

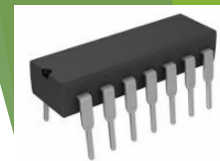
**CONCURSUL DE ROBOTICĂ „ROBOSMART”**

**EDITIA A II-A**

**MICROPROCESOARE, MICROCONTROLERE ȘI MICROSISTEME**

# Clasificarea circuitelor integrate

- ▶ **Small Scale Integration**, sau (SSI) - Un circuit conține până la 10 tranzistori, sau câteva porți logice de tipul AND, OR, NOT.
- ▶ **Medium Scale Integration** or (MSI) - Un circuit conține între 10 și 100 tranzistoare sau de la 10 porți la câteva zeci și execută operații digitale ca însumare, decodare, numărare, multiplexare sau memorare de tip bistabil.
- ▶ **Large Scale Integration** or (LSI) - un circuit conține între 100 și 1,000 tranzistoare sau sute de porți și execută operații digitale tipice , ca unități aritmetice și logice, memorii , operații de intrare/ieșire.
- ▶ **Very-Large Scale Integration** or (VLSI) - un circuit conține între 1,000 și 10,000 tranzistoare sau mii de porți logice și execută operații de calcul ca procesoare, memorii de capacitate mare și dispozitive logice programabile.
- ▶ **Super-Large Scale Integration** or (SLSI) - un circuit conține între 10,000 și 100,000 tranzistoare și execută operații de calcul ca microprocesoare, microcontrolere, calculatoare sau dispozitive periferice.
- ▶ **Ultra-Large Scale Integration** or (ULSI) - un circuit conține peste 1 milion de tranzistoare fiind utilizat în calculatoare ca CPUs, GPUs, videoprocesoare, microcontrolere, FPGAs sau dispozitive periferice complexe.



# Microsistemul de calcul

Microsistem = Sistem de calcul bazat pe microprocesor sau pe microcontroler (*On Chip Computer*).

Structura de bază a unui *microsistem* cuprinde:

1. dispozitive/circuite periferice de intrare/input (I);
2. unitatea de memorie (M);
3. microprocesorul (format din automatul RALU - unitatea aritmetică și logică și regiștri și automatul UC - unitatea de comandă);
4. dispozitive/circuite periferice de ieșire/output (O).

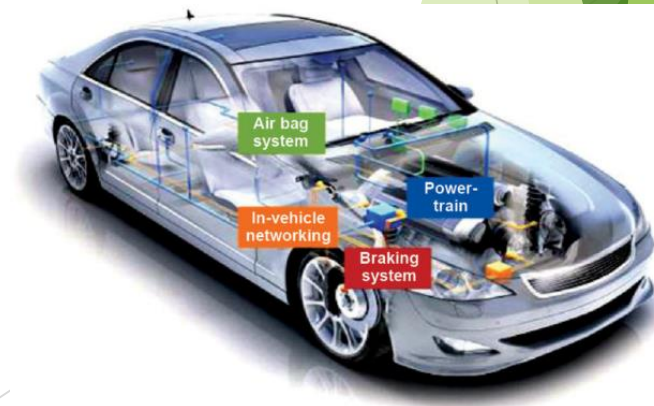
(desen)

Un **sistem embedded** (sistem “încorporat” sau “înglobat”) este un microsistem care face parte dintr-un sistem mai mare, cu componente mecanice sau electromecanice, constituind “inteligența” acestuia.

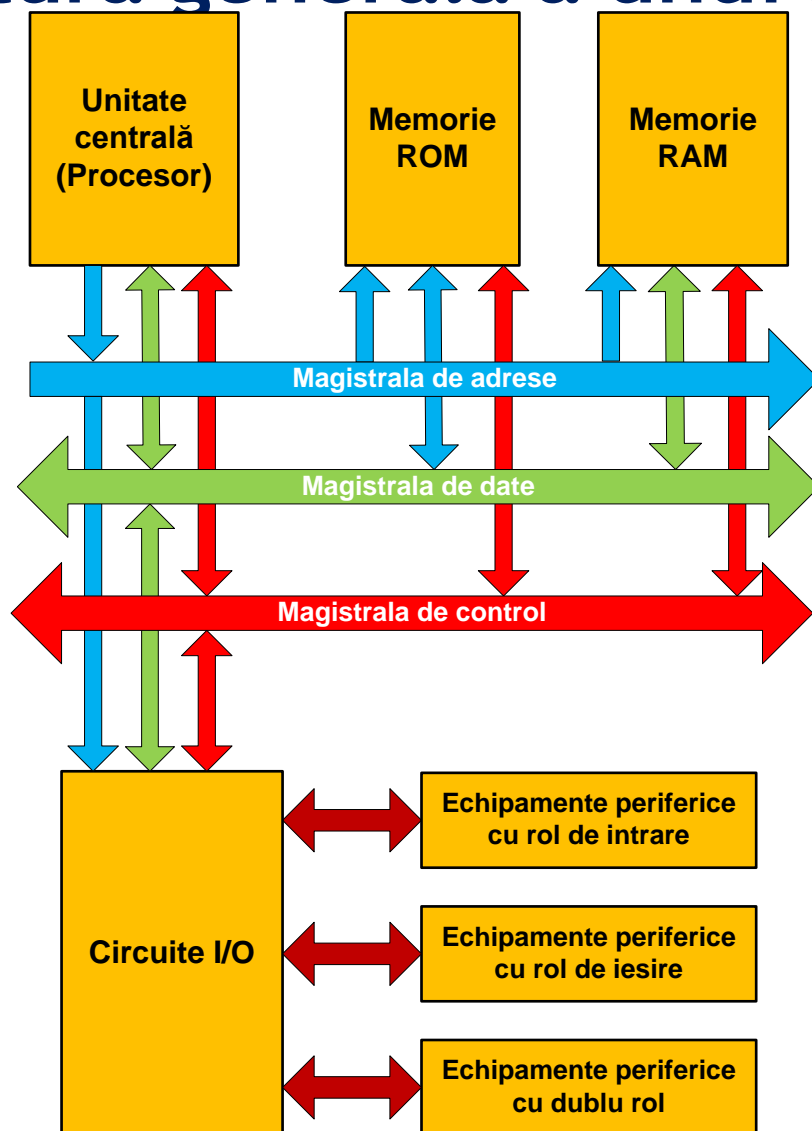
Sistemele embedded comandă și controlează multe aparate și dispozitive funcționale (ex: ceasuri digitale, aparate electrocasnice, televizoare, computerul de bord al automobilelor, roboți, etc).

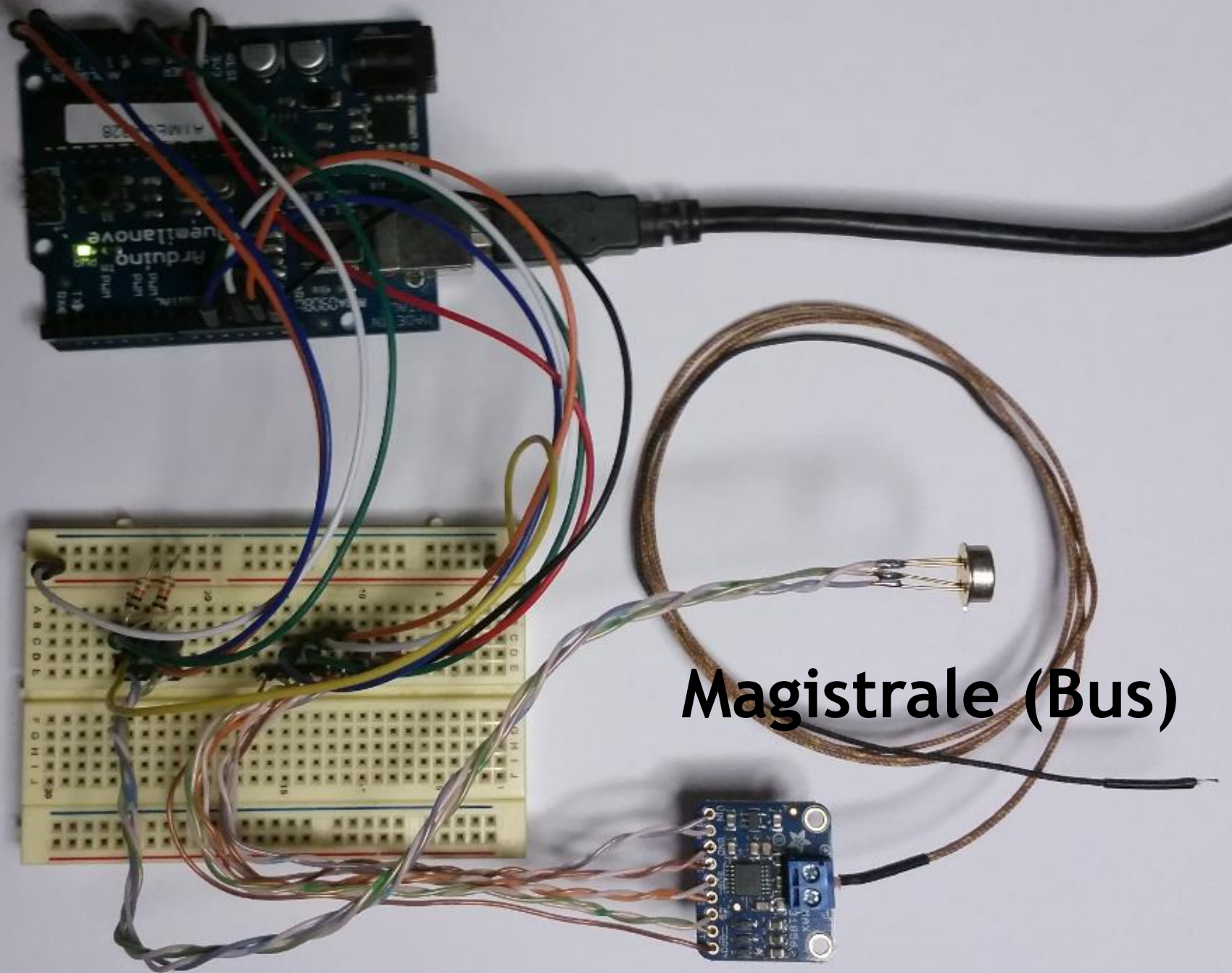
Caracteristile principale ale sistemelor embedded sunt:

- abilitatea de a executa o singură sarcină specifică
- rulare repetată un program
- reacție la evenimente în timp real
- siguranța în funcționare
- miniaturizarea
- viteza
- cost mic, consum redus



# Structura generală a unui sistem de calcul





**Magistrale (Bus)**

# Magistrale (Bus)

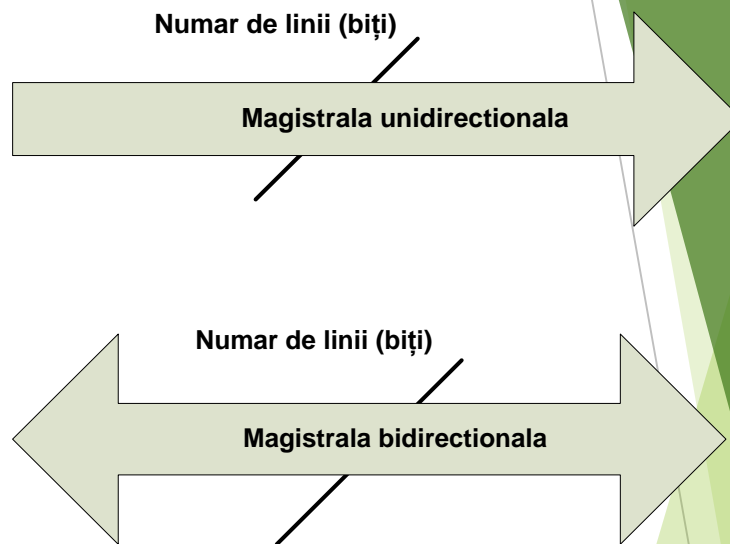
- Magistrala: un grup de linii electrice pe care se transmit semnale electrice digitale care au o aceeași funcție logică.

- Magistralele pot fi :

- **unidirecționale** - transmit semnalele electrice digitale într-un același sens
- **bidirecționale** - transmit semnalele electrice digitale în ambele sensuri

Într-un sistem de calcul se întâlnesc 3 categorii de magistrale:

- magistrala de adrese (unidirecțională)
- magistrala de date (bidirecțională)
- magistrala de comenzi sau control (bidirecțională - unele linii au o direcționalitate, altele altă direcționalitate)





## Memorii RAM 1065 de produse

Ordoneaza dupa:

Cele mai populare ▾

Produse:

60 pe pagina ▾

Tip afisare:



★★★★★ 7 review-uri

Memorie Corsair Vengeance  
LPX 8GB DIMM, DDR4, 2400  
MHz, CL 16, 1.2V, Black

în stoc

3 oferte disponibile

399<sup>99</sup> Lei (-16%)332<sup>57</sup> Lei

Adauga in Cos



★★★★★ 6 review-uri

Memorie Corsair Vengeance  
LPX 16GB (2x8GB) DIMM,  
DDR4, 3000MHz, CL15, 1.35V,

ultimele 3 produse

869<sup>99</sup> Lei (-15%)733<sup>42</sup> Lei

Adauga in Cos



★★★★★ 19 review-uri

Memorie HyperX Fury Black  
8GB, DDR4, 2400MHz, CL15,  
1.2V

în stoc

10 oferte disponibile

489<sup>99</sup> Lei (-28%)349<sup>99</sup> Lei

Adauga in Cos



★★★★★ 12 review-uri

Memorie RAM DIMM Corsair  
Vengeance LPX 16GB  
(2x8GB), DDR4 2400MHz,

în stoc

4 oferte disponibile

665<sup>99</sup> Lei

Adauga in Cos



★★★★★ 24 de review-uri

Memorie HyperX FURY Black  
4GB, DDR4, 2400MHz, CL15,  
1.2V

în stoc

9 oferte disponibile

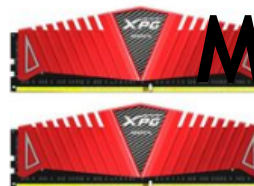


★★★★★ 7 review-uri

Memorie Corsair Vengeance  
LPX 8GB DIMM, DDR4,  
2400MHz, CL14, 1.2V, XMP

21%

21% vandut

409<sup>99</sup> Lei (-36%)

★★★★★ 6 review-uri

Memorie ADATA XPG Z1 Red,  
16GB (2x8GB) DDR4, 2400  
vezi variante: capacitate, latenta, fre...

în stoc

2 oferte disponibile

404<sup>99</sup> Lei (-38%)

☆☆☆☆☆ 0 review-uri

Memorie Corsair Vengeance  
LPX Black 16GB DDR4,  
3000MHz, CL16, Dual

în stoc

3 oferte disponibile

829<sup>99</sup> Lei (-25%)

Memoria

# Memoria

**Memoria** este un circuit digital care permite stocarea

și regăsirea de informații binare, folosind celule de memorare.

Conținutul circuitului de memorie adresat printr-o unică combinație binară aplicată pe liniile de adrese

poartă denumirea de **locăție de memorie**.

Totalitatea locațiilor de memorie formează circuitul de

memorie.

O locăție de memorie este formată dintr-un număr de

celule de memorie, câte una pentru fiecare bit memorat,

Organizarea circuitului de memorie este exprimată prin produsul neefectuat dintre numărul de locații și numărul de biți dintr-o locăție de memorie. Se utilizează următorii multipli pentru numărul de locații de memorie:  $2^{10} = 1k$ ,  $2^{20} = 1M$ ,  $2^{30} = 1G$ ,  $2^{40} = 1T$  Exemple:  $2^{14}$  (locații)  $\times$  8 (biți/locație) =  $16k \times 8 = 16ko = 16kB$ ,  $2^{22} \times 16 = 4M \times 16 = 64Mb$ .

Numărul de biți conținuți în locația de memorie

Pentru a adresa o memorie cu capacitatea de  $N$  locații sunt necesare  $n$  linii de adresă:  $n = \lceil \log_2 N \rceil$  biți.

Exemplu: O memorie cu organizarea  $1MB = 1Mo = 1M \times 8 = 2^{20} \times 8$  trebuie să fie adresată cu  $n = \log_2 2^{20} = 20$  biți (adresarea memoriei se face prin 20 linii de adresă) și prezintă 8 linii de date.

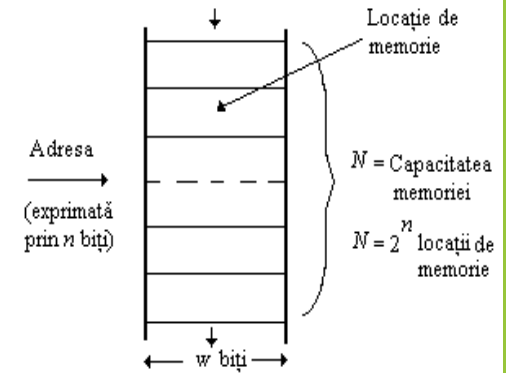
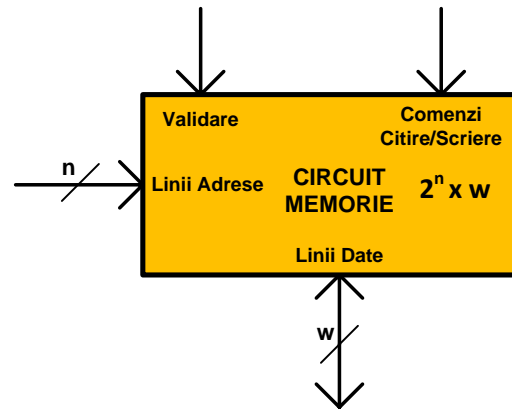
8 biți = octet (*Byte*);

Capacitatea circuitului de memorie se exprimă prin produsul efectuat dintre numărul de locații și numărul de biți dintr-o locăție de memorie și se măsoară în biți și multiplii lor:  $2^{10}b = 1kb$ ,  $2^{20}b = 1Mb$ ,  $2^{30}b = 1Gb$ ,  $2^{40}b = 1Tb$ . Exemple:

$2^{14}$  (locații)  $\times$  8 (biți/locație) =  $2^{17}b = 128kb$ ,  $4M \times 16 = 2^{22} \times 2^4b = 2^{26}b = 64Mb$

32 biți = dublu cuvânt (*double word*);

64 biți = dublu dublu cuvânt (*double double word*).





# Tipuri de circuite de memorie

**Memorii cu conținut fix** (conținutul nu se pierde, chiar dacă tensiunea de alimentare dispare)

**PROM (Programmable Read Only Memory).** Acest tip de memorie poate fi doar citit, iar datele pot fi scrise o singură dată, la momentul inscripționării

**EPROM (Erasable Programmable Only Memory).** poate fi ștearsă cu raze ultraviolete puternice și apoi reînscrișă cu alte valori

**EEPROM (Electrically Erasable Read Only Memory).** poate fi ștearsă și reprogramată electric. mai lentă decât memoria RAM, dar după decuplarea alimentării datele înscrise sunt reținute

**FLASH.** Este o categorie specială de memorie EEPROM, care se poate șterge și reînscrie bloc cu bloc, fără a fi nevoie de trimiterea datelor bit cu bit



**Memorii cu conținut temporar**

***Memorii statice (au nevoie doar de prezența tensiunii de alimentare pentru memorarea informației):***

**Memoria statică (RAM, SRAM).** celula de memorie construită pe baza unor componente electronice active (tranzistori). Avantajul cel mai important este viteza mare de acces, iar dezavantajele sunt: consum mare, capacitate mică, costul mai mare.

**NVRAM (Non Volatile RAM).** Este o memorie RAM nevolatilă, care reține datele și după decuplarea surselor de alimentare cu energie electrică, are consum mic fiind alimentată de la baterie sau acumulator



***Memorii dinamice (necesită și operația de reîmprospătare - refresh pe lângă prezența tensiunii de alimentare pentru memorarea informației):***

**Memoria dinamică (DRAM).** Celula de memorie e construită pe baza unui condensator care memorează „0” sau „1” prin prezența sau absența sarcinii electrice. Are nevoie de reîmprospătare. Avantajele sunt capacitatea mare, consumul mic și costul de asemenea mic. Dezavantajul este viteza mică de acces și de lucru mai mici.



**SDRAM (Synchronous RAM).** Este o memorie rapidă care lucrează sincron cu procesorul (operază

# Tabel cu adresele unor blocuri de memorie frecvent utilizate

Unitățile de măsură pentru memorie sunt exprimate în octeți:

$1k = 2^{10} = 1024$   
octeți;

$1M = 2^{20} = 1024k$ ;

$1G = 2^{30} = 1024M$ ,  $1T = 2^{40} = 1024G$



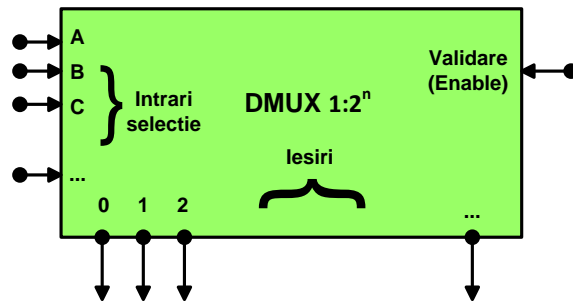
Adresa memoriei în hexa-zecimal	Mărimea memoriei în octeți și kocteți
0000	0
000F	16
0010	16+1
00F0	240
00FF	256
0100	256+1
01FF	512
0200	512+1
0300	768+1
0400	1024+1=1k+1
0800	2k+1
0C00	3k+1
1000	4k+1
2000	8k+1
3000	12k+1
4000	16k+1
8000	32k+1

Adresa memoriei în hexa-zecimal	Mărimea memoriei în octeți și kocteți
FFFF	64k
1 0000	64k+1
1 FFFF	128k
2 FFFF	192k
3 FFFF	256k
7 FFFF	512k
8 FFFF	576k
9 FFFF	640k
A FFFF	704k
B 0000	704k+1
B FFFF	768k
C 0000	768k+1
C FFFF	832k
D 0000	832k+1
D FFFF	896k
E 0000	896k+1
E FFFF	960k

Adresa memoriei în hexa-zecimal	Mărimea memoriei în octeți și kocteți
F 0000	960k
F FFFF	1024k=1M
10 0000	1M+1
1F FFFF	2M
20 0000	2M+1
2F FFFF	3M
30 0000	3M+1
7F FFFF	8M
80 0000	8M+1
FF FFFF	16M
0800 0000	128M+1
0FFF 0000	256M
1000 0000	256M+1
2000 0000	512M+1
4000 0000	1000M+1=1G+1
8000 0000	2G+1
FFFF FFFF	4G

# Structura circuitului de memorie

În structura unui circuit de memorie apar circuitele de tip multiplexor și demultiplexor.

