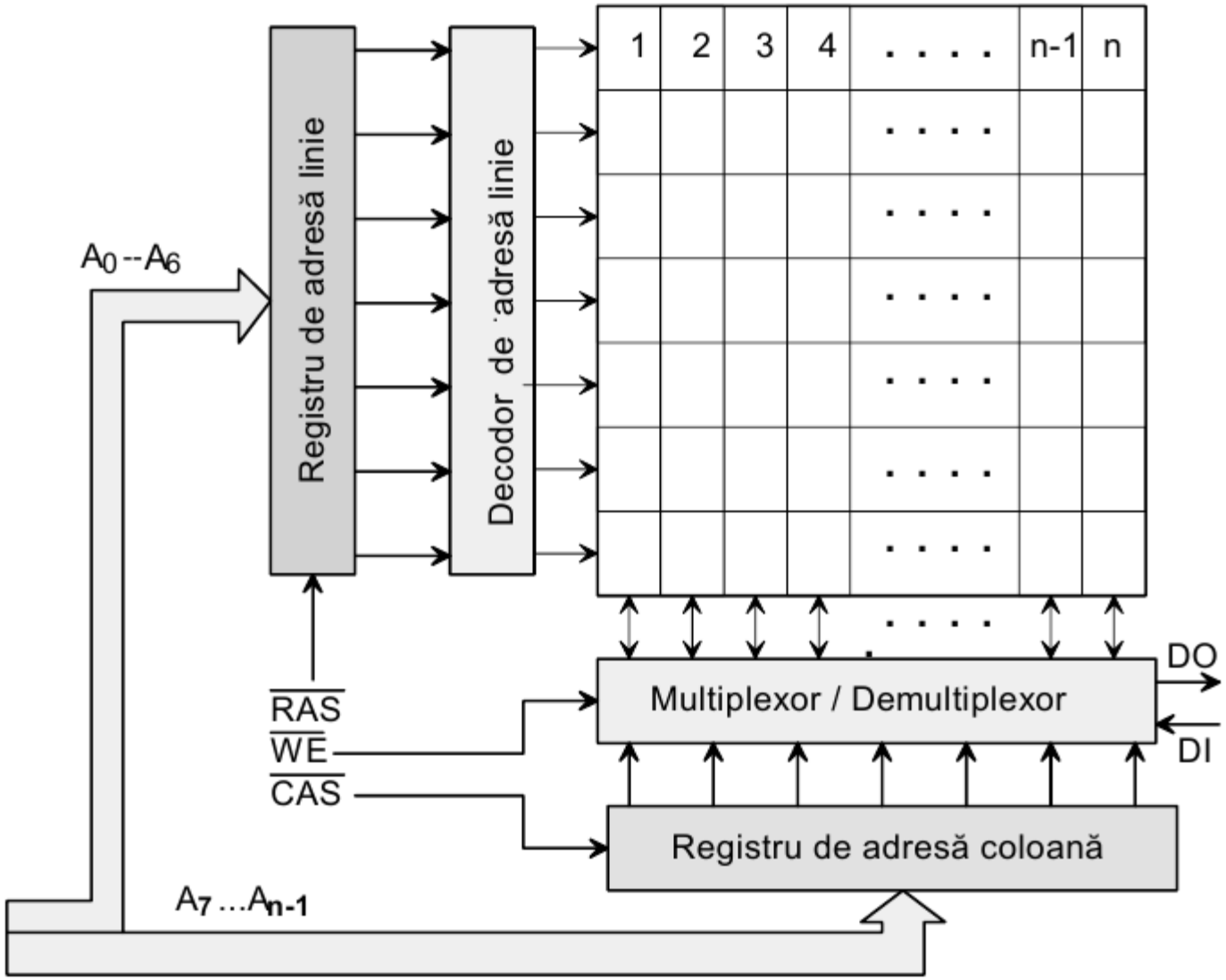
Realizarea operațiilor de Citire și Înscriere SRAM:

Citire din RAM



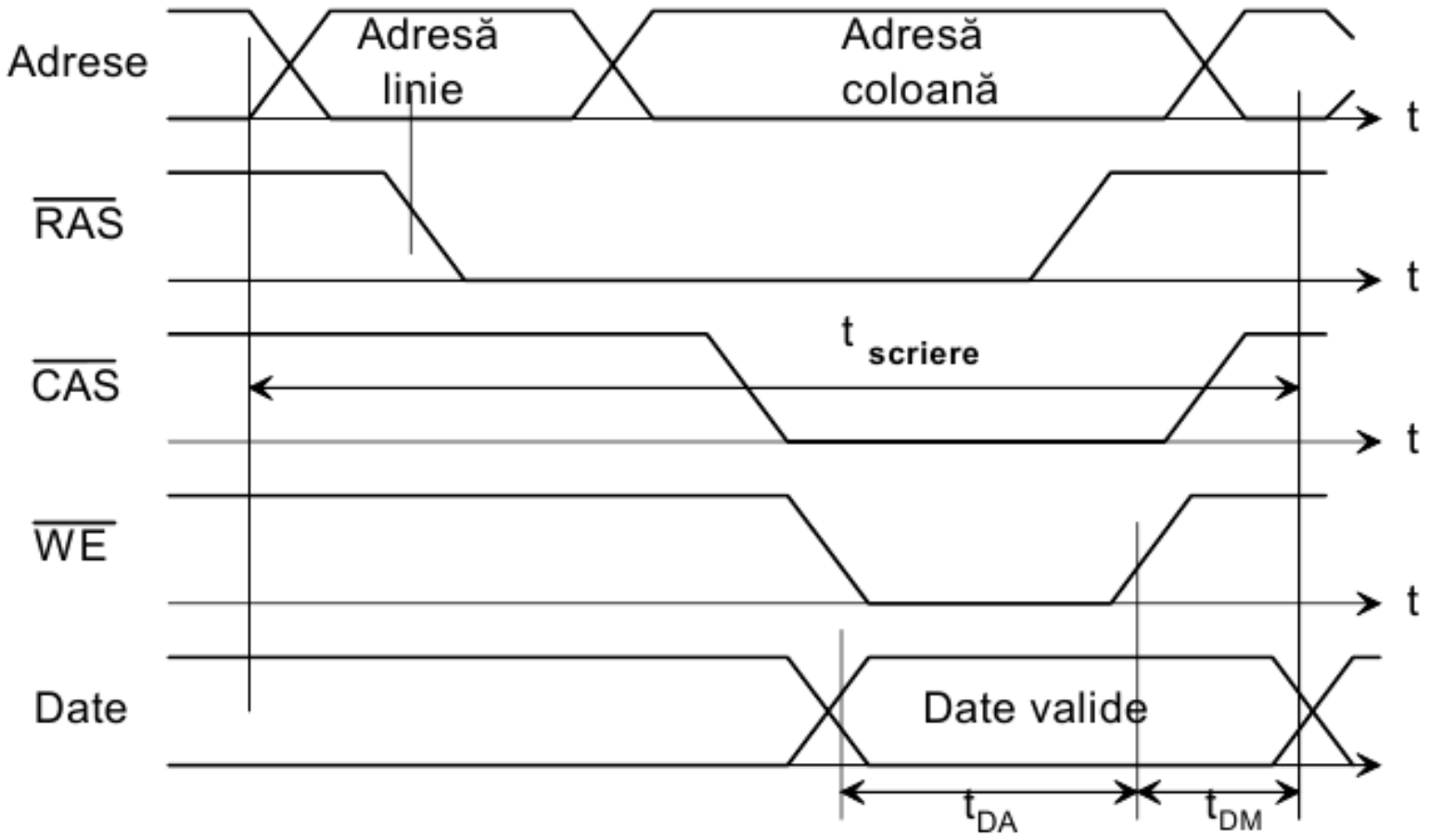
Inscriere in RAM





Realizarea tehnologică DRAM:

Realizarea operației de Înscriere în DRAM:



2.8. Proiectarea memoriei.

Procesul de proiectare a memoriei unui sistem de calcul include următorii pași:

1. Specificarea volumului de memorie RAM și ROM;
2. Specificarea dispozirivelor electronice utilizate pentru proiectarea memoriei;
3. Repartizarea spațiului de adrese rezervate pentru memoria RAM și ROM;
4. Calculul numărului de dispozitive RAM și ROM în arhitectura memoriei;
5. Proiectarea decodificatoarelor pentru memoria RAM și ROM;
6. Proiectarea memoriei RAM și ROM.

2.9. Controloare specializate: Int, DMA.

Majoritatea sistemelor de calcul dispun de 3 controloare specializate:

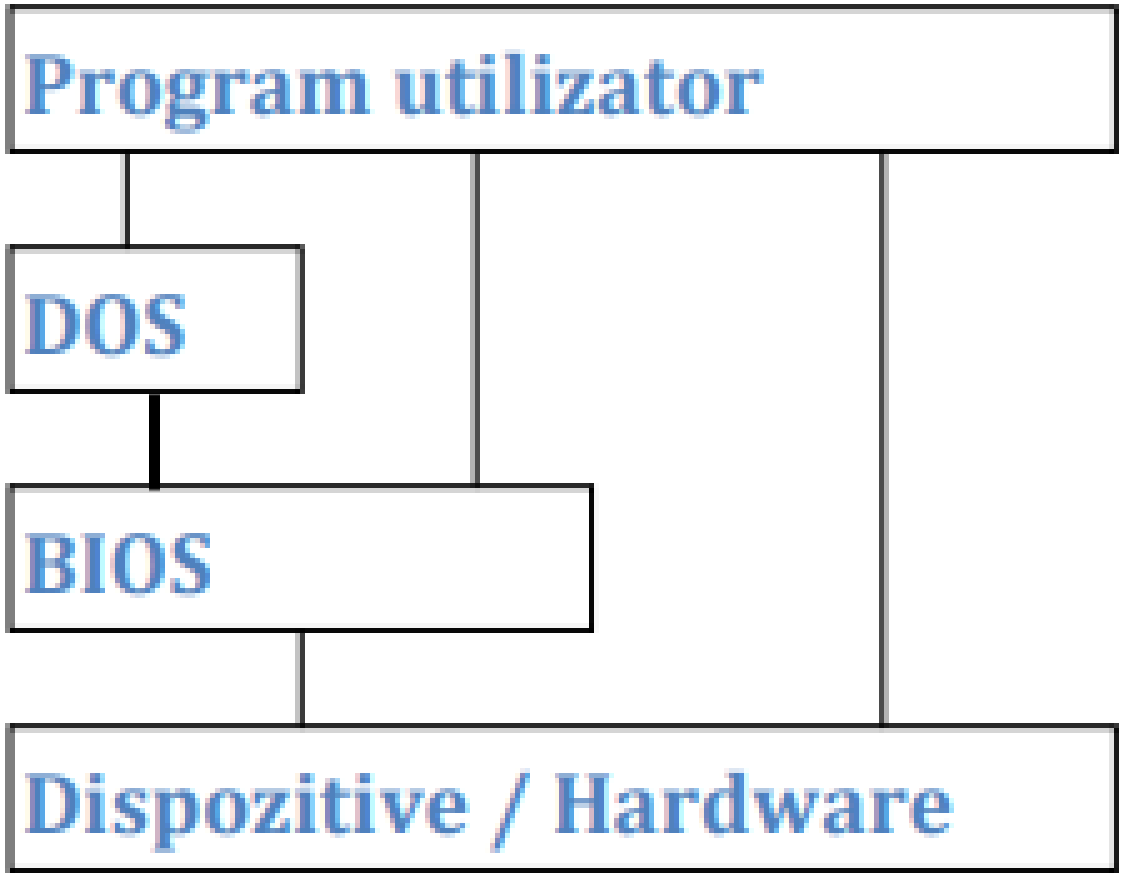
1. **Controlorul de deservire a apelurilor de întrerupere – Int;**
2. **Controlorul de acces disrect la memorie – DMA;**
3. **Controlorul pentru formarea intervalelor de timp – Timer.**

Arhitecturile avansate ale sistemelor de calcul mai utilizează și Arbitrul Magistralei de Sistem pentru a accelera procesul de transfer de date dintre diferite componente ale acestuia: RAM – RAM, RAM – HDD, HDD – HDD, etc.

Structura model a memoriei PC

F0000H	ROM - zonă sistem
C0000H	ROM BIOS
B0000H	Buffere video
A0000H	Buffere video
640K (0FFFFH)	COMMAND.COM – porțiunea tranzientă (programele executate o pot șterge).
	Zonă disponibilă pentru utilizare.
	COMMAND.COM – porțiunea rezidentă
	Fișierele de sistem: IO.SYS, MSDOS.SYS
00500H	Zonă comunicație DOS
00400H	Zona de date BIOS
00000H	Tabela vectorilor de întrerupere

Modelul de interațiune Program <-> DOS <-> BIOS <-> Hardware



DOS – Disc Operation Systems & Directory Operation Systems

BIOS – Basic Input / Output Systems

2.9. Controloare specializate: Int, DMA.

Întreruperea de Program:

1. **Hardware –** inițiate de dispozitivele perifecrice pentru a interveni în cazuri excepționale. Sunt inițiate prin semnale fizice IRQ.
2. **Software –** inițiate de utilizator în ptocesul dezvoltării produselor program pentru a inița un proces de transfer de date, comandă sau control al acestuia. Sunt inițiate prin instrucțiunea în limbajul Assembel INT.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

