Inspectoratul Școlar Județean Argeș

CONCURS DE ROBOTICĂ

RoboSmart

Ediția 1

2017-2018

Universitatea din Pitești

Facultatea de Electronică, Comunicații și Calculatoare

Web info aici:

Sponsori

Continental

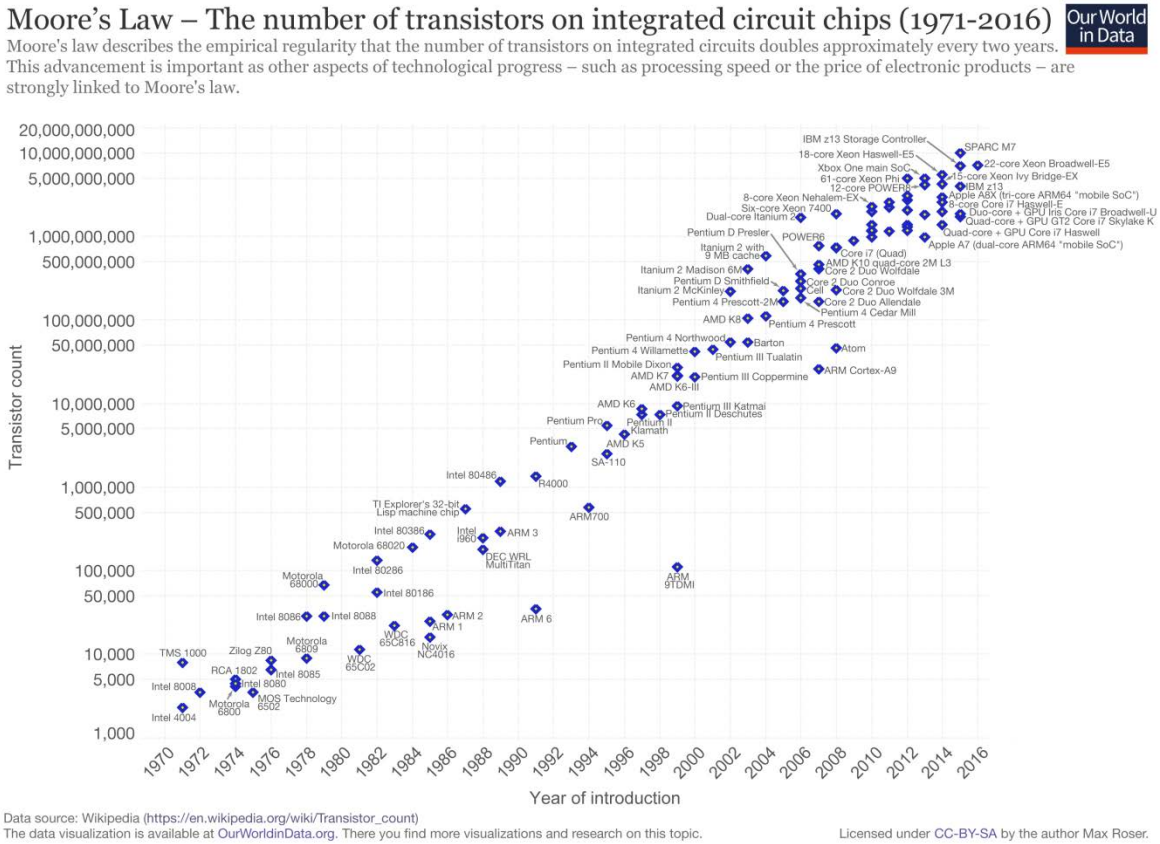
TEXTRON FLEET MANAGEMENT

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 ultimii ani, 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.



Evoluția microprocesoarelor II

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

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

Evoluția microprocesoarelor III

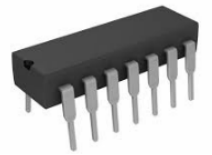
De ce (în ce scop) și când au apărut microprocesoarele?	**În anii 1970, M.E. Hoff de la Intel a reușit înlocuirea a 11 circuite tradiționale cu numai 2-3 circuite, prin folosirea conceptului de logică programată -> primul microprocesor pe 4 biți INTEL 4004**
Care au fost condiționările tehnologice ale 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) înlocuie 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

Clasificarea circuitelor integrate

- **Small Scale Integration**, sau (SSI) – Un circuit conține până la 10 tranzistori; exemple sunt porțile logice (AND, OR, NOT, etc).



- **Medium Scale Integration** or (MSI) – Un circuit conține între 10 și 100 tranzistoare sau de la 10 porți la câteva zeci; exemple sunt decodoarele, demultiplexoarele, multiplexoarele, număratoarele, etc.

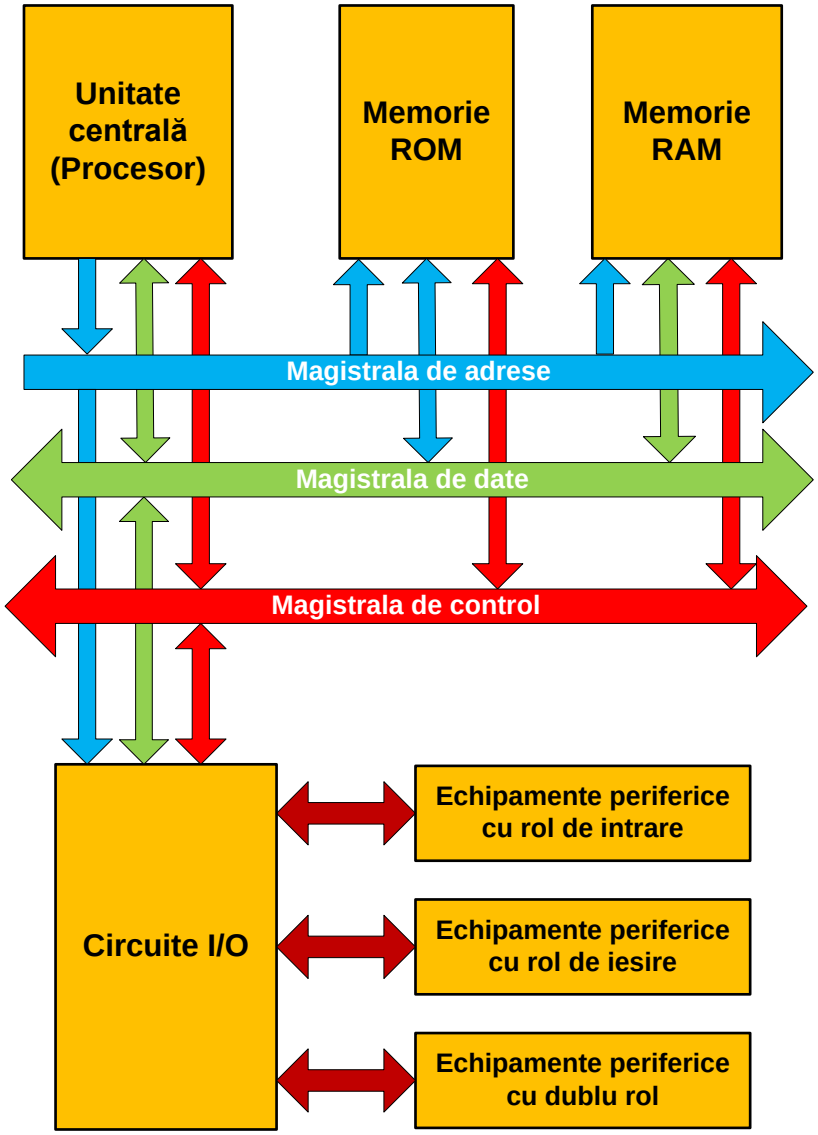
- **Large Scale Integration or** (LSI) – un circuit conține între 100 și 1,000 tranzistoare sau sute de porți; exemple sunt unitățile aritmetico-logice, memoriile, circuitele de intrare/ieșire.

- **Very-Large Scale Integration** or (VLSI) – un circuit conține între 1,000 și 10,000 tranzistoare sau mii de porți logice; exemple sunt procesoarele, memoriile de capacitate mare și circuitele logice programabile.

- **Super-Large Scale Integration** or (SLSI) – un circuit conține între 10,000 și 100,000 tranzistoare și le regăsim sub formă de microprocesoare, microcontrolere sau circuite I/O.

- **Ultra-Large Scale Integration** or (ULSI) – un circuit conține peste 1 million de tranzistoare fiind utilizat în calculatoare în CPUs, GPUs, videoprocesoare, microcontrolere, FPGAs sau circuite I/O complexe.

Structura generală a unui sistem de calcul

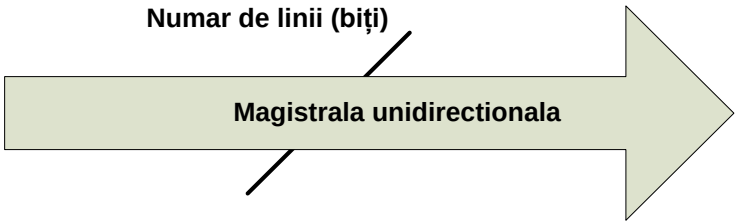


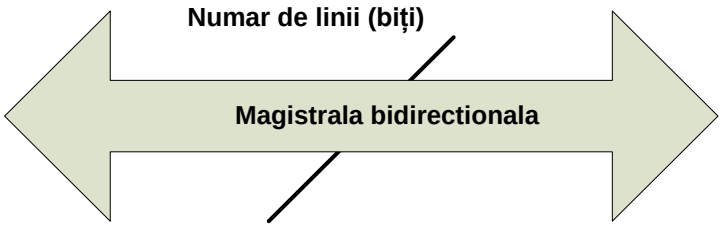
Magistrale (Bus)

- Magistrala: un grup de linii electrice pe care se transmit semnale electrice digitale care au o aceiași funcție logică în sistemul de calcul.

- Magistralele pot fi :

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

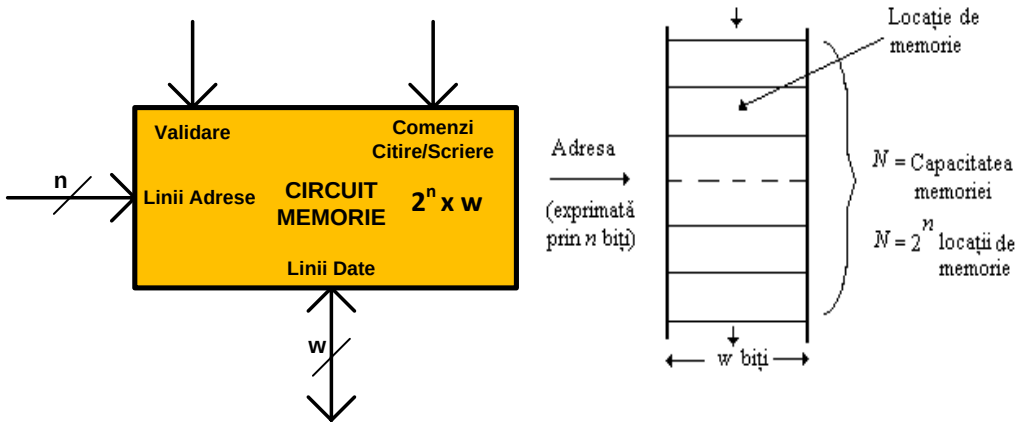




Într-un sistem de calcul se întâlnesc 3 categorii de magistrale:
- magistrala de adrese (unidirecțională)
- magistrala de date (bidirecțională)
- magistrala de comenzi sau control (bidirecțională – unele linii au o direcționalitate, altele altă direcționalitate)

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 **locație de memorie**.
Totalitatea locațiilor de memorie formează circuitul de memorie.
O locație de memorie este formată dintr-un număr de celule de memorie, câte una pentru fiecare bit memorat, care sunt accesate simultan prin aceiași informație de adresă.
Numărul de biţi conţinuţi în locaţia de memorie poate fi:
4 biți = semioctet (*nibble*);
8 biţi = octet (*Byte*);
16 biţi = cuvânt (*word*);
32 biţi = dublu cuvânt (*double word*);
64 biţi = dublu dublu cuvânt (*double double word*).



Organizarea circuitului de memorie se exprimă prin **produsul neefectuat** dintre numărul de locații și numărul de biți dintr-o locație 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) x 8 (biți/locație)=16kx8=16ko=16kB, 2^{22} x 16=4Mx16=4Mw.

Pentru a adresa o memorie cu capacitatea de *N* locaţii sunt necesare *n* linii de adresă: $n = [\log_2 N]$ biţi.
Exemplu: O memorie cu organizarea 1MB=1Mo=1M x 8=2^{20} x 8 trebuie să fie adresată cu $n = \log_2 2^{20}$ = 20 biţi (adresarea memoriei se face prin 20 linii de adresă) şi prezintă 8 linii de date.

Capacitatea circuitului de memorie se exprimă prin **produsul efectuat** dintre numărul de locații și numărul de biți dintr-o locație 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) x 8 (biți/locație)= 2^{17}b=128kb, 4Mx16= 2^{22}x2^4b= 2^{26}b=64Mb

Tipuri de circuite de memorie

***Memorii cu conţinut fix* (conţinutul 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, în momentul inscripţionării (programării).

EPROM (*Erasable Programmable Only Memory*). Poate fi ştearsă cu raze ultraviolete puternice şi apoi reînscrisă cu alte valori.

EEPROM (*Electricaly Erasable Read Only Memory*). Poate fi ştearsă şi reprogramată prin semnale electrice. Este mai lentă decât memoria RAM, dar după decuplarea alimentării datele înscrise sunt reţinute pe timp nelimitat.

FLASH. Este o categorie specială de memorie EEPROM, care se poate şterge şi reînscrie bloc cu bloc, fără a fi necesară trimiterea datelor bit cu bit

NVRAM (*Non Volatile RAM*). Este o memorie RAM nevolatilă, care reţine datele şi după decuplarea sistemului de la alimentarea cu energie electrică, are consum mic fiind alimentată de la baterie sau acumulator.

Memorii cu conţinut temporar

Memorii statice (au nevoie doar de prezenţa tensiunii de alimentare pentru memorarea informaţiei):

***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 crescut, capacitate mică, cost mai mare.

Memorii dinamice (necesită şi operaţia de reîmprospătare – refresh pe lângă prezenţa tensiunii de alimentare pentru memorarea informaţiei):

***Memoria dinamică* (DRAM).** Celula de memorie e construită pe baza unui condensator atașat unui tranzistor unipolar care memorează „0“ sau „1“. Necesită reîmprospătare. Avantajele sunt capacitatea mare, consumul mic și costul de asemenea mic. Dezavantajul este o viteză de acces și de lucru mai mică.

SDRAM (*Syncronous RAM*). Este o memorie rapidă care lucrează sincron cu procesorul (operează la tactul extern al acestuia) fără timpi de aşteptare.

DDR-SDRAM (*Double Data Rate*). Funcţionează utilizând ambele fronturi ale impulsurilor de ceas (crescător şi descrescător), în acest fel practic dublând viteza de transfer a datelor la memorie

DDR-SDRAM 2 (DDR2). Utilizează o tehnologie denumită ”4 *bit prefetch*”, astfel încât poate transmite 4 biţi pe ciclu de ceas, pe fiecare linie de bit

DDR-SDRAM 3 (DDR3), DDR-SDRAM 4 (DDR4)

Tabel cu adresele unor blocuri de memorie frecvent utilizate

Unităţile de măsură pentru memorie sunt exprimate în octeţi:

$1k = 2^{10} = 1024$ octeţi;

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

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

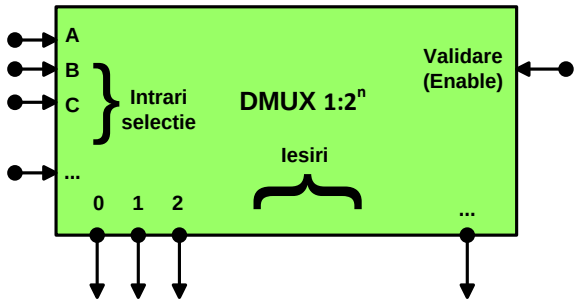
Adresa memoriei în hexa-zecimal	Mărimea memoriei în octeţi şi kocteţi
0000	0
000F	16
0010	16+1
00F0	240
00FF	256
0100	256+1
01FF	512
0200	512+1
0300	768+1
0400	1024+1=1k+1
0800	2k+1
0C00	3k+1
1000	4k+1
2000	8k+1
3000	12k+1
4000	16k+1
8000	32k+1

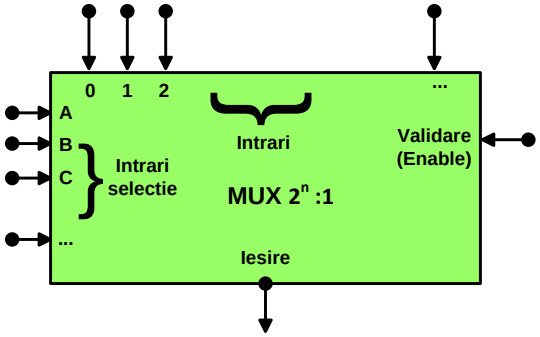
Adresa memoriei în hexa-zecimal	Mărimea memoriei în octeţi şi kocteţi
FFFF	64k
1 0000	64k+1
1 FFFF	128k
2 FFFF	192k
3 FFFF	256k
7 FFFF	512k
8 FFFF	576k
9 FFFF	640k
A FFFF	704k
B 0000	704k+1
B FFFF	768k
C 0000	768k+1
C FFFF	832k
D 0000	832k+1
D FFFF	896k
E 0000	896k+1
E FFFF	960k

Adresa memoriei în hexa-zecimal	Mărimea memoriei în octeţi şi kocteţi
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.



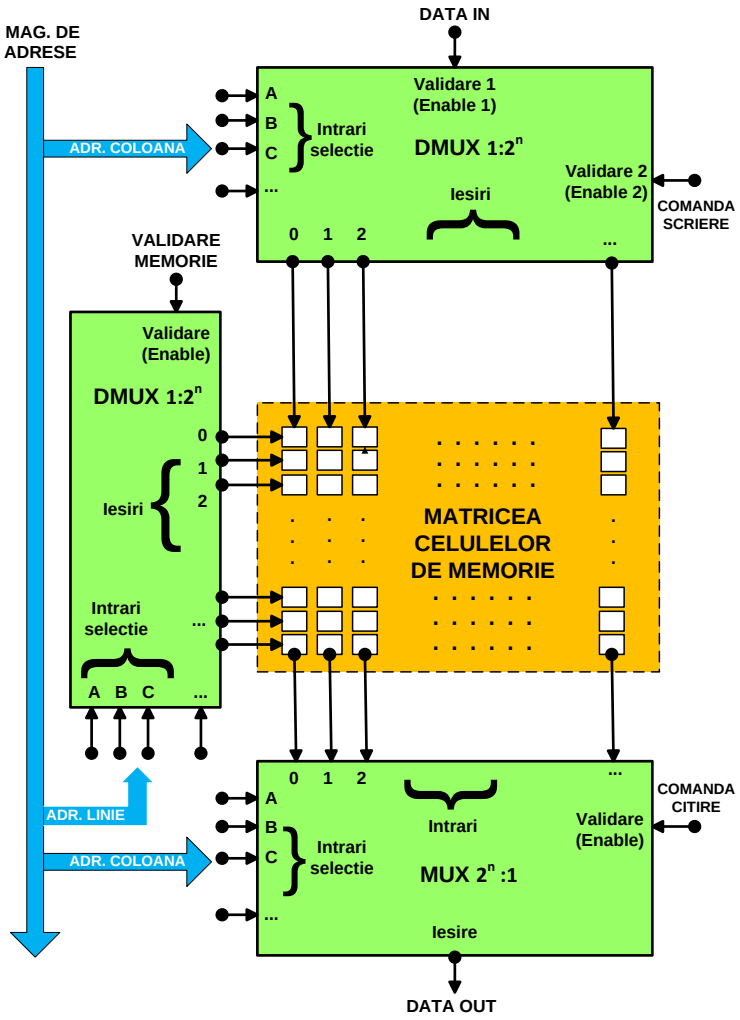


Demultiplexorul este un circuit digital care distribuie (repartizează) semnalul digital aplicat pe o intrare (denumită validare - enable) şi îl transferă către una dintre ieşirile circuitului, identificată prin intermediul semnalelor digitale aplicate intrărilor de selecţie.
O altă interpretare a funcţionării este aceea de a identifica o combinaţie de semnale binare aplicate intrărilor de selectie prin activarea corespunzătoare a uneia dintre ieşiri, atât timp cât circuitul este validat.
DMUX-ul are rol de distribuitor sau repartitor, respectiv de identificator de coduri binare.
Notaţia circuitului DMUX indică raportul de distribuţie (repartiție) 1:2^n corespunzător circuitului.