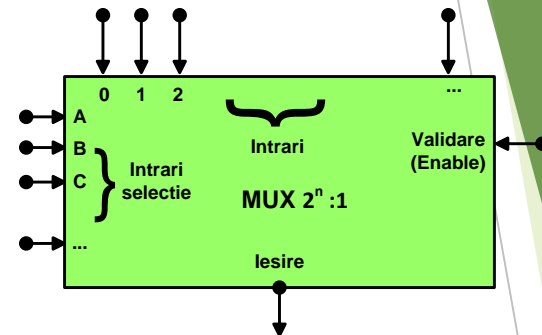


Demultiplexorul este un circuit digital care distribuie semnalul digital aplicat pe o intrare (denumită validare - enable) și îl transferă către una dintre ieșirile circuitului, identificată prin intermediul semnalelor digitale aplicate intrărilor de selecție.

O altă interpretare a funcționării este aceea de a identifica o combinație de semnale binare aplicate intrărilor de selecție prin activarea corespunzătoare a uneia dintre ieșiri, atât timp cât circuitul este validat.

DMUX-ul are rol de selector sau concentrator, respectiv de identificator de coduri binare.

Notăția circuitului DMUX indică raportul de distribuție 1:2<sup>n</sup> corespunzător circuitului.



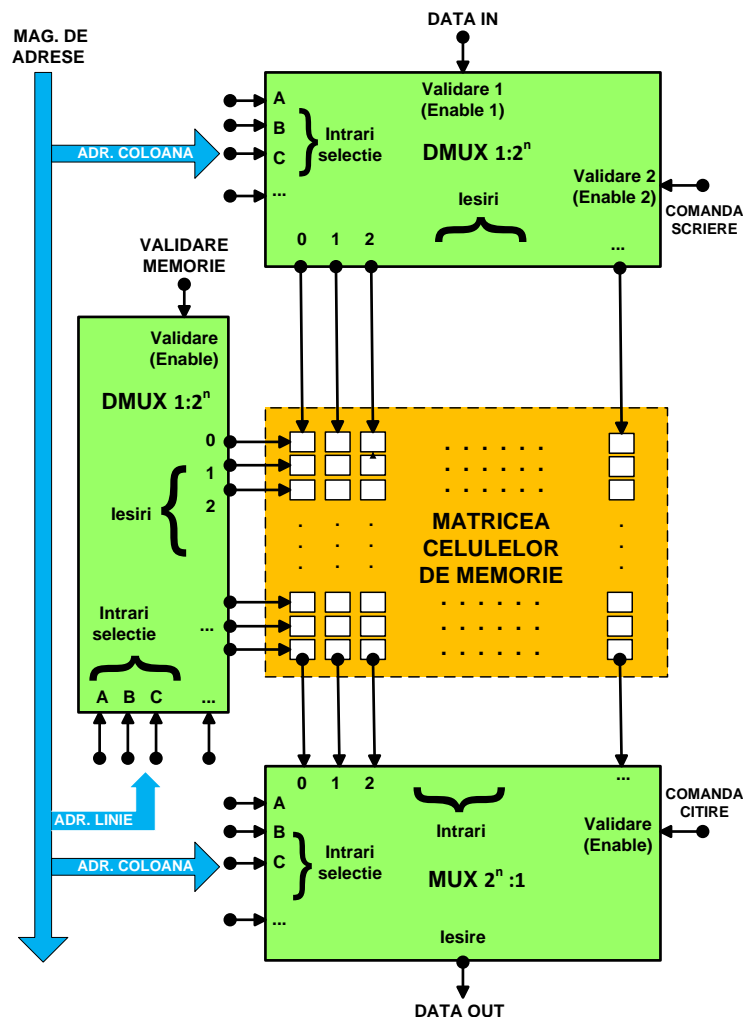
Multiplexorul este un circuit digital care selectează semnalul digital aplicat pe o intrare identificată prin intermediul semnalelor digitale aplicate unor intrari de selecție și îl transferă către ieșirea unică a circuitului, atât timp cât circuitul este validat.

MUX-ul are rol de selector sau de concentrator de semnale digitale.

Notăția circuitului MUX indică raportul de selecție 2<sup>n</sup>:1 corespunzător circuitului.

# Memoria - Schema bloc - Funcționare I

Schema bloc a unei memorii cu organizare  $2^n \times 1$



Asupra circuitului de memorie se pot efectua 2 tipuri de operații:

## 1. Citirea unei locații prin acțiunile:

- transmiterea adresei locației căutate pe magistrala de adrese către memorie; o parte a adresei este folosită de DMUX-ul răspunzător de accesarea liniei din matricea celulelor de memorie, iar cealaltă parte de MUX-ul ce selectează coloana pe care se află celula căutată;
- aplicarea comenzii de citire care conduce la apariția datei căutate la linia DATA OUT.

Operația de citire se derulează în mod similar atât la memorii ROM, cât și la memorii RAM.

## 2. Scrierea unei locații prin acțiunile:

- transmiterea adresei locației ce urmează a fi scrisă pe magistrala de adrese către memorie; o parte a adresei este folosită de DMUX-ul răspunzător de accesarea liniei din matricea celulelor de memorie, iar cealaltă parte de DMUX-ul ce selectează coloana pe care se află celula căutată;
- aplicare datei ce urmează a fi scrisă pe linia DATA IN;
- aplicarea comenzii de scriere care provoacă memorarea datei de intrare în locația adresată.

Operația de scriere este proprie memoriilor RAM.

Totuși, și în memoriile ROM se poate realiza "operația de scriere", dar această poartă denumirea de programare și se realizează în condiții speciale.

Duratele de execuție ale operațiilor de citire, scriere conduc la un parametru important al circuitelor de memorie - timpul de acces. Acest timp este legat de tehnologia de realizare a circuitelor, tehnologia ce utilizează tranzistoare bipolare fiind cea care oferă timpii cei mai mici (memoriile cele mai rapide). Totuși, tehnologia bipolară consumă multă energie și nu

# Memoria - Schema bloc - Funcționare II

Tipul celulelor de memorie folosite în matricea celulelor de memorie diferă, funcție de circuitul de memorie.

La memoriile ROM realizate cu tranzistori bipolari, celulele de memorie sunt de fapt fuzibile (asemănătoare siguranțelor electrice), care se ard la programare, procesul fiind ireversibil.

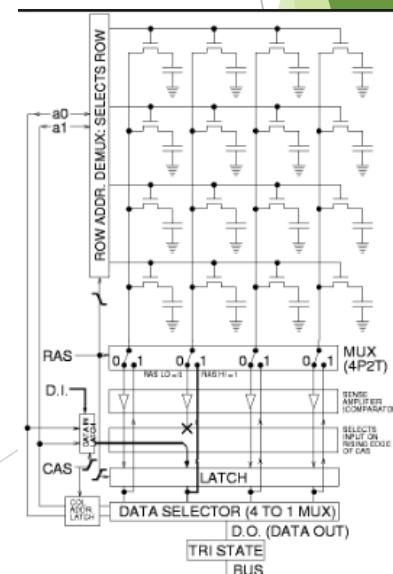
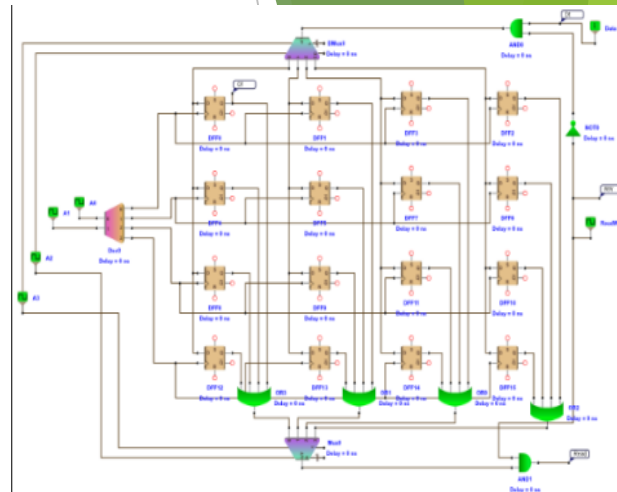
La memoriile ROM realizate cu tranzistori unipolari, celulele de memorie se formează prin sarcina electrică aplicată între 2 terminale speciale, prin diferența de potențial care apare materializându-se efectul de memorie. La memoriile ROM uniplare, este posibilă ștergere și reprogramarea acestora. Operațiile de ștergere se pot face prin expunere la raze UV (la anumite tipuri de memorii ROM) sau prin aplicarea de semnale electrice la ROM-urile de tip flash.

La memoriile SRAM (realizate cu tranzistori bipolari sau unipolari), celulele de memorie sunt bistabili sau latch-uri, realizate cu 4 - 6 tranzistori bipolari sau unipolari, elemente care au proprietatea de memorare cât timp se aplică tensiunea de alimentare, fără a mai fi nevoie de acțiuni suplimentare asupra circuitului de memorie.

În memoriile DRAM, celulele de memorie sunt condensatori care însoțesc tranzistorii unipolari. Aceștia au capacități foarte mici ( $<1\text{pF}$ ) și au pierderi mari. Stocarea sarcinii electrice pe armăturile acestora, de fapt ceea ce produce și efectul de memorare prin diferența de potențial care apare, este o sarcină dificilă. La aceste circuite se impune efectuarea repetitivă unei acțiuni suplimentare - reînprospătarea sau refresh-ul prin care se reîncarcă sarcina electrică pierdută la armături. Operația de refresh este suplimentară în raport cu operațiile de citire și scriere și micșorează timpul de disponibilitate al memoriei pentru operațiile cu procesorul. Pentru o funcționare corectă a memoriilor DRAM se impune atât menținerea tensiunii de alimentare, cât și efectuarea ritmică a refresh-ului.

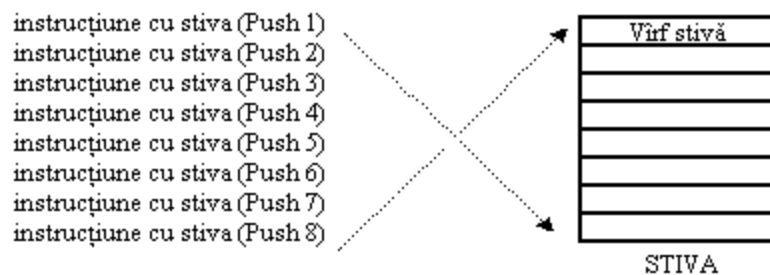
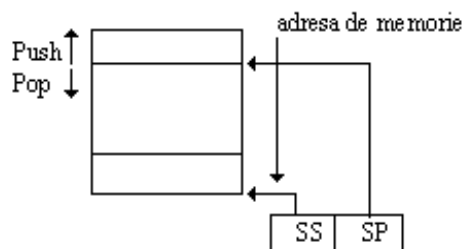
La memoriile DRAM o celulă de memorie folosește un singur tranzistor unipolar. Prin diferența numărului de tranzistori folosiți pentru formarea unei celule de memorie rezultă eficiența mult mărită a memoriilor DRAM față de SRAM.

Comparativ cu toate tehnologiile actuale de fabricare ale memoriilor, circuitele DRAM sunt cele mai avantajoase atât din punct de vedere al prețului de cost (pentru 1 bit), cât și din punct de vedere al densităților obținute pe suprafața de siliciu (număr de tranzistori/unitatea de suprafață).



# Stiva

- ▶ Stiva este o zonă de memorie a cărei dimensiune poate, după tipul procesorului, să crească odată cu adresa de memorie sau să scadă odată cu adresa de memorie. Caracteristica sa este că datele sale pot fi citite doar strict în ordinea inversă în care au fost scrise.
- ▶ Stiva este implementată în memorie și se adresează prin registrul de stivă (RS) care conține baza stivei și registrul pointerului de stivă (SP).
- ▶ Exemplu:
- ▶ În cazul stivei care crește odată cu scăderea adresei de memorie (crește “în sus”), încărcarea unui cuvânt în stivă se face prin decrementarea registrului SP cu o locație și scrierea cuvântului dorit să fi reținut în stivă la noul vârf al stivei. Scoaterea unui cuvânt din stivă este realizată prin copierea lui din vârful stivei și incrementarea registrului SP cu o locație.
- ▶ Stiva conține de obicei adresa de revenire în programul apelant, indicatori, parametri.







# ► PROCESORUL

# Evoluția microprocesoarelor I

## LEGEA LUI MOORE

- În 1965, Gordon Moore, co-fondator al Corporației Intel, a prezis că *“Numărul de tranzistori și rezistori pe un singur chip se va dubla la fiecare 18 luni”* în legătură cu dezvoltarea tehnologiei semiconductoare digitale. Când Gordon Moore a făcut această faimoasă afirmație, în anul 1965, existau doar aproximativ 60 de tranzistori individuali pe un singur chip.

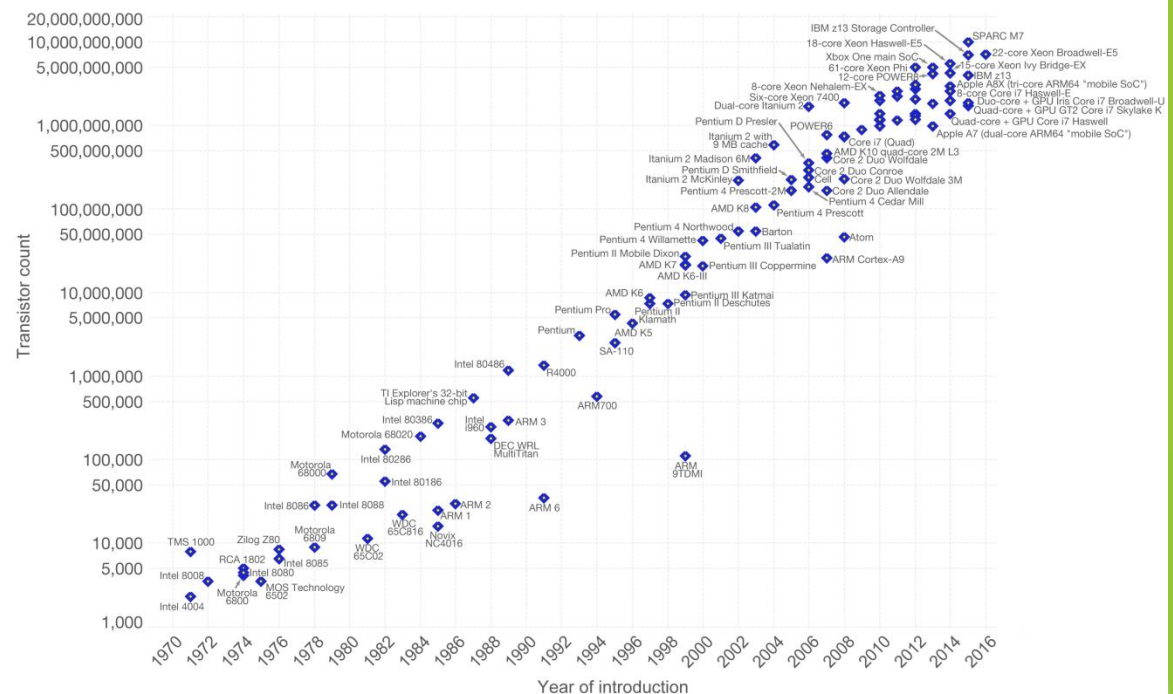
- În 2011, Corporația Intel a reușit să plaseze **1.2 Miliarde de tranzistori** individuali în microprocesorul *Quad-core i7-2700K Sandy Bridge* de 64 de biți care operează la aproape 4GHz, iar numărul de tranzistori pe un singur chip continuă să crească, la fel și viteza noilor microprocesoare și microcontrolere.

- Astăzi, după KabyLake (generația 7) a urmat CofeeLake (generația 8) ambele în tehnologie de 14 nm (dimensiunea unui tranzistor). Aceasta înseamnă un număr de 2 ori mai mare de tranzistori pe capsulă față de 2011...

## Moore's Law – The number of transistors on integrated circuit chips (1971-2016)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are strongly linked to Moore's law.

Our World  
in Data



# Evoluția microprocesoarelor I

[Products](#) [Solutions](#) [Support](#)



USA (English) [Sign In](#)



[Intel® Core™ Processors](#)

[Features and Performance](#)

[Related Technologies](#)

[Shop Systems](#)

[Related Materials](#)

Computers with the new 9th Generation Intel® Core™ desktop processors are packed with performance for mainstream and competitive gamers. With up to 8 cores, 16 threads, 5.0 GHz, and 16 MB cache the 9th Generation Intel® Core™ desktop processors are built for gaming. Overall system performance is boosted up to 15%<sup>1</sup> vs. 8th Gen Intel® Core™ processors and you can Game + Stream + Record with up to 40%<sup>2</sup> more FPS on Total War: WARHAMMER II vs. a 3 yr. old PC.

[9th Gen Intel® Core™ S-series family brief >](#)

[View all Intel® Core™ processors](#)

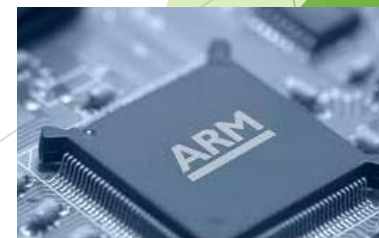
All Processors



# Evoluția microprocesoarelor II

Familiile cele mai reprezentative de microprocesoare, prezentate în ordinea generațiilor apărute, au fost/sunt:

- ▶ 1971 - INTEL 4004 (tehnologie PMOS), 4 biți
- ▶ 1972 - INTEL 8008 (tehnologie NMOS), 8 biți
- ▶ 1974 - MOTOROLA 6800, 8 biți
- ▶ 1976 - ZILOG Z-80, 8 biți
- ▶ 1978 - INTEL 8086, 8088, 16 biți
- ▶ 1978 - MOTOROLA 6809, 8 biți
- ▶ 1979 - ZILOG Z8000, 16 biți
- ▶ 1979 - MOTOROLA 68000, 68020 (1984), 68030, 32 biți
- ▶ 1983 - ARM 32/64 biți -> Cea mai mare cantitate de procesoare produsa
- ▶ 1985 - INTEL 80286, 80386, 80486, 80586, 32 biți
- ▶ 1990 - RS/6000 (IBM RISC) 32/64 biți
- ▶ 1995 - INTEL Pentium I, II, III, IV, ....., etc., 32 /64 biți.
- ▶ 1996 - Atmel AVR
- ▶ 2010 - I3, I5, 2009 - I7, 2018 - I9 (direct CoffeLake) ...



# Evoluția microprocesoarelor III

De ce (în ce scop) și când au apărut microprocesoarele?

În anii 1970, M.E. Hoff de la Intel a reușit înlocuirea a 11 circuite tradiționale cu numai 2-3 circuite, prin folosirea conceptului de **logică programată** -> primul microprocesor pe 4 biți INTEL 4004

Care au fost condiționările tehnologice ale apariției microprocesoarelor?

Apariția tehnologiei “circuitelor integrate pe scară largă” (LSI) în anii 1970 (sute până la mii de porți pe un chip).

Ce avantaje aduc microprocesoarele?

Sistemul de calcul dezvoltat în jurul unui microprocesor își **realizează funcțiile pe baza unui program propriu**.

Un singur chip (microprocesorul) înlocuiește mai multe circuite mai simple, **reducând costurile și complexitatea sistemului pe ansamblu**.

Câte porți logice conține un microprocesor?

Până la zeci de milioane de porți logice în procesoarele complexe.

# Producători actuali de microprocesoare

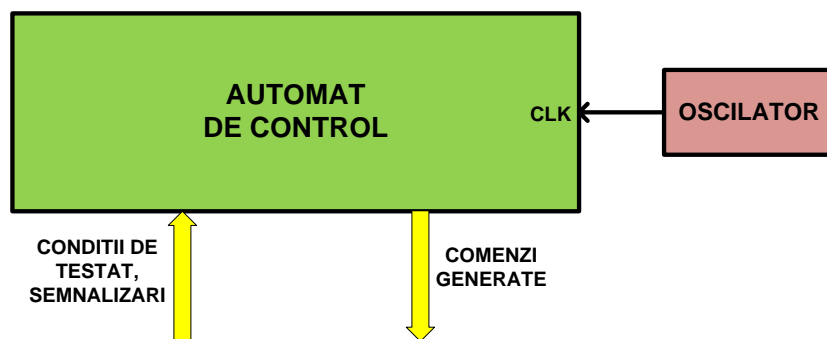
- ▶ INTEL
- ▶ ST Microelectronics
- ▶ MICROCHIP
- ▶ TEXAS INSTRUMENTS
- ▶ Freescale Semiconductor
- ▶ HITACHI
- ▶ AMD



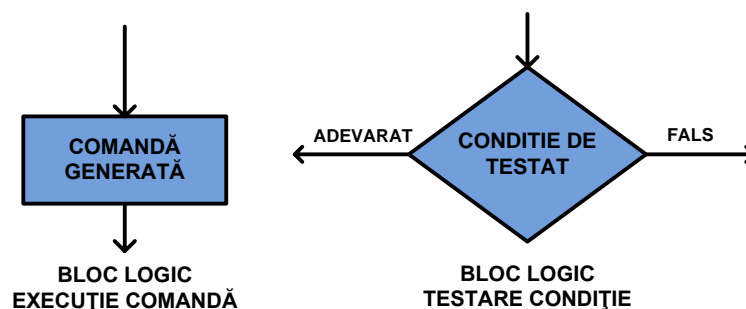


# Automate de control

Un automat de control reprezintă un ansamblu de circuite digitale (structură digitală) capabil să evolueze în timp, trecând prin stări bine definite, în ritmul unui semnal de ceas provenit de la un oscilator, în conformitate cu o schemă logică sau algoritm, având capacitatea de a interpreta valorile logice ale unor semnale digitale care i se aplică (condiții de testat sau diverse semnalizări) și, funcție de acestea și de algoritmul implementat (stările prin care trece), să genereze comenzi în consecință sub formă de semnale digitale.



Schemele logice sunt alcătuite din blocuri logice elementare de tip execuție comenzi, respectiv testare condiție:



Un automat este capabil să execute blocurile logice elementare care apar în schemele logice sau algoritmi, indiferent de modul lor de interconectare. Dimensiunea fizică a automatului este legată de mărimea schemei logice pe care trebuie să o implementeze.