2.9. Controloare specializate: Int, DMA.

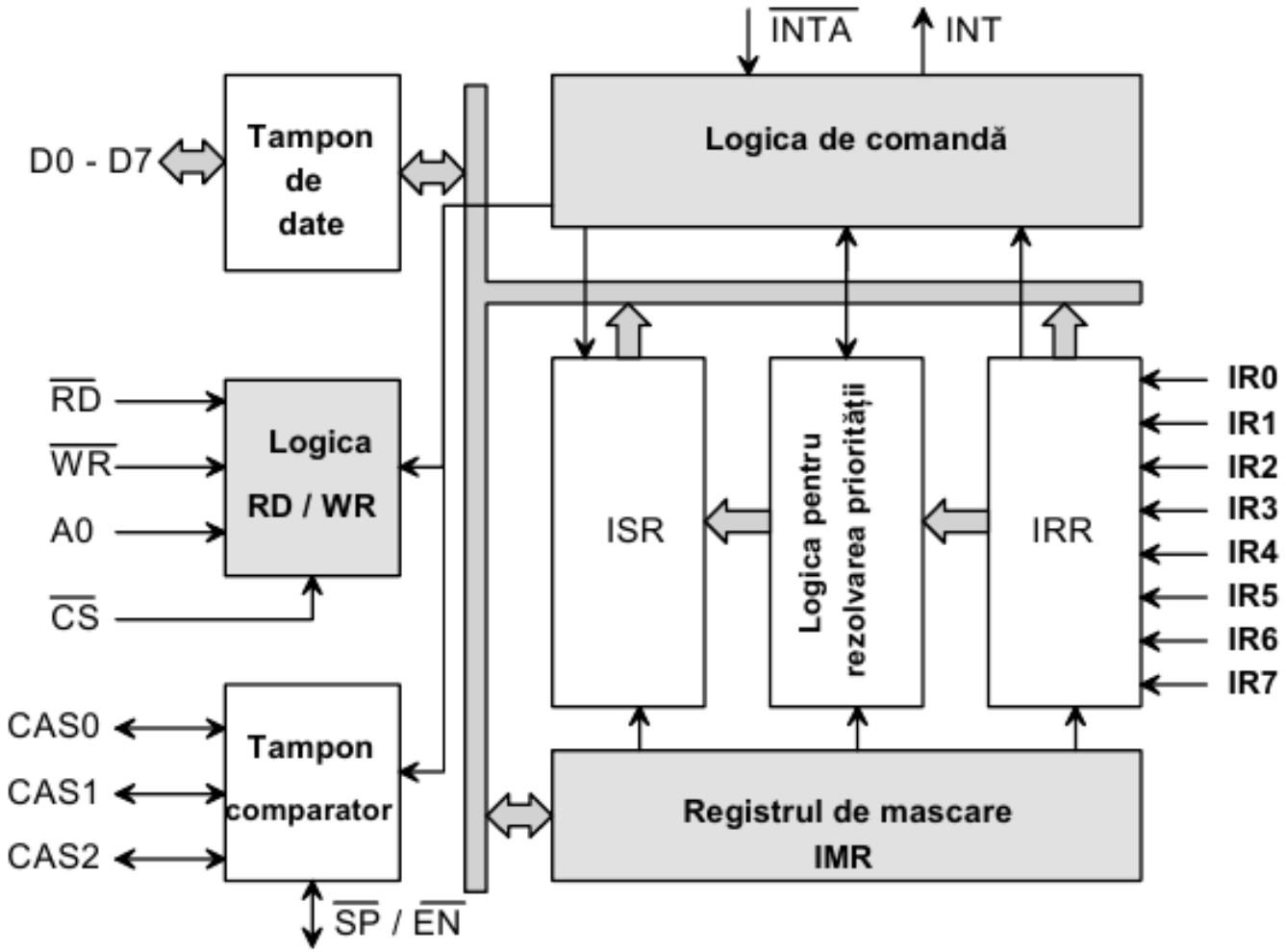
Întreruperile de Program Hardware, sunt evenimente **asincrone** față de codul aflat în execuție, care trebuie să capteze atenția procesorului pentru a fi rezolvate imediat. Altfel, în timpul execuției unui program, dacă este generată o întrerupere, procesorul va **întrerupe firul normal de execuție după terminarea instrucțiunii curente** și va trata întreruperea respectivă.

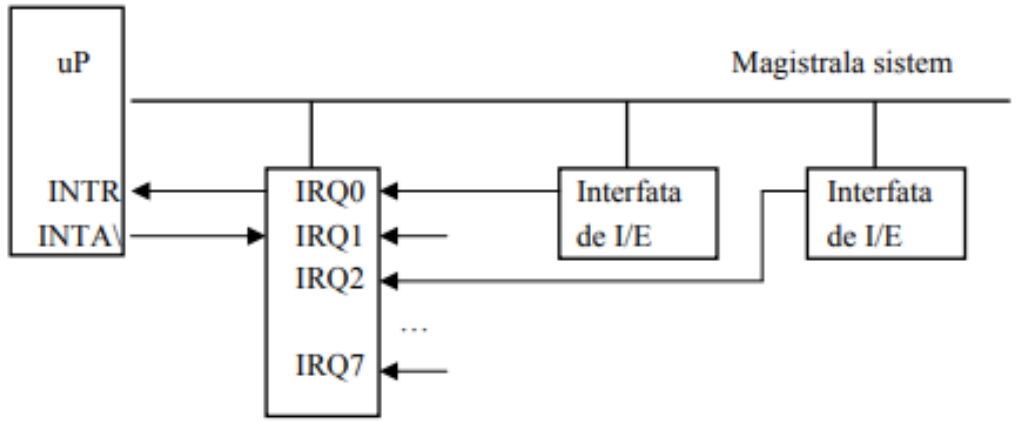
Pentru a determina dacă trebuie să trateze o întrerupere, procesorul verifică, după fiecare instrucțiune, linia de cereri de întreruperi (interrupt request sau **IRQ**). Dacă această linie este activă, atunci starea curentă a procesorului este salvată pe stivă și se transferă execuția către o rutină de tratare a întreruperii respective, după un mecanism foarte asemănător instrucțiunii **CALL**.

Tratarea întreruperilor se face de către funcții dedicate, numite rutine de tratare a întreruperilor (Interrupt Service Routine - **ISR**). Acestea sunt rutinele care fac parte din **BIOS**.

Prin **vector de întrerupere** se subînțelege adresa de memorie la care se afla începutul rutinei.