Multiplexorul este un circuit digital care selectează semnalul digital aplicat pe una dintre intrări, identificată prin intermediul semnalelor digitale aplicate unor intrari de selectie şi îl transferă către ieşirea unica a circuitului, atât timp cât circuitul este validat.
MUX-ul are rol de selector sau de concentrator de semnale digitale.
Notaţia circuitului MUX indică raportul de selecţie 2^n:1 corespunzător circuitului.

Memoria - Schema bloc – Funcţionare I

Schema bloc a unei memorii cu organizare 2^n x 1



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

1. **Citirea unei locaţii** prin acţiunile:

- transmiterea adresei locaţiei 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 care selectează coloana pe care se află celula căutată;
- aplicarea comenzii de citire care conduce la apariţia datei căutate la linia DATA OUT.

Operaţia de citire se derulează în mod similar atât la memorii ROM, cât şi la memorii RAM.

2. Scrierea unei locaţii prin acţiunile:

- transmiterea adresei locaţiei 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 memoriei, 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 locaţia adresată.

Operaţia de scriere este proprie memoriilor RAM.

Totuşi, şi în memoriile ROM se poate realiza ”operaţia de scriere”, dar aceasta poartă denumirea de programare şi se realizează în condiţii speciale.

Duratele de execuţie ale operaţiilor 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). Totuşi, tehnologia bipolară consumă multă energie şi nu conduce la densităţi mari de integrare pe suprafaţa de siliciu.

Memoria - Schema bloc – Funcţionare II

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

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

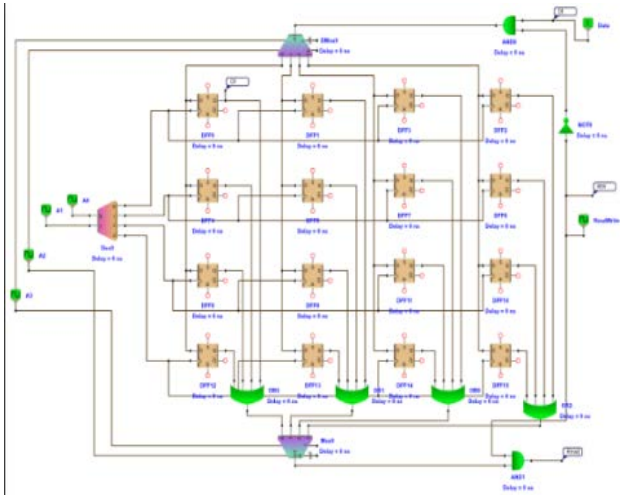
La memoriile ROM realizate cu tranzistori unipolari, celulele de memorie se formează prin sarcina electrică aplicată pe 2 terminale speciale ale tranzistorilor, diferenţa de potenţial care apare conducând la efectul de memorare. La memoriile ROM uniplare este posibilă ştergerea şi reprogramarea acestora. Operaţiile de ştergere 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.

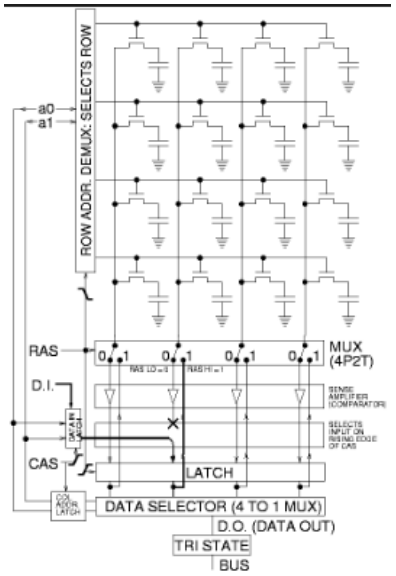
La memoriile SRAM (realizate cu tranzistori bipolari sau unipolari), celulele de memorie sunt formate din bistabili sau latch-uri, realizate cu 4 – 6 tranzistori bipolari sau unipolari, elemente care au proprietatea de memorare cât timp se aplică tensiunea de alimentare, fără a mai fi nevoie de acţiuni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

În memoriile DRAM, celulele de memorie sunt condensatori care însoţesc tranzistorii unipolari. Aceştia au capacităţi foarte mici (<1pF) şi au pierderi mari. Stocarea sarcinii electrice pe armăturile acestora, de fapt ceea ce produce şi efectul de memorare prin diferenţa de potenţial care apare, este o sarcină dificilă. La aceste circuite se impune efectuarea repetitivă unei acţiuni suplimentare – reîmprospătarea sau refresh-ul prin care se reîncarcă sarcina electrică pierdută la armături. Operaţia de refresh este suplimentară în raport cu operaţiile de citire/scriere şi micşorează timpul de disponibilitate al memoriei pentru operaţiile cu procesorul. Pentru o funcţionare corectă a memoriilor DRAM se impune atât menţinerea tensiunii de alimentare, cât şi efectuarea ritmică a refresh-ului (la intervale de timp de până la 2 – 4 ms).

La memoriile DRAM o celulă de memorie foloseşte un singur tranzistor unipolar. Prin diferenţa numărului de tranzistori folosiţi pentru formarea unei celule de memorie rezultă eficienţa mult mărită a memoriilor DRAM faţă de SRAM.

Comparativ cu toate tehnologiile actuale de fabricare ale memoriilor, circutele DRAM sunt cele mai avantajoase atât din punct de vedere al preţului de cost (pentru 1 bit), cât şi din punct de vedere al densităţilor obţinute pe suprafaţa de siliciu (număr de tranzistori/unitatea de suprafaţă). Din păcate, progresele tehnologice legate de micşorarea timpului de acces la operațiile cu memoriile sunt mici și de mai mult timp a apărut o diferență între vitezele procesoarelor moderne și răspunsul memoriilor semiconductoare.





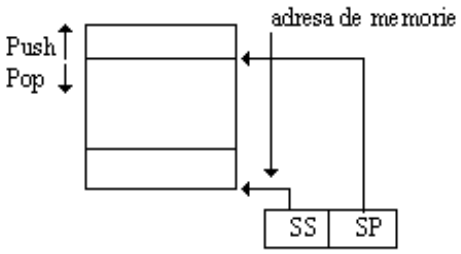
Stiva

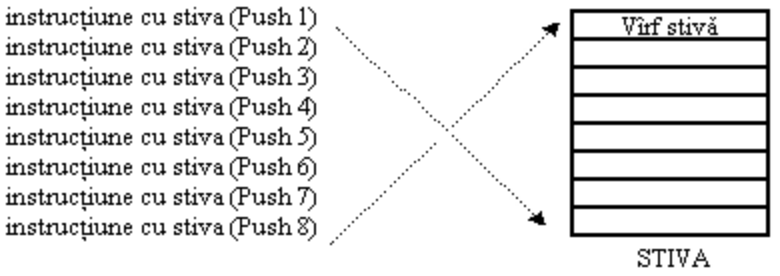
- Stiva este o zonă de memorie a cărei dimensiune poate, după tipul procesorului, să crească odată cu adresa de memorie sau să scadă odată cu adresa de memorie. Caracteristica sa este că datele sale pot fi citite doar strict în ordinea inversă în care au fost scrise (LIFO – last in first out).

- Stiva este implementată în memorie şi se adresează prin registrul de stivă (RS) care conține baza stivei și registrul 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 şi scrierea cuvântului dorit a fi reţinut în stiva la noul vârf al stivei. Scoaterea unui cuvânt din stivă este realizată prin copierea lui din vârful stivei şi incrementarea registrului SP cu o locație.

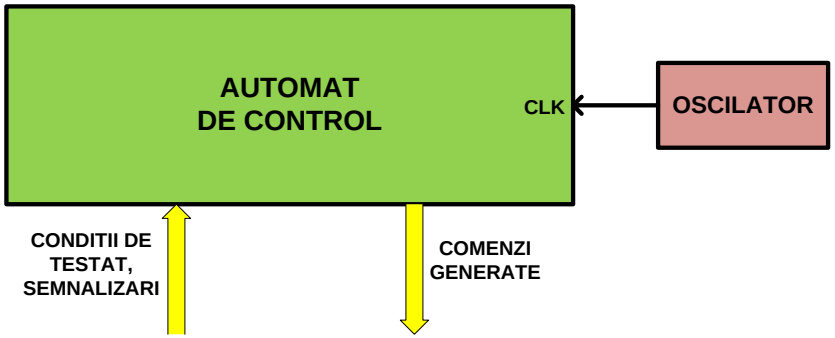
- Stiva conţine de obicei adresa de revenire în programul apelant, indicatori, parametri.



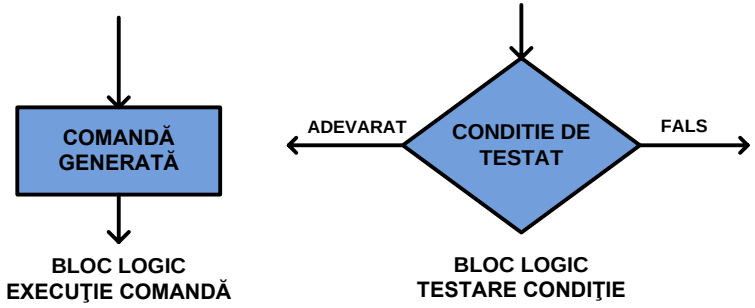


Automate de control

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 (condiţii de testat sau diverse semnalizări) şi, funcţie de acestea şi de algoritmul implementat (stările prin care trece), să genereze comenzi în consecinţă sub formă de semnale digitale.



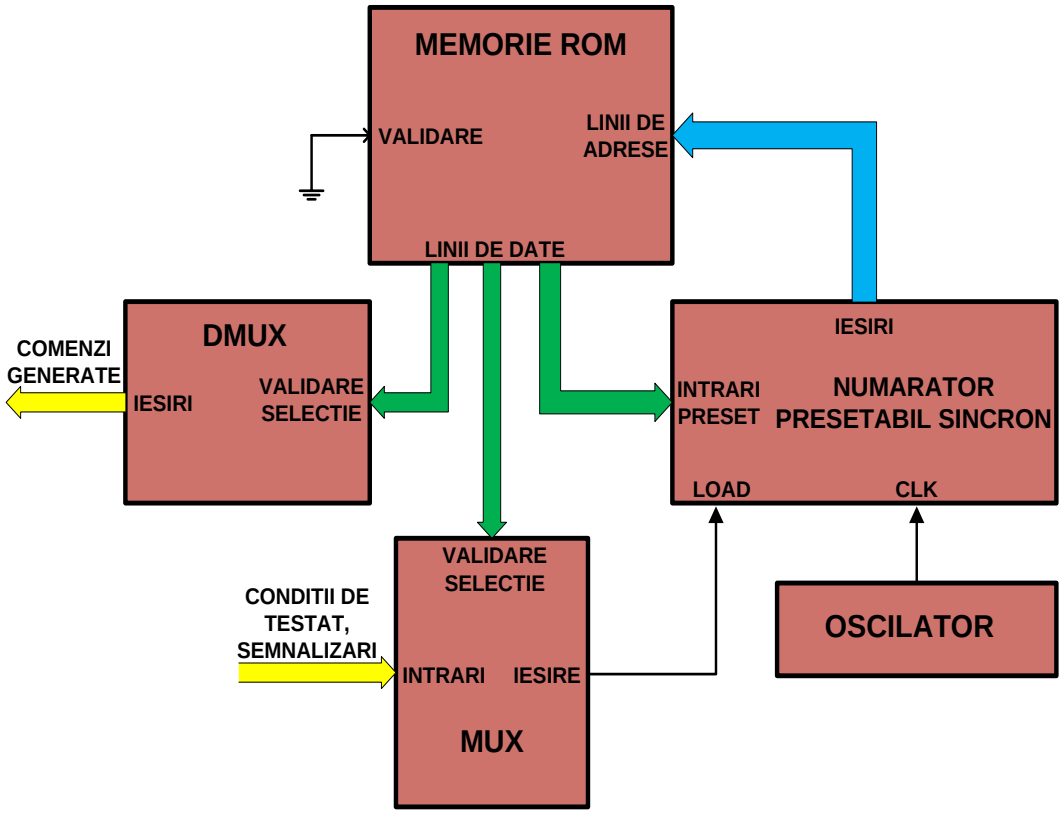
Schemele logice sunt alcătuite din blocuri logice elementare de tip execuţie comenzi, respectiv testare condiţie:



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 locaţii conţin secvenţe de biţi care transcriu blocurile din schema logică implementată. Fiecare locaţie de memorie conţine implementarea binară a unui singur bloc logic elementar din algoritm.
Parcurgerea schemei logice se face prin adresarea memoriei, realizată prin intermediul unui numărător cu dublu rol: numărare, respectiv presetare (încărcare cu o valoare de la care poate să reînceapă procesul de numărare) şi care este acţionat de un semnal de ceas provenit de la un oscilator.
Blocurile logice de execuţie 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 condiţiilor sau a semnalizărilor primite sunt imlementate prin MUX-ul din figură, comanda fiind asigurată de un alt câmp de date corespunzător de la memoria ROM. Funcţie de valoarea logică a condiţiei testate, ieşirea acestui circuit comandă funcţia de numărare la numărător, specifică blocurilor logice de execuţie comenzi sau a blocurilor de test pentru situaţia de condiţie neîndeplinită, respectiv funcţia de încărcare paralelă – presetare (comanda LOAD) pentru blocurile de test în situaţia condiţiei testate îndeplinite – situaţie corespunzătoare salturilor.
Se pot face astfel parcurgeri liniare sau ramificate în schemele logice.
O adresă se aplică memoriei ROM la fiecare semnal de ceas generat de oscilator, în acest ritm executându-se fiecare bloc din schema logică (algoritm).



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 şi logică și regiștri și automatul UC – unitatea de comandă);

4. dispozitive/circuite periferice de ieşire/output (O).

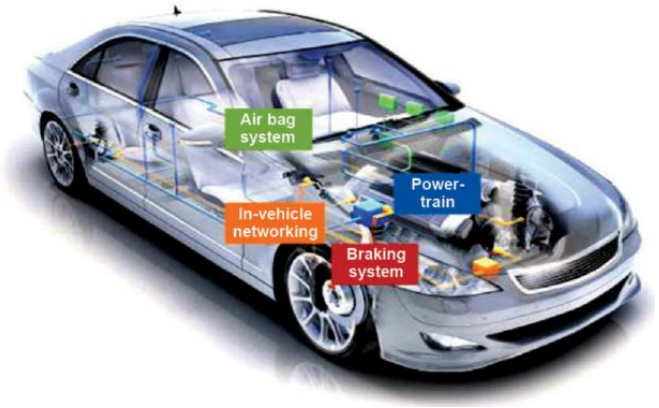
(desen)

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:

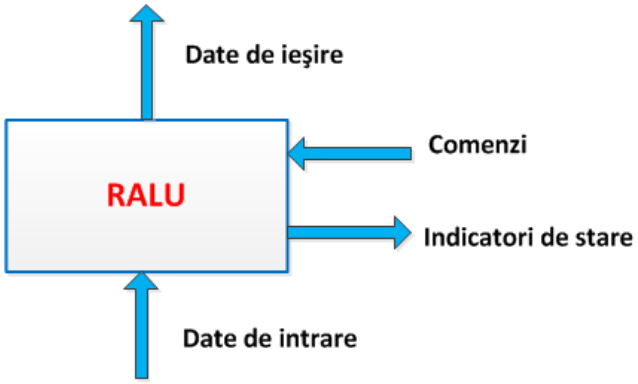
- abilitatea de a executa o singură sarcină specifică
- rulare repetată un program
- reacție la evenimente în timp real
- siguranța în funcționare
- miniaturizarea
- viteza
- cost mic, consum redus



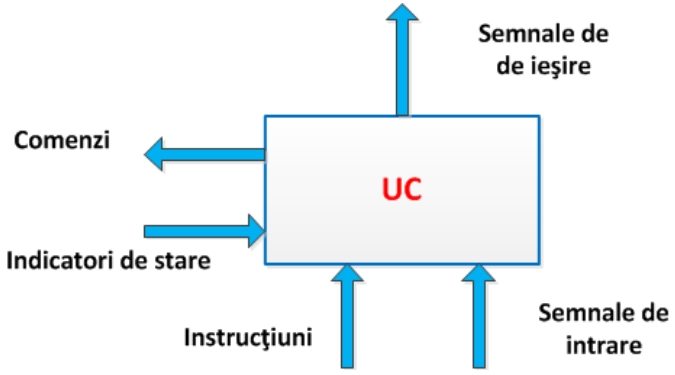
Microprocesorul I

Microprocesorul este un sistem microprogramabil compus din două automate interconectate astfel încât să comunice bidirecţional:
Automatul de Comandă (AC) ;
Automatul de Prelucrare (AP) .
La acestea se adaugă şi o serie de **mecanisme specifice**:
- mecanismul registrului PC (Program Counter sau Instruction Pointer) – pentru adresarea şi preluarea (fetch) a instrucţiunilor de executat
- mecanismul registrului SP pentru adresarea stivei şi rezolvarea apelului, respectiv revenirilor din subrutine
- mecanismul de întreruperi pentru preluarea şi tratarea evenimentelor prioritare ce apar pe parcursul execuţiei programelor
- mecanismul de reset sau iniţializare (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 obţinere o reducere timpului de obținere a a datelor din memorie de către procesor
- mecanismul semnalului de ceas

Prin interconectarea celor două automate se obţine un sistem microprogramat care funcţionează astfel:
1. Automatul de comandă generează o secvenţă de comenzi (prin execuţia unor microinstrucţiuni) către automatul de prelucrare.
2. Automatul de prelucrare procesează datele de intrare în funcţie de comenzile primite şi transmite indicatori despre natura rezultatelor obţinute care influenţează evoluţia ulterioară a automatului de comandă.