# Structura unui automat de control

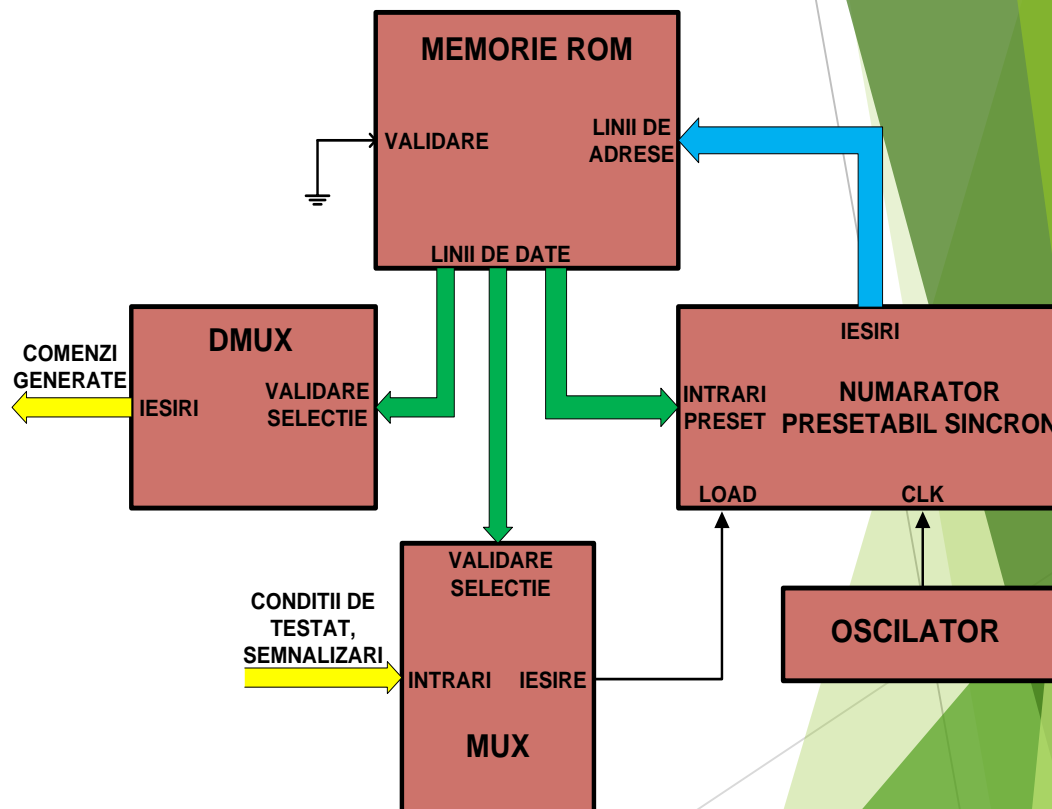
Un automat de control este organizat în jurul unei memorii ROM, ale cărei locații conțin secvențe de biți care transcriu blocurile din schema logică implementată. Fiecare locație de memorie conține implementarea binară a unui singur bloc logic elementar din algoritm.

Parcursul schemei logice se face prin adresarea memoriei, realizată prin intermediul unui numărator cu dublu rol: numărare, respectiv presetare (încărcare cu o valoare de la care poate să reînceapă procesul de numărare) și care este acționat de un semnal de ceas provenit de la un oscilator.

Blocurile logice de execuție comenzi se realizează prin DMUX-ul prezent în schemă, comandat de un câmp de date corespunzător de la memoria ROM.

Blocurile logice de testare a condițiilor sau a semnalizărilor primite sunt implementate prin MUX-ul din figură. Funcție de valoarea logică a condiției testate, ieșirea acestui circuit comandă funcția de numărare la numărator, specifică blocurilor logice de execuție comenzi sau a blocurilor de test pentru situația de condiție neîndeplinită, respectiv funcția de încărcare paralelă - presetare pentru blocurile de test în situația condiției testate îndeplinite.

Se pot face astfel parcurgeri liniare sau ramificate în schemele logice.



# Microprocesorul I

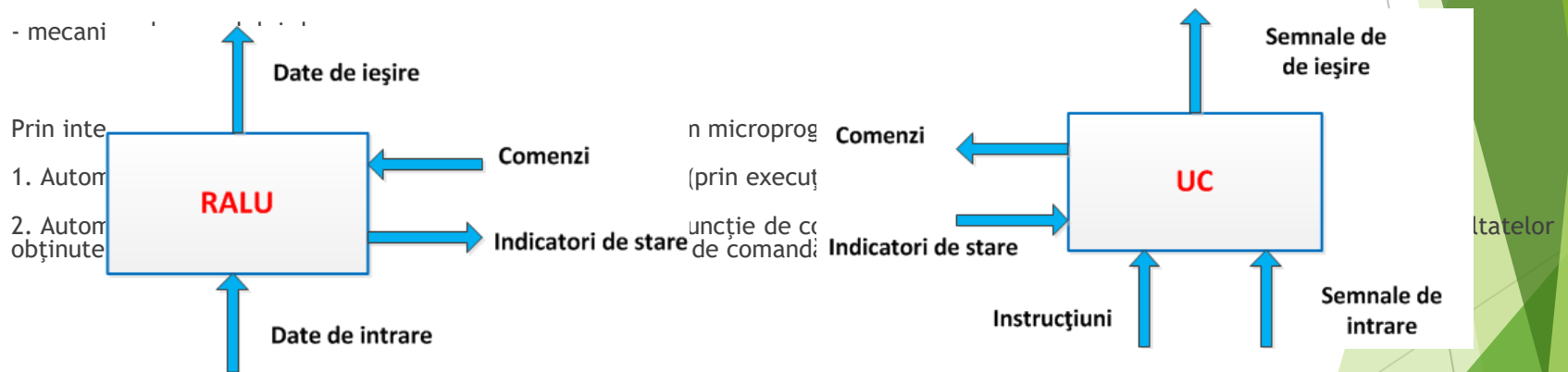
Microprocesorul este un sistem microprogramabil compus din două automate interconectate astfel încât să comunice bidirecțional:

Automatul de Comandă (AC) ;

Automatul de Prelucrare (AP) .

La acestea se adaugă și o serie de **mecanisme specifice**:

- mecanismul registrului PC (Program Counter sau Instruction Pointer) - pentru adresarea și preluarea (fetch) a instrucțiunilor de executat
- mecanismul registrului SP pentru adresarea stivei și rezolvarea apelului, respectiv revenirilor din subrutine
- mecanismul de întreruperi pentru preluarea și tratarea evenimentelor prioritare ce apar pe parcursul execuției programelor
- mecanismul de reset sau inițializare (ce include circuite de tip POR - power on reset, PBOD - Programmable Brown Out Detector, watchdog, etc)
- mecanismul de management al memoriei prin care se extinde capacitatea de adresare
- mecanismul memoriei cache prin care se obține o creștere a vitezei de transfer a datelor din și în memorie
- mecani



Automatul de prelucrare se numește tradițional **Unitate Aritmetico-Logică cu Registre (Register Arithmetic-Logic Unit, RALU)**. El conține și indicatorii rezultatelor Z, S, C.

Automatul de comandă este denumit „**Unitate de Comandă**” (UC)

# Microprocesorul II

## Criterii de clasificare a microprocesoarelor

- după numărul de biți; exprimă dimensiunea operanzilor (în biți) ce pot fi procesați printr-o singură instrucțiune, de obicei reprezentând și mărimea magistralei de date; conform acestui criteriu cu cât un procesor este pe un număr mai mare de biți, cu atât este mai puternic;
- după arhitectură; pot fi procesoare Von Neumann (Princeton) sau Harvard;
- după setul de instrucțiuni; pot fi procesoare CISC (complex instruction set computer) sau RISC (reduced instruction set computer);
- după numărul de instrucțiuni executate simultan; pot fi procesoare scalare (o singură instrucțiune executată la un moment dat) sau superscalare (mai multe instrucțiuni executate la un moment dat, eventual în stadii diferite);
- după modul, locul utilizării și al accesibilității pentru mai mulți utilizatorii (procesoare pentru laptop, desktop, workstation, server, mainframe, supercalculator)
- după clasificarea lui Michael Flynn:

**SISD** (Single Instruction Single Data – o singură instrucțiune, o singură dată de prelucrat); sunt microprocesoarele clasice Von Nuemann;

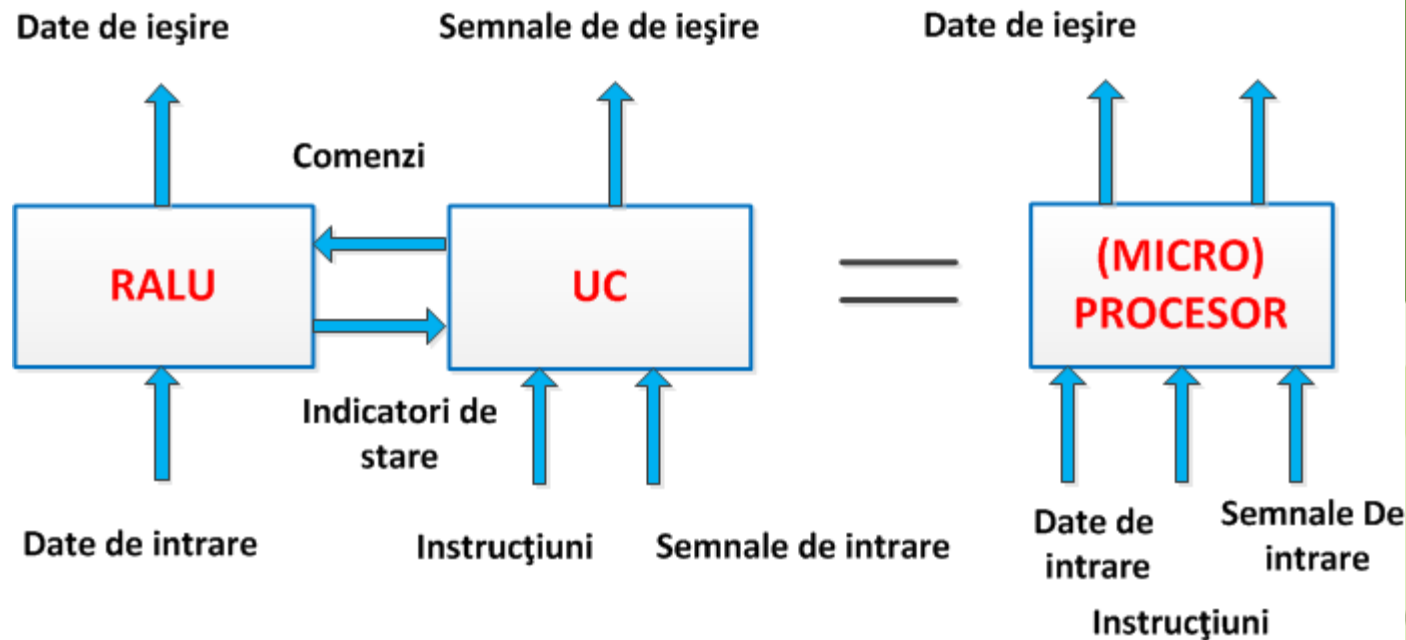
**SIMD** (Single Instruction Multiple Data – o singură instrucțiune, mai multe date); sunt procesoarele matriceale (vectoriale);

**MISD** (Multiple Instruction Single Data - mai multe instrucțiuni, o singură dată); sunt procesoarele care utilizează conceptul de pipe-line (execuții de mai multe instrucțiuni simultan, dar în stadii diferite);

**MIMD** (Multiple Instruction Multiple Data - mai multe instrucțiuni, mai multe date); sunt procesoare dedicate pentru supercalculatoare, folosind arhitecturi paralele de calcul.

# Microprocesorul III

## Modelul Harvard

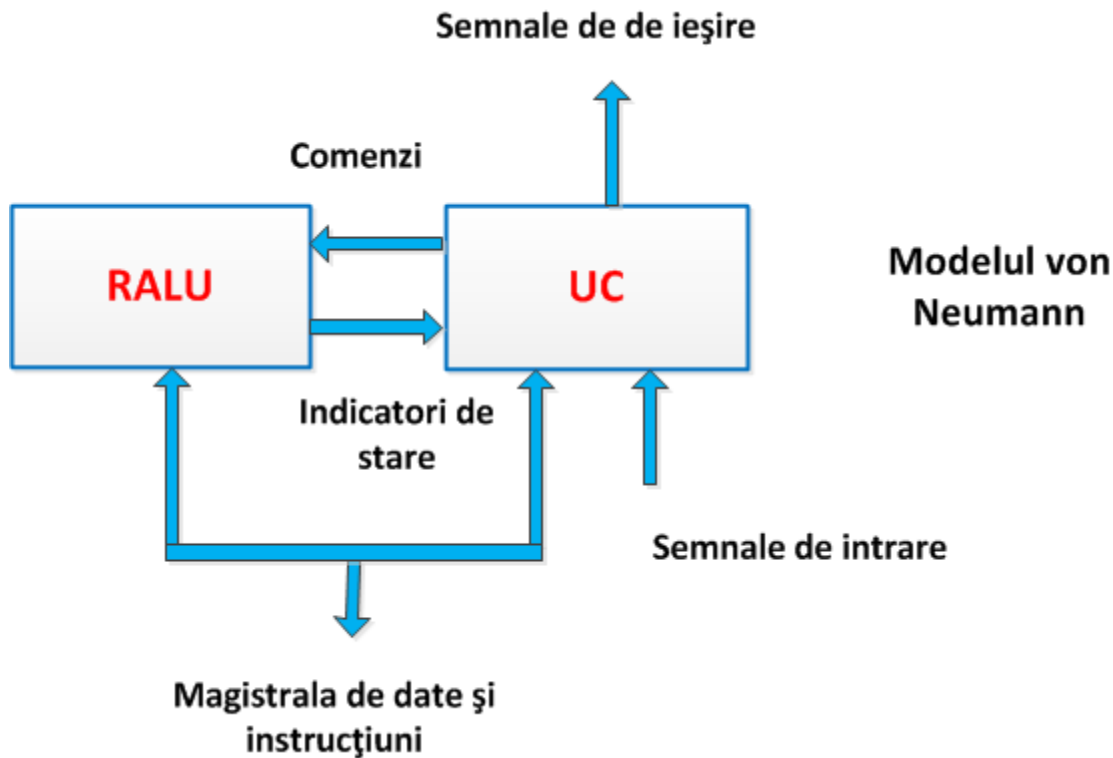


Modelul (arhitectura) Harvard se caracterizează prin separarea căilor de transfer pentru datele de intrare/ieșire față de cea a instrucțiunilor, **liniile de transfer a datelor sunt separate de cele a transferului instrucțiunilor.**

Prin acest paralelism al căilor de comunicație de date și instrucțiuni se asigură o viteză de transfer și (implicit) o viteză de procesare mai mari.

# Microprocesorul IV

## Modelul von Neumann



Dacă se unesc, pentru reducerea numărului total de linii de semnal, liniile de date de intrare, de date de ieșire și instrucțiuni într-o singură magistrală bidirecțională (bus de date și instrucțiuni) se obține modelul von Neumann (sau Princeton).



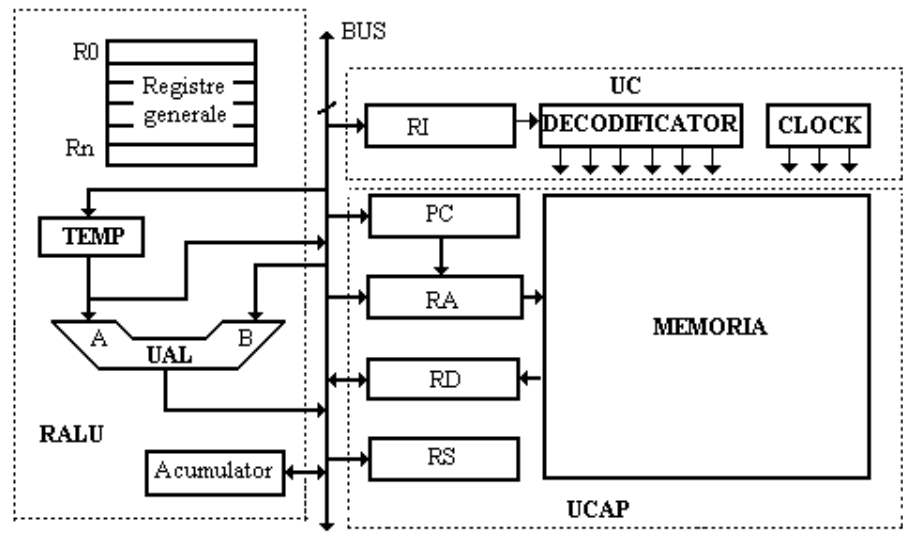
# Microprocesorul V

## Modelul von Neumann

### ► Un exemplu de model Von Neumann (8086)

#### Setul de regiștri

- Regiștrii sunt circuite de memorare utilizate pentru stocarea temporară a datelor.
- Cu cât un procesor are mai mulți regiștri, cu atât este mai ușoară munca sa.
- Regiștrii pot fi pe 8, 16 sau 32 biți.



ACUMULATORUL (WORK Register) memorează rezultatele intermediare. De exemplu, poate fi utilizat la însumarea unui șir de numere.

PROGRAM COUNTERUL (PC ) calculează adresa următoarei instrucțiuni de executat din program.

REGISTRUL DE STARE conține printre altele indicatorii de stare Z(zero), C(carry), N(negativ), V(overflow), I (intrerupere)

# Funcționarea microprocesorului I

## -instrucțiunea-

- ▶ *Instrucțiunea este o componentă a unui program prin care se cere ca una sau mai multe operații să fie executate de procesor.*
- ▶ Structura instrucțiunii în memorie este o succesiune de locații de memorie conținând obligatoriu codul instrucțiunii și opțional una sau două adrese de operand. Instrucțiunile unui anumit procesor pot fi în unul din două tipuri de format:
- ▶ - instrucțiuni cu **format fix** (toate instrucțiunile au aceeași lungime în memorie);
- ▶ - instrucțiuni cu **format variabil** (instrucțiunile pot fi diferite ca lungime în număr de octeți ocupați în memorie).

Structura instrucțiunii:

Opcode

Opcode	Mod	Operand 1	
Opcode	Mod	Operand 1	Operand 2

- Unele instrucțiuni nu au operanzi, sau au doar unul
- Opcode spune ce operațiune se efectuează
- Modul indică tipul instrucțiunii: tip registru, tip memorie
- Operanzii pot fi **imediate** sau pot fi găsiți în memorie prin **adresa** lor

Cod operație

Adresă operand 1

Cod operație

Cod operație

Adresă operand 1

Adresă operand 2

Cod operație

Adresă operand 1

Cod operație

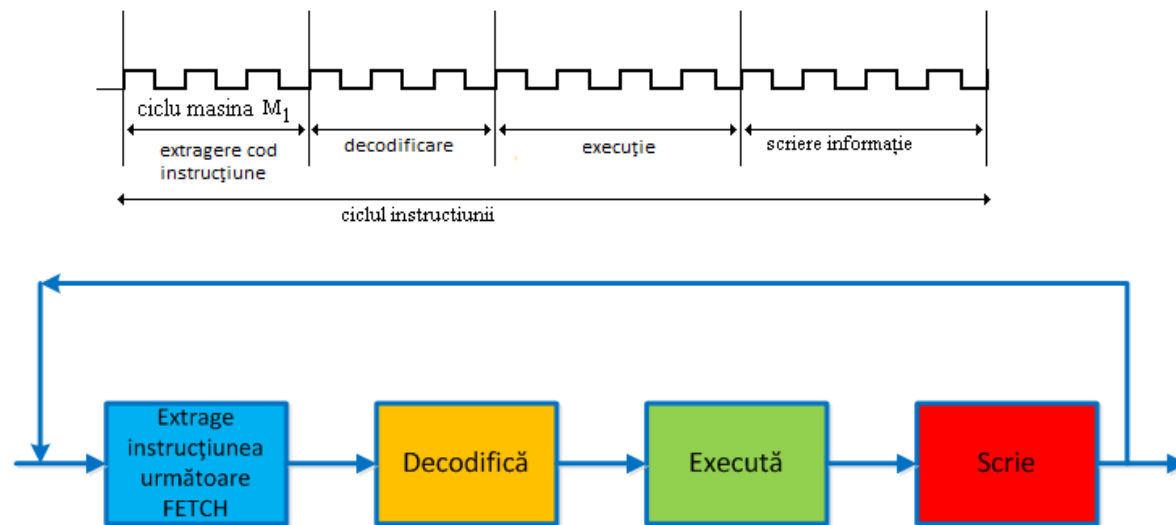
Adresă operand 1

Adresă operand 2

# Funcționarea microprocesorului II

## - ceasul, cicluri -

- *Operațiile de bază* sunt componentele instrucțiunilor (*fetch*, citire-scriere în memorie, etc.). Acestor componente le corespund una sau mai multe perioade de ceas.
- *Starea* este perioada de ceas ( $T$ ) a sistemului, fiind o componentă a operației de bază. Operația de bază poate să conțină mai multe stări.
- *Ciclul mașină* este format din una sau mai multe stări  $T$  reprezentând intervalul de timp în care se execută o operație de bază a instrucțiunii. Ciclurile mașină pot avea un număr fix sau variabil de stări.
- *Ciclul instrucțiune* este o succesiune de cicluri mașină în care primul ciclu  $M_1$  este ciclul de extragere din memorie a codului instrucțiunii ce urmează a fi executat.



# Funcționarea microprocesorului III

## -ciclul instrucțiune-

Ciclu l	1	2	3	4	5	6	7	8	9
Instr1	Fetch	Decode	Execu e	Write					
Instr2					Fetch	Decode	Execu e	Write	
Instr3									Fetch

Execuția ciclică secvențială a instrucțiunii în procesor: o singură instrucțiune la un moment dat

# Funcționarea microprocesorului IV

## -ciclul instrucțiune-

Ciclu	1	2	3	4	5	6	7	8	9
Instr1	Fetch	Decode	Execute	Write					
Instr2		Fetch	Decode	Execute	Write				
Instr3			Fetch	Decode	Execute	Write			
				Fetch	Decode	Execute	Write		

Execuția ciclică a instrucțiunii tip “linie de asamblare/pipeline”. Prin suprapunerea fazelor se produce creșterea numărului de instrucțiuni operate în unitatea de timp.

# Funcționarea microprocesorului V

## -CISC și RISC-

### RISC - *Reduced Instruction Set Computer*

- ▶ Set mai mic (număr mai mic) și mai simplu (programare mai ușoară, de exemplu în cazul lungimii fixe) de instrucțiuni;
- ▶ Toate instrucțiunile au aceeași durată de timp la execuție și se execută la fel indiferent de regiștrii cărora li se aplică.

