Sisteme de calcul - generalităţi

În cea mai simplă formă a sa, un sistem de calcul constă din cinci părţi principale funcţionale : blocurile de intrare şi ieşire, memoria de date, memoria program, aritmetica/logica de calcul (CPU), aşa cum se arată în Fig. 1. Unitatea de intrare acceptă informaţia codificată de la operatorii umani, de la dispozitive electromagnetice, sau de la alte computere conectate la ea prin linii de comunicaţie digitale. Informaţia este stocată în memorie pentru a fi referită ulterior sau este tratată imediat de către unitatea aritmetică şi logică realizând operaţia dorită. Paşii de procesare sunt determinaţi de un program ce se află stocat în memorie. În final rezultatele sunt trimise înapoi în lumea exterioară cu ajutorul unităţii de ieşire.

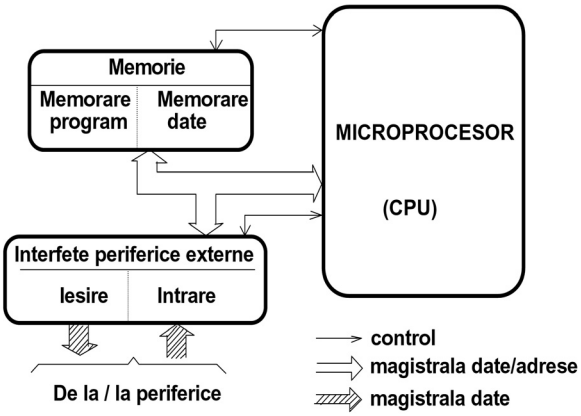

Fig. 1 Arhitectura unui sistem de calcul

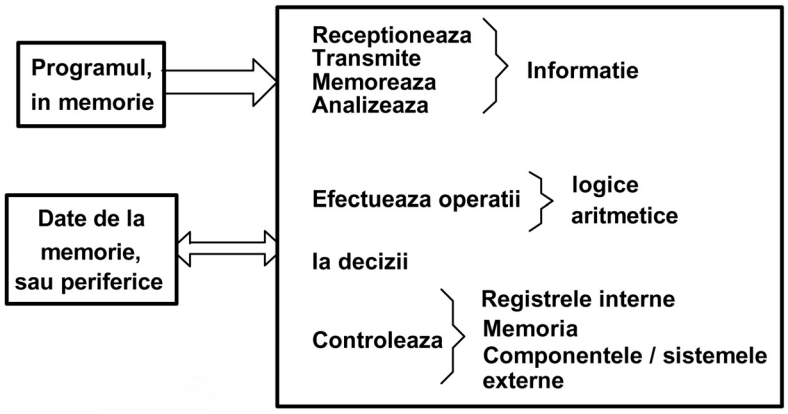

Fig. 2 Funcţiile CPU

Unitatea de procesare a informaţiei (CPU) are cel mai important rol din acest sistem, şi trebuie să îndeplinească funcţiile reprezentate în Fig. 2.
Să considerăm un exemplu tipic de adunare a două numere ce se află în memoria principală. Aceste numere sunt aduse în unitatea aritmetică unde adunarea va avea fizic loc după care suma poate fi stocată în memorie.

In Fig. 3 este reprezentată structura funcţională şi fluxul de informaţii într-un sistem cu microprocesor.

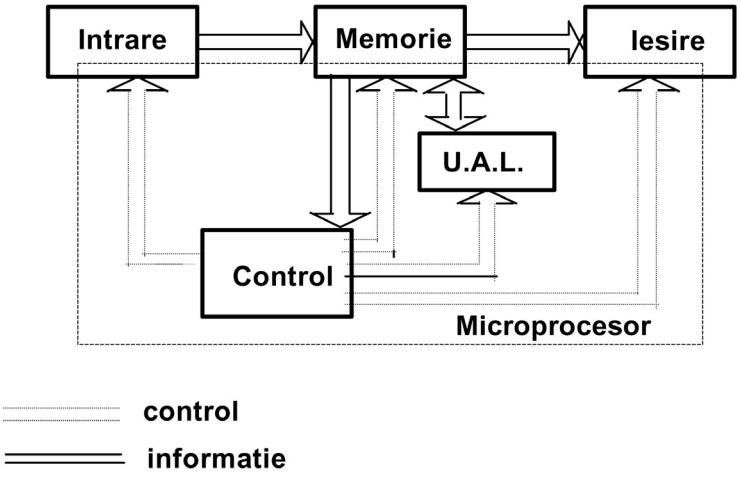

Fig. 3 Structura unui sistem de calcul

Sistem de calcul utilizând microprocesorul I8080

Structura generală a unui sistem cu microprocesor este compusă din:
1. Unitatea centrală de prelucrare (conţine microprocesorul)
2. Memorie operativă
3. Interfaţa de intrare/ieşire (porturi). (vezi pagina anterioară)

Aceste dispozitive sunt interconectate printr-o serie de semnale şi anume:

- semnale de adresă: organizate într-o magistrală de adresare, pe care se transmite, codificat binar, adresa locaţiei de memorie la care are loc accesul;
- semnale de date: grupate în magistrala de date – de lungime egală cu cea a cuvântului procesat – pe care circulă codul binar al cuvântului din memorie adresat în momentul respectiv;
- semnale de control: care formează magistrala de control – mai puţin omogenă decât primele 2 grupări – reuneşte toate semnalele care determină tipul accesului la memorie; citire, scriere, precum şi semnale cu rol de control în comunicaţia microprocesorului cu dispozitivele periferice.