Automatul de prelucrare se numește tradiţional **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 biţi**; exprimă dimensiunea operanzilor (în biţi) ce pot fi procesaţi printr-o singură instrucţiune, de obicei reprezentând şi mărimea magistralei de date; conform acestui criteriu cu cât un procesor este organizat pe un număr mai mare de biţi, cu atât este mai puternic;

- **după arhitectură**; pot fi procesoare Von Neumann (Princeton) sau Harvard;

- **după setul de instrucţiuni**; pot fi procesoare CISC (complex instruction set computer) sau RISC (reduced instruction set computer); setul de instrucțiuni diferă de la procesor la procesor reprezentând un element de individualizare a acestora;

- **după numărul de instrucţiuni executate simultan;** pot fi procesoare scalare (o singură instrucţiune executată la un moment dat) sau superscalare (mai multe instrucţiuni executate la un moment dat, eventual în stadii diferite);

- **după modul, locul utilizării şi al accesabilităţii pentru mai mulţi utilizatorii;** sunt procesoare pentru laptop, desktop, workstation, server, mainframe, supercalculator;

- **după clasificarea lui Michael Flynn:**

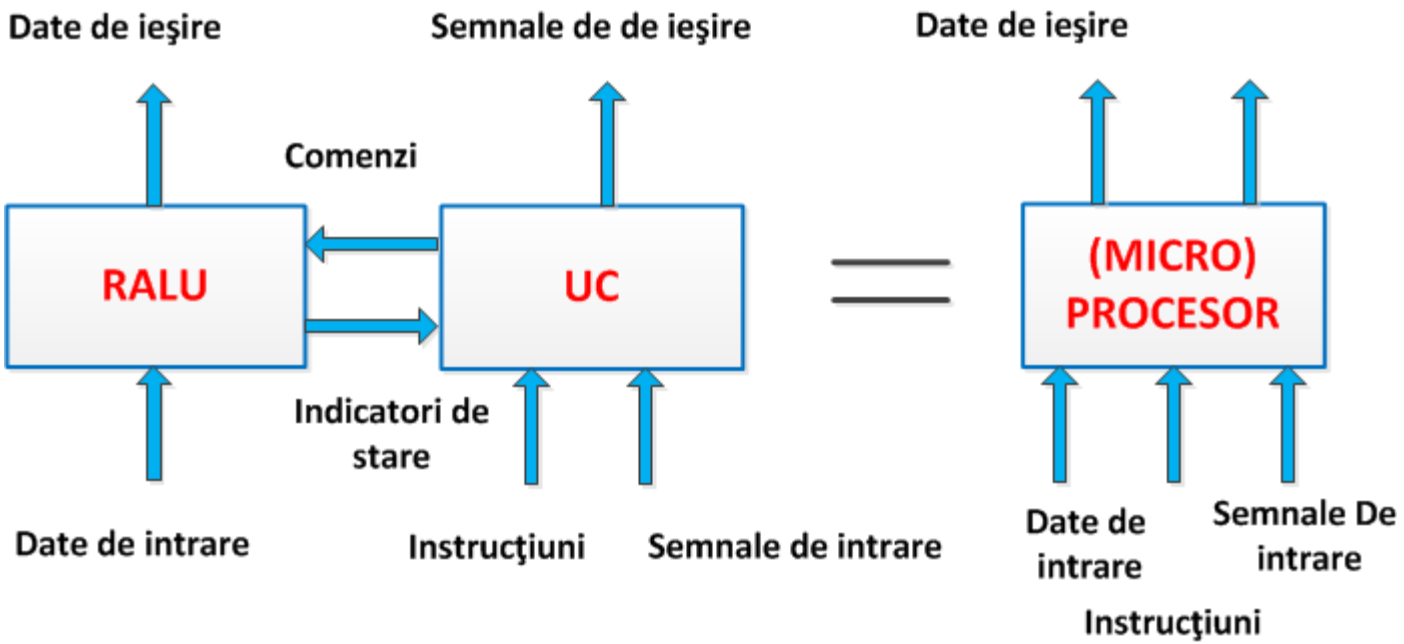
SISD (Single Instruction Single Data – o singură instrucţiune, o singură dată de prelucrat); sunt microprocesoarele clasice Von Neumann;

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

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

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

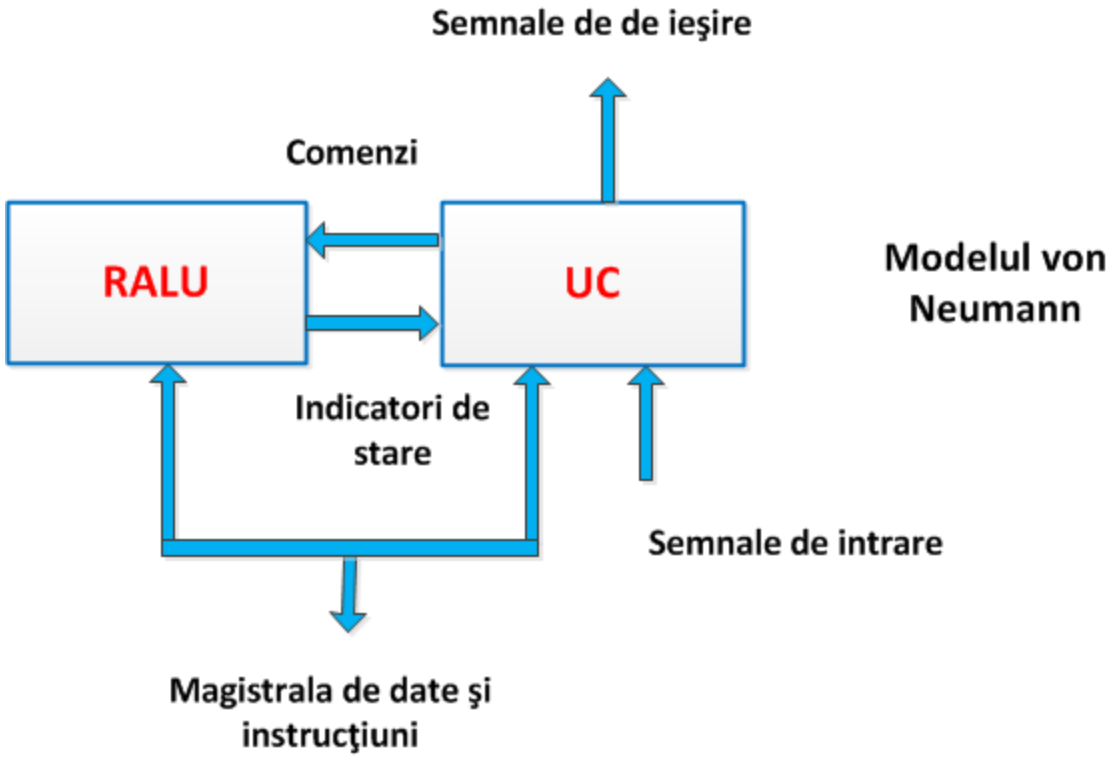
Microprocesorul III
Modelul Harvard



Modelul (arhitectura) Harvard se caracterizează prin separarea căii de transfer pentru datele de intrare/ieşire faţă de cea a instrucţiunilor (liniile de transfer ale datelor sunt separate de cele pentru transferul instrucţiunilor).

Prin acest paralelism (separaţie) a căilor de comunicaţie de date şi instrucţiuni se asigură o viteză de transfer şi (implicit) o viteză de procesare mai mare.

Microprocesorul IV
Modelul von Neumann

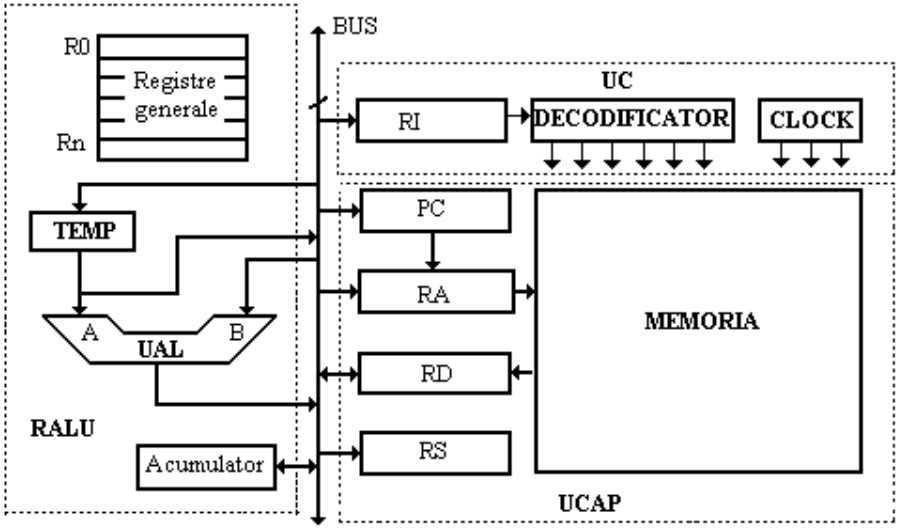


Dacă se unesc, pentru reducerea numărului total de linii de semnal, liniile de date de intrare, de date de ieşire şi instrucţiuni într-o singură magistrală bidirecţională (bus de date şi instrucţiuni) se obţine modelul von Neumann (sau Princeton).

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 utilizarea sa.
- Regiștrii pot fi pe 8, 16 sau 32 de biți.

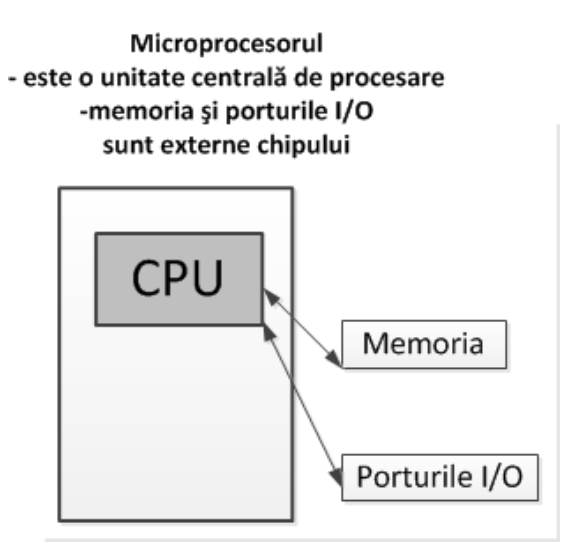


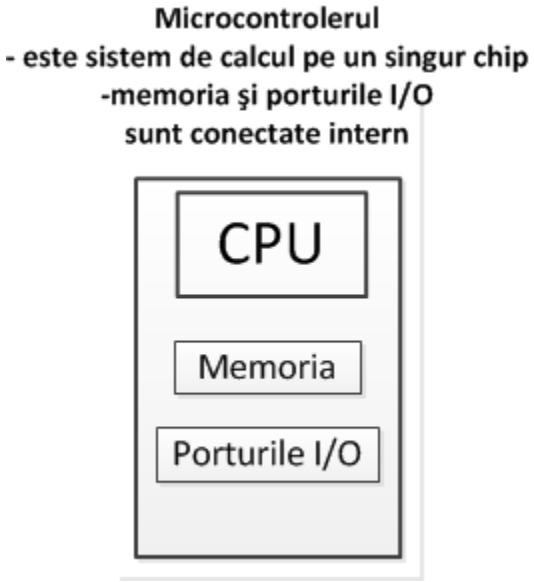
ACUMULATORUL (WORK Register) memorează rezultatele intermediare. De exemplu, poate fi utilizat la însumarea unui șir de numere.
PROGRAM COUNTERUL (PC) calculează adresa următoarei instrucțiuni de executat din program.
REGISTRUL DE STARE conține printre altele indicatorii de stare Z(zero), C(carry), N(negativ), V(overflow), I (interrupt)

Microprocesor vs. Microcontroler

Microcontrolerul (MCU – microcontroller unit) este de regulă un circuit complex obţinut prin adăugarea unor dispozitive (circuite) suplimentare la un microprocesor**, *totul fiind integrat pe acelaşi chip***.

Un **microprocesor (MPU – micro proceesor unit)** conține pe același cip:
- unitatea centrală de procesare – UCP (registre, ALU, indicatori, magistrale);
-unitatea de gestionare a întreruperilor interne şi externe;
- circuite pentru generarea şi gestionarea impulsurilor ceasului (modificări ale frecvenţei, reducerea radiaţiei, etc);
circuite de alimentare cu tensiuni de valori diferite, etc;
unitatea de management a memoriei.



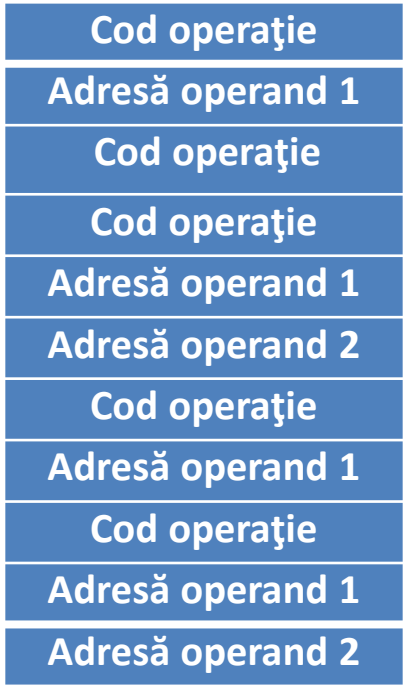


Un **microcontroler cuprinde pe același chip un microprocesor** și, în plus, o serie de circuite necesare unui sistem de calcul (practic, poate fi un microcalculator complet):
- memorii (RAM, ROM, EPROM/EEPROM, FLASH);
- periferice de comunicaţie a datelor paralelă şi serială - (porturi paralele, porturi seriale asincrone şi sincrone – UART, USART, SCI, SPI, I^2C);
- periferice de gestionarea timpului (timere pentru generarea întârzierilor de timp, circuite pentru captură de timp/ măsurarea şi compararea timpului);
- circuite antiblocare de tip watchdog;
- circuite de securitate pentru protecţia softwarelui înscris în memoria ROM;
- circuite pentru numărarea impulsurilor sau generarea de impulsuri PWM (*pulse width modulation* - impulsuri modulate în durată);
- convertoare analog-digitale şi digital-analogice;

Funcționarea microprocesorului I
-instrucțiunea-

- ***Instrucţiunea este o componentă a unui program prin care se cere ca una sau mai multe operaţii să fie executate de procesor.***
- Structura instrucţiunii în memorie este o succesiune de locaţii de memorie conţinând obligatoriu codul instrucţiunii şi opţional una sau două adrese de operand. Instrucţiunile unui anumit procesor pot fi în unul din două tipuri de format:
- - instrucţiuni cu **format fix** (toate instrucţiunile au aceeaşi lungime în memorie);
- - instrucţiuni cu **format variabil** (instrucţiunile pot fi diferite ca lungime în număr de octeţi ocupaţi în memorie).

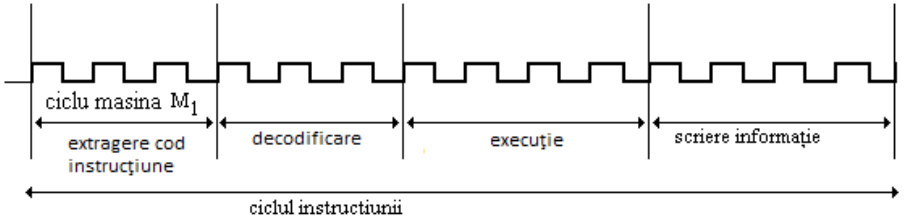
Structura instrucțiunii:

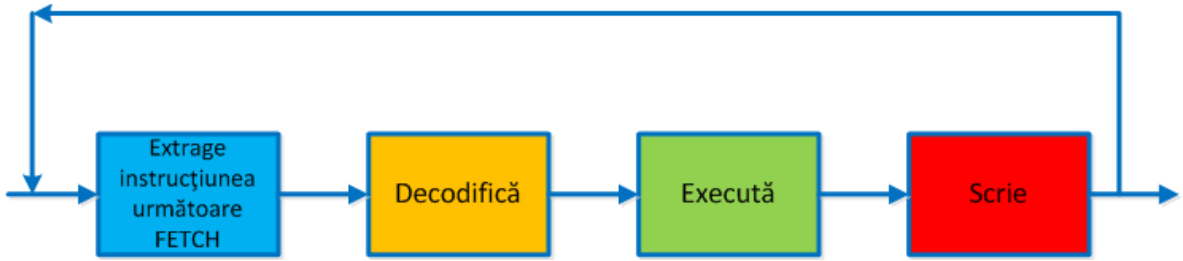


- Unele instrucțiuni nu au operanzi sau au doar unul
- Opcode spune ce operațiune se efectuează
- Modul indică tipul instrucțiunii: tip registru, tip memorie
- Operanzii pot fi **imediați** sau pot fi găsiți în memorie prin **adresa** lor

Funcționarea microprocesorului II
- ceasul, cicluri -

- *Operaţiile de bază* sunt componentele instrucţiunilor (*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 operaţiei de bază. Operaţia de bază poate să conţină mai multe stări.
- *Ciclul maşină* este format din una sau mai multe stări T reprezentând intervalul de timp în care se execută o operaţie de bază a instrucţiunii. Ciclurile maşină pot avea un număr fix sau variabil de stări.
- *Ciclul instrucţiune* este o succesiune de cicluri maşină în care primul ciclu M_1 este ciclul de extragere din memorie a codului instrucţiunii ce urmează a fi executat.





Funcționarea microprocesorului III
-ciclul instrucțiune-

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

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

Funcționarea microprocesorului IV
-ciclul instrucțiune-

Ciclul	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.

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)
  - Z: zero
  - N: Negativ
- *Stack Pointer* (SP)
- *Input/Output register*

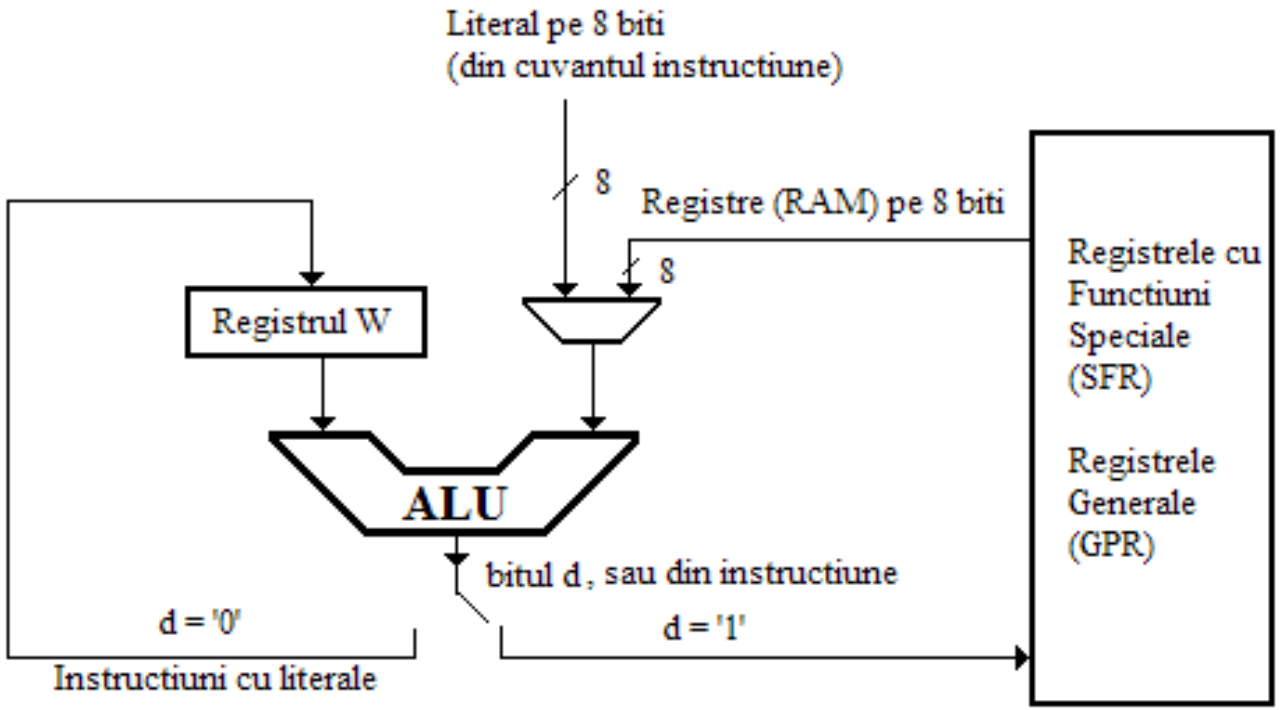
Cu destinație generală (*General Purpose Register*). Conținutul lor este în totalitate accesibil adică poate fi citit și scris. Sunt utilizate pentru memorarea temporară a unor date sau rezultate

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”.

Exemplu: modelul proramatorului pentru procesoarele PIC16 (Microchip)



Funcționarea microprocesorului V
-CISC și RISC-

RISC – *Reduced Instruction Set Computer*

- Set mai mic (număr mai mic) şi mai simplu (programare mai uşoară, de exemplu în cazul lungimii fixe) de instrucţiuni;
- Toate instrucţiunile au aceeaşi durată de timp la execuţie şi se execută la fel indiferent de regiştrii cărora li se aplică.

CISC – *Complex Instruction Set Computer*

- O instrucţiune poate executa mai multe operaţii de nivel inferior: de exemplu o singură instrucţiune poate face operaţiuni de încărcare a memoriei, operaţii aritmetice şi de stocare a rezultatului în memorie;
- Instrucţiunile au o durată diferită de timp la execuţie şi se pot aplica doar asupra unor regiştri.

Arhitectura cea mai potrivită pentru un procesor rapid:

- este dependentă de aplicaţiile ce vor fi executate;
- foarte puţine instrucţiuni;
- multe registre;
- acces simplificat la încărcarea şi depozitarea datelor în memoria principală;
- posibilitatea ca majoritatea instrucţiunilor să fie executate într-o singură perioadă de ceas.