### CISC - *Complex Instruction Set Computer*

- O instrucțiune poate executa mai multe operații de nivel inferior: de exemplu o singură instrucțiune poate face operațiuni de încărcare a memoriei, operații aritmetice și de stocare a rezultatului în memorie;
- Instrucțiunile au o durată diferită de timp la execuție și se pot aplica doar asupra unor regiștri.

### *Arhitectura cea mai potrivită pentru un procesor rapid:*

- este dependentă de aplicațiile ce vor fi executate;
- foarte puține instrucțiuni;
- multe registre;
- acces simplificat la încărcarea și depozitarea datelor în memoria principală;
- posibilitatea ca majoritatea instrucțiunilor să fie executate într-o singură perioadă de ceas.



# Registrele

## Două tipuri de registre

**Cu destinație specială**  
(*Special Purpose Register*).  
Conținutul lor poate fi doar citit, sau parțial citit-scris

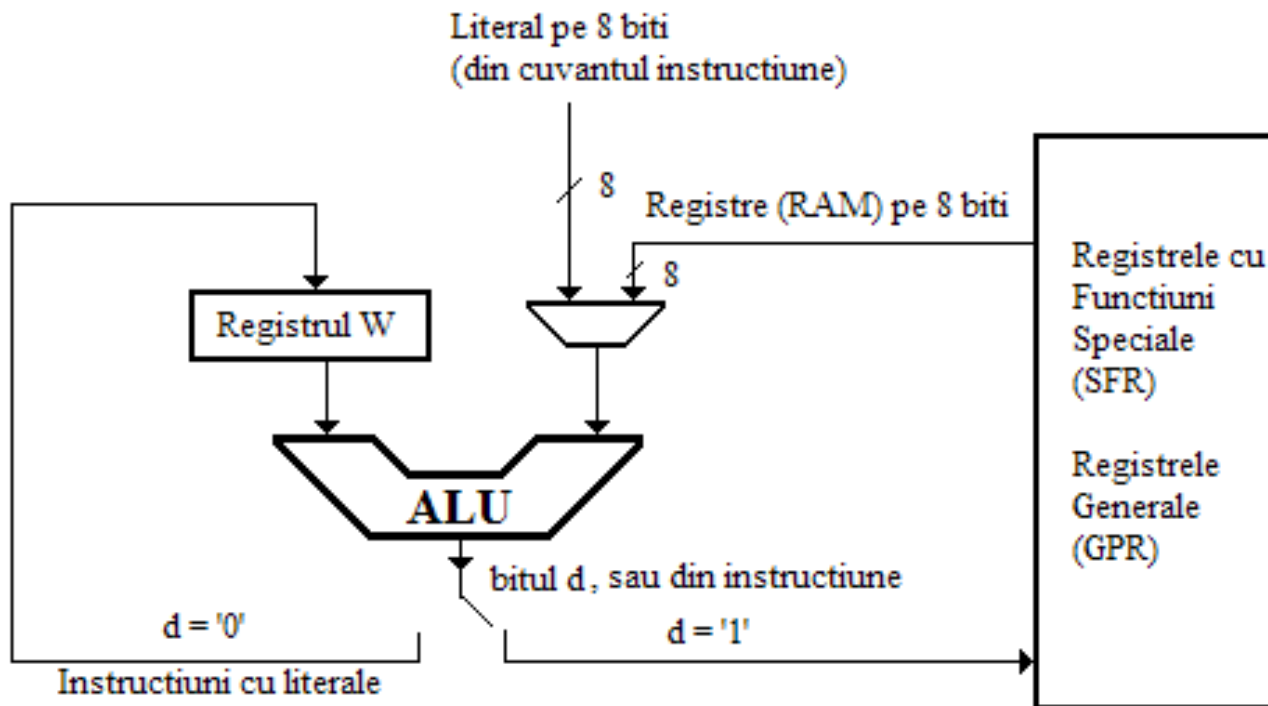
- ▶ Exemple:
- ▶ *Program Counter (PC)*  
Este folosit pentru memorarea adresei curente a instrucțiunii care se execută. De regulă se auto-incrementează
- ▶ *Registrul de Stare*
  - ▶ Conține un număr de biți care se asociază cu starea operațiilor executate de procesor (CPU)
  - ▶ Biții tipici de stare:
  - ▶ V: Overflow (depășire)
  - ▶ C: Carry (transport)

**Cu destinație generală** (*General Purpose Register*). Conținutul lor este în totalitate accesibil adică poate fi citit și scris. Sunt utilizate pentru memorarea temporară a unor date sau rezultate

*Registrele pot avea dimensiuni diferite: 8, 16, 32 biți și pot fi în număr diferit după tipul de procesor*  
*Registrele pot fi structurate diferit (pot fi în spații separate de memorie)*

# Un model al programatorului

- Fluxul de date și instrucțiuni mai este numit și “modelul programatorului”



# Arhitectura setului de instrucțiuni I

## - clasificare-

### ► Instrucțiunile se clasifică după:

- operațiunile efectuate;
- modurile de adresare.

Clasificarea instrucțiunilor diferă după tipul de procesor

### **Clasificare după operațiile efectuate:**

- Aritmetice
- Logice
- De lucru cu regiștrii și memoria (de transfer, de încărcare sau stocare date în memorie)
- De salt
- De control
- Etc

### **Clasificare după modurile de adresare la memorie:**

- Inerent (fără adresare)
- Imediat
- Direct cu regiștrii
- Indirect
- Cu adresare pe bit
- Cu adresare bazată
- Cu index
- Cu index și deplasare
- Cu bază, index și deplasare
- Etc

# Arhitectura setului de instrucțiuni II

## -moduri de adresare-

Instrucțiunile trebuie să specifice de unde se iau operanzii și cum se formează adresa lor. Modul cum se formează adresa de memorie determină variantele de adresare, dintre care cele mai importante sunt:

- "adresarea directă" ;
- "adresarea indirectă".

Exemple:

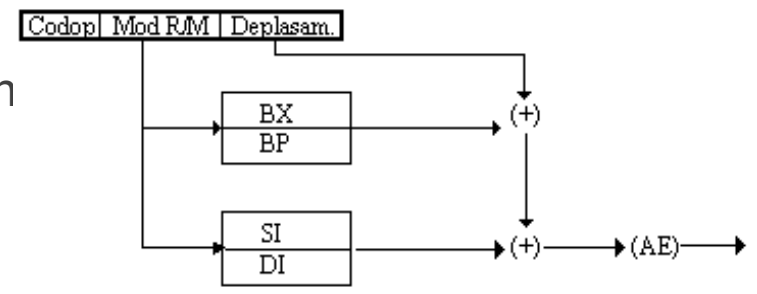
- ▶ Valorile operanzilor sunt în instrucțiune.
- ▶ Valorile sunt în registre - numărul registrului este în instrucțiune.
- ▶ Valorile sunt în memorie:

-adresa este în registru - numărul registrului este în instrucțiune;

-adresa este în instrucțiune;

-adresa este valoarea din registru plus o diferență/offset - numărul registrului este în instrucțiune, iar offsetul este în instrucțiune, sau în registru (adresare indirectă).

- ▶ → aceste căi de specificare a locațiilor operanzilor sunt denumite "moduri de adresare"



# Arhitectura setului de instrucțiuni

## III

### Adresare imediată-

- ▶ Operandul provine din instrucțiunea însăși => operandul este imediat disponibil din instrucțiune

Exemplu :

`addw #99, d7`

Se face adunarea:  $d7 \leftarrow d7 + 99$

Valoarea 99 provine din instrucțiune

-D7 este un registru

Exemplu (uP PIC16 Microchip)

`addlw 0x20`

Se face adunarea  $w \leftarrow w + 32_{10}$

W este registrul de lucru (acumulatorul)

0x20 (32 în baza 10) este valoarea din instrucțiune

# Arhitectura setului de instrucțiuni

## IV

### -adresarea directă la registri-

#### Adresarea directă la registru:

- Numărul registrului în care găsește operandul este dat direct de instrucțiune

Exemplu :

`addw d0, d7`

Se face operația:  $d7 \leftarrow d0 + d7$

Valoarea din d0 se adună la 99 provine din instrucțiune

d7 este un registru

Exemplu (uP PIC16 Microchip)

`movwf 0x30`

Se memorează (w) în f, unde W este acumulatorul și 0x30 ( $48_{10}$ ) este registrul f

# Arhitectura setului de instrucțiuni V

## -adresarea indirectă -

- ▶ Adresarea indirectă
- ▶ Operandul este în memorie, adresa memoriei fiind dată de un registru, iar numărul de registru este conținut în instrucțiune

Exemplu:

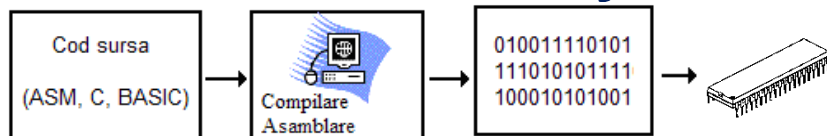
```
addw (a0), d7
```

Se face operația:  $d7 \leftarrow d7 + (a0)$

Valoarea din memoria a cărei adresă se află în registrul a0 se adună registrul d7

De exemplu, dacă a0 este 20 și (20)=321, se adună 321 la registrul d7

# Limbaje de programare -asamblorul și compilatorul-



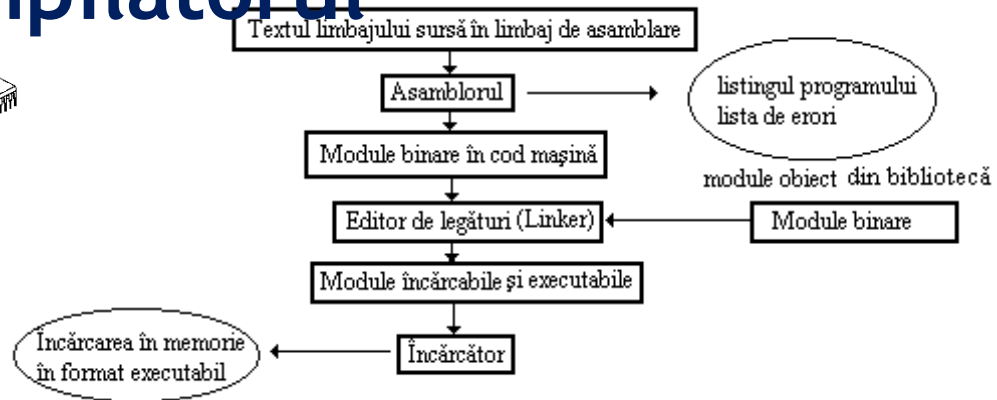
## Asamblorul/Compilatorul

este un program care acceptă la intrare un text numit “program sursă”, scris în limbaj de asamblare sau C și produce la ieșire un program executabil

Un asamblor, spre deosebire de un compilator, face să corespundă, de regulă, în mod direct unei linii sursă, un cod binar executabil.

Programele sursă conțin etichete, instrucțiuni și directive.

Directive ale asamblorului:



## Reprezentarea numerelor în limbajul asamblor hexazecimal:

- 37h (litera h urmează după valoarea numerică):
- 0x37h (prefixul 0x este utilizat în special în limbajul C);
- 37 (implicit). Un număr fără prefix sau sufix este considerat automat hexazecimal.

## binar:

- b'00100110'. Prefixul este litera b urmată de numărul în binar delimitat de apostrof.

## zecimal:

- d'45'. Prefixul este litera d urmată de numărul în format zecimal delimitat de apostrof.
- .45. Punctul utilizat ca prefix arată că numărul este în format zecimal.

ASCII: Caracterele ASCII sunt delimitate de apostrof: 'A'.

Etapele necesare pentru obținerea unui program executabil în memorie pornind de la un program sursă



# Circuitele de intrare-ieșire I/O

Sunt circuite digitale, de obicei programabile, prin intermediul cărora se pot conecta la magistralele unui sistem de calcul diverse echipamente periferice. Din acest motiv se mai pot numi cuploare.

De obicei, fiecare tip de echipament periferic necesită un circuit I/O propriu.

Spre exemplu, un hard-disk necesită un circuit I/O specializat propriu, destul de complex, programabil.

Alte periferice, spre exemplu cele care comunică prin USB, pot să utilizeze un același circuit I/O, programabil.

Unele periferice, mult mai simple (led-uri, taste), necesită circuite I/O la rândul lor simple - chiar regiștri de tip paralel-paralel, și care nu este nevoie să fie programabili.

Totuși, în marea majoritate a cazurilor, circuitele I/O sunt circuite “inteligente”, care pot să își desfășoare activitatea fără a mai fi nevoie în evoluția lor de intervenții dese ale procesorului. Acest lucru este posibil datorită automatelor de control pe care le conțin, dar care necesită o programare prealabilă.

Programarea prealabilă se realizează de obicei la începutul execuției programelor și constă în configurarea conținuturilor unor regiștri interni, prin instrucțiuni pe care procesorul le transmite, ca urmare a programului de lucru scris de către utilizator. Deci “inteligenta” circuitului I/O este de fapt transmisă de utilizator, folosind ca intermediar procesorul.

De obicei acești regiștri poartă denumirea de regiștri de comandă și control, regiștri de stare, regiștri de date și se identifică în datele de catalog.

Fiecare mare producător de procesoare produce și un set specific de circuite I/O și de aici o mare diversitate a acestora. Cele mai întâlnite **circuite I/O** sunt: **porturile paralele** (prin intermediul cărora se pot genera și recepta semnale digitale, dar și transmite și recepționa date binare în format paralel, simultan de obicei pe 8 biți), **timer-ele** (care permit numărarea de impulsuri și indirect măsurarea timpului, generarea de forme de undă legate de numărarea de impulsuri, etc), **circuitele pentru controlul comunicației seriale** (care permit transferuri de date între diverse echipamente)

Astfel de circuite I/O sunt conținute și în microcontrolerul utilizat de modulul Arduino (produs de ATMEL, Atmega 328P). Pentru aceste circuite I/O au fost create librării cu funcții specifice pentru mediul de programare, prin care munca de programare să fie ușurată, dar câteodată este nevoie și de setări directe de biți în regiștrii de comandă și control.

# Interfațarea I/O a microprocesoarelor I

- ▶ O **interfață** este un concept care se referă la interacțiunea dintre componentele din interiorul microsistemului și exteriorul său.
- ▶ Un procesor utilizează circuite (dispozitive) I/O pentru transferul datelor între interiorul său și lumea exterioară.
- ▶ Deoarece circuitele (dispozitivele) I/O sunt de obicei lente, este necesar ca la ieșirea procesorului datele de ieșire să fie valide mai mult timp. Cum datele sunt prezente pe magistrală o scurtă perioadă de timp, ele trebuie să fie memorate în registre (latch-uri) externe.
- ▶ Și în cazul intrărilor, datele trebuie memorate într-un registru. Un semnal de întrerupere poate sesiza prezența datelor la intrare. Pentru conectarea dispozitivelor de intrare la magistrală este necesar un buffer **tri-state** (adică un circuit cu etaj de ieșire care permite conectare în paralel a mai multor ieșiri de circuite digitale).

# Interfațarea I/O a microprocesoarelor II

## -porturi și metode de transfer-

- ▶ Prin utilizarea circuitelor I/O denumite **porturi**, datele pot fi transferate între procesor și echipamentele periferice (lumea exterioară).
- ▶ Transferul poate fi făcut în grupuri de 8 biți utilizând magistrale. Acest mod se numește I/O paralel, iar denumirea dispozitivului este “**port paralel**”.
- ▶ Altă metodă este transmiterea serială I/O în care un bit este transferat la un moment dat pe pinii TX și RX ai microcontrolerului. Circuitul I/O care gestionează acești pini și de fapt comunicația serială este un “**port serial**”.
- ▶ Metodele de transferare a datelor între microprocesor și lumea exterioară sunt:
  - Intrare/ieșire (I/O) programată
  - Intrare/ieșire (I/O) utilizând sistemul de întreruperi
  - Intrare/ieșire (I/O) cu DMA (Direct Memory Access)

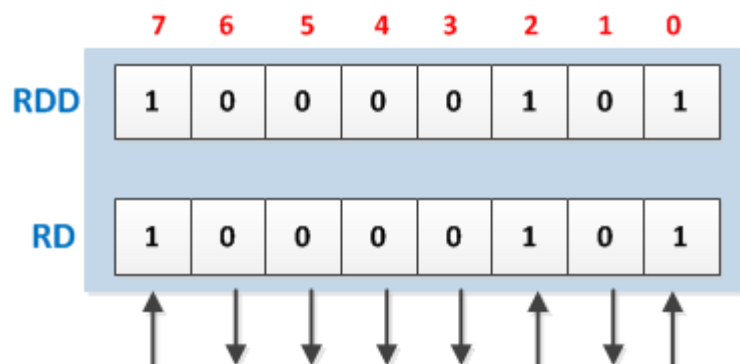
# Interfațarea I/O a microprocesoarelor III

## - I/O programat -

- ▶ uP execută un program pentru a face transferul între uP și un circuit I/O printr-unul sau mai multe registre ale circuitului I/O. uP controlează complet transferul datelor prin programul executat.
- ▶ La un circuit I/O port paralel apar de obicei două registre prin care se gestionează transferul I/O:
  - RD, **Registrul de Date** va conține datele reale care intră sau ies în/din uP.
  - RDD, **Registrul Direcției Datelor** configurează fiecare bit din registrul de date ca intrare sau ca ieșire, astfel:

1 în RDD indică bitul corespondent din RD ca intrare.

0 în RDD indică bitul corespondent din RD ca ieșire.

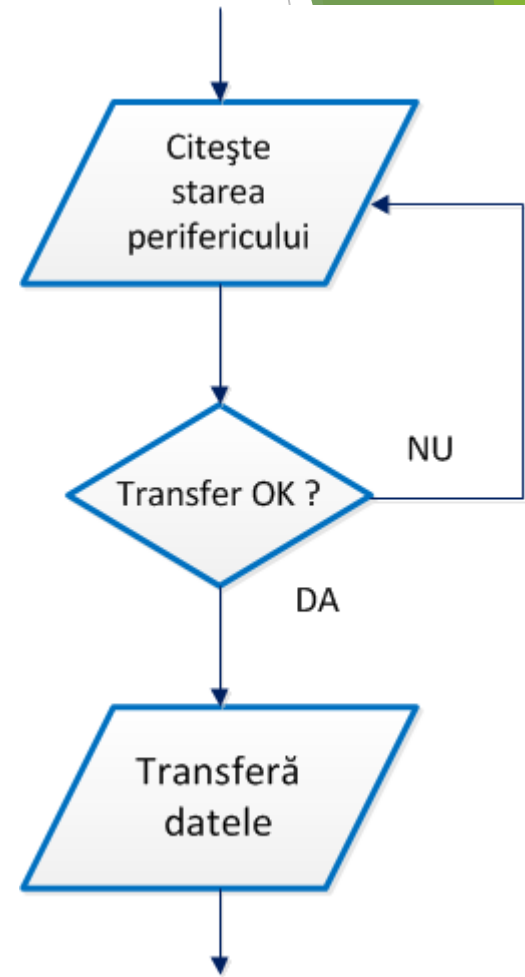


# Interfațarea I/O a microprocesoarelor IV

## - I/O programat -

Transferul I/O programat se poate face **condiționat** sau **necondiționat**.

- Transferul **necondiționat** al datelor se poate face oricând; perifericul extern trebuie să fie ready întotdeauna pentru transferul datelor. În acest caz receptorul nu informează emițătorul despre starea primirii informațiilor.
- Transferul **condiționat** al datelor între uP și perifericul extern se face prin **hand-shaking**. Receptorul informează emițătorul despre reușita sau nereușita transferului de informații, dar și despre pregătirea pentru altă recepție. Practic, Hand-shaking-ul presupune și transmisia de informații suplimentare, față de cele propriu-zise prin care să se controleze validitatea datelor



# Sistemul de întreruperi al microprocesoarelor

O întrerupere este de fapt o semnalizare (semnal digital) generat de un circuit extern procesorului sau intern acestuia prin care se anunță apariția unui eveniment important care necesită tratare din partea procesorului.

Dacă întreruperea este acceptată de procesor, atunci acesta suspendă execuția programului în curs, salvând în stivă adresa de revenire, dar și eventuale alte informații despre starea programului suspendat și trece la tratarea evenimentului apărut prin execuția unui alt program denumit "*interrupt service rutine* - ISR". După ce încheie acest program de tratare a întreruperii, uP se reîntoarce la execuția programului suspendat reîncărcând din stivă elementele salvate anterior necesare continuării corecte.

Întreruperile pot fi: hardware (aplicate pe pini specializați ai procesorului și care provin din surse externe) sau software (care provin din surse interne procesorului; exemplu forțarea unei operații de împărțire cu 0)

Întreruperile hardware, la rândul lor pot fi nemascabile - se primesc pe un pin notat de obicei NMI - non maskable interrupt - și necesită tratare imediată sau mascabile. Întreruperile mascabile se pot dezactiva și activa pe cale software prin rularea de instrucțiuni specifice care acționează asupra unui bistabil intern procesorului (interrupt flag).

O întrerupere mascabilă poate fi dezactivată sau activată prin execuția unor instrucțiuni de tipul DI (*Disable interrupt*) sau EI (*Enable Interrupt*). Dacă întreruperea este dezactivată (disable, uP ignoră întreruperea mascabilă).

Producătorii de procesoare au inclus în setul de circuite I/O produse și circuite I/O de tipul controlor de întreruperi care au facilități în ceea ce privește operațiunile care au loc între procesor și circuitul I/O generator de întreruperi pentru tratarea acesteia. Alte circuite I/O cu funcții diverse (timer, port paralel, port serial) au fost proiectate astfel încât să conțină o logică minimală de întreruperi care să asigure operarea în acest mod cu procesorul.

În sistemele de calcul cu mai multe surse de întreruperi se utilizează sisteme de priorități programabile pentru a ierarhiza solicitările primite.

Se întâlnesc mai multe modalități de tratare a întreruperilor. Un astfel de mod este tratarea vectorială prin folosirea unui tabel al vectorilor de întreruperi, în care se indică adresa rutinei de tratare a întreruperii apărute. Un astfel de mod folosește și microcontrolerul ATMEL ATmega 328P din modulul Arduino (vezi planșa următoare).