Controlorul de intreruperi I8259





IRQ0-IRQ7/15 De la Hardware

2.9. Controloare specializate: Int, DMA.

Întreruperi Hardware

IRQ	Prioritate	Dispozitivului hard rezervat
0	00	Timer
1	01	Tastatură
2	02	Extindere IRQ8 – IRQ 15
3	11	Port serial 2; Port serial 4; Comunicaţii SDLC; Comunicaţii BSC; LAN
4	12	Port serial 1; Port serial 3; Comunicaţii SDLC; Comunicaţii BSC; Comunicaţii vocale
5	13	Port paralel 2; Audio
6	14	Controller FDD
7	15	Port paralel 1
8	03	Ceas de timp real
9	04	Redirectare software la INT 0AH; Video; LAN
10	05	Rezervat
11	06	Rezervat
12	07	Rezervat mouse integrat
13	08	Coprocesor
14	09	Controller (IDE) Primar HDD
15	10	Controller Secundar HDD

Întrruperi Software

Principalele întreruperi BIOS sunt:

05h Se emite la apăsarea tastei PrintScreen; Instrucţiunea BOUND (ce verifică dacă valoarea unui index de tablou se află între limitele specificate) apelează şi ea această întrerupere dacă condiţiile verificate nu sunt îndeplinite.
10h Servicii de lucru cu sistemul video, în mod text şi în mod grafic.
11h Returnează lista echipamentelor BIOS instalate în sistem
12h Returnează dimensiunea memoriei RAM.
13h Pune la dispoziţie servicii de lucru cu harddisk-ul şi cu discheta.
14h Permite accesul la porturile seriale.
15h Pune la dispoziţie funcţii de acces la memoria extinsă, de citire a dispozitivelor de tip joystick etc.

Întrruperi Software

Principalele întreruperi BIOS sunt:

16h Această întrerupere oferă servicii de manipulare a tastaturii.
17h Această întrerupere oferă servicii de manipulare a imprimantei.
18h Activează interpretorul ROM BASIC.
19h Oferă servicii de încărcare a sistemului de operare.
1Ah Servicii legate de ceasul de sistem.
1Bh Această întrerupere este apelată la apăsarea combinaţiei de taste <CTRL/BREAK>. Ca efect al RTI corespunzătoare, în bufferul tastaturii se va pune CTRL-C ca următorul caracter. La citirea acestuia se va invoca întreruperea 23h, care termină execuţia programului curent şi redă controlul sistemului de operare.
1Ch Această întrerupere este apelată de 18.2 ori / secundă de RTI 8. Rutina de tratare a acestei întreruperi nu face nici o acţiune, lăsând posibilitatea utilizatorului de a scrie propria rutină de tratare. Aceasta este o întrerupere utilizator.

Întrruperi Software

Principalele întreruperi DOS sunt:

Principala întrerupere DOS este **21h**. Ea înmagazinează practic întreaga componentă BDOS a sistemului de operare DOS. Secţiunea următoare va fi destinată exclusiv acestei întreruperi.
20h Acesta este unul dintre apelurile care pot terminarea execuţia unui program.
25h Permite citirea fizică de pe disc de la o anumită locaţie de memorie, începând cu un anumit sector, într-o anumită locaţie de memorie.
26h Permite scrierea fizică pe disc dintr-o anumită locaţie de memorie, începând cu un anumit sector.
27h Termină execuţia programului curent lăsând rezidentă în memorie o parte sau întreg programul, astfel încât această zonă de memorie să nu fie suprascrisă de un alt program.
33h Această întrerupere grupează toate funcţiile necesare lucrului cu mouse-ul.

Transferul de date prin acces direct la memorie (DMA - Direct memory access):

Scopul DMA:

Pentru cresterea vitezei de transfer.
Pentru eliberarea procesorului de sarcina transferului.

Mod de implementare:

Circuit specializat pentru operatii de transfer – controlor DMA.
Controlorul este programat de microprocesor, dupa care efectueaza transferul intre interfata si memorie (sau invers), fara implicarea procesorului.

Fazele transferului:

Initializarea: - procesorul programeaza transferul unui bloc de date: (controlorul DMA este slave).

- Adresa zonei de memorie.
- Directia de transfer.
- Numarul de date transferate.