Arhitectura setului de instrucţiuni 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

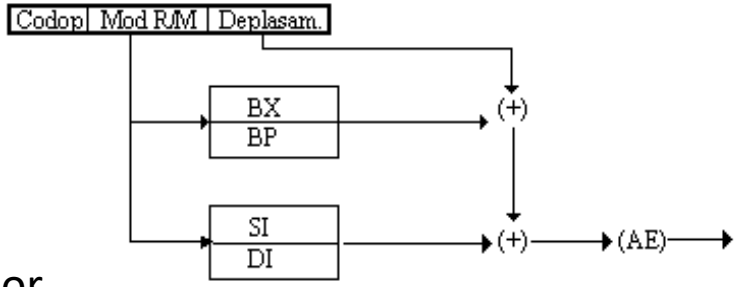
Arhitectura setului de instrucţiuni II
-moduri de adresare-

Instrucţiunile trebuie să specifice de unde se iau operanzii şi 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 instrucţiune.
- Valorile sunt în registre - numărul registrului este în instrucţiune.
- Valorile sunt în memorie:
  - -adresa este în registru - numărul registrului este în instrucţiune;
  - -adresa este în instrucţiune;
  - -adresa este valoarea din registru plus o diferenţă/offset - numărul registrului este în instrucţiune, iar offsetul este în instrucţiune, sau în registru (adresare indirectă).



- ➔ aceste căi de specificare a locaţiilor operanzilor sunt denumite **“moduri de adresare”**

Arhitectura setului de instrucţiuni III -adresarea imediată-

Adresarea imediată:

- Operandul provine din instrucţiunea însăşi => operandul este imediat disponibil din instrucţiune

Exemplu :

addw #86, d5
Se face adunarea: d5 <- d5 + 86

Valoarea 86 provine din instrucţiune

d5 este un registru

Exemplu (uP PIC16 Microchip)

addlw 0x20
Se face adunarea w<- w + 32_{10}

W este registrul de lucru (acumulatorul)

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

Arhitectura setului de instrucţiuni IV -adresarea directă la registri-

Adresarea directă la registru:

- Numărul registrului în care găseşte operandul este conţinut în instrucţiune

Exemplu :

addw d0, d6

Se face operaţia: d6 <- d0 + d6

Valoarea din d0 se adună la d6

d6 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 instrucţiuni V
-adresarea indirectă -

- Adrsarea indirectă
- Operandul este în memorie, adresa memoriei fiind conţinută într-un registru, iar numărul registrului este conținut în instrucțiune

Exemplu:

addw (a0), d3

Se face operaţia: d3 <- d3 + (a0)

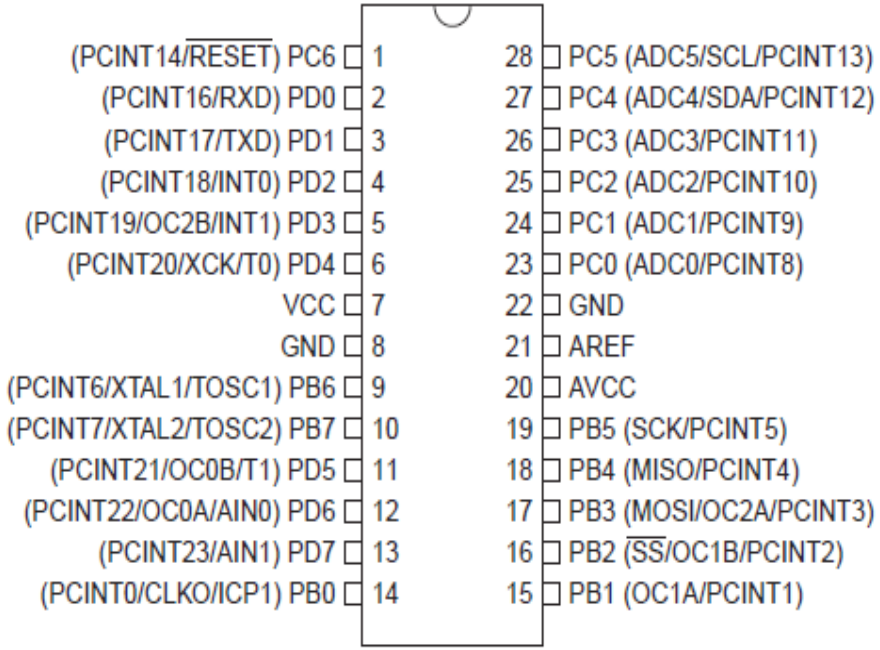
Valoarea din locația de memorie a cărei adresă se află în registrul a0 se adună registrul d3
De exemplu, dacă a0 este 20 și (20)=321, se adună 321 la registrul d3

Microcontrolerul ATMEL ATMega 328P

Caracteristici generale

Conține:

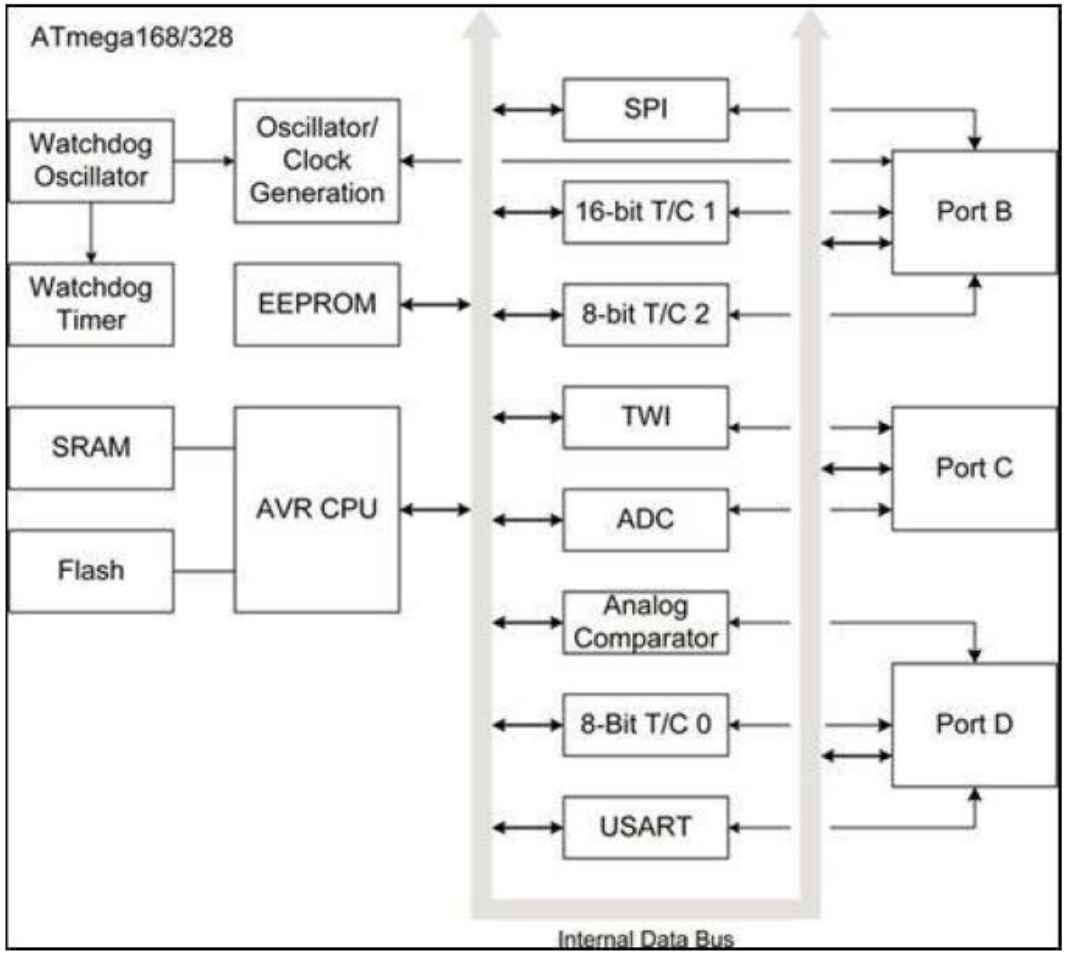
- procesor RISC pe 8 biți, 131 instrucțiuni
- arhitectură Harvard modificată
- 32 regiștri de uz general pentru CPU
- 64 regiștri pentru programarea circuitelor I/O
- 32k x 8 memorie ROM flash (10.000 cicluri ștergere)
- 2k x 8 memorie SRAM
- 1k x 8 memorie EEPROM (100.000 cicluri ștergere)
- 3 canale 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.

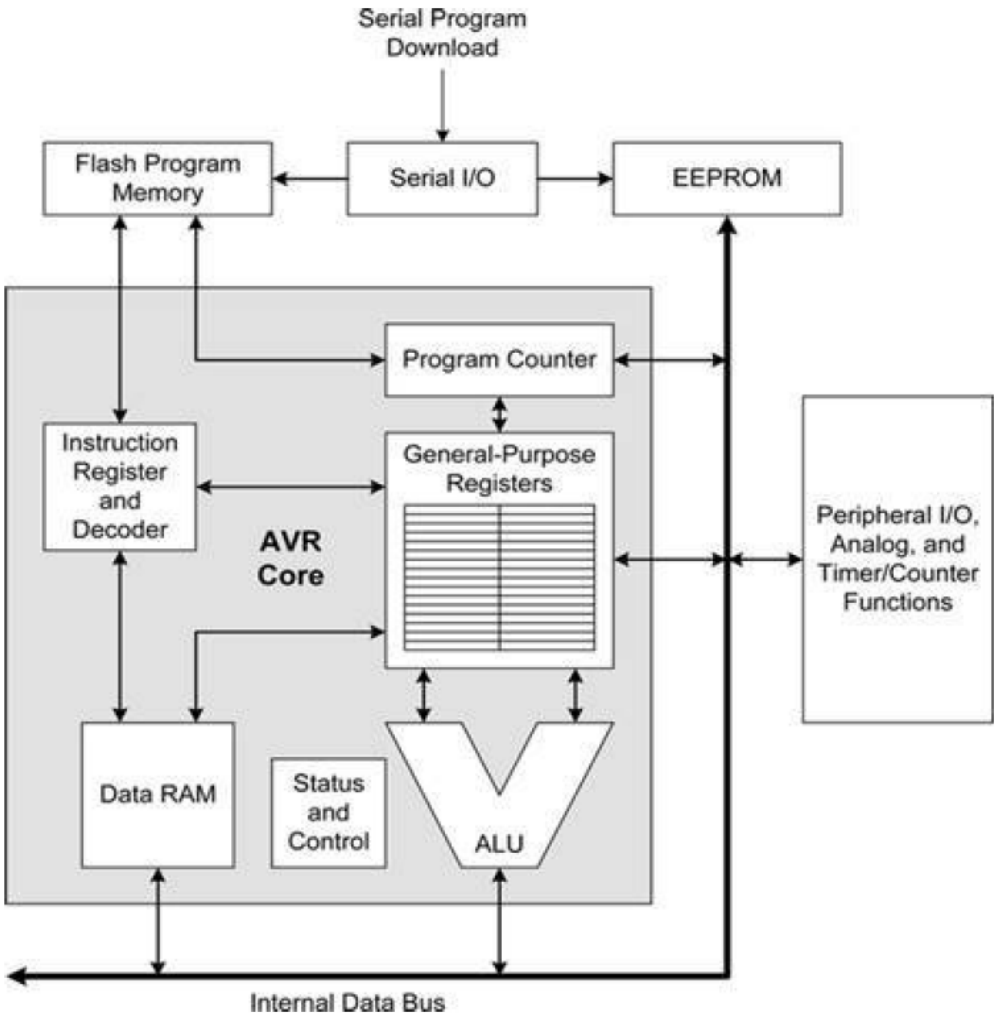
Microcontrolerul ATMEL ATMega 328P

Schema bloc



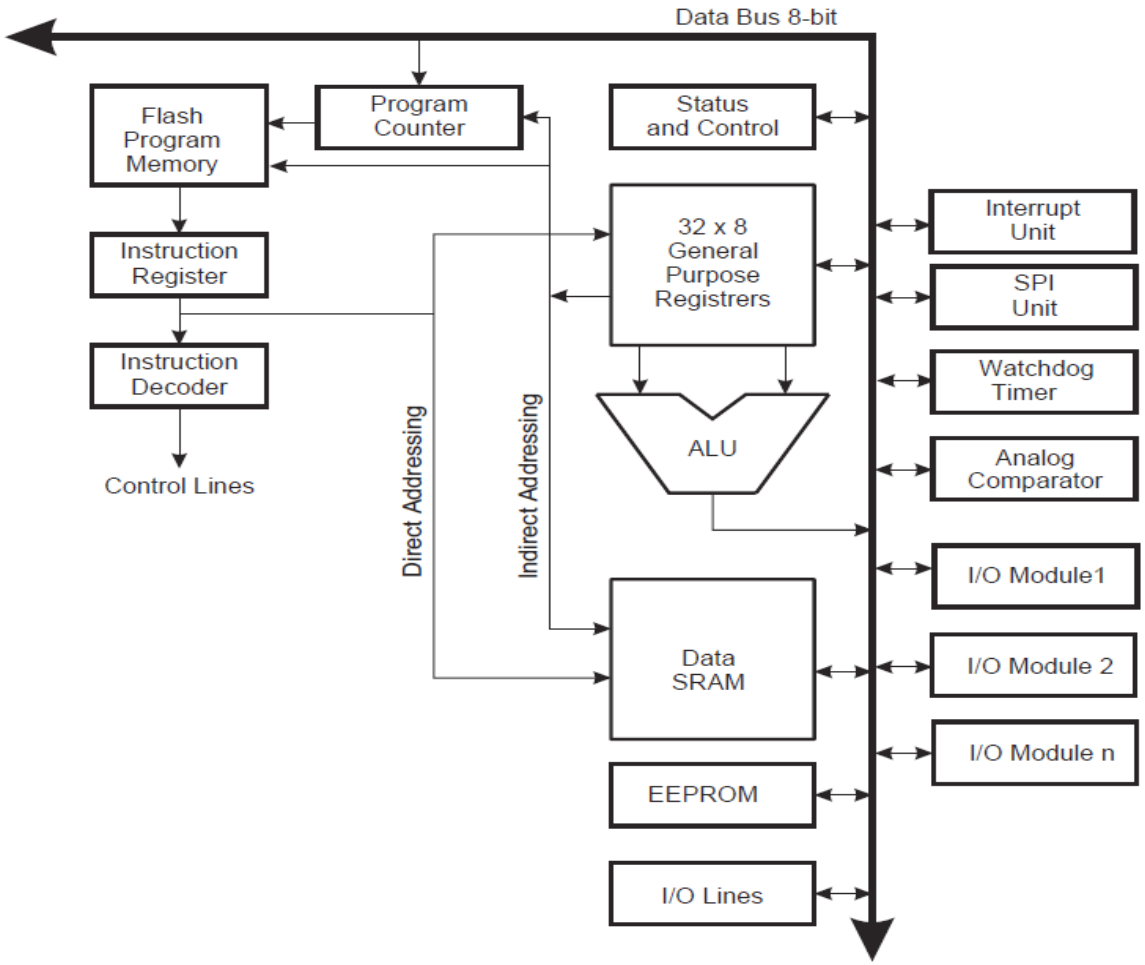
Microcontrolerul ATMEL ATMega 328P

Procesorul - CPU



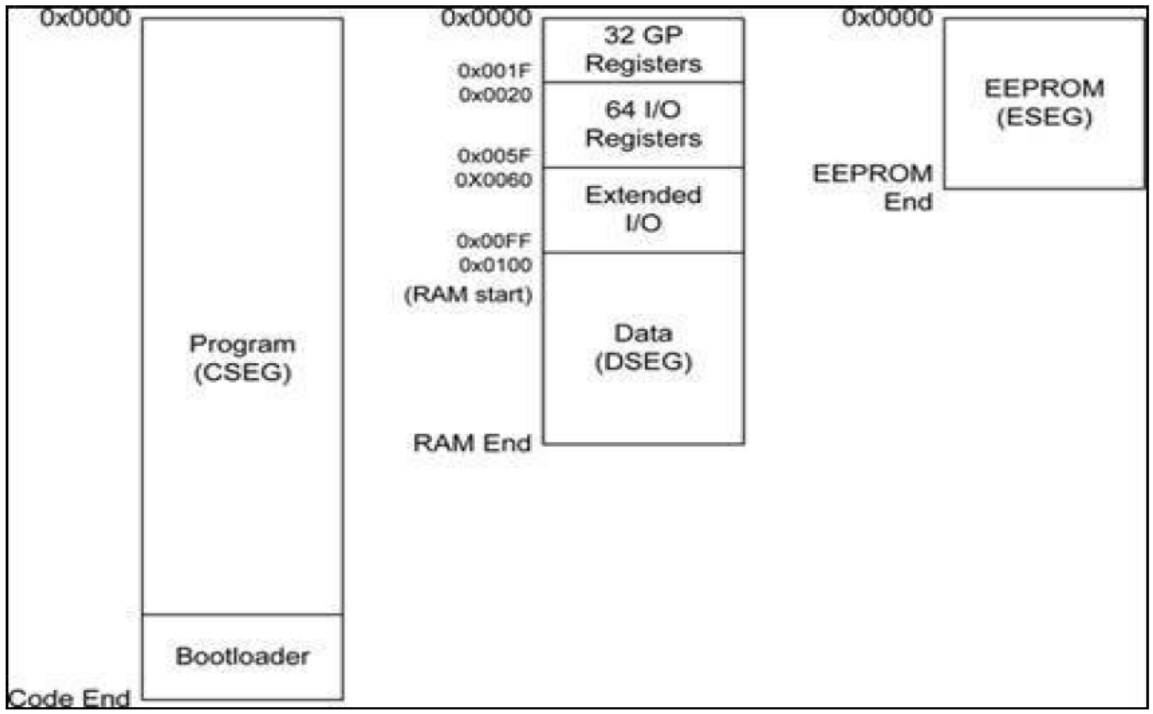
Microcontrolerul ATMEL ATMega 328P

Procesorul – CPU – catalog ATMEL



Microcontrolerul ATMEL ATMega 328P

Organizarea memoriei interne



Memoria internă 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 reaplicarea 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
	R27		0x1B
	R28		0x1C
	R29		0x1D
	R30		0x1E
	R31		0x1F

X, Y și Z sunt regiștri de 16 biți utilizați pentru adresarea indirectă

X-register Low Byte	
X-register High Byte	
Y-register Low Byte	
Y-register High Byte	
Z-register Low Byte	
Z-register High Byte	

Circuitele de intrare-ieşire I/O

Sunt circuite digitale, de obicei programabile, prin intermediul cărora se pot conecta la magistralele unui sistem de calcul divese 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 acelaşi circuit I/O, programabil.

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

Totuşi, în marea majoritate a cazurilor, circuitele I/O sunt circuite "inteligente", care pot să îşi desfăşoare activitatea fără a mai fi nevoie în evoluţia lor de intervenţii dese ale procesorului. Acest lucru este posibil datorită automatelor de control pe care le conţin, dar care necesită o programare prealabilă.

Programarea prealabilă se realizează de obicei la începutul execuţiei programelor şi constă în configurarea conţinuturilor unor regiştri interni, prin instrucţiuni pe care procesorul le transmite, ca urmare a programului de lucru scris de către utilizator. Deci "inteligenţa" circuitului I/O este de fapt transmisă de utilizator, folosind ca intermediar procesorul.

De obicei aceşti regiştri poartă denumirea de regiştri de comandă şi control, regiştri de stare, regiştri de date şi se identifică în datele de catalog.

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

Astfel de circuite I/O sunt conţinute şi în microcontrolerul utilizat de modulul Arduino (produs de ATMEL, Atmega 328P). Pentru aceste circuite I/O au fost create librării cu funcţii specifice pentru mediul de programare, prin care munca de programare să fie uşurată, dar câteodată este nevoie şi de setări directe de biţi în regiştrii de comandă şi 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.