Mediul de programare al ARDUINO permite implementarea lucrului cu întreruperi prin care să fie semnalizate procesorului evenimente prioritare care necesită tratare. Totodată, prin întreruperi este posibil și time-sharing-ul (împărțirea timpului de procesare între mai multe activități, acestea părând că se execută în paralel) - sursa de întreruperi fiind în acest caz un canal

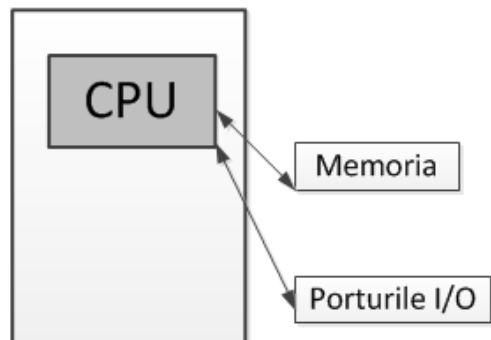
# Microprocesor vs. Microcontroler

**Microcontrolerul (MCU - microcontroller unit)** este de regulă un circuit complex obținut prin adăugarea unor dispozitive suplimentare la un microprocesor, *toate fiind integrate pe același chip.*

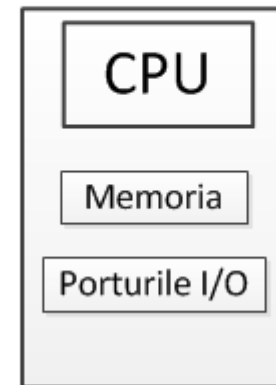
Un **microprocesor (MPU - micro processor unit)** conține pe același cip:

- unitatea centrală de procesare - UCP (registre, ALU, indicatori, magistrale);
- unitatea de gestionare a întreruperilor interne și externe;
- circuite pentru generarea și gestionarea impulsurilor ceasului (modificări ale frecvenței, reducerea radiației, etc.);
- circuite de alimentare cu tensiuni de valori diferite, etc.;
- unitatea de management a

**Microprocesorul**  
- este o unitate centrală de procesare  
-memoria și porturile I/O  
sunt externe chipului



**Microcontrolerul**  
- este sistem de calcul pe un singur chip  
-memoria și porturile I/O  
sunt conectate intern



Un **microcontroler** cuprinde pe același chip un **microprocesor** și, în plus, o serie de circuite necesare unui sistem de calcul (practic, poate fi un microcalculator complet):

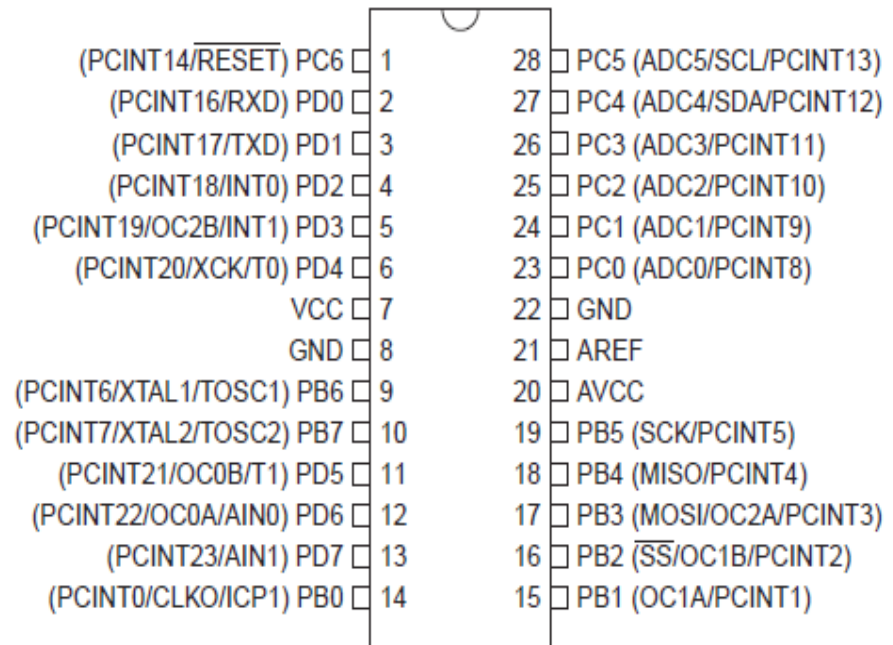
- memorii (RAM, ROM, EPROM/EEPROM, FLASH);
- periferice de comunicație a datelor paralelă și serială - (porturi paralele, porturi seriale asincrone și sincrone - UART, USART, SCI, SPI, I<sup>2</sup>C);
- periferice de gestionarea timpului (timere pentru generarea întârzierilor de timp, circuite pentru captură de timp/ măsurarea și compararea timpului);
- circuite antiblocare de tip watchdog;
- circuite de securitate pentru protecția softwarelui înscris în memoria ROM;
- circuite pentru numărarea impulsurilor sau generarea de impulsuri PWM (pulse width modulation - impulsuri modulate în durată);

# Microcontrolerul ATMEL ATmega 328P

## Caracteristici generale

### Conține:

- procesor RISC pe 8 biți, 131 instrucțiuni
- arhitectură Harvard modificată
- 32 regiștri de uz general pentru CPU
- 64 regiștri pentru programarea circuitelor I/O
- 32k x 8 memorie ROM flash (10.000 cicluri ștergere)
- 2k x 8 memorie SRAM
- 1k x 8 memorie EEPROM (100.000 cicluri ștergere)
- 3 canale timer (1 pe 16 biți, 2 pe 8 biți)
- 6 canale PWM
- 6 canale cu 10 biți ADC
- 1 canal UART
- interfață serială 2 wire (TWI - I2C)
- interfață serială SPI
- 23 pini programabili I/O (tip port paralel)
- surse de întreruperi interne și externe
- facilități automate de reset (Power on reset - POR, Programmable Brown Out Detector - PBOD, watchdog)
- operare cu semnal de ceas de până la 20MHz, 20 MIPS
- programare *In Circuit Self Programming* (ICSP) pentru memoria flash
- consum < 15 mA / 5V

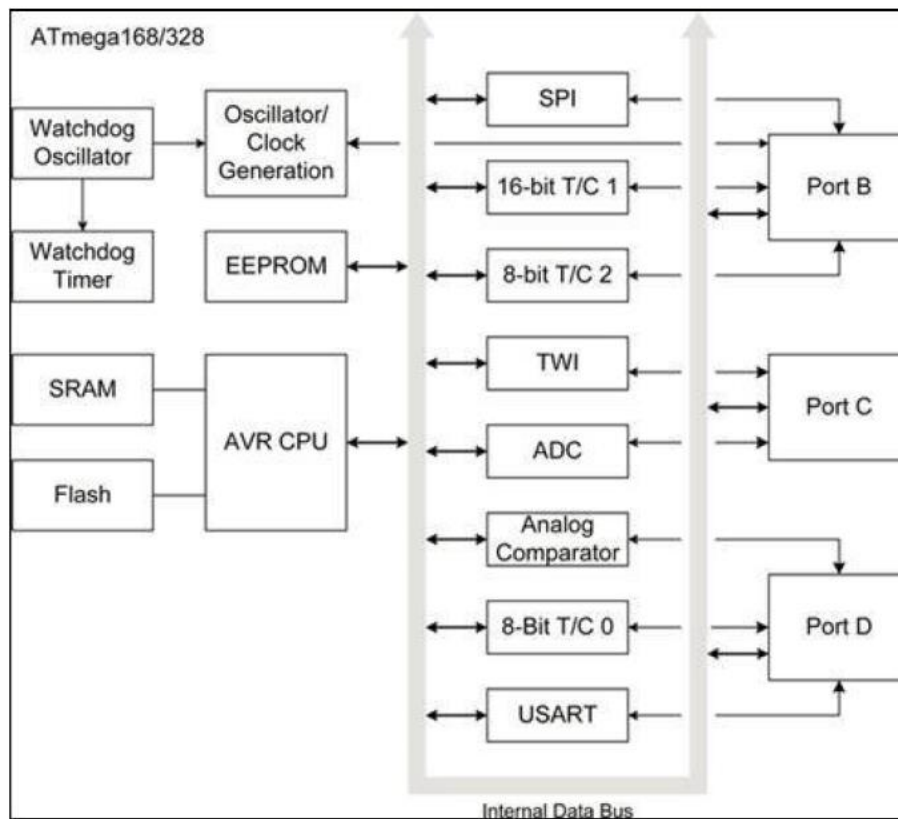


Arhitectura inițială a microcontrolerului a fost proiectată la începutul anilor '90 de 2 studenți, Alf-Egil Bogen și Vergard Wollan, de la Norwegian Institute of Technology, pe când lucrau la o societate de realizare a circuitelor integrate în Trondheim. Ulterior, proiectul l-au vândut companiei Atmel din USA, unde ei lucrează în prezent.



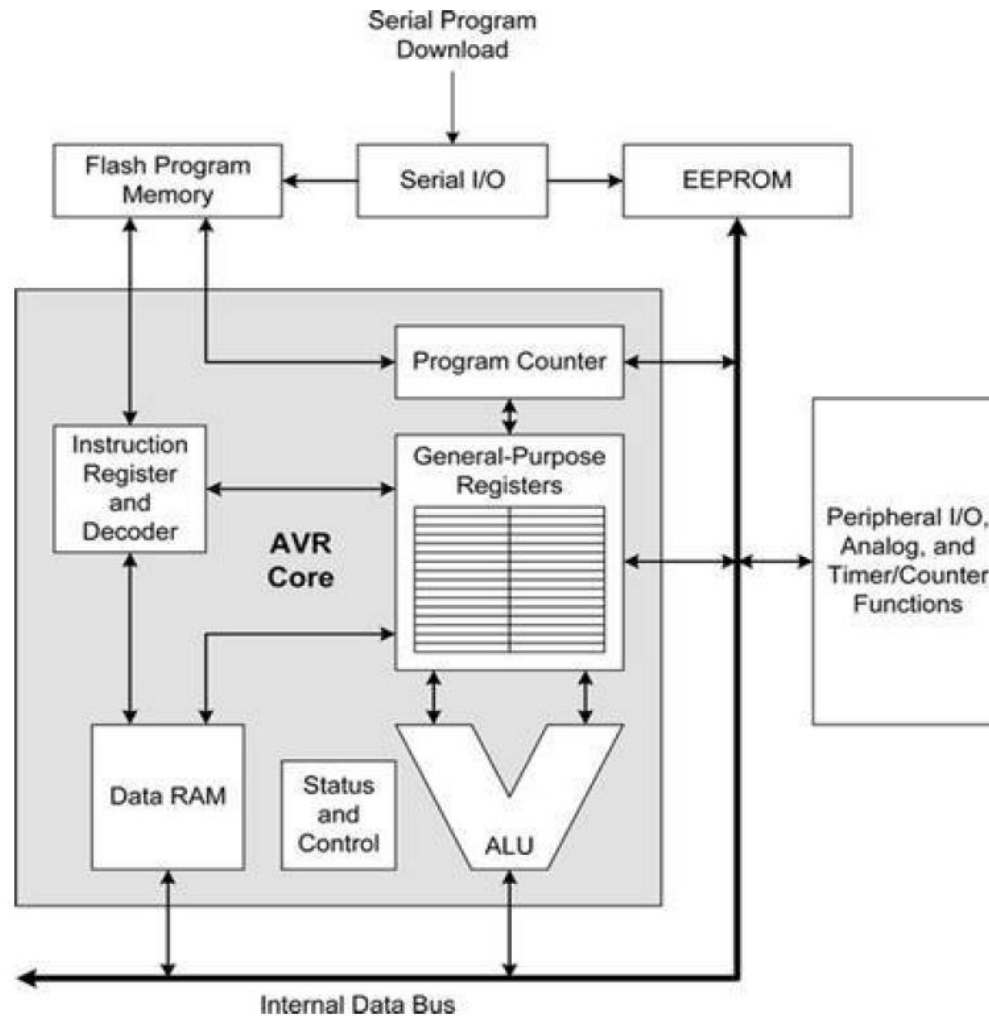
# Microcontrolerul ATMEL ATmega 328P

## Schema bloc



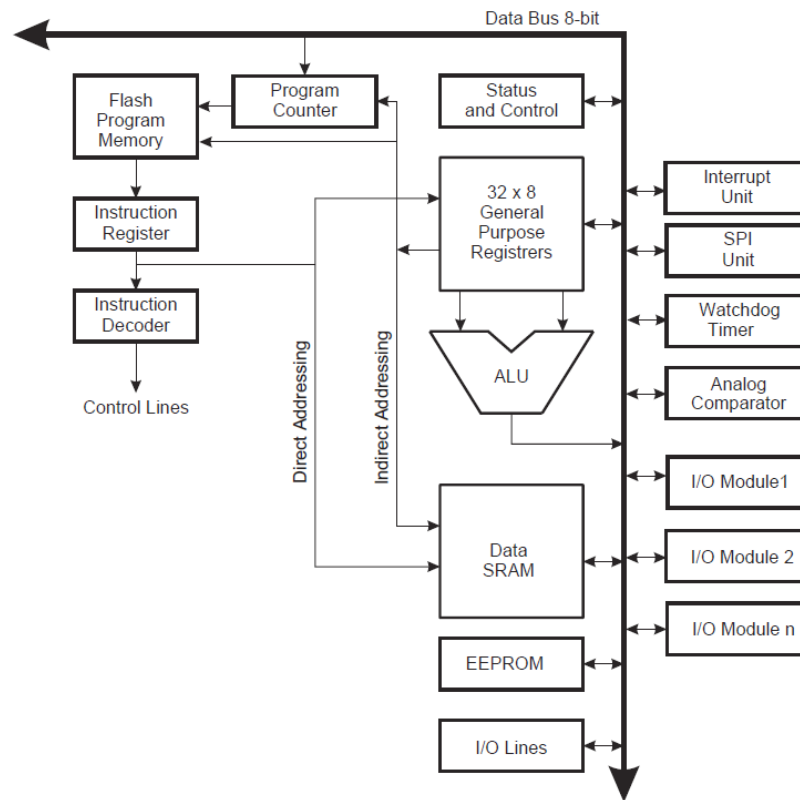
# Microcontrolerul ATMEL ATMega 328P

## Procesorul - CPU



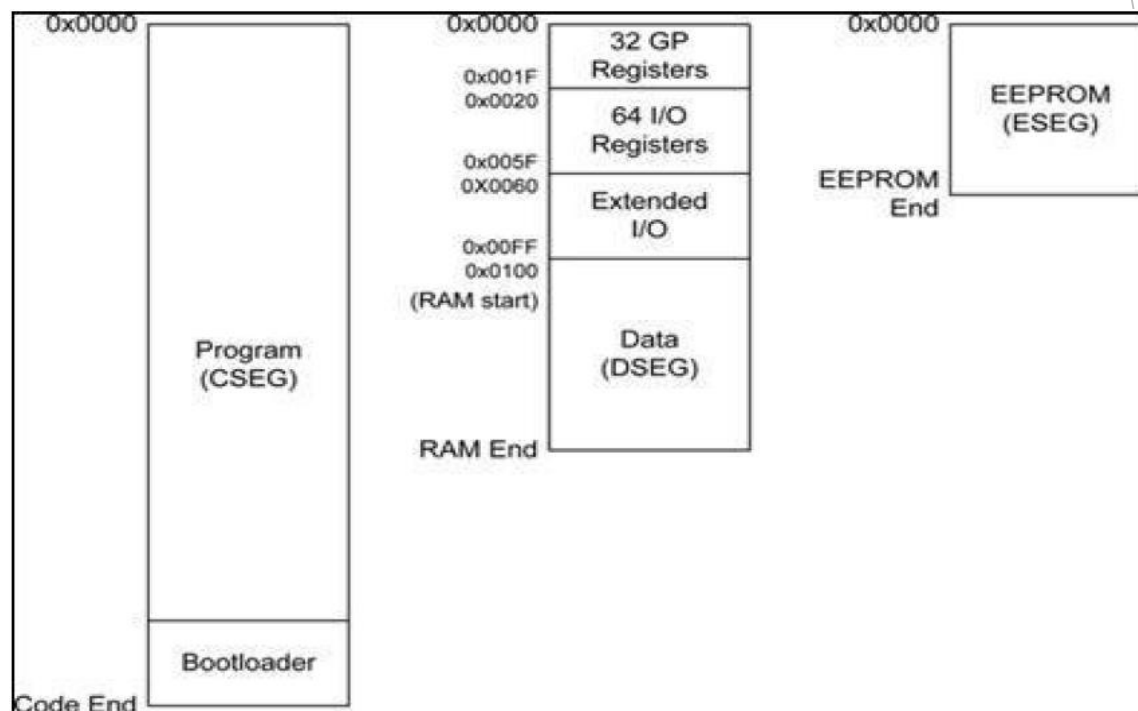
# Microcontrolerul ATMEL ATmega 328P

## Procesorul - CPU - catalog ATMEL



# Microcontrolerul ATMEL ATmega 328P

## Organizarea memoriei interne



Memoria interna este formată din 3 componente:

- Memoria ROM - flash - în care se păstrează Programele executate de CPU (Code Segment) și Bootloader-ul
- Memoria SRAM - folosită pentru formarea și păstrarea celor 32 regiștri de uz general CPU și cei 64 regiștri ai circuitelor I/O, dar și pentru păstrarea și actualizarea variabilelor din programele executate de CPU (Data Segment)
- Memoria EEPROM - folosită pentru păstrarea de informații necesare programelor după reaplicare tensiunii de alimentare (EEPROM Segment)

# Microcontrolerul ATMEL ATmega 328P

## Setul de regiștri de uz general

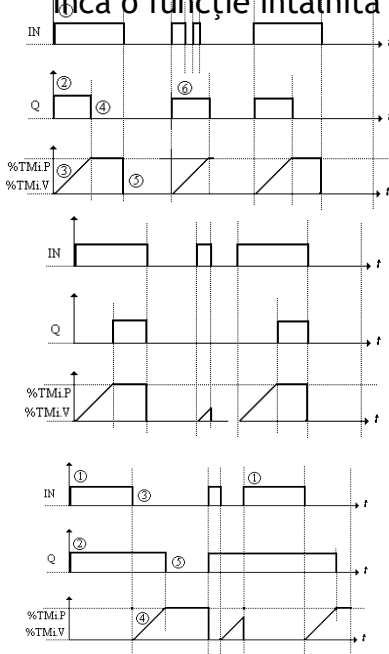
AVR CPU General Purpose Working Registers

	7	0	Addr.	
General Purpose Working Registers	R0		0x00	
	R1		0x01	
	R2		0x02	
	...			
	R13		0x0D	
	R14		0x0E	
	R15		0x0F	
	R16		0x10	
	R17		0x11	
	...			
	R26		0x1A	X-register Low Byte
	R27		0x1B	X-register High Byte
	R28		0x1C	Y-register Low Byte
	R29		0x1D	Y-register High Byte
	R30		0x1E	Z-register Low Byte
	R31		0x1F	Z-register High Byte

# Microcontrolerul ATMEL ATmega 328P

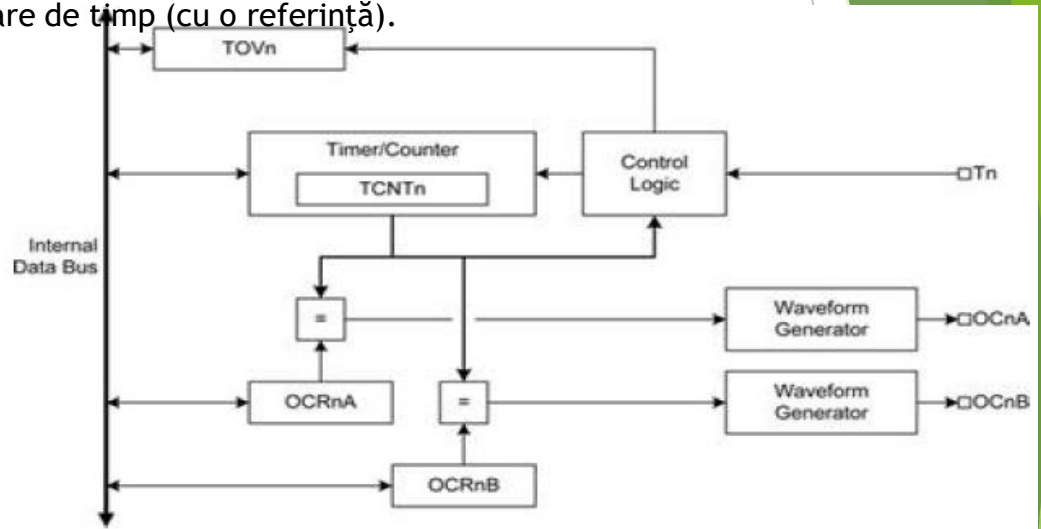
## Structură generală canal timer programabil

Un circuit I/O (canal) timer este organizat în jurul unui numărător, de obicei pe 16 biți, care poate fi configurat în diverse moduri de lucru, prin regiștrii de comandă și control ai circuitului. Acest numărător poate număra impulsuri digitale din surse diverse (interne sau externe). Un canal timer poate fi folosit pentru a genera la ieșire semnale digitale, funcție de configurația programată (exemplu semnal PWM). Circuitul poate fi folosit pentru măsurarea timpului, dacă se cunoaște perioada semnalelor de numărare aplicate. O altă funcție este aceea de captura de timp, adică măsurarea timpului între două evenimente. Încă o funcție întâlnită este cea de comparare de timp (cu o referință).



În general, funcția de generare impulsuri întârziate – *Timer Function* – are trei forme de utilizare:

- TON (*Timer On-Delay*): acțiunea de întârziere este declanșată de începutul evenimentului de intrare;
- TOF (*Timer Off-Delay*): acțiunea de întârziere este declanșată de sfârșitul evenimentului de intrare;
- TP (*Timer – Pulse*): acționează ca un monotabil și se utilizează pentru crearea de impulsuri de durată precisă.