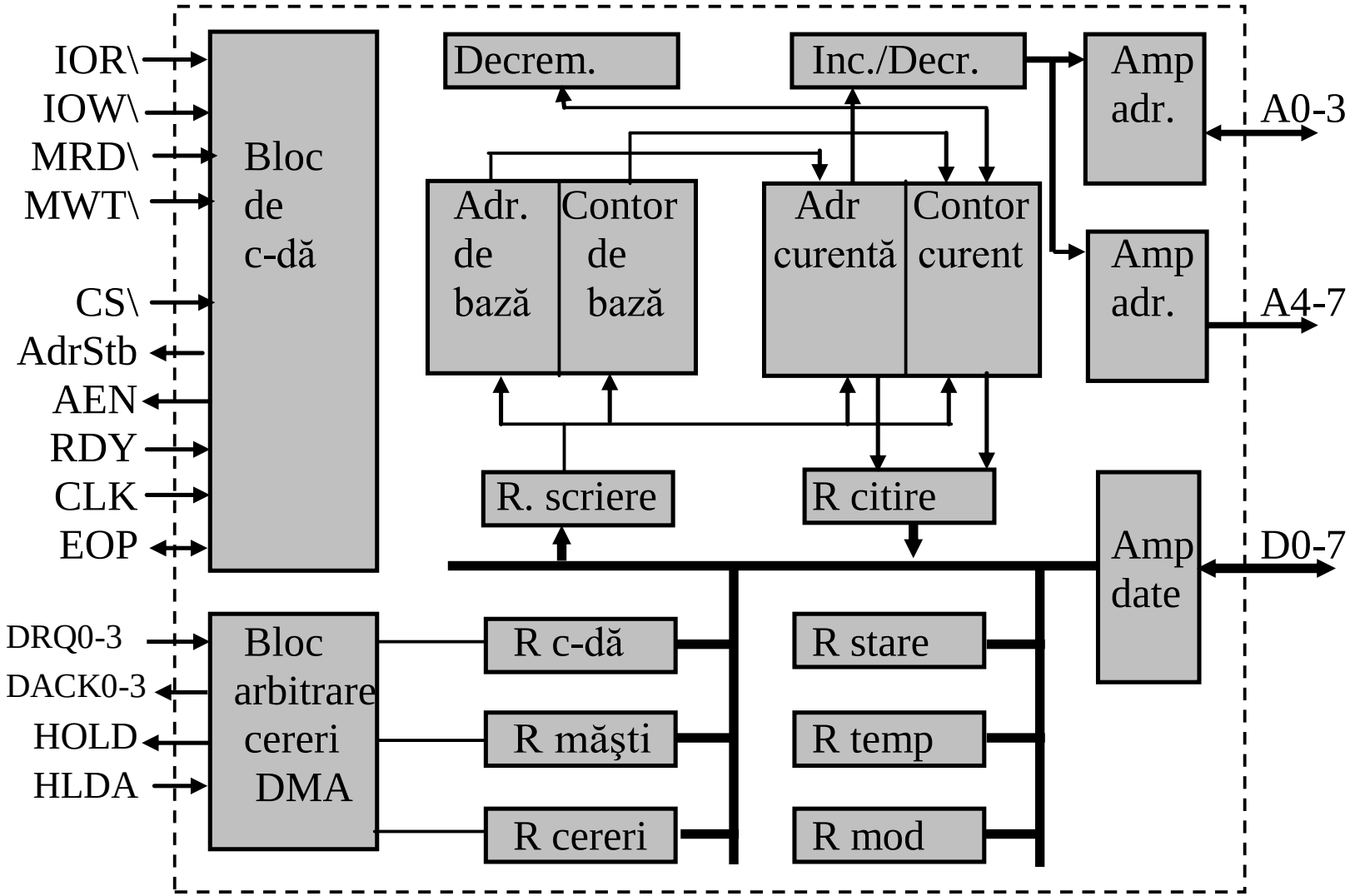
Transferul proproiu-zis: controlorul efectueaza transferul si asigura sincronizarea cu dispozitivul periferic (contr. DMA este master).

Finalizarea transferului: procesorul verifica corectitudinea transferului prin citirea starii controlorului DMA si a interfetei implocate (contr. DMA este slave).

Caracteristicile Controlorului DMA I8259 (DMA - Direct memory access):