- Şi 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 ieşire care permite conectarea în paralel a mai multor ieşiri 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 aceşti pini şi de fapt comunicaţia 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 -

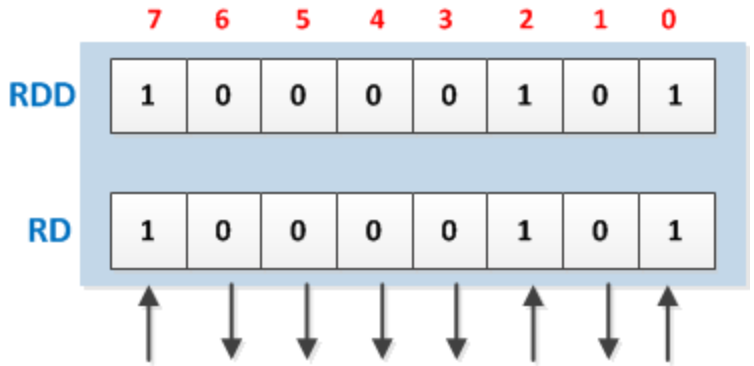
- uP execută un program pentru a face transferul între µP și un circuit I/O printr-unul sau mai multe registre ale circuitului I/O. µP controlează complet transferul datelor prin programul executat.
- La un circuit I/O port paralel apar de obicei două registre prin care se gestionează transferul I/O:

- **RD, Registrul de Date** va conține datele reale care intră sau ies în/din µP.

- **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.

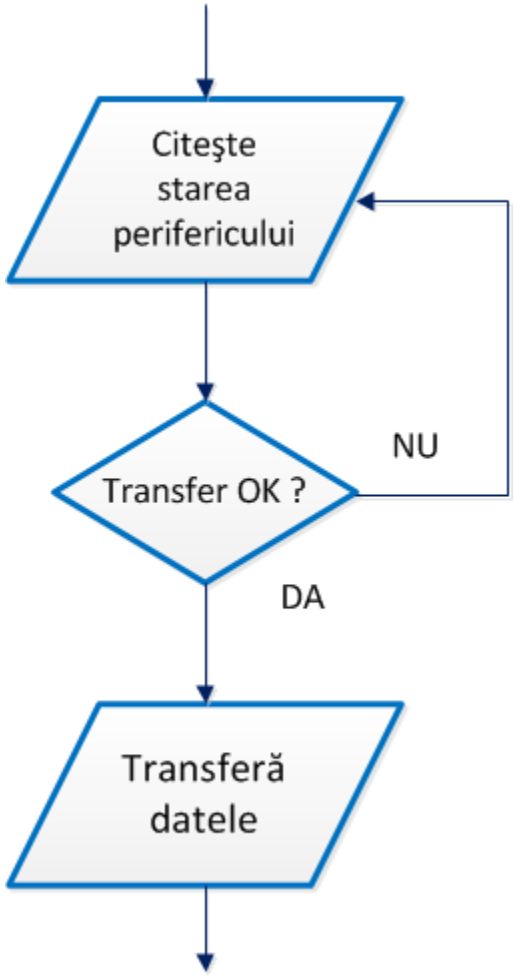


Interfațarea I/O a microprocesoarelor IV
- I/O programat -

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

- Transferul **necondiționat** al datelor se poate face oricând; perifericul extern trebuie să fie ready întotdeauna pentru transferul datelor. În acest caz receptorul nu informează emiţătorul despre starea primirii informaţiilor.

- Transferul **condiționat** al datelor între uP și perifericul extern se face prin **hand-shaking**. Receptorul informează emiţătorul despre reuşita sau nereuşita transferului de informaţii, dar şi despre pregătirea pentru altă recepţie. Practic, hand-shaking-ul presupune şi transmisia de informaţii suplimentare, faţă de cele propriu-zise prin care să se controleze validitatea datelor transferate şi procesul de comunicaţie.



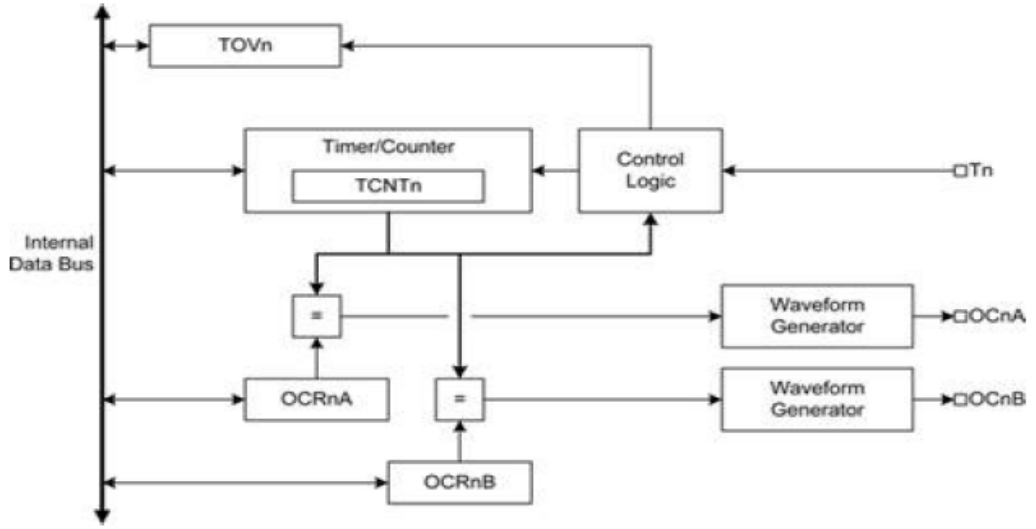
Microcontrolerul ATMEL ATMega 328P

Structură generală canal timer programabil

Un circuit I/O canal timer este organizat în jurul unui numărător, de obicei pe 8/16 biţi, care poate fi configurat în diverse moduri de lucru, prin regiştrii de comandă şi 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 ieşire semnale digitale, funcţie de configuraţia programată (exemplu semnal PWM). Circuitul poate fi folosit pentru măsurarea timpului, dacă se cunoaşte perioada semnalelor de numărare aplicate. O altă funcţie este aceea de captură de timp, adică măsurarea timpului între două evenimente. Încă o funcţie întâlnită este cea de comparare de timp (cu o referinţă).

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

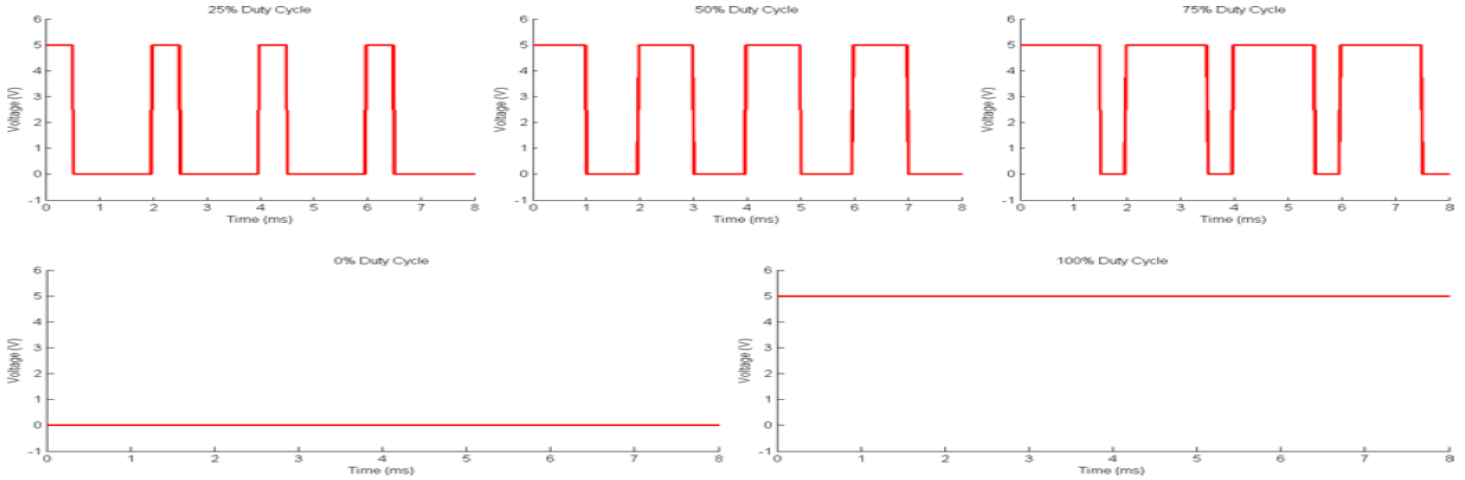
- TON (*Timer On-Delay*): acţiunea de întârziere este declanşată de începutul evenimentului de intrare;
- TOF (*Timer Off-Delay*): acţiunea de întârziere este declanşată de sfârşitul evenimentului de intrare;
- TP (*Timer – Pulse*): acţionează ca un monotabil şi se utilizează pentru crearea de impulsuri 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

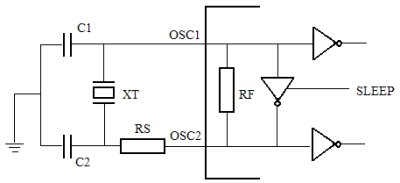


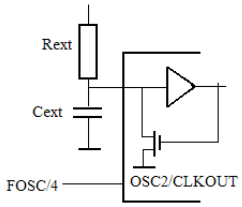
Factorul de umplere este definit ca raportul dintre durata părţii din impuls diferită de zero si perioada impulsurilor.

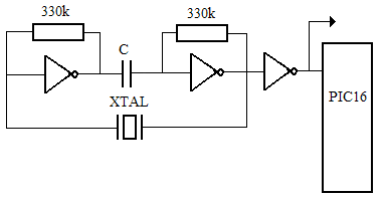
Semnalele PWM, utile la comanda motoarelor de curent continuu, se pot genera prin folosirea canalelor timer programabile.

Sistemul de ceas (clock)

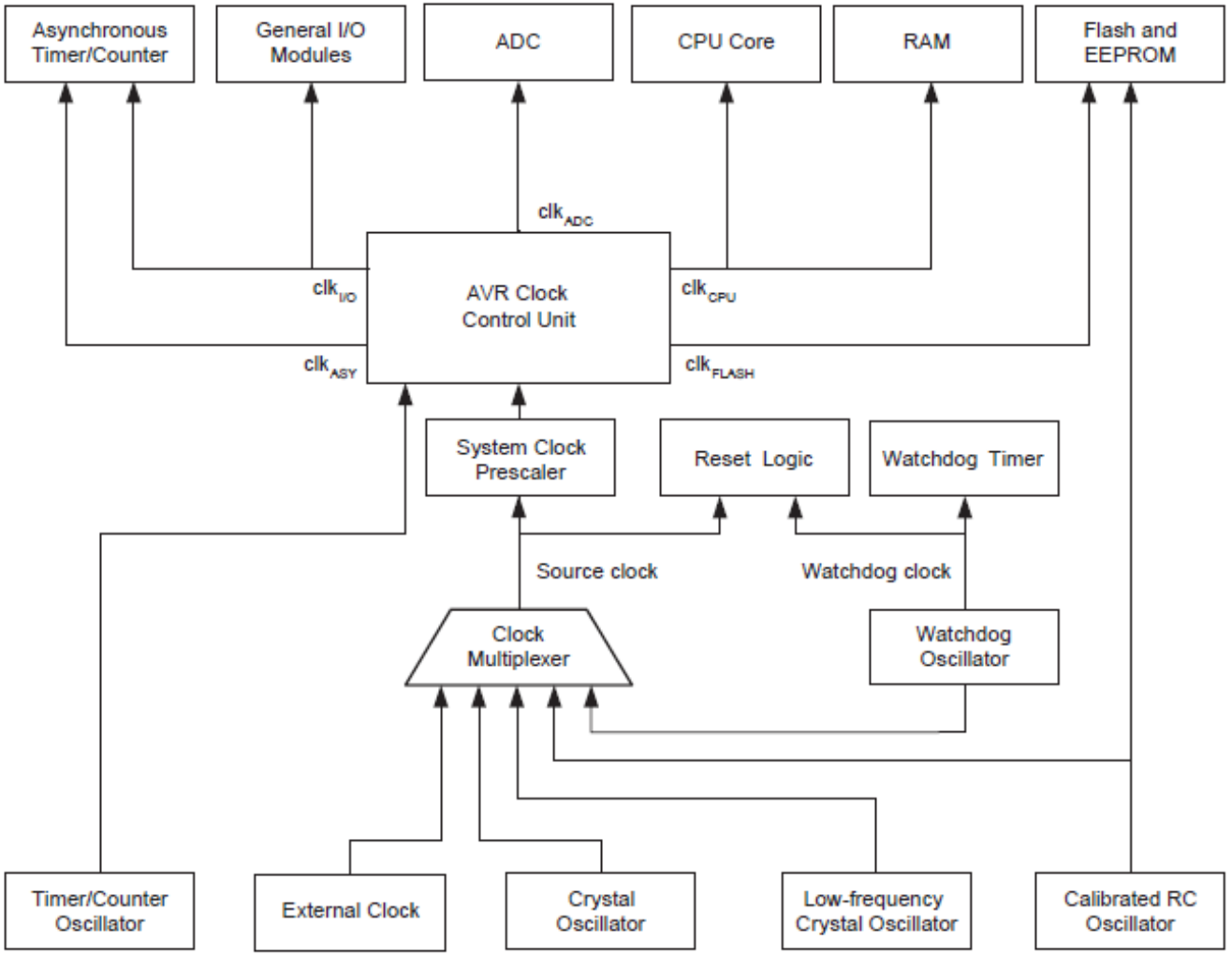
Rolul oscilatorului este să furnizeze semnalele de ceas necesare execuţiei instrucţiunilor şi sincronizării cu circuitele I/O, respectiv perifericele.
Semnalul de ceas este folosit de multe circuite I/O în evoluţia lor.







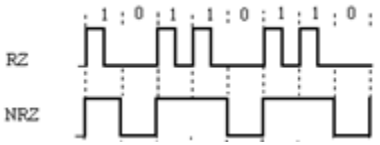
Distribuţia semnalelor de ceas la uP AVR



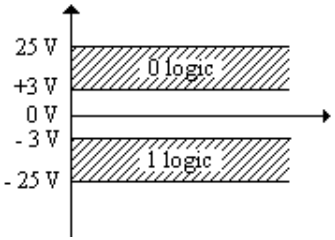
Microcontrolerul ATMEL ATMega 328P

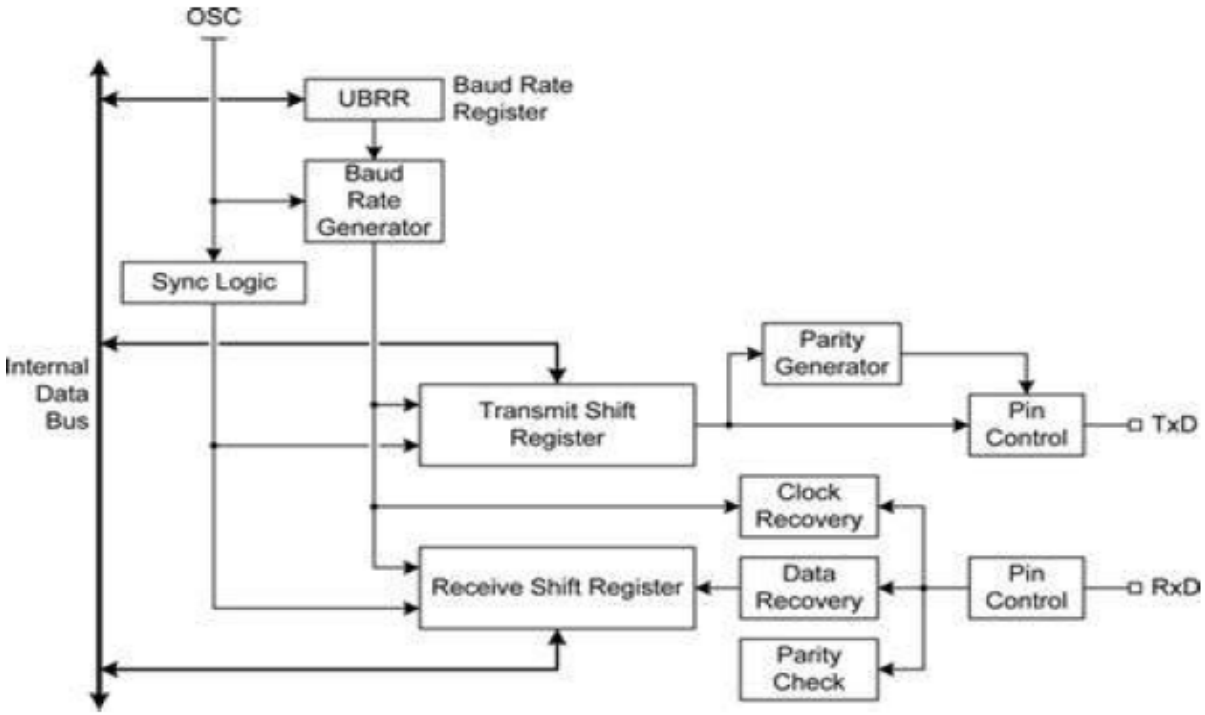
Structură generală canal de comunicație serială USART

Tipuri de coduri binare utilizate în comunicaţiile seriale



Exemplu: corespondenţa între valorile bitului şi nivelurile tensiunii în standardul RS232

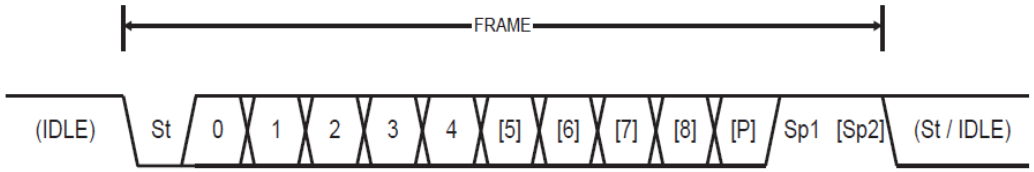




Se programează prin regiștrii de tip I/O:
UCSR – USART Control and Status Register
UBRR – USART Baud Rate Register
UDR – USART I/O Data Register

Microcontrolerul ATMEL ATMega 328P

Structură pachet date în comunicația serială USART



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

Calculul bitului de paritate

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

$\mathbf{P_{even}}$	Parity bit using even parity
$\mathbf{P_{odd}}$	Parity bit using odd parity
$\mathbf{d_n}$	Data bit n of the character

Microcontrolerul ATMEL ATMega 328P

Structură generală canal de comunicație serială TWI – I2C

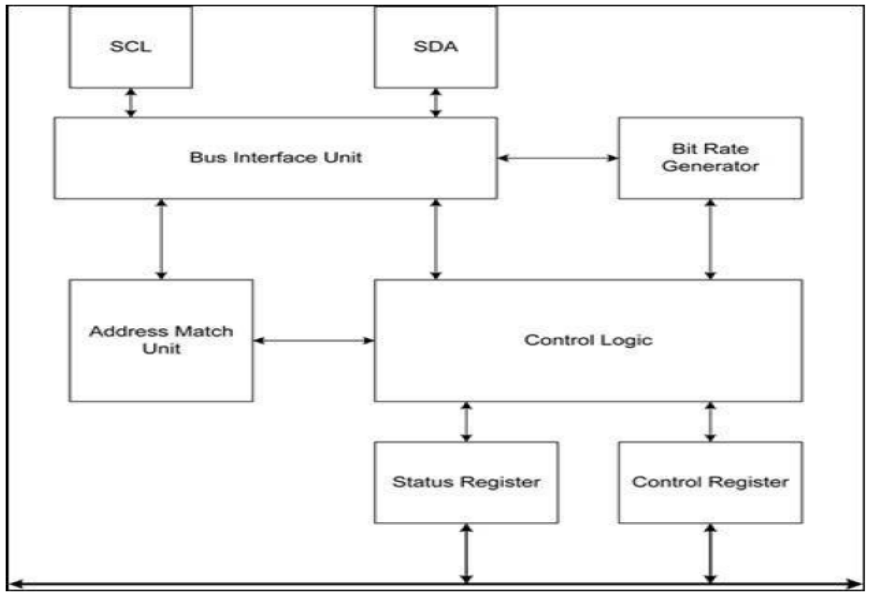
TWI este acronimul pentru *Two Wire Interface*, iar I²C înseamnă *Inter-Integrated Circuit,* cele 2 prescurtări reprezentând același standard de comunicație. Standardul a fost proiectat ca mod de comunicaţie pe **două fire**, cu imunitate crescută la perturbații, între diferite circuite integrate cu viteze de transfer de 100kbps şi 400kbps (*Fast mode*). Ulterior a fost extins până la 1Mbps şi chiar la 3,4Mbps (*High-speed,* sau *Hs-mode*).

I²C este o interfaţă *multimaster*, ceea ce înseamnă că admite mai mult de un *master* pentru controlul liniei. Protocolul I²C presupune că fiecare dispozitiv are o adresă.