Se programează prin regiștrii de tip I/O:

TCNTn - Registru numărător (Timer - Counter Register) n = 0, 1, 2

TCCRn - Timer-Counter Control Registers

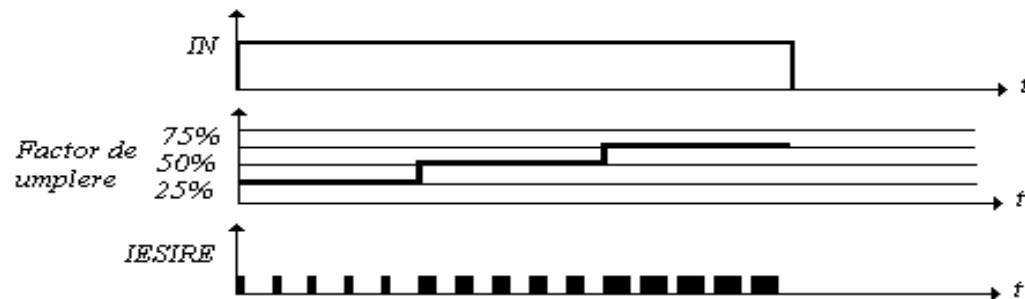
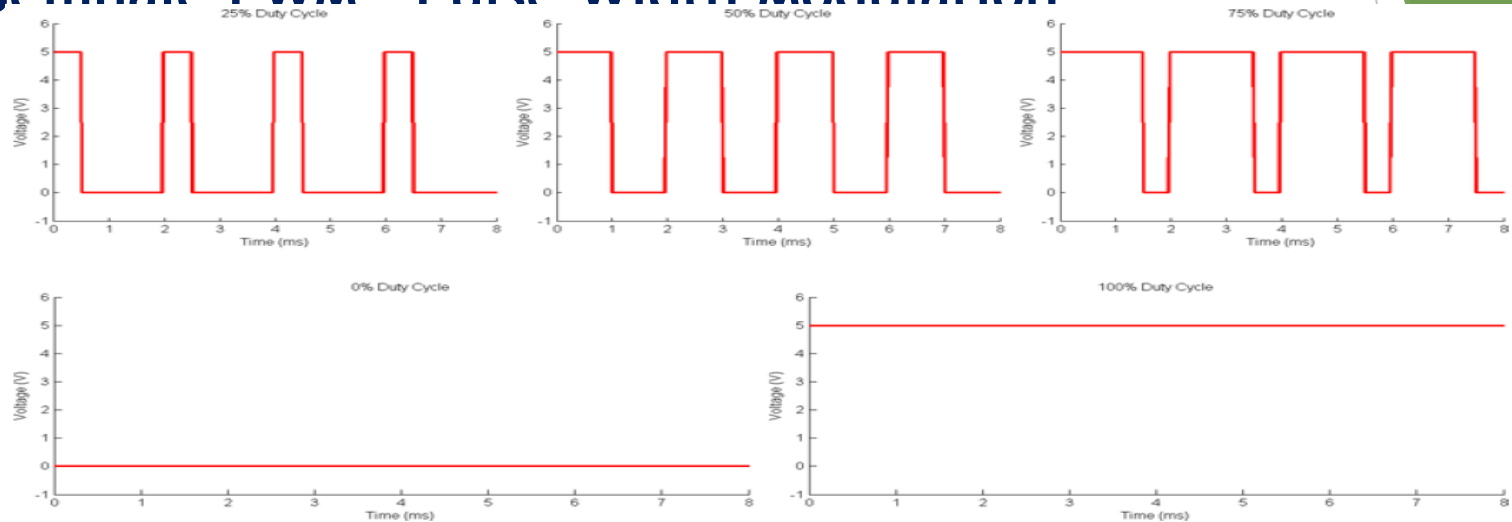
OCRnA, B - Output Compare Registers

TIFRn - Timer-Counter Interrupt Flag Registers

TOVn - Indicator depășire (Timer Overflow Flag) bit încadrul registrului TIFRn

# Microcontrolerul ATMEL ATMega 328P

## Semnale PWM - Pulse Width Modulation



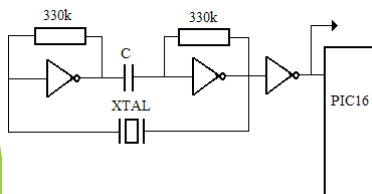
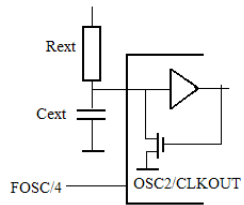
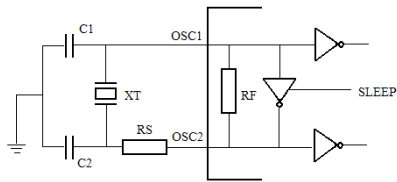
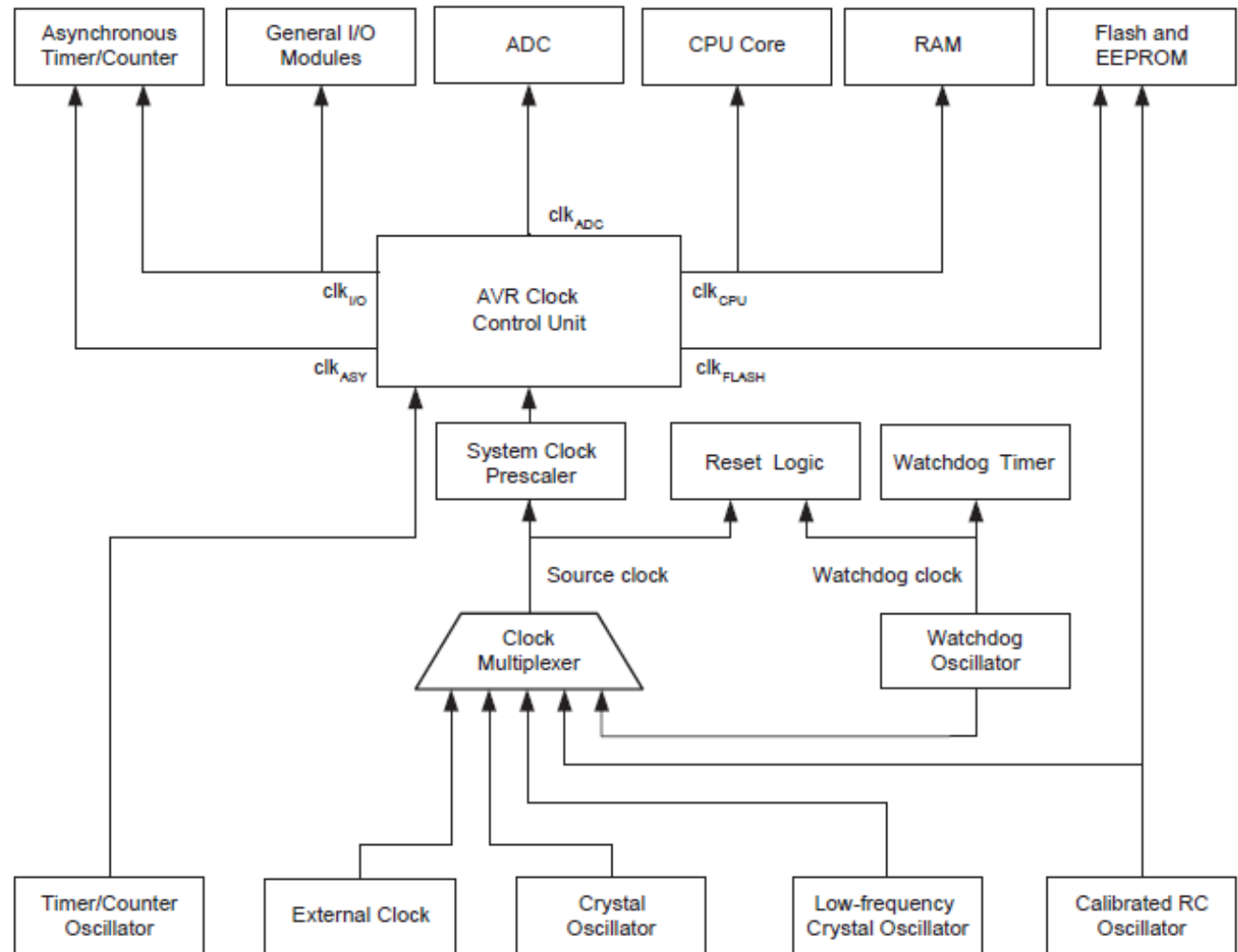
Semnalele PWM, utile la comanda motoarelor de curent continuu, se pot genera prin folosirea canalelor timer programabile.

# Sistemul de ceas (clock)

Rolul oscilatorului este să furnizeze semnalele de ceas necesare execuției instrucțiunilor și sincronizării cu circuitele I/O, respectiv perifericele.

Semnalul de ceas este folosit de multe circuite I/O în evoluția lor.

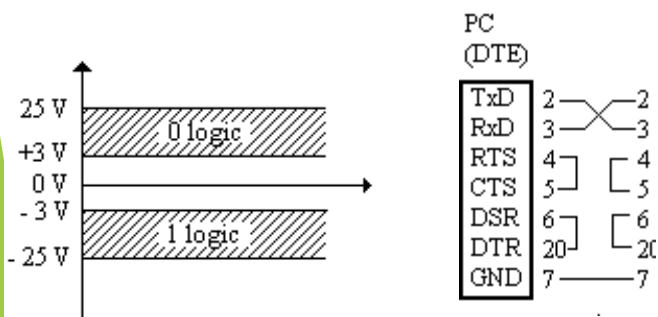
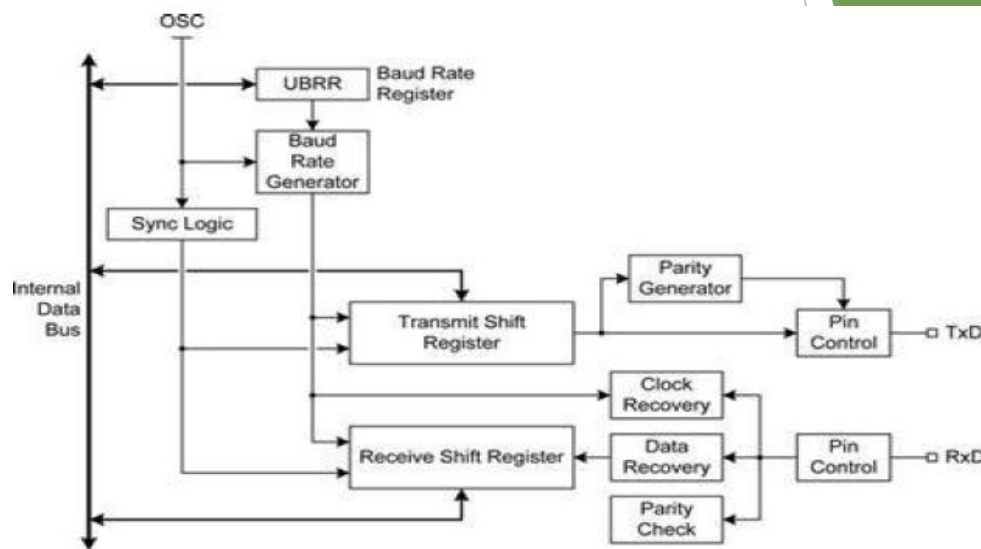
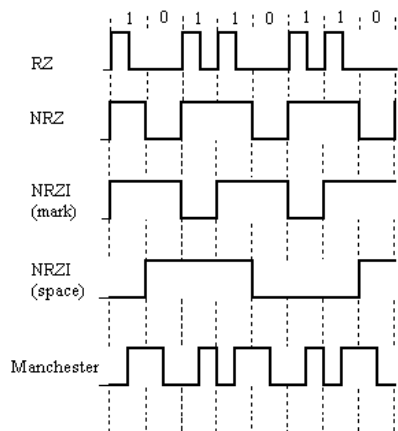
## Distribuția semnalelor de ceas la uP AVR





# Microcontrolerul ATMEL ATmega 328P

## Structură generală canal de comunicație serială USART

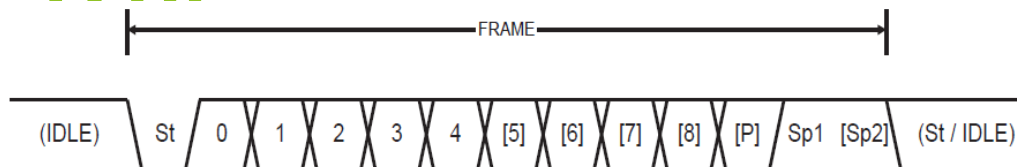


Se programează prin regiștrii de tip I/O:  
 UCSR - USART Control and Status Register  
 UBRR - USART Baud Rate Register  
 UDR - USART I/O Data Register

# Microcontrolerul ATMEL

## ATMega 328P

### Structură serială



<b>St</b>	Start bit, always low.
<b>(n)</b>	Data bits (0 to 8).
<b>P</b>	Parity bit. Can be odd or even.
<b>Sp</b>	Stop bit, always high.
<b>IDLE</b>	No transfers on the communication line (RxDn or TxDn). An IDLE line must be high.

### Calculul bitului de paritate

$$P_{even} = d_{n-1} \oplus \dots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 0$$

$$P_{odd} = d_{n-1} \oplus \dots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 1$$

<b>P<sub>even</sub></b>	Parity bit using even parity
<b>P<sub>odd</sub></b>	Parity bit using odd parity
<b>d<sub>n</sub></b>	Data bit n of the character

# Microcontrolerul ATMEL ATMega 328P

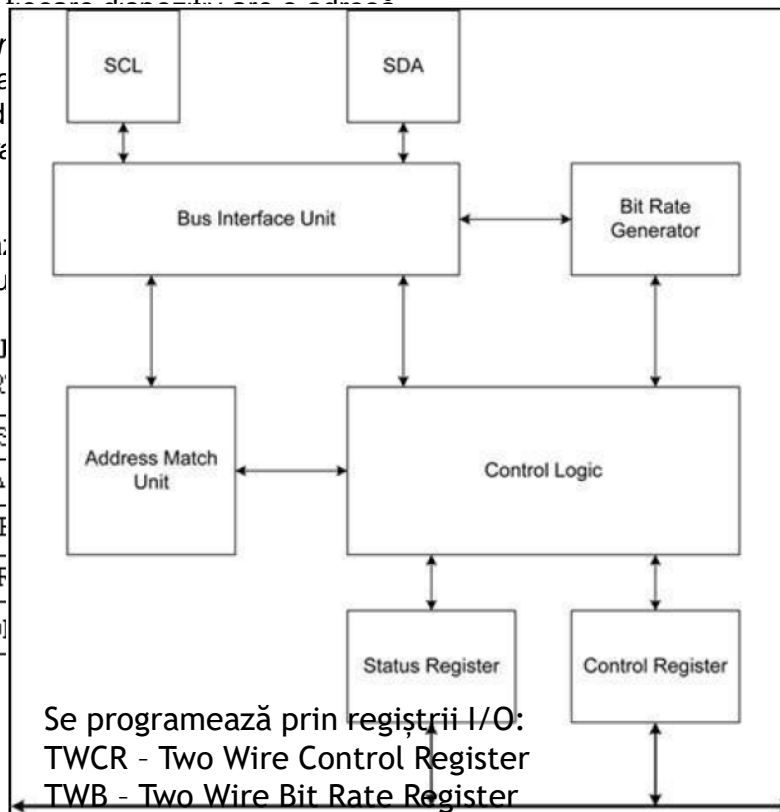
## Structură generală canal de comunicație serială TWI - I2C

I<sup>2</sup>C înseamnă *Inter-Integrated Circuit*. Standardul a fost destinat ca o interfață de comunicație pe **două fire** între diferite circuite integrate pe vitezele de transfer de 100kbps și 400kbps (*Fast mode*). Ulterior a fost extins până la 1Mbps și chiar la 3,4Mbps (*High-speed*, sau *Hs-mode*).

I<sup>2</sup>C este o interfață *multimaster*, ceea ce înseamnă că admite mai mult de un *master* pentru controlul liniei. Protocolul I<sup>2</sup>C presupune că fiecare dispozitiv are o adresă unică.

Când un *master* vrea să comunice. Toate slavele trebuie să fie în stare de atenție. Dacă *master*-ul detectează că nu este singurul *master*, el trebuie să aștepte până când linia este liberă pentru a transmite semnal de ceas.

Se utilizează un singur fir pentru a transfera datele și un alt fir pentru a transfera semnalul de ceas.



Se programează prin registre I/O:  
TWCR - Two Wire Control Register  
TWBR - Two Wire Bit Rate Register  
TWSR - Two Wire Status Register  
TWDR - Two Wire Data Register  
TWAR - Two Wire Address Register

Un *master* transmite adresa dispozitivului *slave*, cu care vrea să comunice. În această adresă, un bit specifică dacă este *master* sau *slave*. În ambele cazuri *master*-ul generează un semnal de ceas.

De interfață dispune de proceduri de arbitraj pentru a evita conflictele de transmisie ca urmare a unei arbitrări.

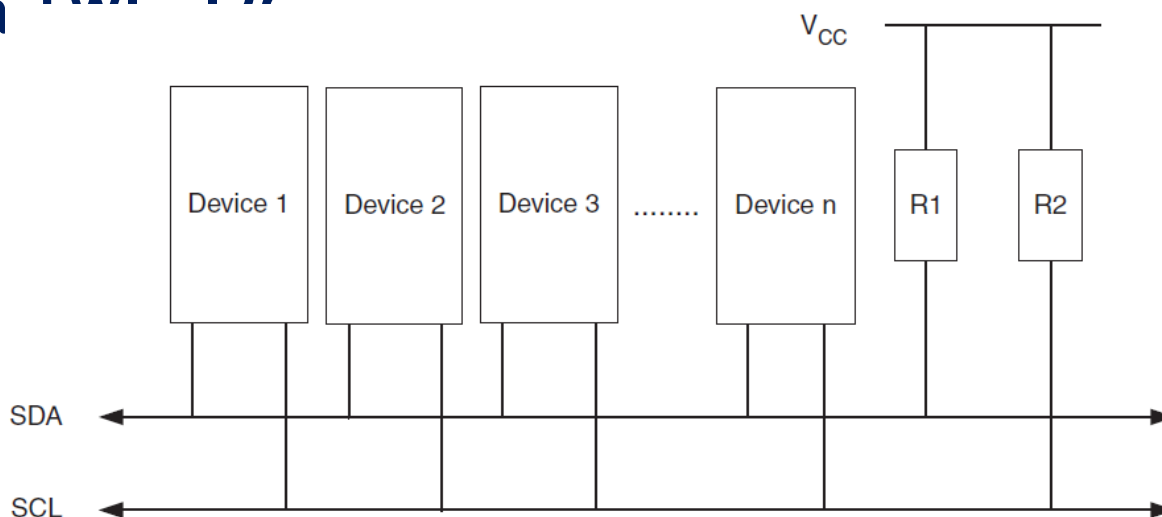
Master

MESSAJ
START
ADRESA
CONFIRMARE
DATE
CONFIRMARE
STOP

# Microcontrolerul ATMEL

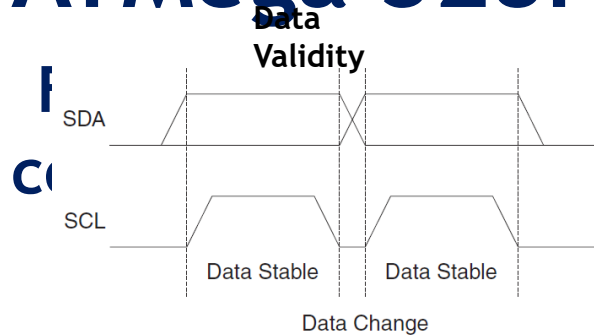
## ATMega 328P

Conectarea circuitelor I/O în comunicația serială



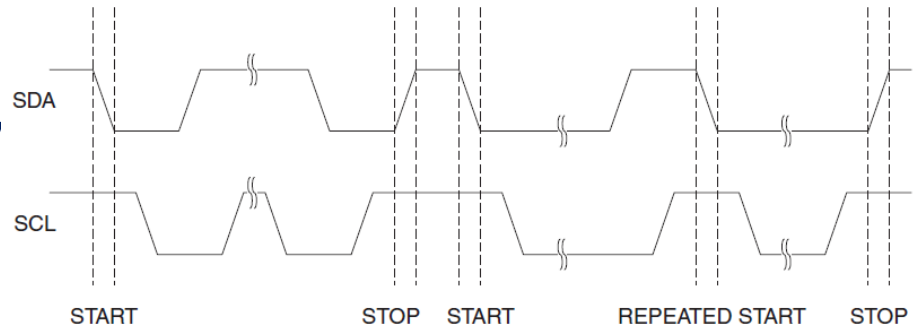
Semnale  
folosite:  
SCL - Serial Clock  
SDA - Serial Data

# Microcontrolerul ATMEL ATMega 328P

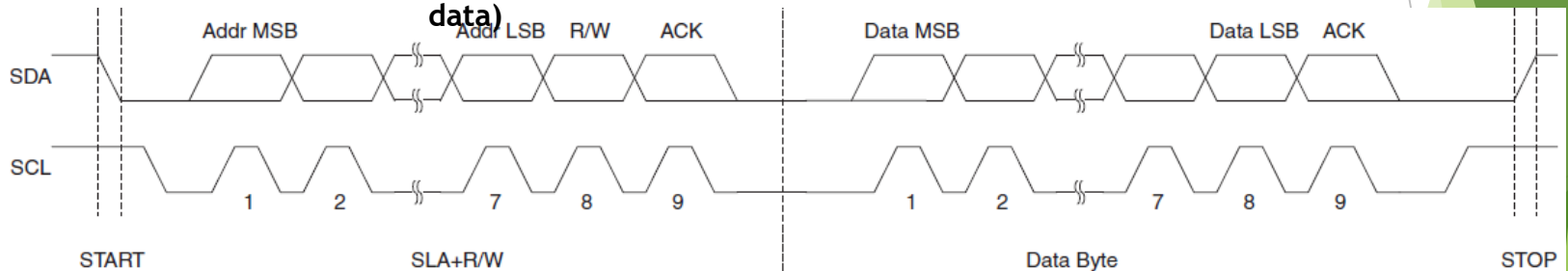


Info  
TW

START, REPEATED START and STOP conditions



Format pachet complet informații (adresa și data)

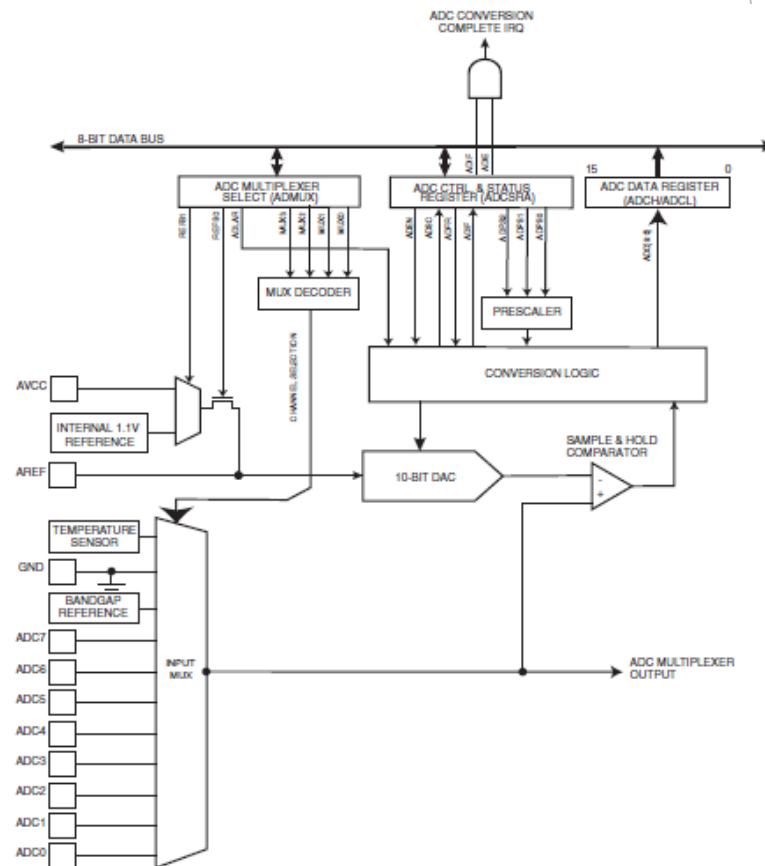


# Microcontrolerul ATMEL ATmega 328P

## Structură generală convertor analog - digital ADC

### Caracteristici:

- 10 biți rezoluție
- 13 - 260 us timp de conversie
- 76,9 kSPS (eșant/sec)
- 6 intrări simple
- 0 - Vcc domeniul tensiunii de intrare ( $V_{FS}$ )
- întrerupere la sfârșit conversie