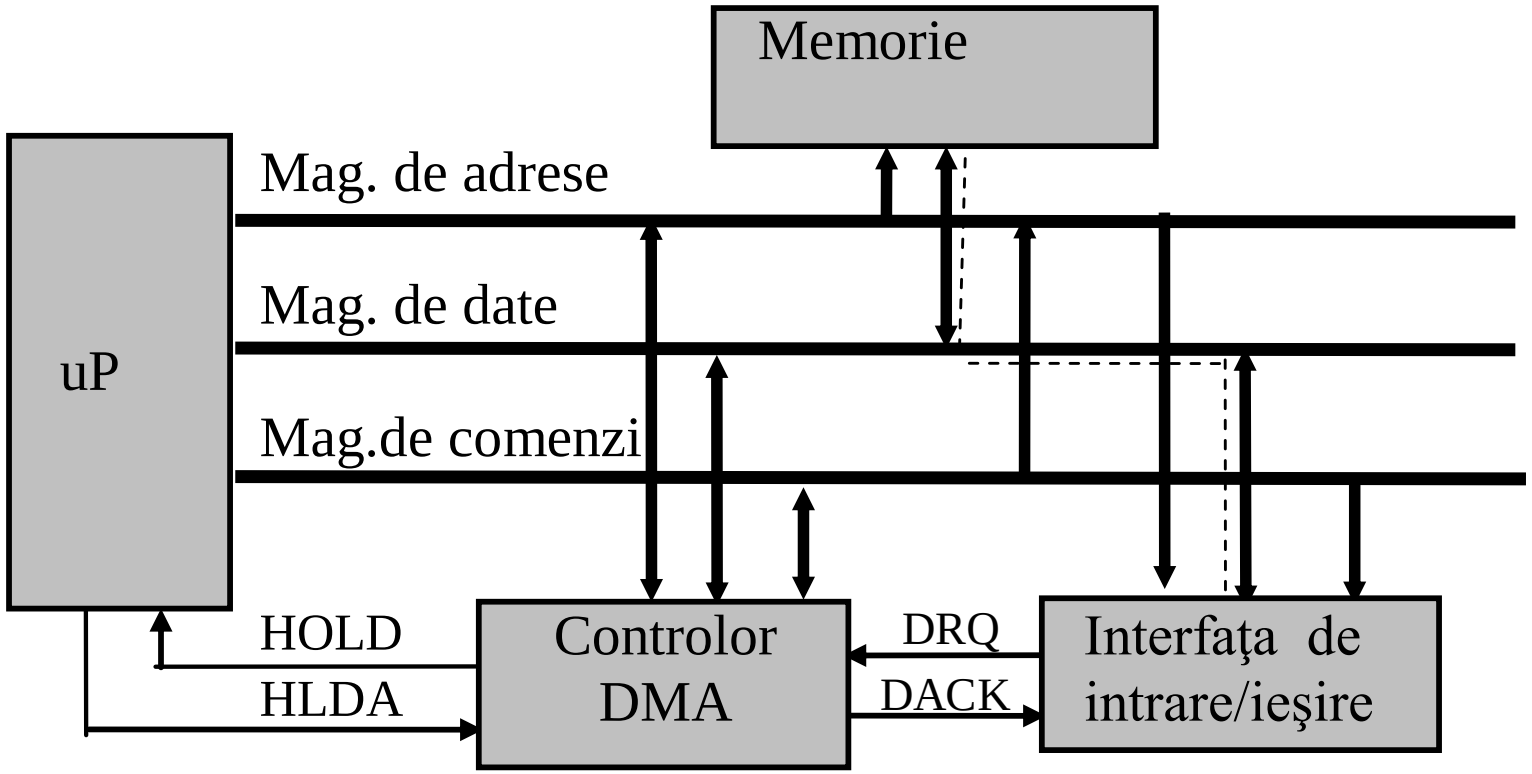
- Poate deservi 4 periferice independente (are 4 canale DMA).
- Viteza de transfer maximă.
- Mai multe controloare pot fi conectate în “cascadă” pentru a mări numărul de canale DMA disponibile, pînă la 16.
- Dimensiunea maximă a blocului de date transferat este de 64KB.
- Poate efectua transfer de tip memorie-memorie.
- Mai multe moduri de transfer:
  - Transfer singular,
  - Transfer pe blocuri de date (burst),
  - Transfer cu autoinitializare,
  - Transfer memorie-memorie.

2.9. Controloare specializate: Int, DMA.



Schema internă a controlorului DMA:

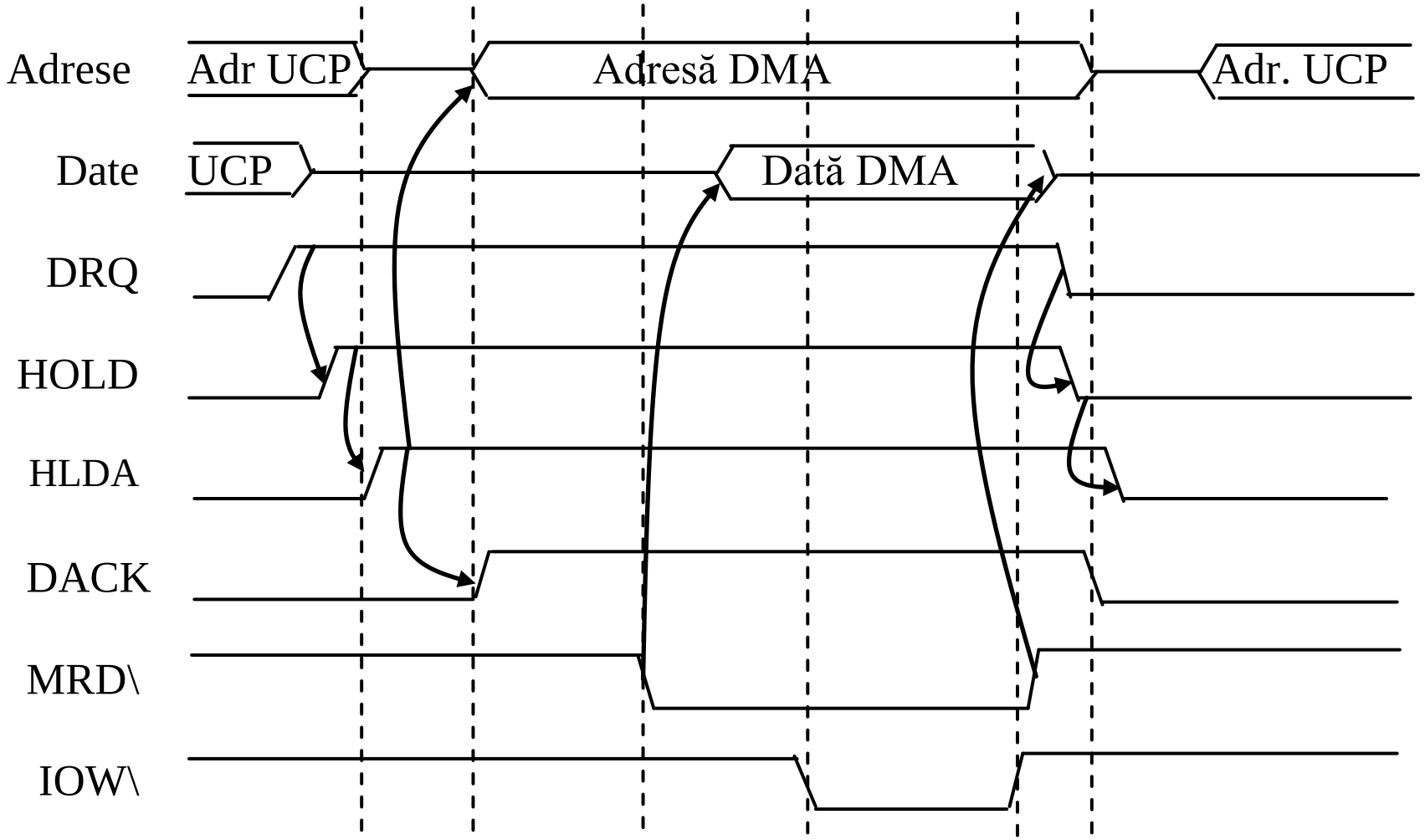
Conectarea controlorului DMA la MS:



Procesul de deservire a unui apel Regim DMA:

1. O interfata solicita un transfer prin activarea unui semnal DRQi catre controlorul DMA.
2. Controlorul DMA verifica daca transferul este valid (programat) si nu este altul mai prioritar in desfasurare.
3. Daca cererea este valida se solicita accesul pe magistala prin activarea semnalului HOLD catre procesor.
4. Procesorul termina executia ultimului ciclu de transfer dupa care isi invalideaza amplificatoarele de magistrala si semnalizeaza cedarea magistralei prin activarea semnalului HLDA (Hold Acknowledge) catre controlor.
5. Controlorul DMA isi activeaza amplificatoarele de adrese si de comanda si genereaza adesa locatiei de memorie unde/de unde se face transferul; activeaza semnalul DACK (DMA acknowledge) catre interfata.
6. Controlorul DMA genereaza semnal de citire memorie (MRDC) sau citire interfata (IORC)
7. Se genereaza data de catre memorie sau de interfata.
8. Controlorul DMA genereaza semnal de scriere interfata (IOWC) sau de scriere memorie (MWTC); astfel se realizeaza transferul de la memorie la interfata sau invers.
9. Controlorul DMA dezactiveaza semnalele de comanda si apoi cele de adresa.
10. Controlorul DMA dezactiveaza semnalul HOLD, cedand astfel procesorului controlul magistralei.
11. Procesorul dezactiveaza semnalul HLDA, isi reactiveaza amplificatoarele de magistrala si preia controlul magistralei.

Diagrama de timp a procesului de transfer date RAM – HDD în Regim DMA:



Distribuirea canalelor DMA și prioritarea acestora pentru diferite dispozitive periferice:

Canal DMA	Dimensiune	Asignare
DMA0	8 sau 16 biţi	Audio
DMA1	8 sau 16 biţi	Audio sau LAN
DMA2	8 sau 16 biţi	FDD
DMA3	8 sau 16 biţi	Port ECP/EPP
DMA4	16 biţi	Canal de legătură
DMA5	16 biţi	Neasignat
DMA6	16 biţi	Neasignat
DMA7	16 biţi	ISA IDE

Caracteristici calitative și cantitative ale metodelor de transfer date:

Transfer	Complexitate	Cost	Viteza	Implicare procesor
Prin program	Mica	Mic	Mica	totala
Prin intreruperi	Medie	Mediu	Medie	Mare
Prin DMA	Mare	Mare	Mare	Mica
Prin procesor de I/E	Foarte mare	Foarte mare	Mare	Foarte mica

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Tema 2. Microprocesoare și Microcontrolere. Limbajul de programare Assembler :

1. Clasificarea MPU și MCU.
2. MPU cu arhitectura x86.
3. MCU Intel, AVR, MicroChip.
4. Limbajul de programare Assembler. Setul de instrucțiuni x86.
5. Medii pentru dezvoltarea, compilarea și testarea produselor program.
6. Memoria internă a sistemelor de calcul.
7. Magistrala de sistem. Ciclul magistralei de sistem.
8. Proiectarea memoriei.
9. Controloare specializate: Int, DMA, Timet.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Tematica disciplinei Arhitectura Calculatoarelor:

Tema 1. Introducere. Bazele fundamentale ale Arhitecturii Calculatoarelor;

Tema 2. Microprocesoare și Microcontrolere. Limbajul de programare Assembler;

Tema 3. Dispozitive pentru achiziția datelor;

Tema 4. Dispozitive pentru afișarea și imprimarea datelor;

Tema 5. Dispozitive pentru stocarea datelor.

UNIVERSITATEA TEHNICĂ A MOLDOVEI

Mulțumesc pentru atenție