Când un *master* doreşte să iniţieze un transfer de date, mai întâi transmite adresa dispozitivului *slave*, cu care vrea să comunice. Toate dispozitivele „ascultă" linia, pentru a-şi identifica propria adresă. În această adresă, un bit specifică dacă masterul doreşte să citească din/scrie în dispozitivul slave. Master-ul şi slave-ul sunt întotdeauna în mod complementar (*transmitter/receiver*) în timpul unei operaţii de transfer de date. În ambele cazuri master-ul generează un semnal de ceas.

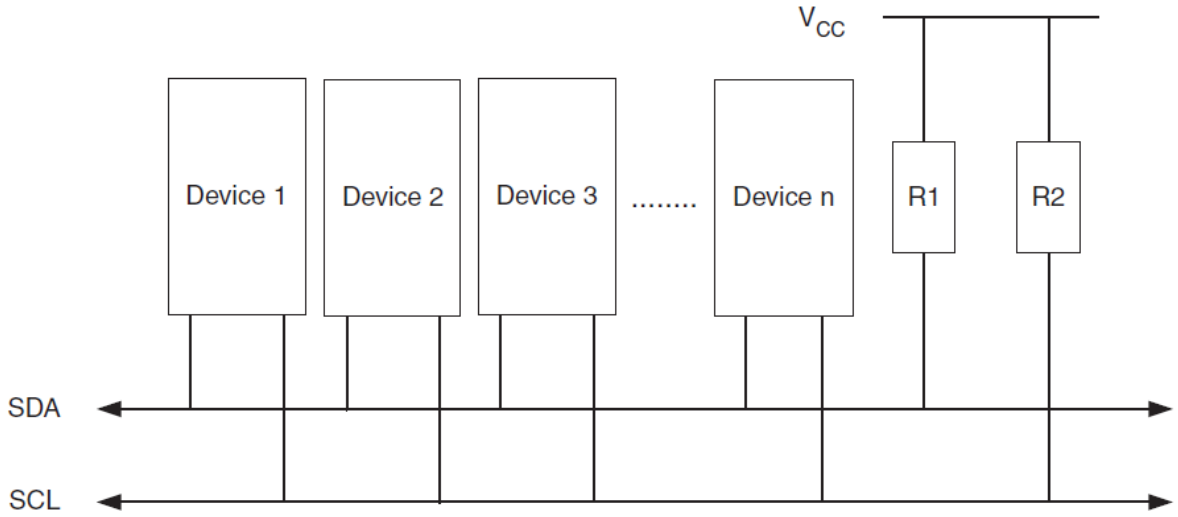
Se utilizează două formate de adresă: pe 7 biţi şi pe 10 biţi. Protocolul de interfaţă dispune de proceduri de arbitrare a transferurilor multimaster şi de sincronizare a ceasului atunci când începe transmisia ca urmare a unei arbitrări.

Se programează prin regiștrii I/O:
TWCR – Two Wire Control Register
TWB – Two Wire Bit Rate Register
TWSR – Two Wire Status Register
TWDR - Two Wire Data Register
TWAR – Two Wire Adress Register



Microcontrolerul ATMEL ATMega 328P

Conectarea circuitelor I/O în comunicația serială TWI – I2C

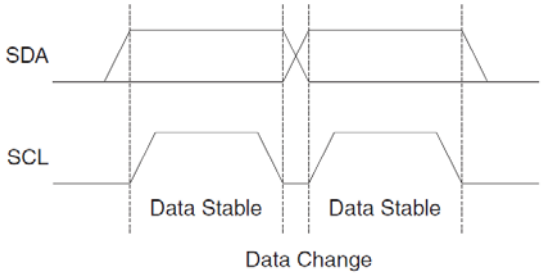


Semnale folosite:
SCL – Serial Clock
SDA – Serial Data

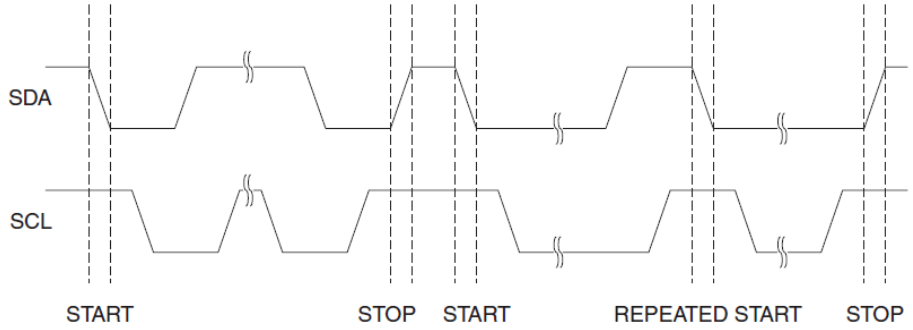
Microcontrolerul ATMEL ATMega 328P

Format pachet de informații în comunicația serială TWI – I2C

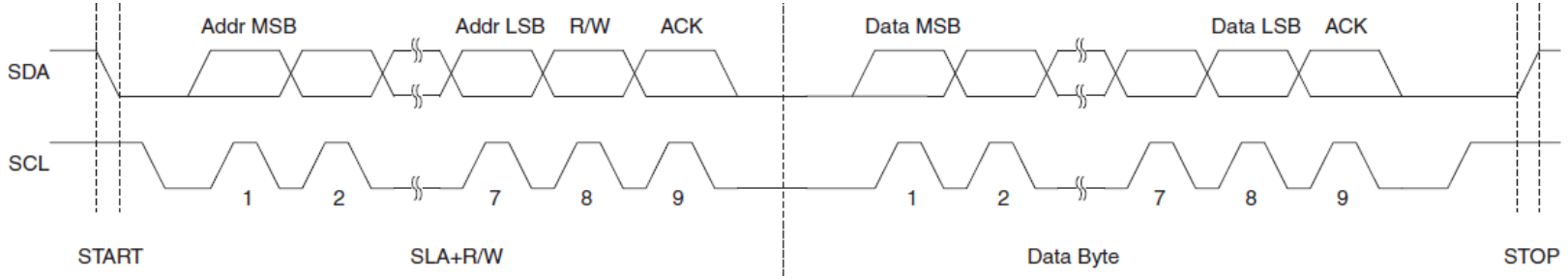
Data Validity



START, REPEATED START and STOP conditions



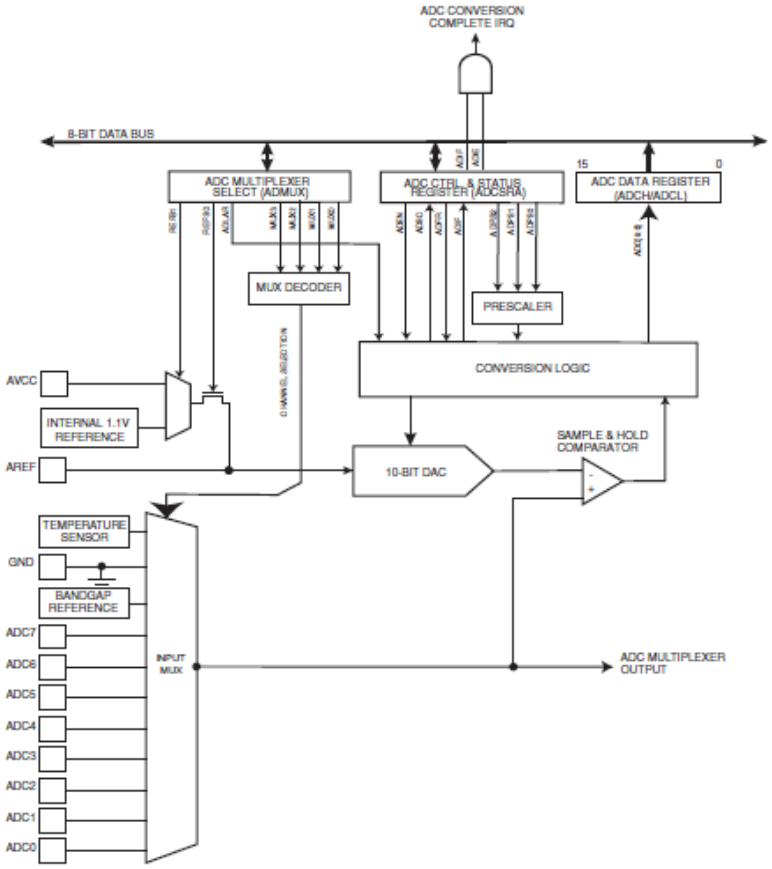
Format pachet complet informații (adresa și data)



Microcontrolerul ATMEL ATMega 328P

Structură generală convertor analog – digital ADC

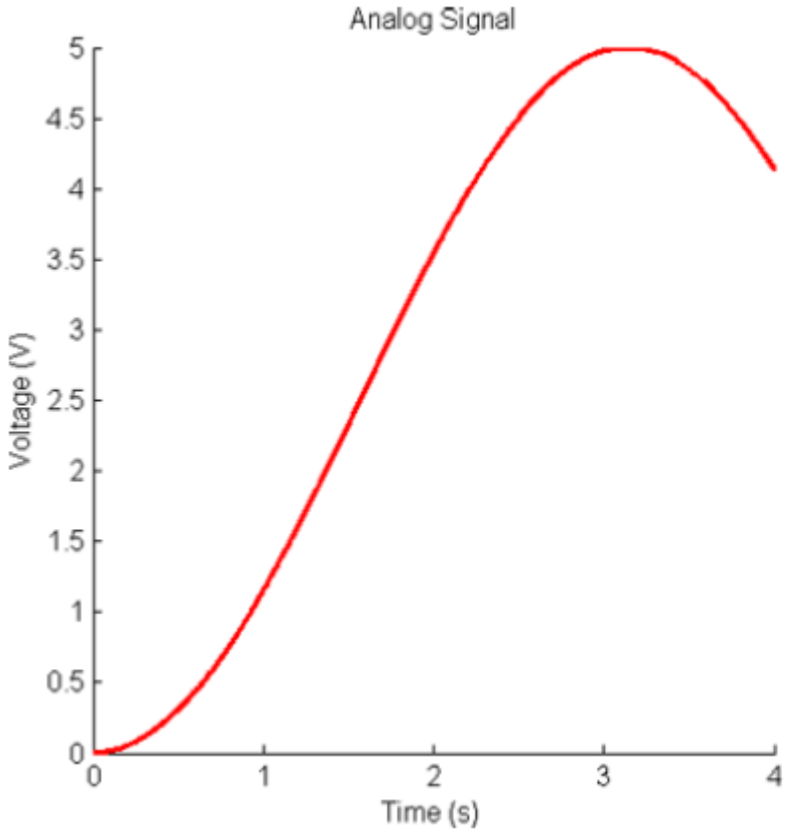
Caracteristici:
- 10 biţi rezoluţie
- 13 – 260 us timp de conversie
- 76,9 kSPS (eşant/sec)
- 6 intrări simple
- 0 – Vcc domeniul tensiunii de intrare (V_{FS})
- întrerupere la sfârşit conversie

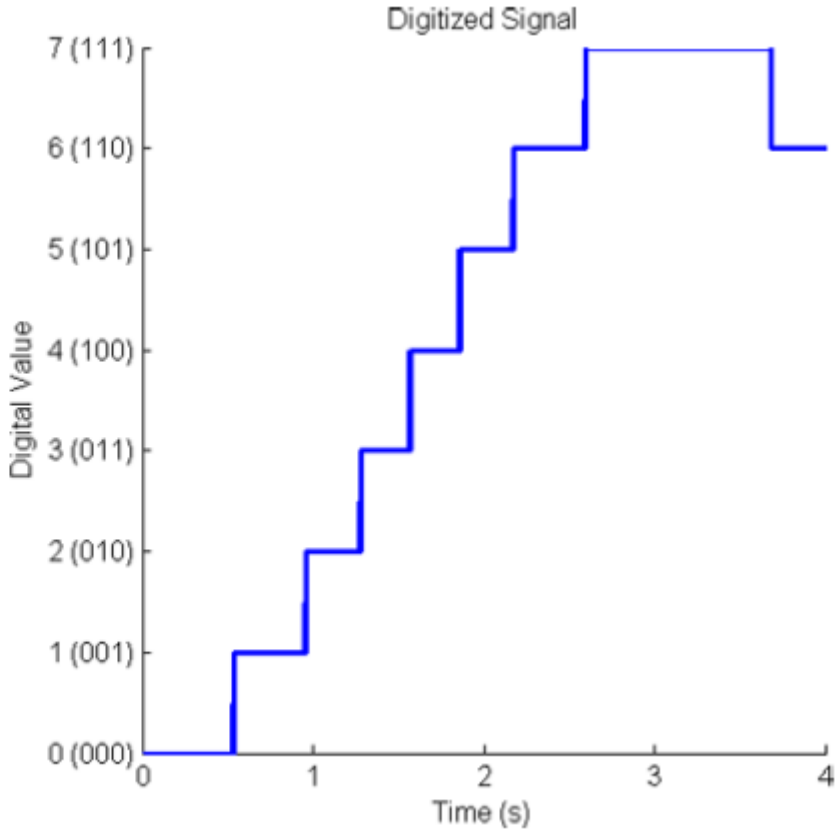


Se programează prin regiștrii I/O:
ADMUX – ADC Multiplexer Selection Register
ADCSRA – ADC Control and Status Register
ADC L, H – ADC data Registers
DIDR – Digital Input Disable Register

Microcontrolerul ATMEL ATMega 328P

Conversia analog – digitală pe 3 biți a unui semnal analogic





Sistemul de întreruperi al microprocesoarelor

O *cerere de întrerupere* este de fapt o semnalizare (semnal digital) generată de un circuit extern procesorului sau intern acestuia prin care se anunţă apariţia unui eveniment important care necesită tratare din partea procesorului.

Dacă cererea de întreruperea este acceptată de procesor, atunci acesta suspendă execuţia programului în curs, salvând în stivă adresa de revenire, dar şi eventuale alte informaţii despre starea programului suspendat şi trece la tratarea evenimentului apărut prin execuţia 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 a acestuia.

Cererile de întrerupere pot fi: hardware (aplicate pe pini specializaţi ai procesorului şi care provin din surse externe) sau software (care provin din surse interne procesorului; exemplu forţarea unei operaţii de împărire cu 0)

Întreruperile hardware, la rândul lor pot fi nemascabile - se primesc pe un pin unic notat de obicei NMI – non maskable interrupt - şi necesită tratare imediată sau mascabile. Întreruperile mascabile se pot dezactiva şi activa pe cale software prin rularea de instrucţiuni specifice care acţionează asupra unui bistabil intern procesorului (interrupt flag).
Întreruperile mascabile se primesc pe pini notați INT (pot fi mai mulți pini) și pot fi dezactivate sau activate prin execuția unor instrucțiuni de tipul DI (*Disable interrupt*) sau EI (*Enable Interrupt*). Dacă întreruperea este dezactivată (disable, uP ignoră primirea respectivului semnal de întrerupere).

Producătorii de procesoare au inclus în setul de ciruite I/O produse şi circuite I/O de tipul controlor de întreruperi care au facilităţi în ceea ce priveşte operaţiunile care au loc între procesor şi circuitul I/O generator de întreruperi pentru tratarea acesteia. Alte circuite I/O cu funcţii diverse (canale timer, port paralel, port serial, etc) au fost proiectate astfel încât să conţină o logică minimală ce asigură suportul mecanismului de întreruperi prin care să asigură operarea în acest mod cu procesorul.

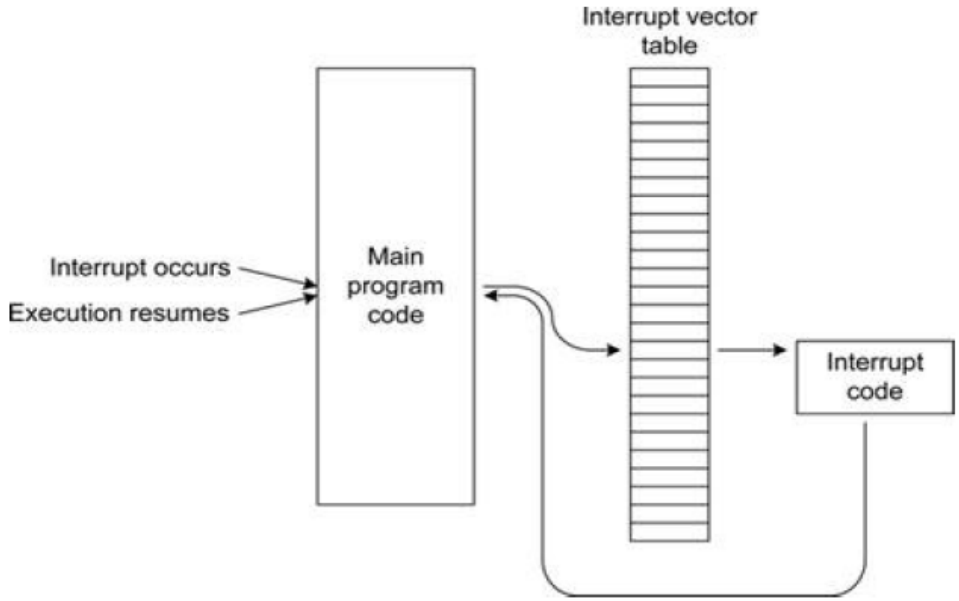
În sistemele de calcul cu mai multe surse de întreruperi se utilizează sisteme de priorităţi programabile pentru a ierarhiza solicitările primite.

Se întâlnesc mai multe modalităţi de tratare a întreruperilor. Un astfel de mod este tratarea vectorială prin folosirea unei tabele a vectorilor de întreruperi, în care se indică adresa rutinei de tratare a întreruperii apărute. Un astfel de mod foloseşte şi microcontrolerul ATMEL ATmega 328P din modulul Arduino (vezi planşa următoare).