Se programează prin regiștrii I/O:

ADMUX - ADC Multiplexer Selection Register

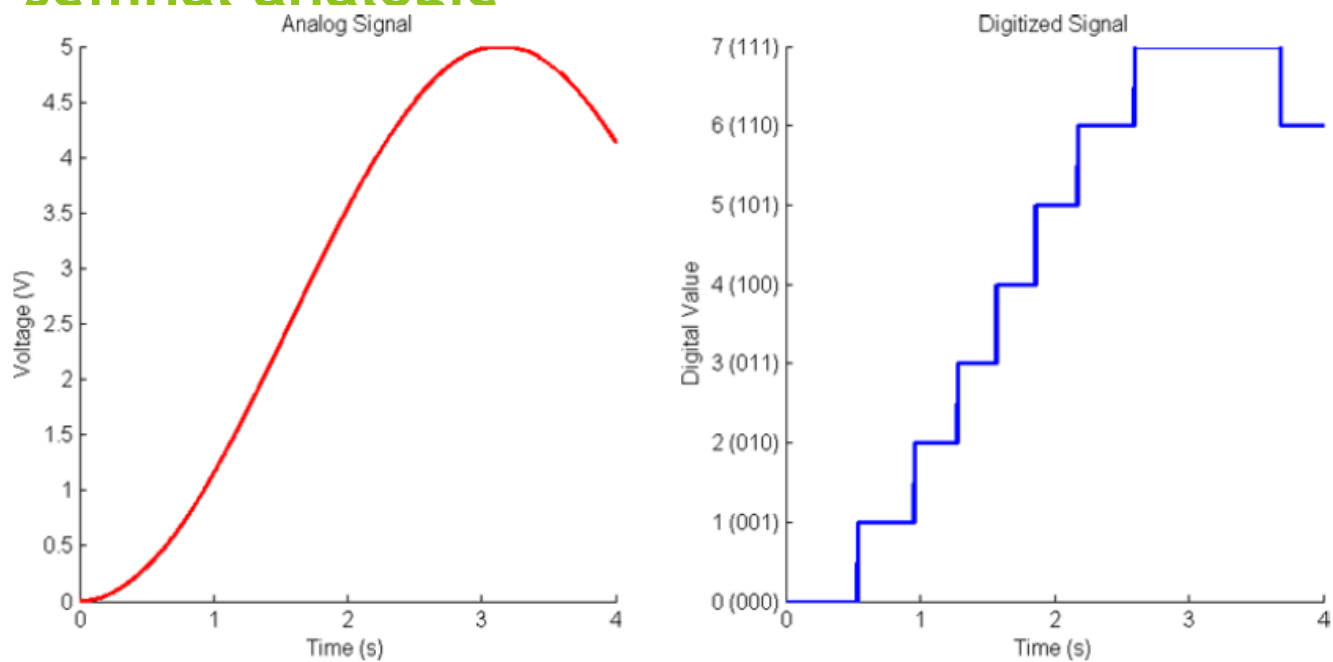
ADCSRA - ADC Control and Status Register

ADC L, H - ADC data Registers

DIDR - Digital Input Disable Register

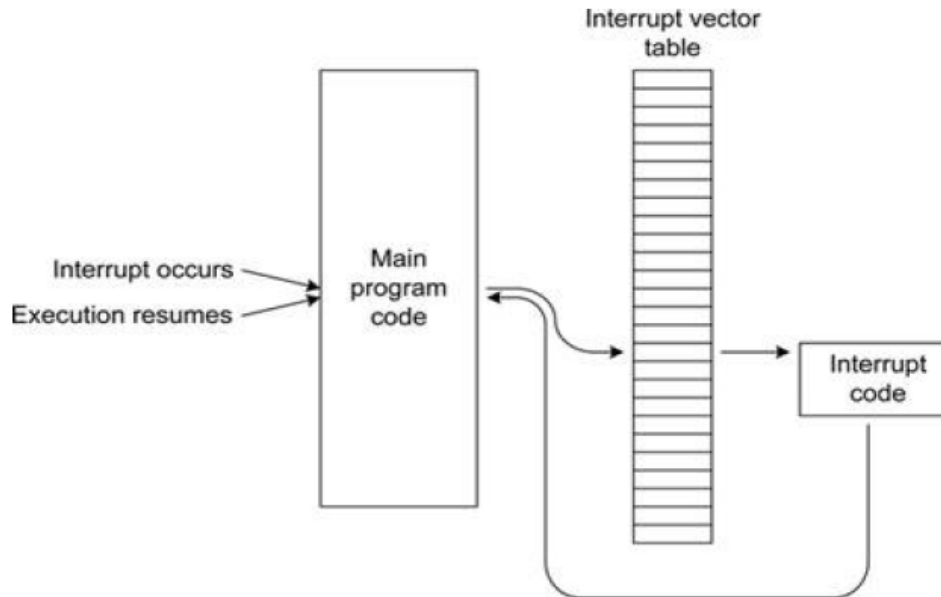
# Microcontrolerul ATMEL ATMega 328P

Conversia analog - digitală pe 3 biți a unui  
semnal analogic



# Microcontrolerul ATMEL ATmega 328P

## Schema de principiu pentru operarea în întreruperi



ATmega 328P gestionează 26 surse de întrerupere interne și externe (2 externe, 3 de la pinii porturilor paralele, 10 de la cele 3 canale timer, 1 de la watchdog, 3 de la canalul USART, 1 de la canalul SPI, 1 de la canalul TWI, 1 de la ADC, 1 de la un bloc comparator, 1 de la EEPROM, 1 de la programarea memoriei ROM - flash și 1 de la RESET).

Pentru fiecare sursă de întrerupere în Tabela vectorilor de întrerupere trebuie scriși 2 octeți (adresa rutinei de tratare).

Programarea mecanismului de întreruperi se face prin următorii regiștri I/O:

MCUCR - Microcontroller Control Register

EICRA - External Interrupt Control Register

EIMSK - External Interrupt Mask Register

EIFR - External Interrupt Flag Register

PCICR - Pin Change Interrupt Control Register

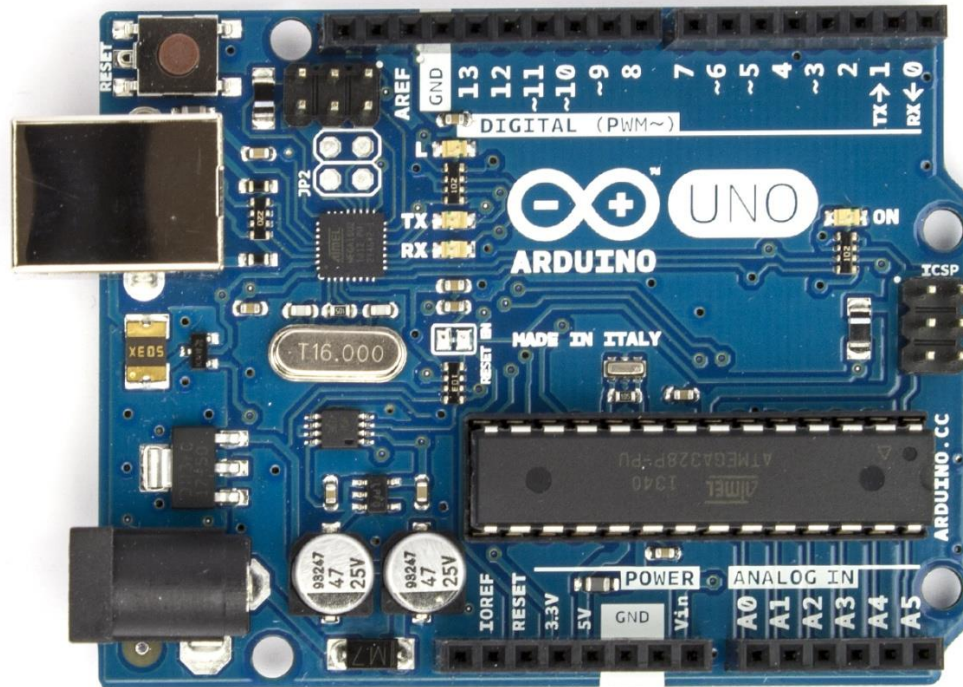
PCIFR - Pin Change Interrupt Flag Register