Mediul de programare în C al modulului ARDUINO permite implementarea lucrului cu întreruperi prin care să fie semnalizate procesorului evenimente prioritare care necesită tratare prioritară. Totodată, prin întreruperi este posibil şi time-sharing-ul (împărţirea timpului de procesare între mai multe activităţi, acestea apărând că se execută simultan/în paralel) – sursa de întreruperi fiind în acest caz un canal timer. Sarcinile multiple pe care robotul line-follower trebuie să le îndeplinească pot fi realizate apelând la tehnica time-sharing

Microcontrolerul ATMEL ATMega 328P

Schema de principiu pentru operarea în întreruperi

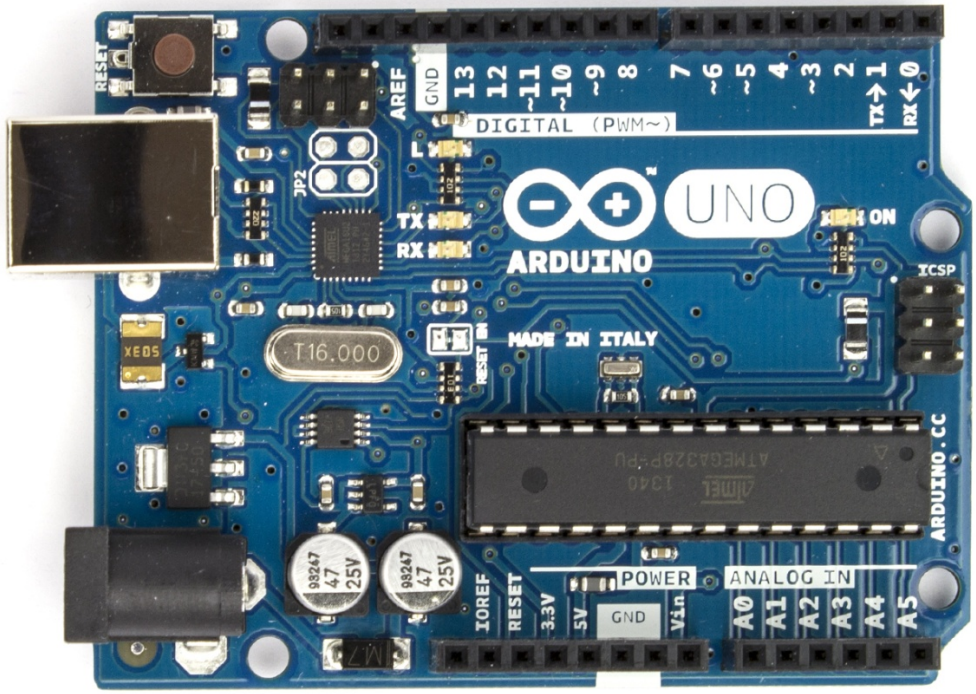


ATMega 328P gestionează vectorial 26 surse de întrerupere interne și externe (2 externe, 3 de la pinii porturilor paralele, 10 de la cele 3 canale timer, 1 de la watchdog, 3 de la canalul USART, 1 de la canalul SPI, 1 de la canalul TWI, 1 de la ADC, 1 de la un bloc comparator, 1 de la EEPROM, 1 de la 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

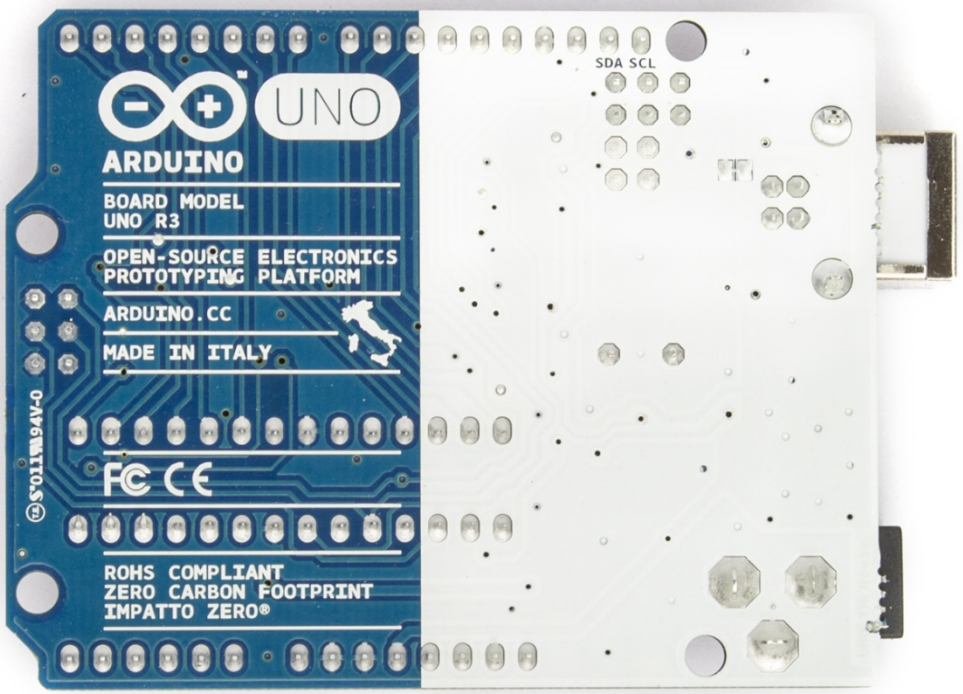
Microsistemul ARDUINO UNO R3

Fața cu piese



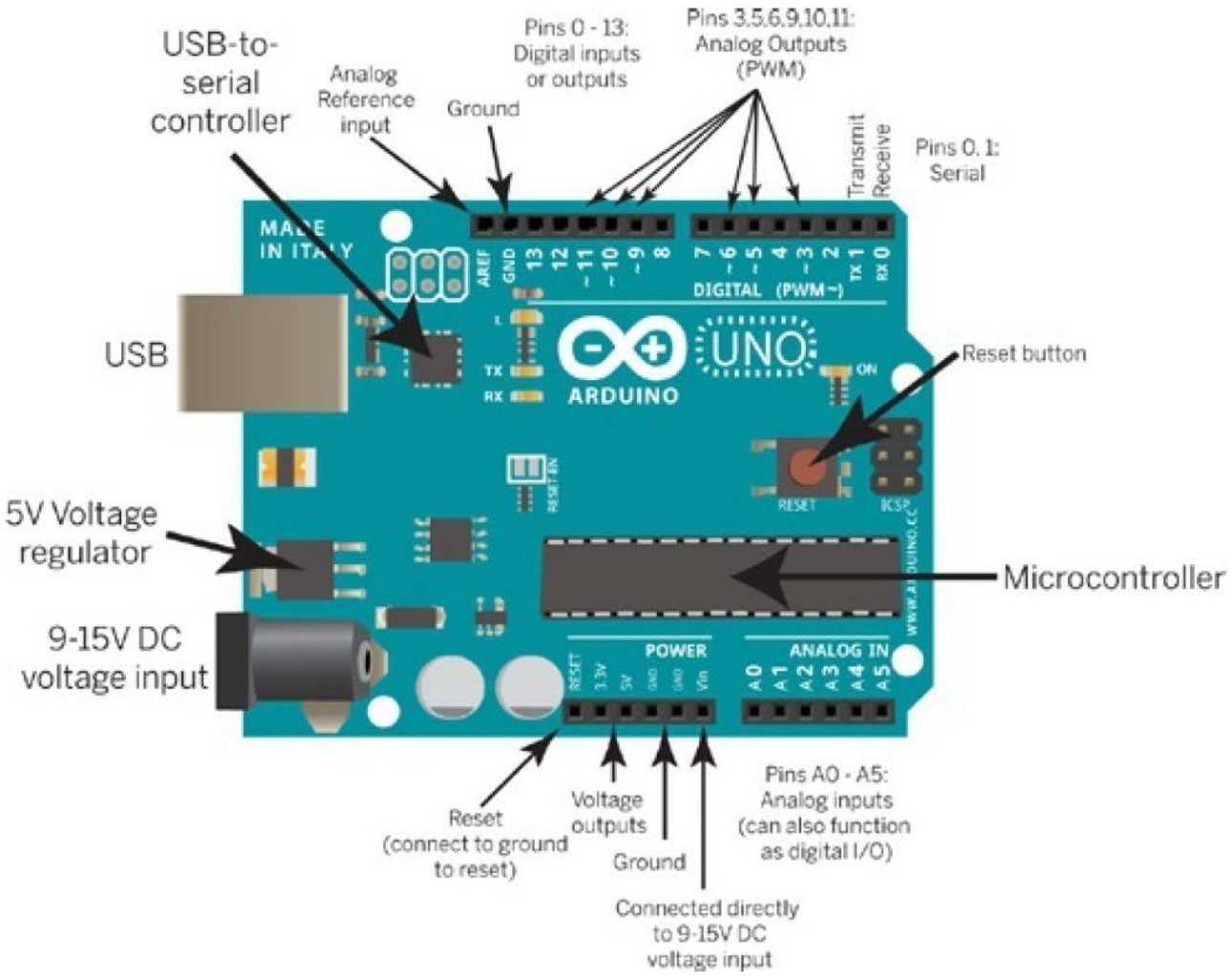
Microsistemul ARDUINO UNO R3

Spatele cablajului



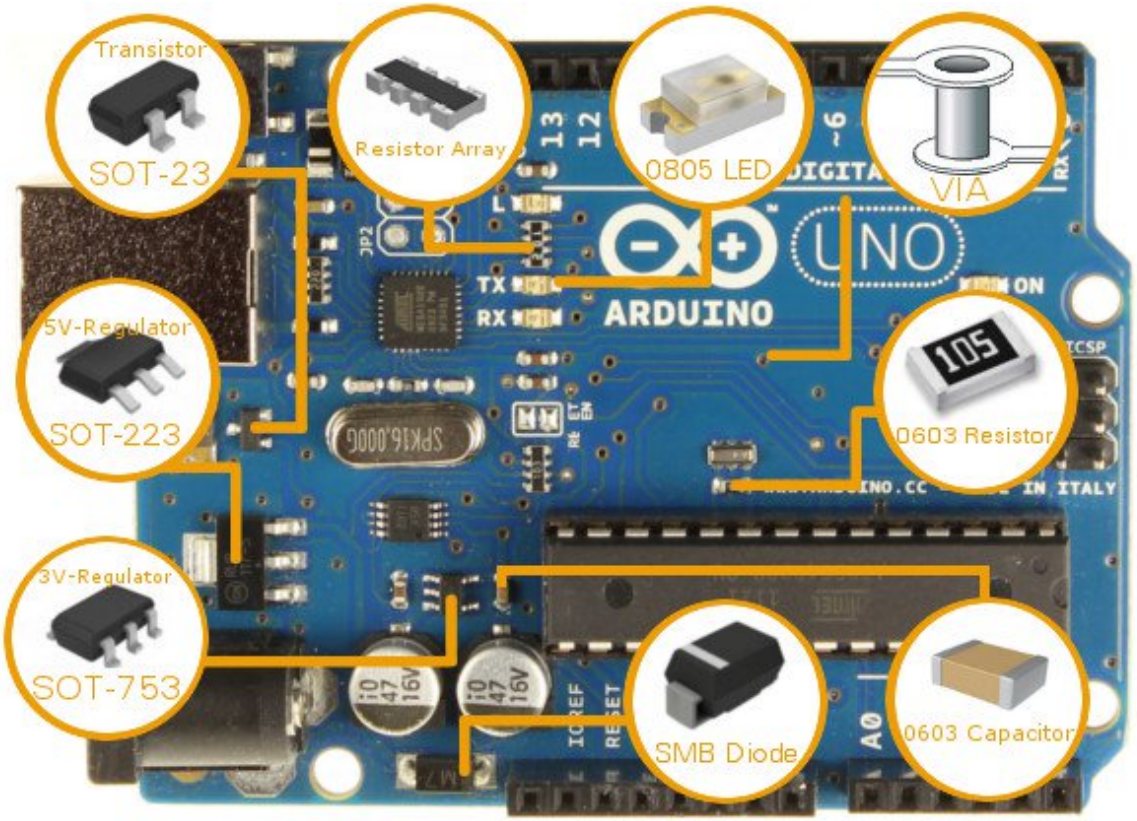
Microsistemul ARDUINO UNO R3

Identificarea conectorilor și ai pinilor pe modul

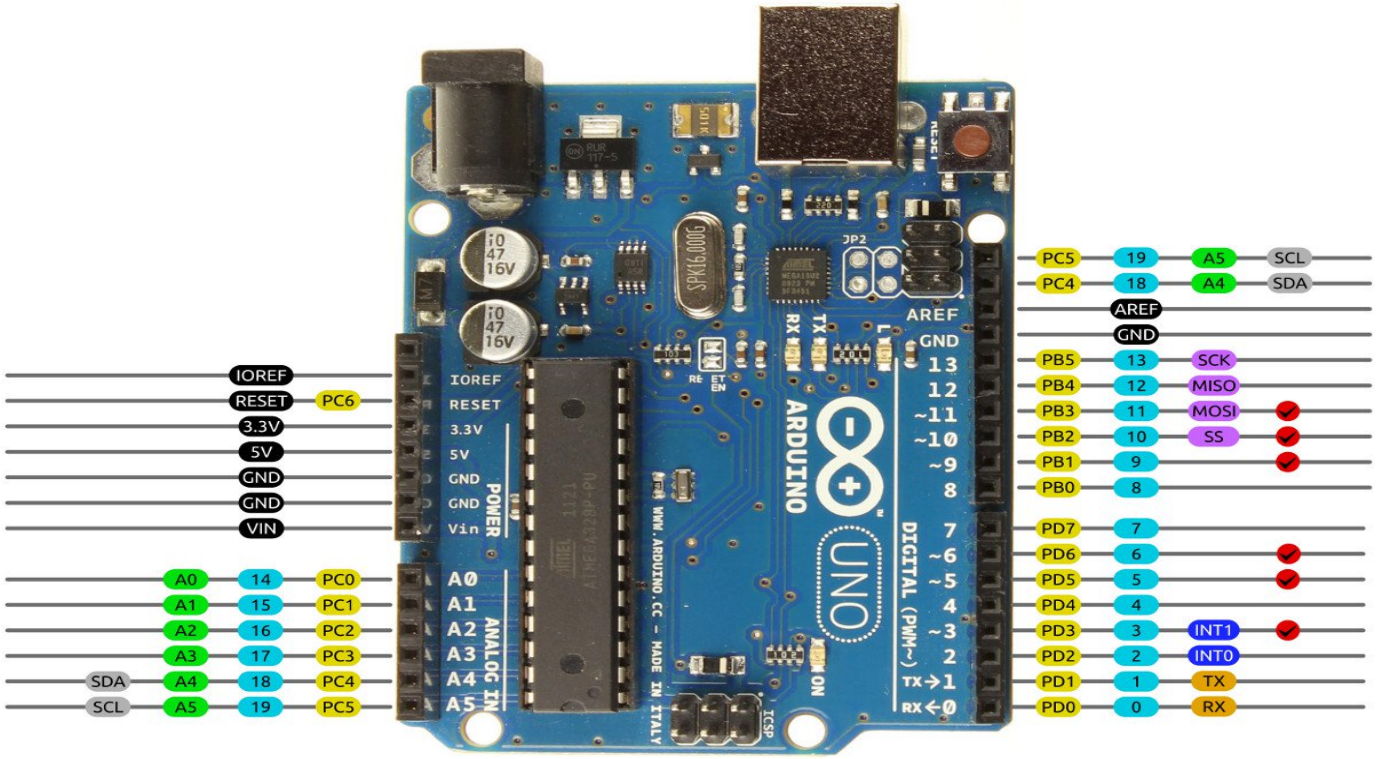


Microsistemul ARDUINO UNO R3

Componente electronice pe modul

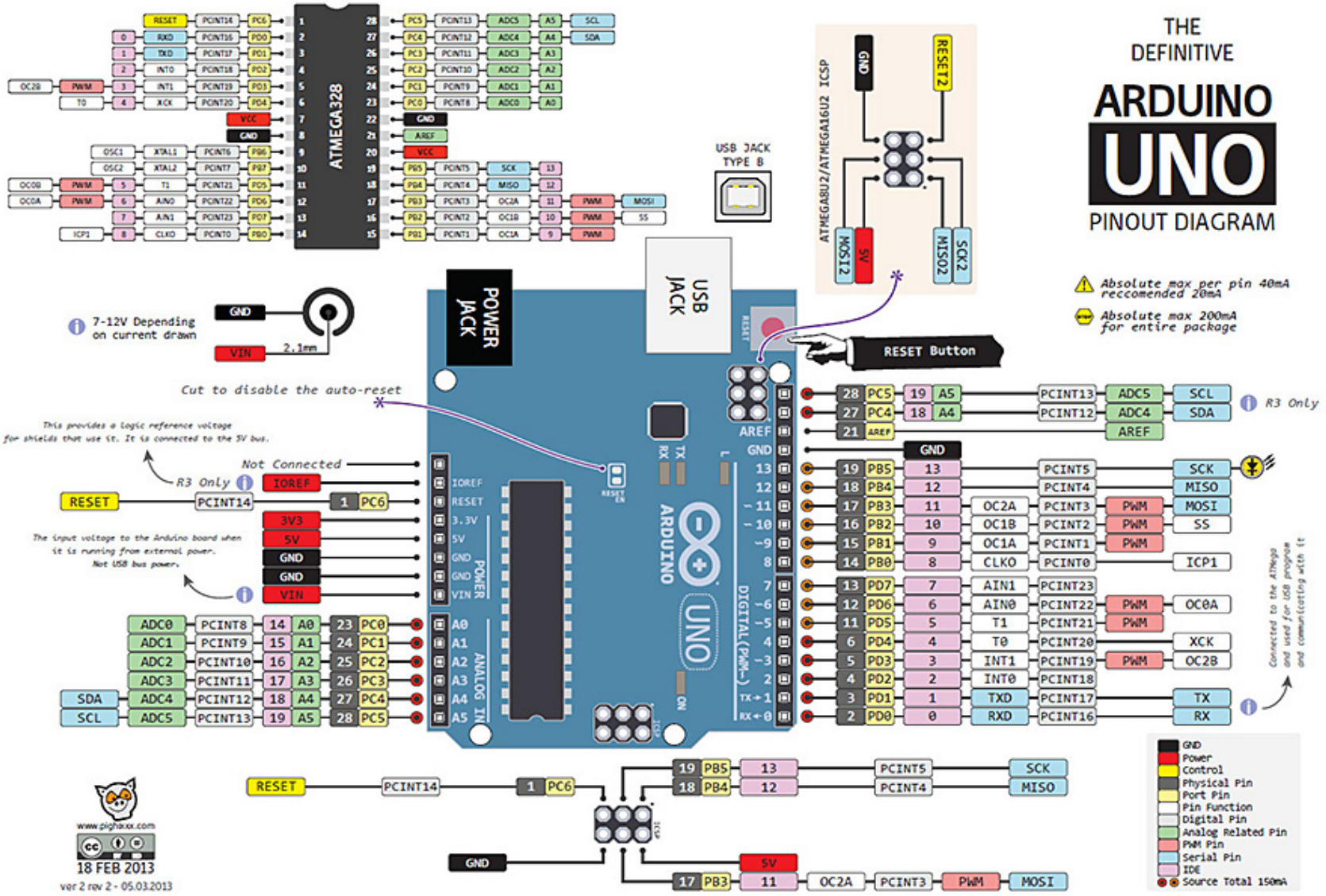


Arduino Uno R3 Pinout

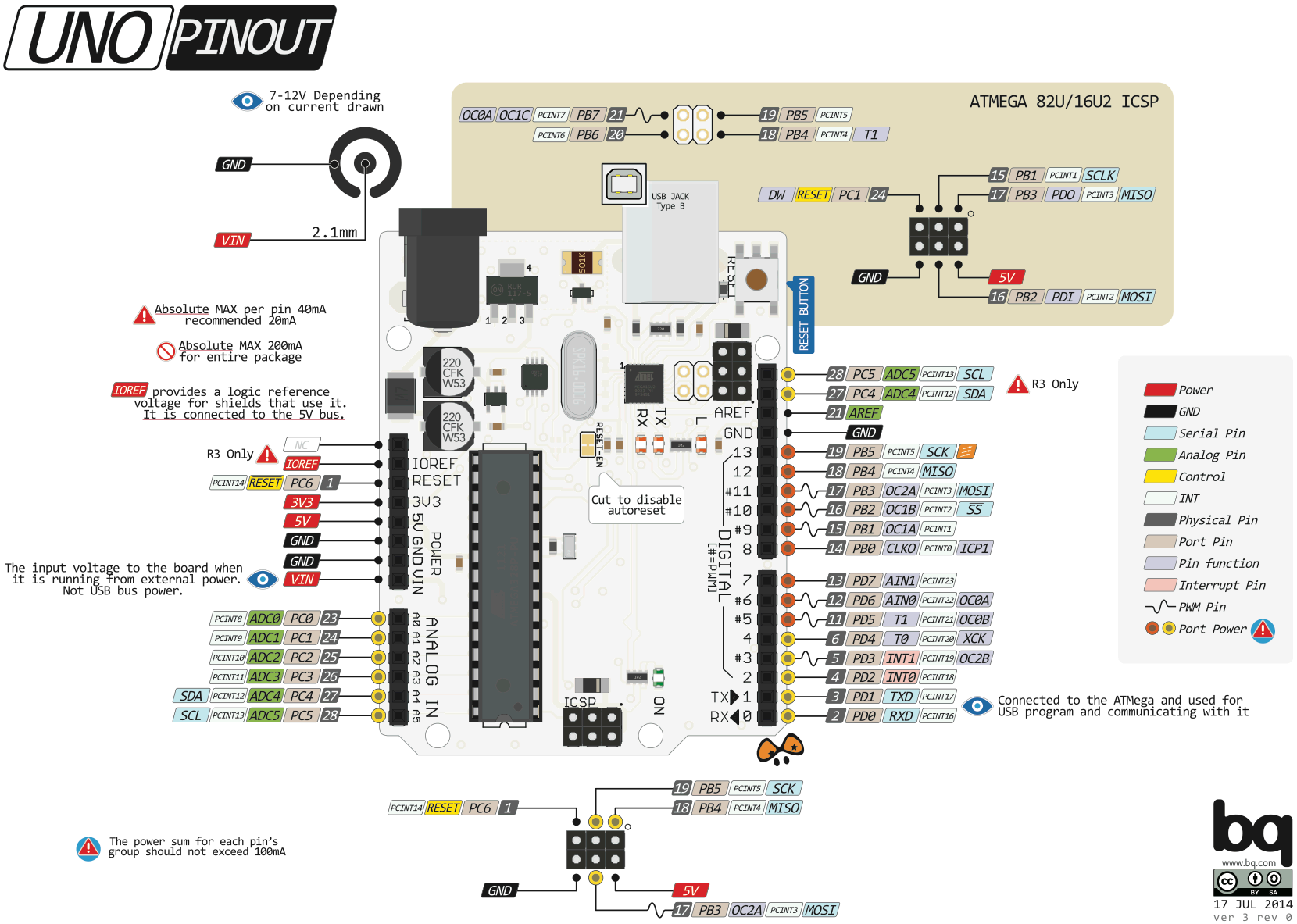


AVR DIGITAL ANALOG POWER SERIAL SPI I2C PWM INTERRUPT

Microsistemul ARDUINO UNO R3



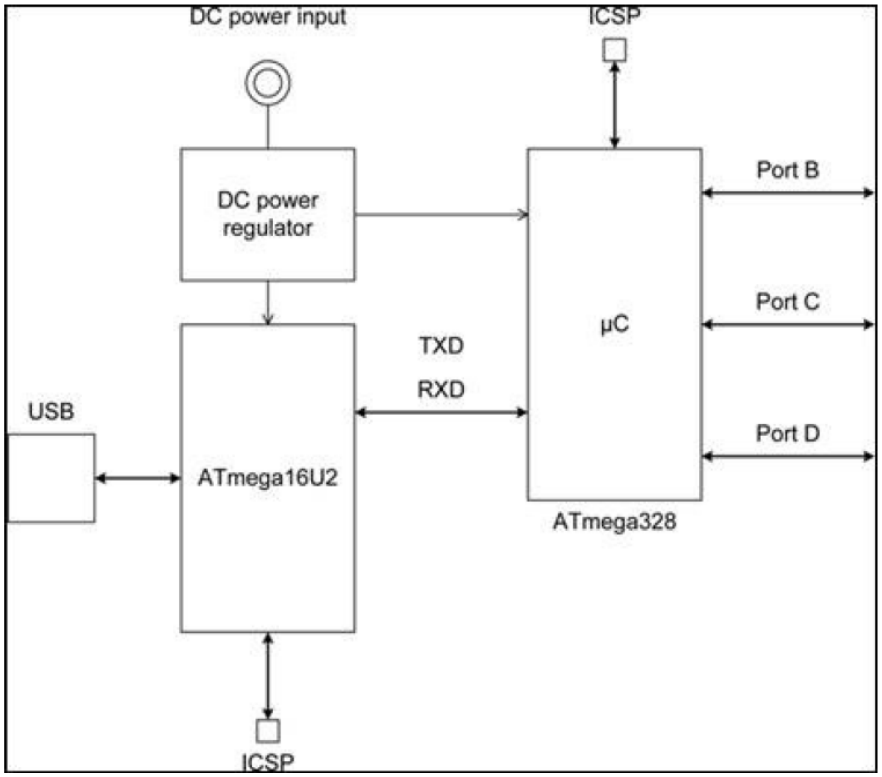
Microsistemul ARDUINO UNO R3



17 JUL 2014
ver 3 rev 0

Microsistemul ARDUINO UNO R3

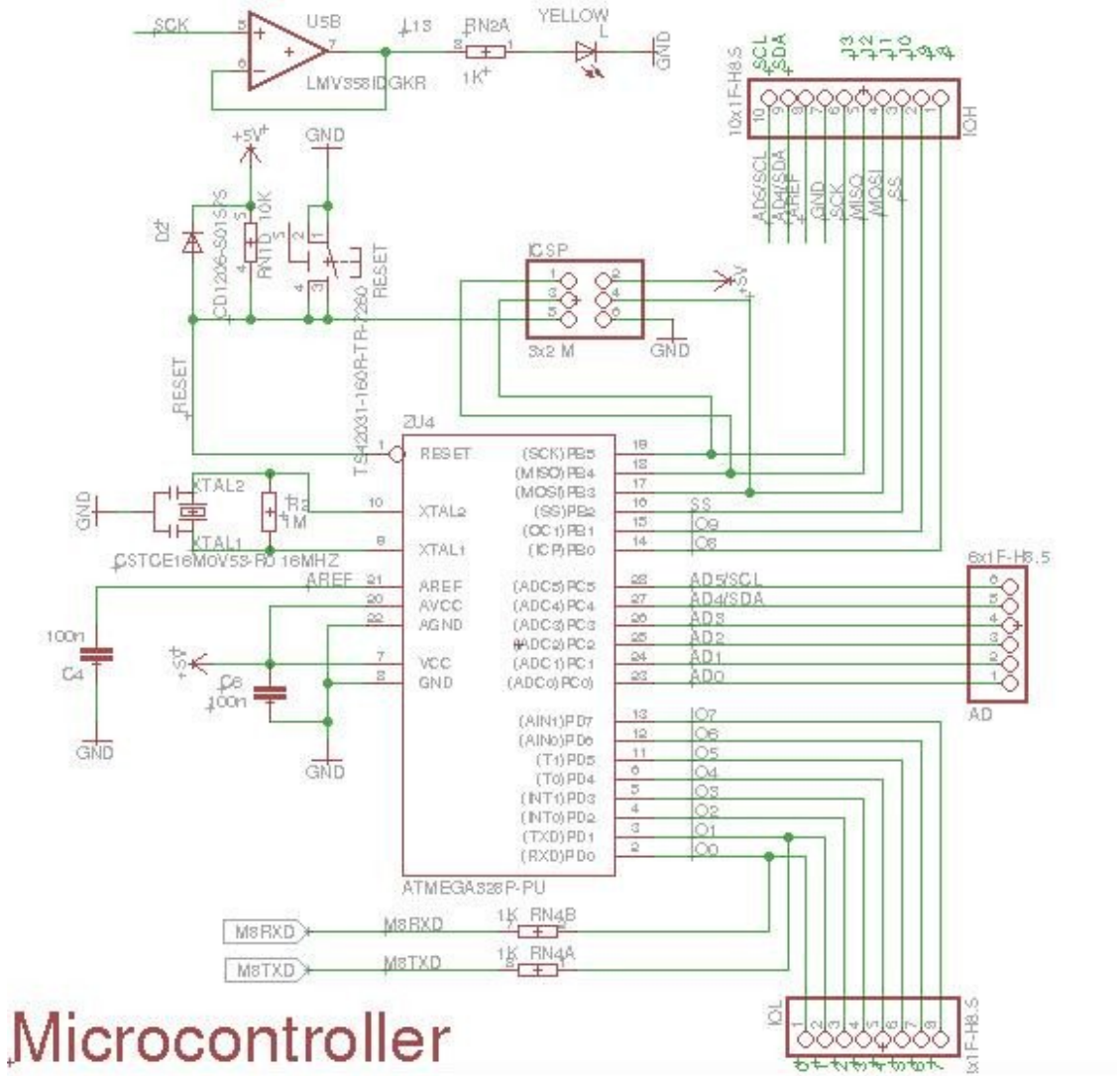
Schema bloc a modulului ARDUINO UNO R3



Apar 3 componente majore:
- Microcontrolerul ATMega 328P
- Interfața USB (realizată cu un alt microcontroler ATMega 16U2)
- Sistemul de alimentare

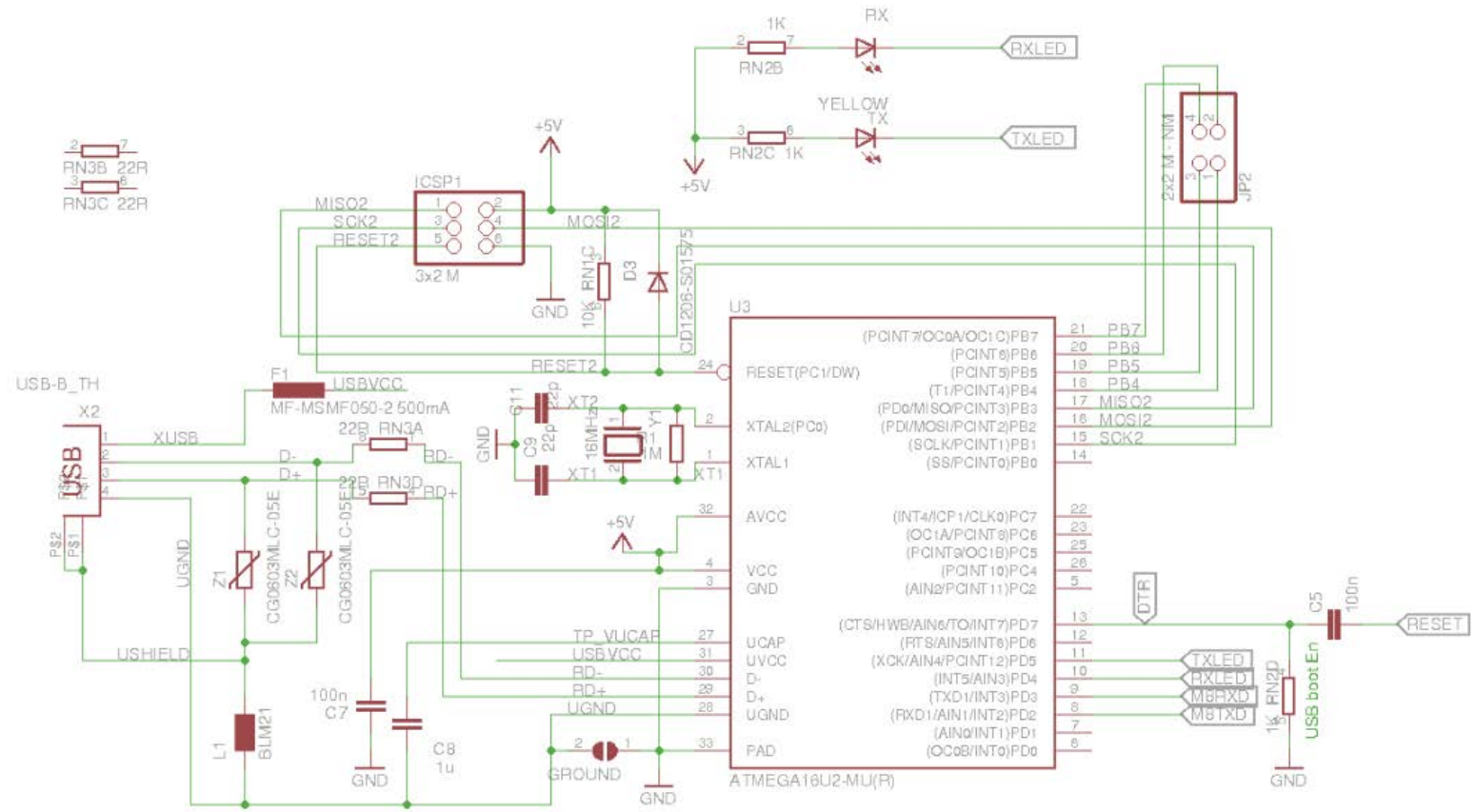
ICSP – In Circuit Self Programming (permite programarea memoriei ROM flash)

Microsistemul ARDUINO UNO R3



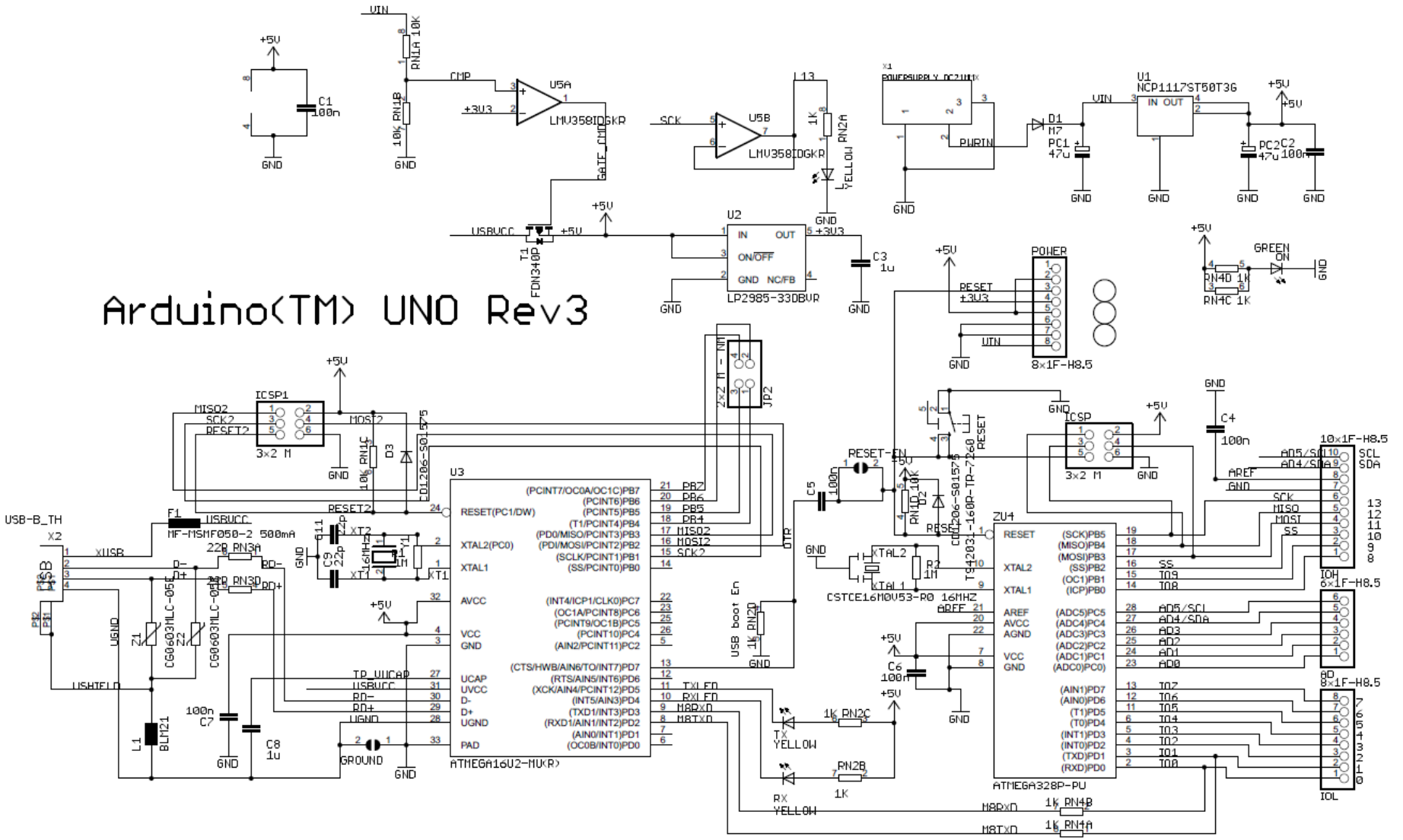
Microcontroller

Microsistemul ARDUINO UNO R3

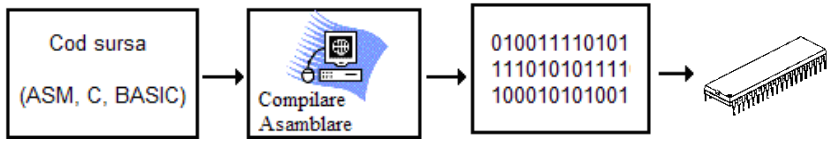


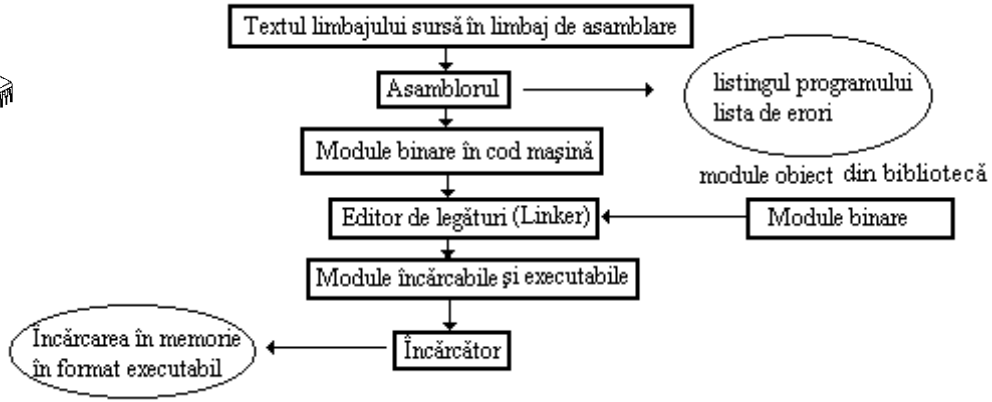
USB Bridge

Microsistemul ARDUINO UNO R3



Limbaje de programare
-asamblorul şi compilatorul-





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

Asamblorul/Compilatorul este un program care acceptă la intrare un text numit "program sursă", scris în limbaj de asamblare sau C şi produce la ieşire 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:

- Cblock ... Endc, db, equ, org, global, #define, #include

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:

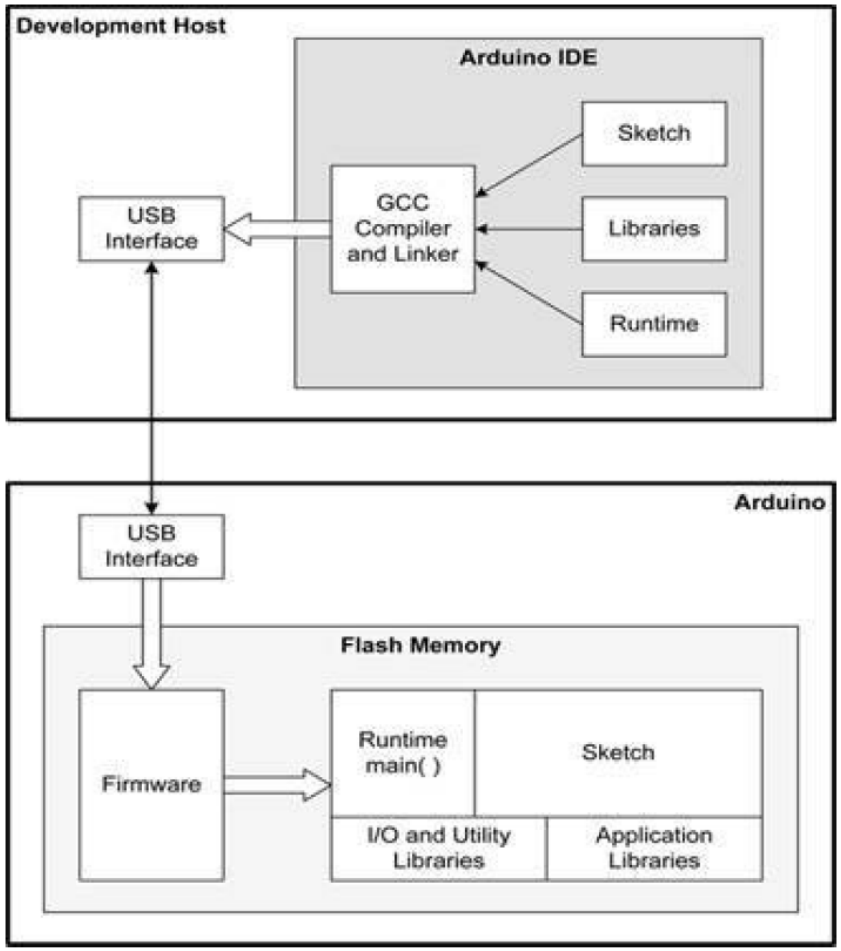
→ d'45'. Prefixul este litera d urmată de numărul în format zecimal delimitat de apostrof.

→ .45. Punctul utilizat ca prefix arată că numărul este în format zecimal.

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

Microsistemul ARDUINO UNO R3

Dezvoltarea aplicațiilor pentru ARDUINO UNO R3



Bibliografie

- http://www.atmel.com/images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf
- http://www.microchip.com/wwwproducts/en/ATmega328P
- http://www.microchip.com/design-centers/8-bit
- https://www.arduino.cc
- https://ro.wikipedia.org/wiki/Arduino
- http://www.electronicshub.org/arduino-line-follower-robot/
- https://circuitdigest.com/microcontroller-projects/line-follower-robot-using-arduino
- https://www.robofun.ro/arduino_uno_v3
- https://en.wikiversity.org/wiki/Introduction_to_Computers

Vă mulţumim pentru atenţie !