PCMSK - Pin Change Mask Register



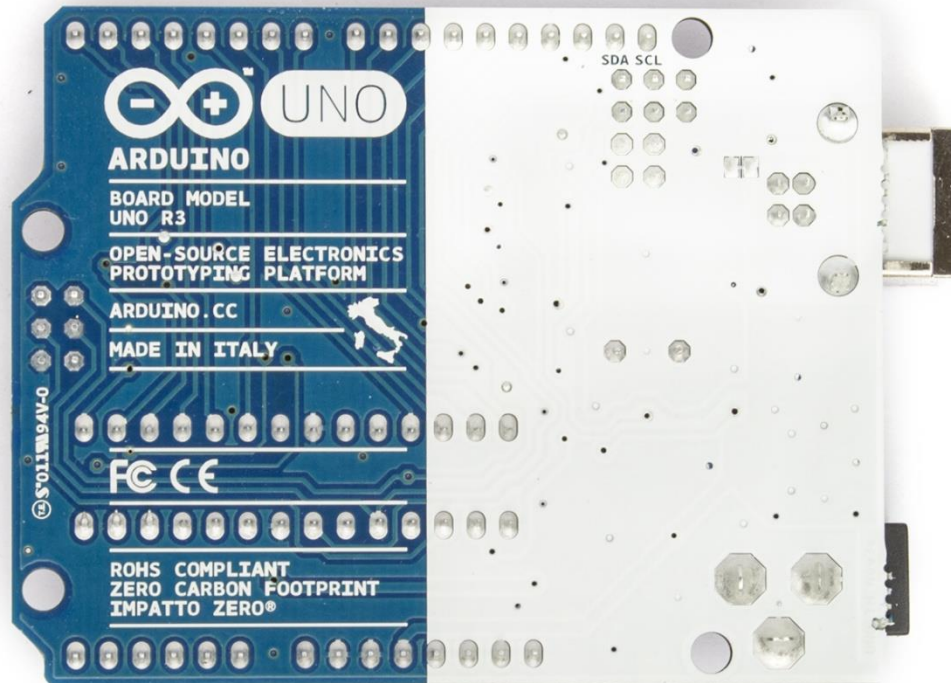
# Microsistemul ARDUINO UNO R3

Fața cu  
piese



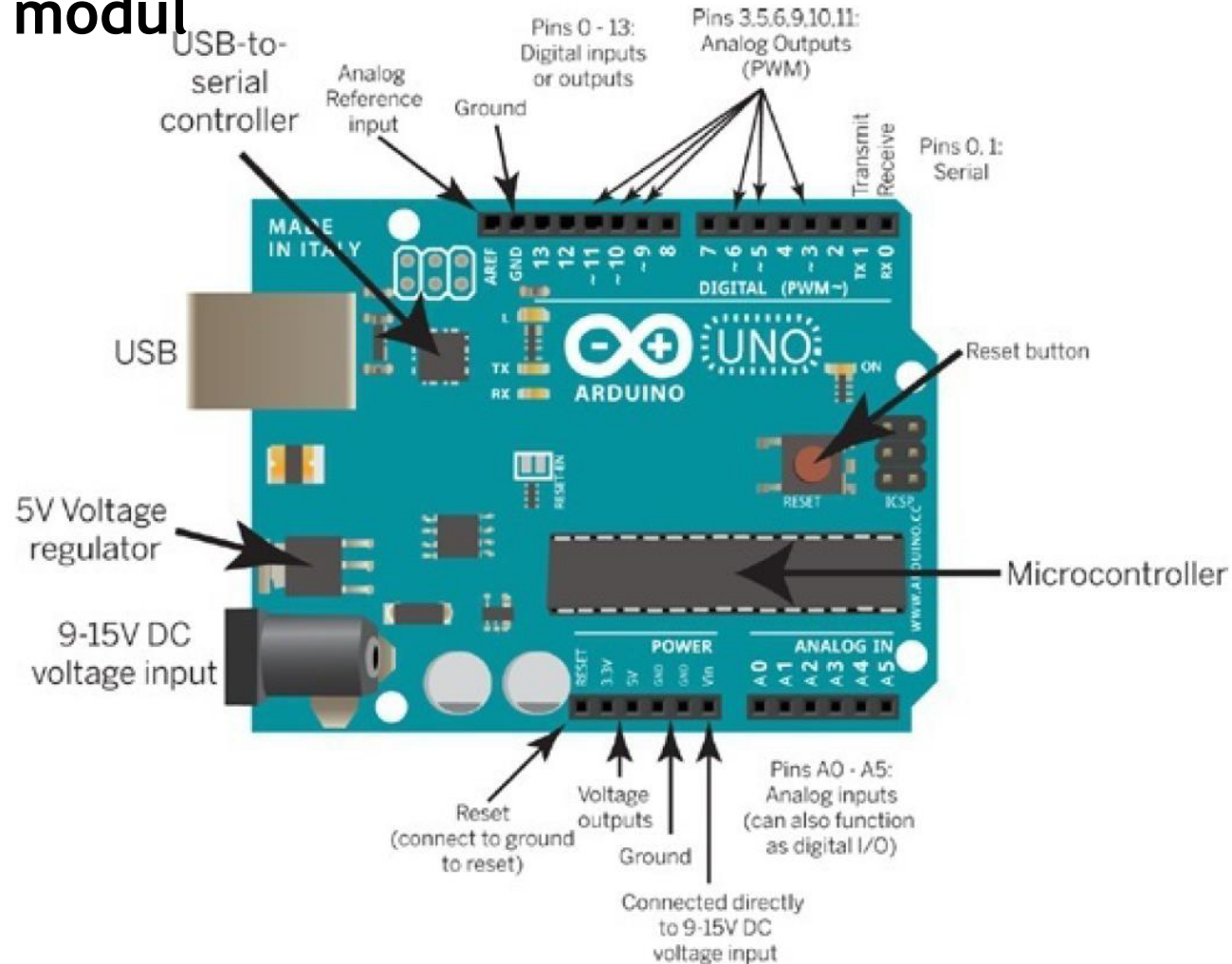
# Microsistemul ARDUINO UNO R3

Spatele  
cablajului



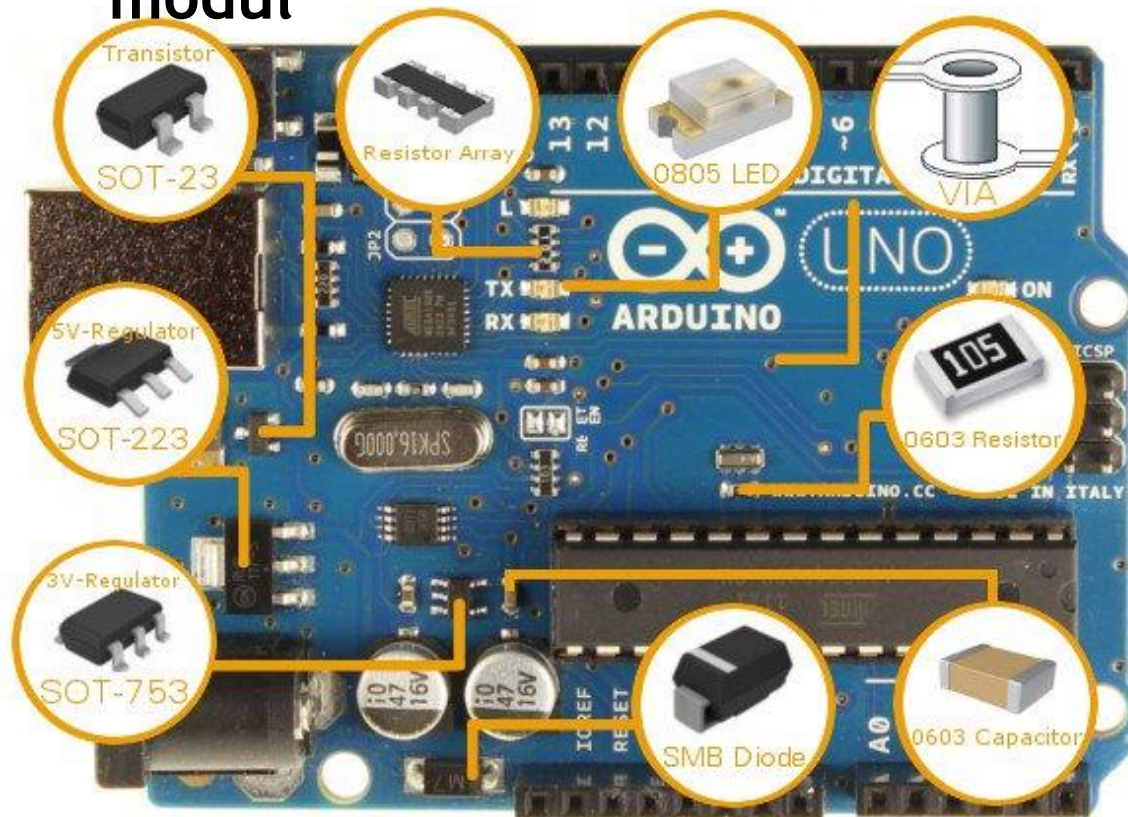
# Microsistemul ARDUINO UNO R3

## Identificarea conectorilor și ai pinilor pe modul



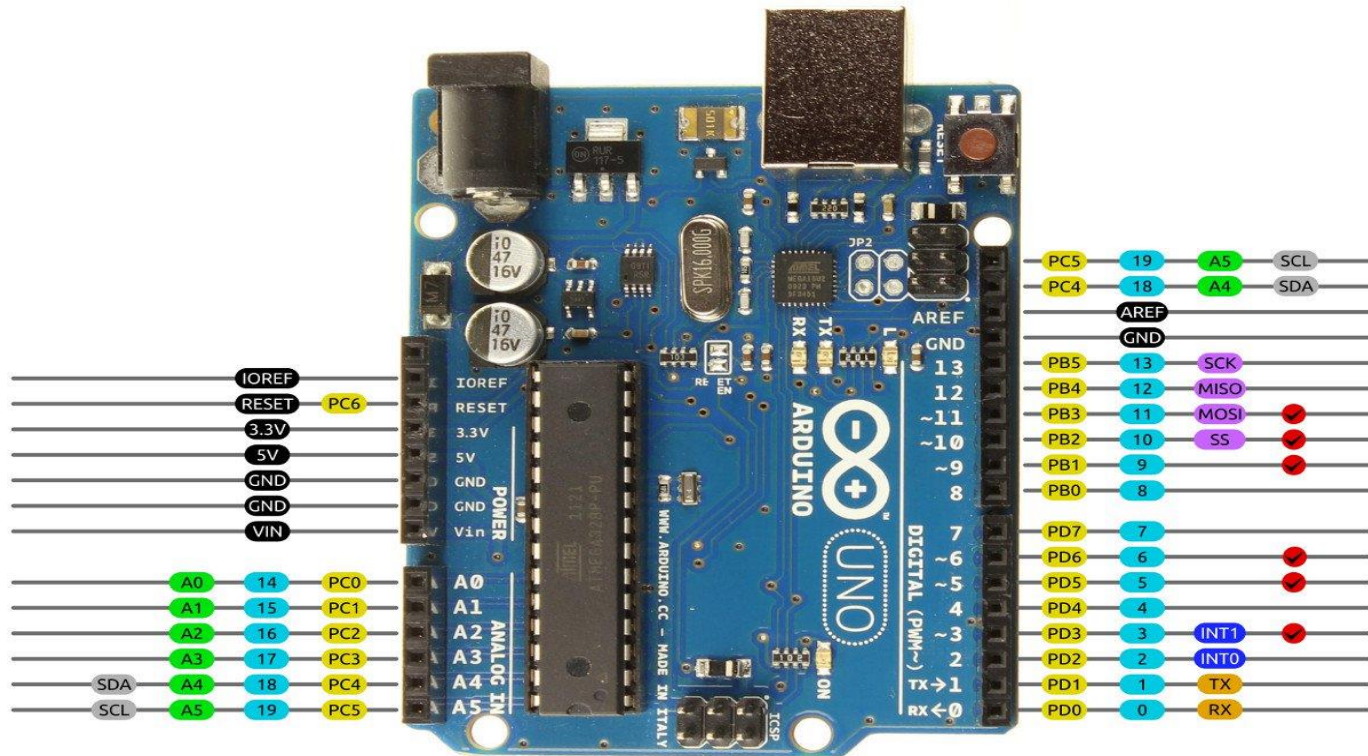
# Microsistemul ARDUINO UNO R3

## Componente electronice pe modul





# Arduino Uno R3 Pinout

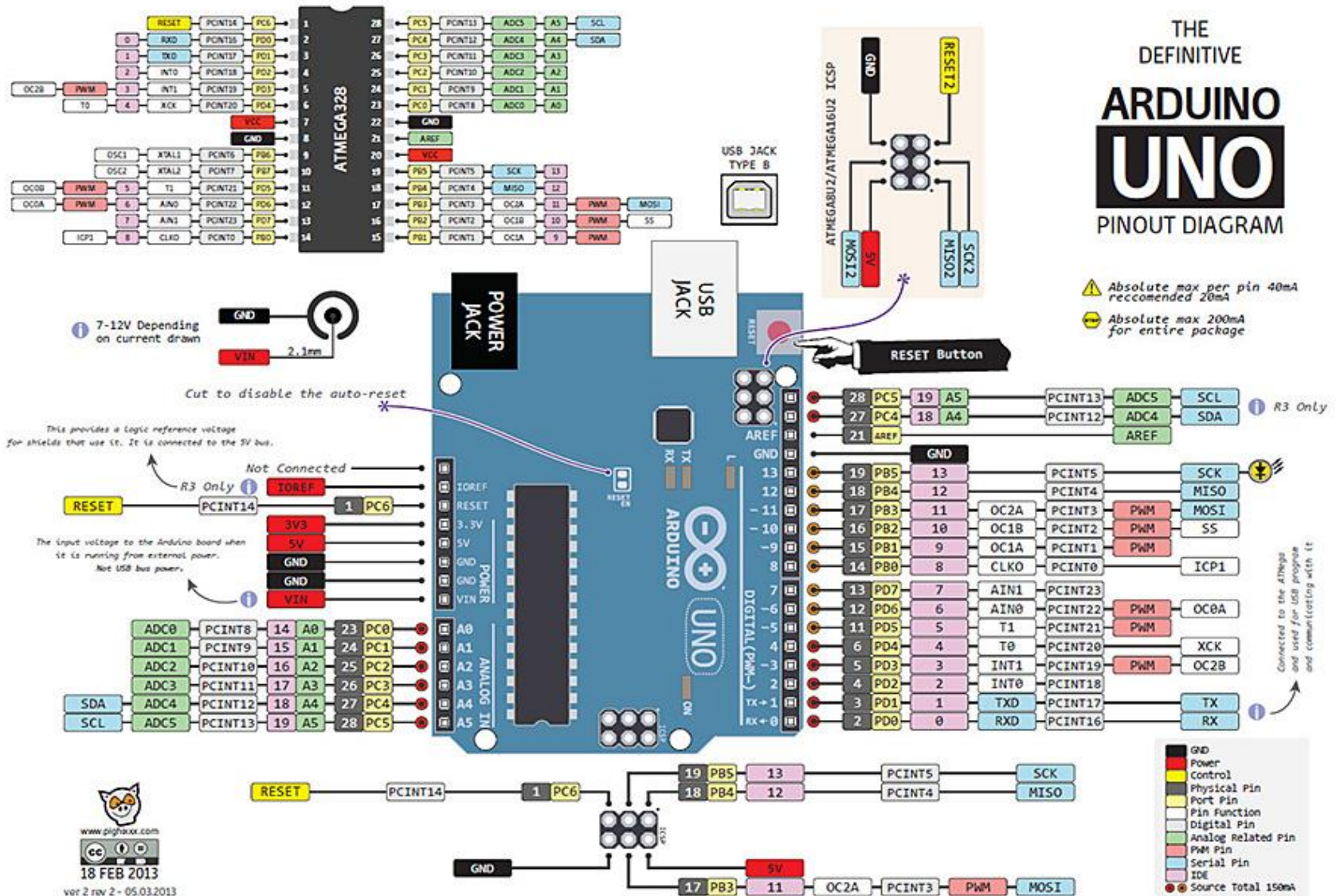


AVR DIGITAL ANALOG POWER SERIAL SPI I2C PWM INTERRUPT



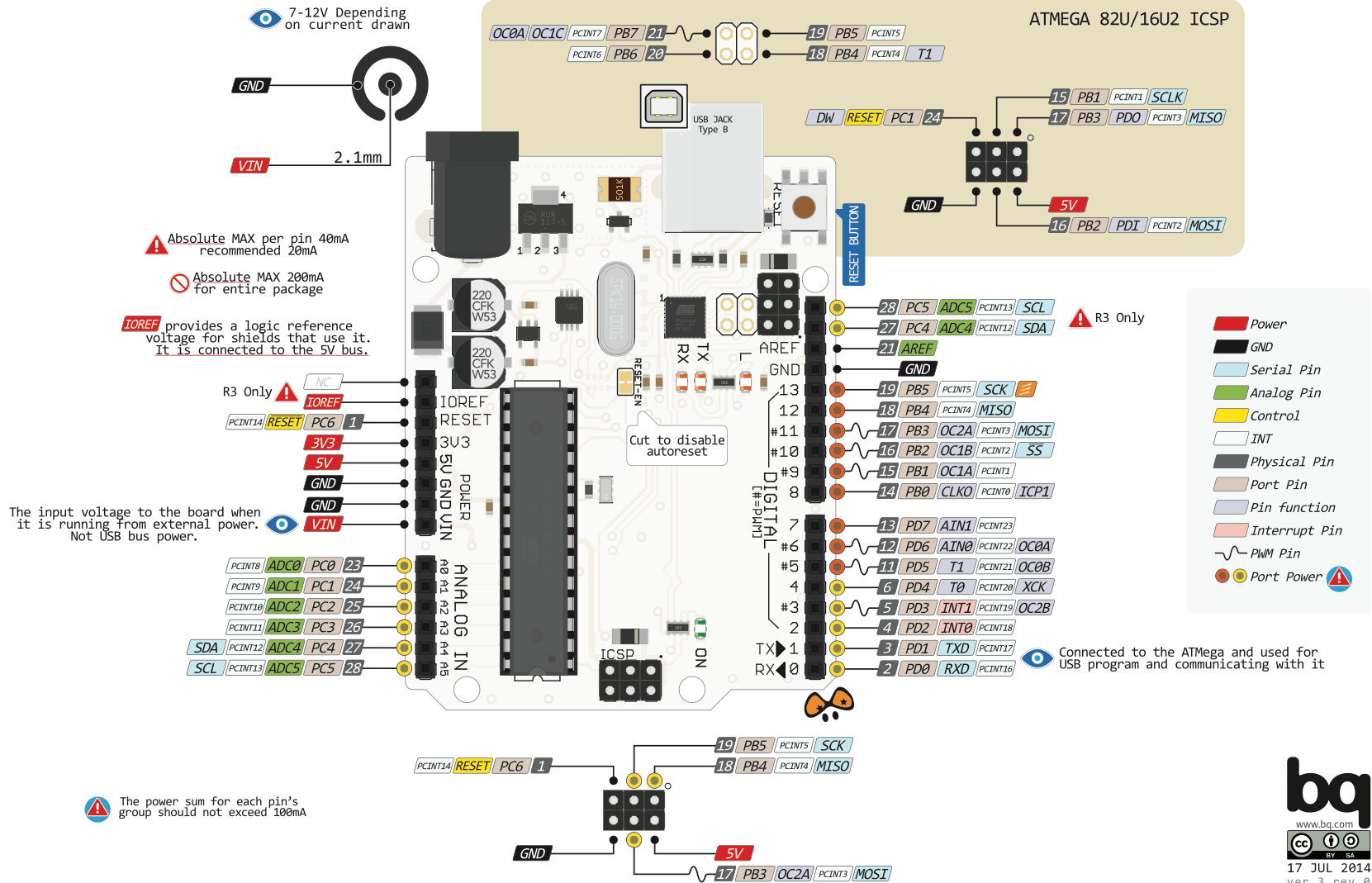
2014 by Bouni  
Photo by Arduino.cc

# Microsistemul ARDUINO UNO R3



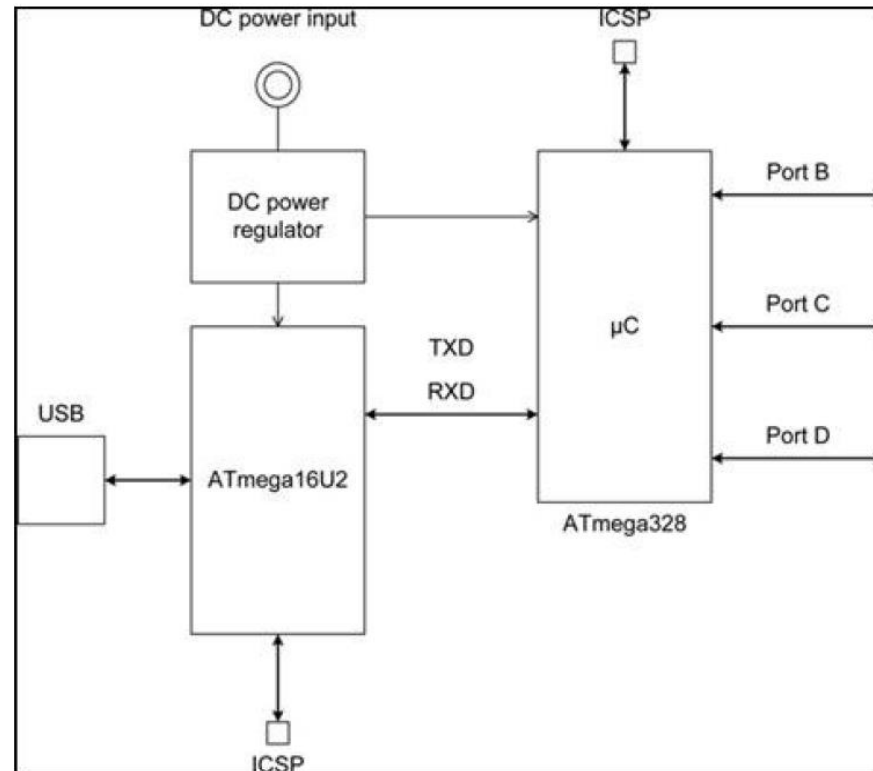
# Microsistemul ARDUINO UNO R3

## UNO PINOUT



# Microsistemul ARDUINO UNO R3

## Schema bloc a modului ARDUINO UNO R3



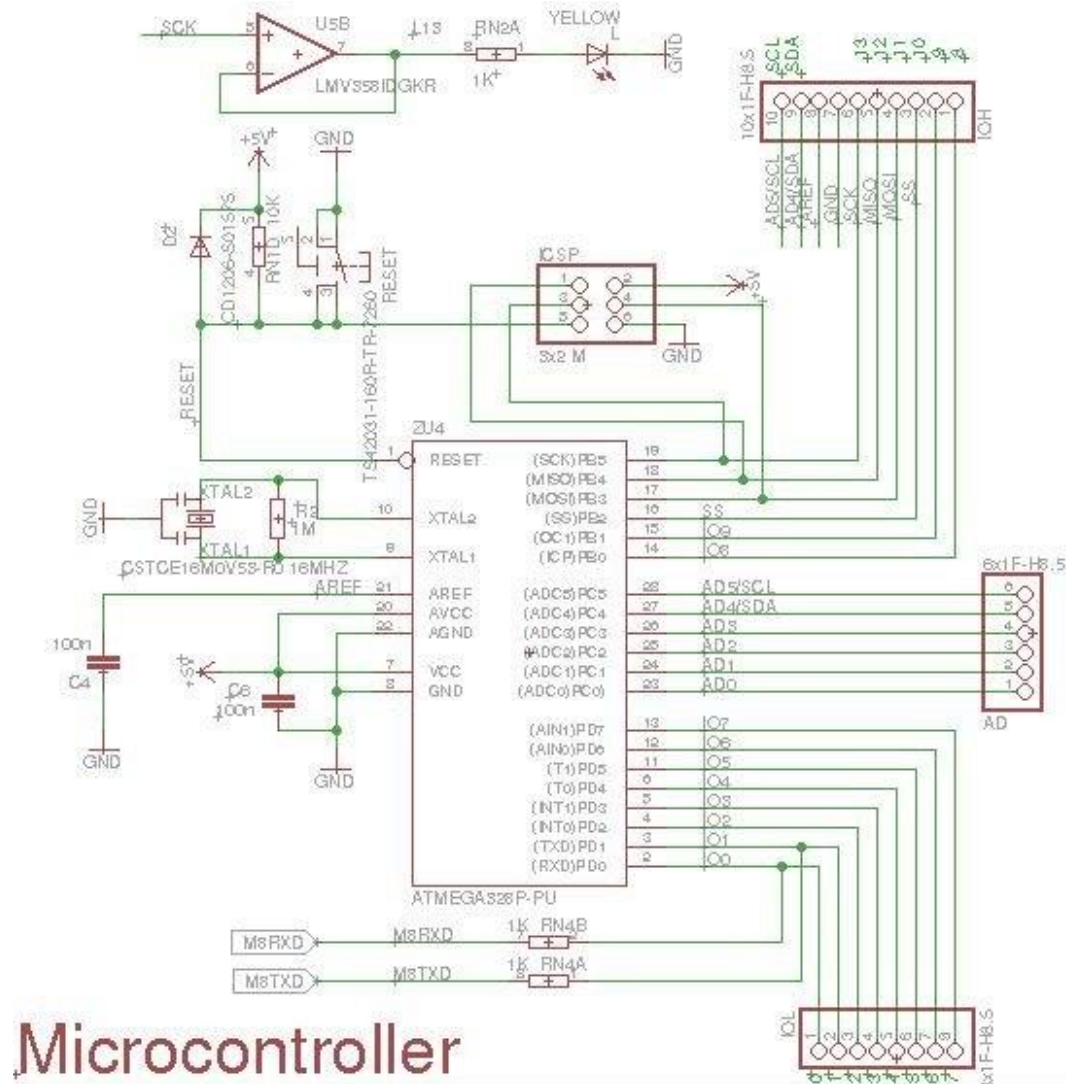
Apar 3 componente majore:

- Microcontrolerul ATmega 328P
- Interfața USB (realizată cu un alt microcontroler ATmega 16U2)
- Sistemul de alimentare

ICSP - In Circuit Self Programming (permite programarea memoriei ROM flash)

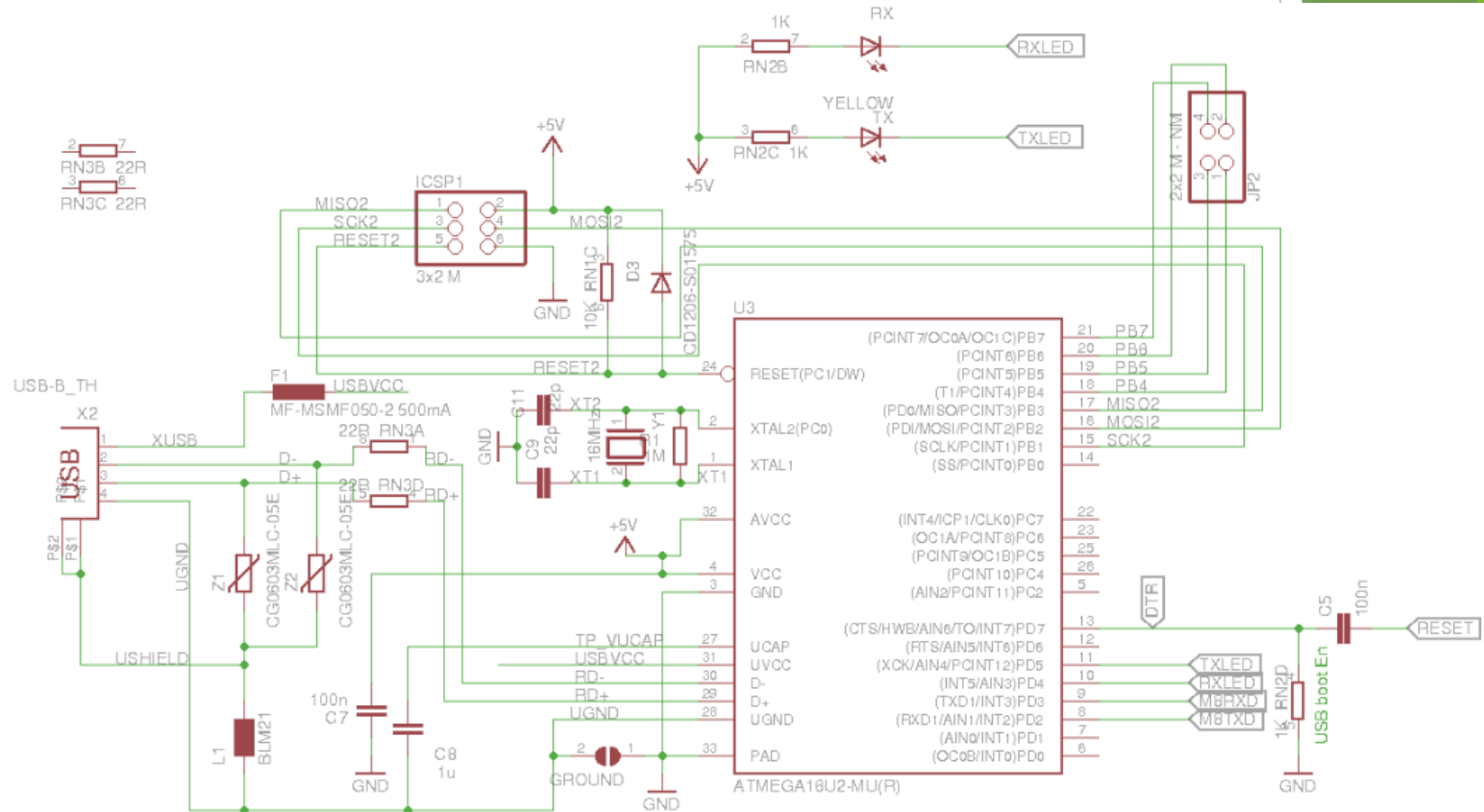


# Microsistemul ARDUINO UNO R3



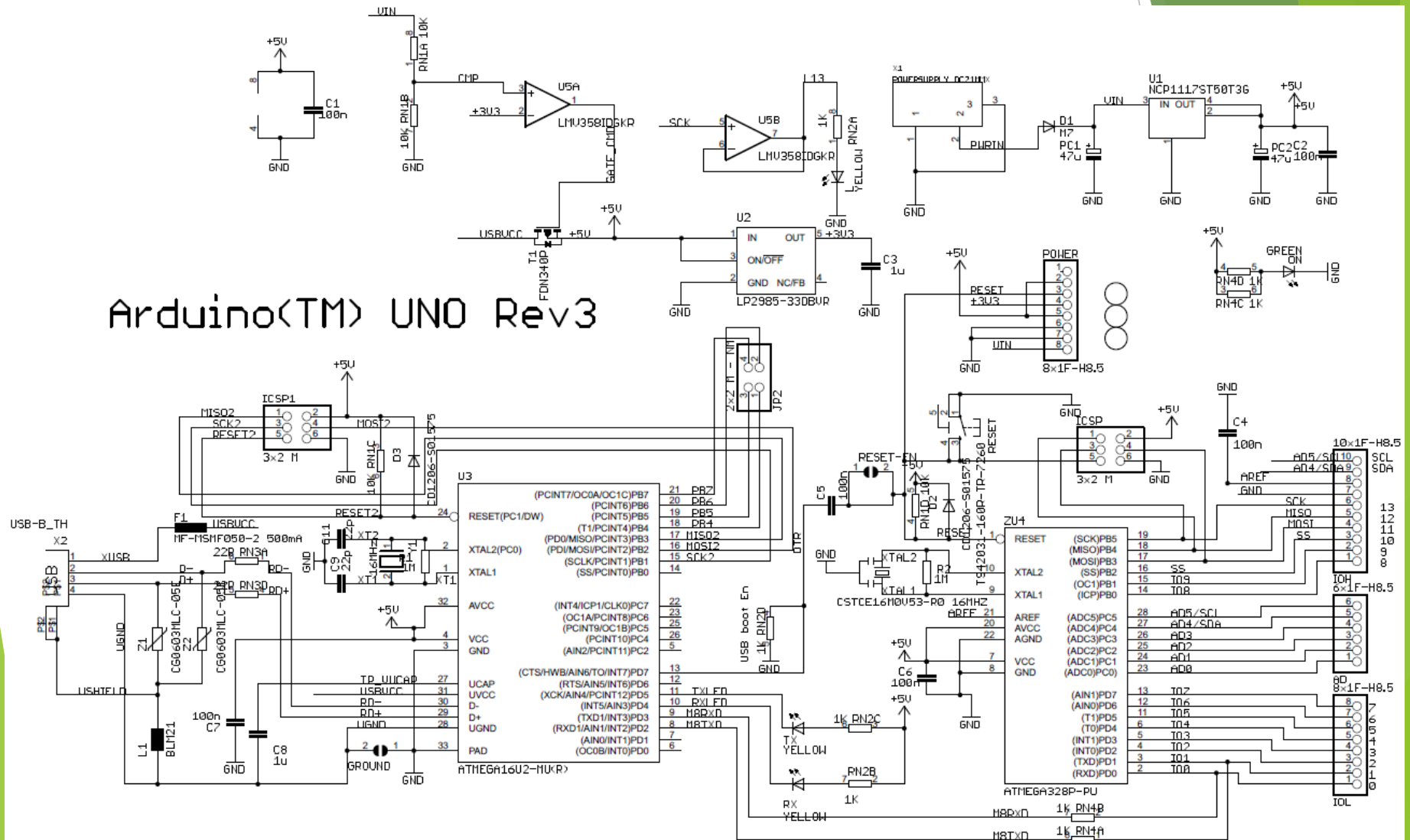
Microcontroller

# Microsistemul ARDUINO UNO R3



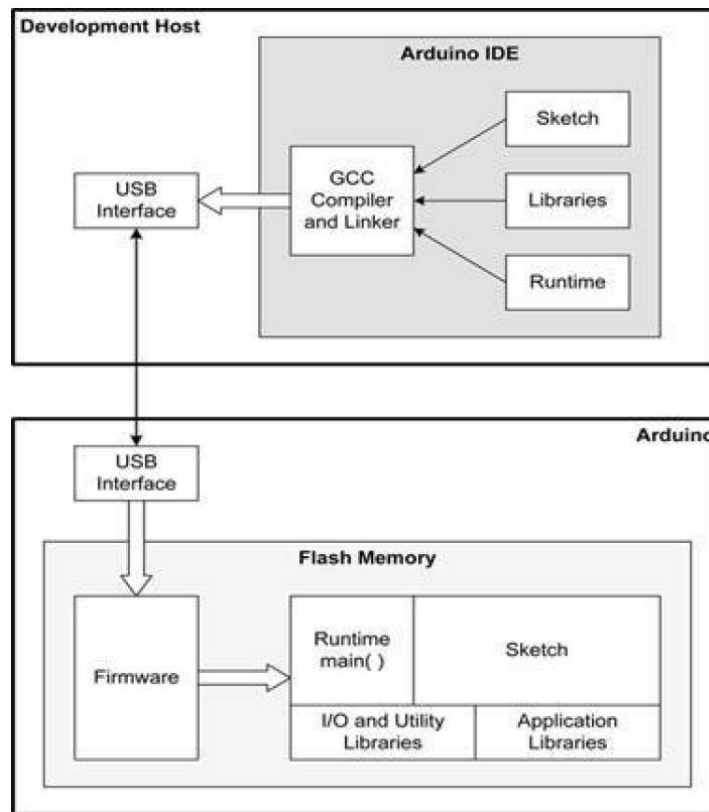
USB Bridge

Arduino(TM) UNO Rev3



# Microsistemul ARDUINO UNO R3

## Dezvoltarea aplicațiilor pentru ARDUINO UNO R3



Vă mulțumim pentru atenție !