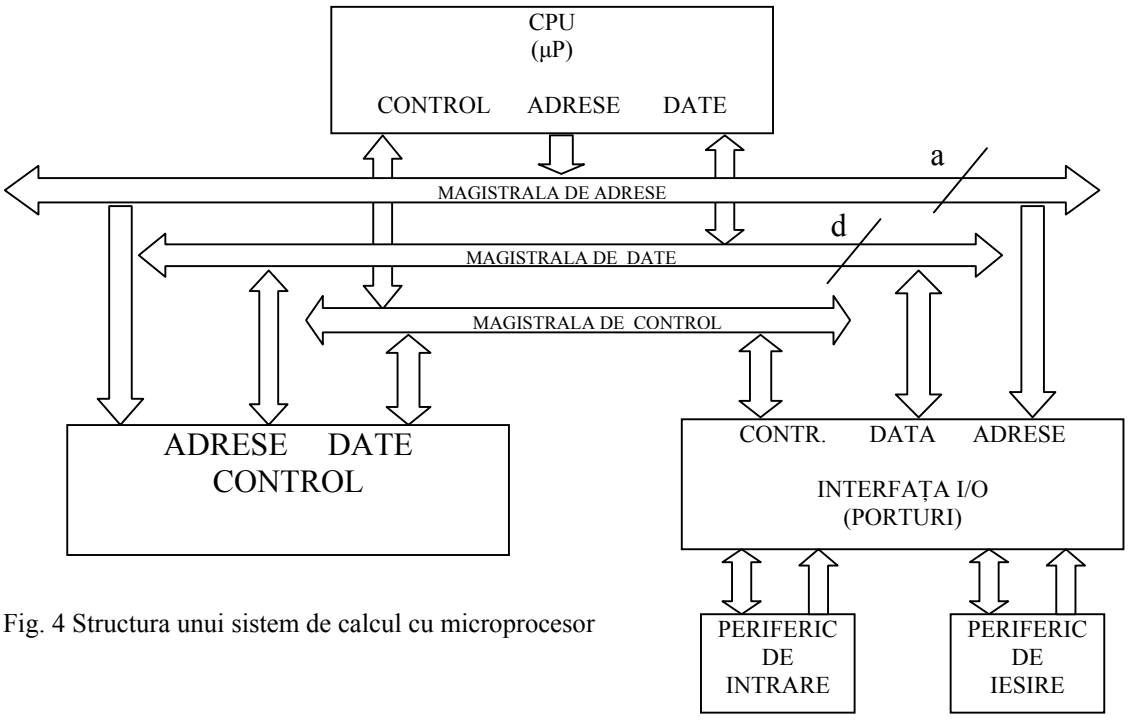

Fig. 4 Structura unui sistem de calcul cu microprocesor

Lucrul cu memoria operativă se desfăşoară astfel: fiecare locaţie de memorie are o adresă proprie; conţinutul locaţiei (data) este accesat numai după ce microprocesorul depune respectiva adresă pe magistrala de adrese. Datele vor apare pe magistrala de date, iar semnalele de control activate la momentul respectiv dau sensul de transfer.

Între microprocesor si periferice obişnuit se interpune un bloc de interfaţă compus din circuite standard, specializate, denumite **porturi,** care preiau procedurile de lucru specifice fiecărui tip de periferic. Astfel, microprocesorul trebuie să asigure comunicaţia cu porturile interfeţelor. Un port de intrare/ieşire este compus dintr-un registru împreună cu logica de control aferentă.

Comunicaţia cu porturile I/O are loc după aceleaşi principii ca şi lucrul cu memoria operativă: - fiecare port are o **adresă** la care poate fi accesat conţinutul său de către microprocesor. Un cuvânt va fi transferat pe magistrala de date – spre/dinspre port – în conformitate cu semnalele de control activate

care determină transferul. De obicei, aceste semnale sunt altele decât cele active la dialogul microprocesorului cu memoria.

3. Memoria operativă

Un microprocesor nu poate funcţiona decât dacă i se transmite pas cu pas succesiunea de operaţii pe care trebuie să le execute.

Memoria operativă constituie spaţiul de lucru al microprocesorului. Aici sunt stocate programele pentru a fi executate şi tot aici sunt stocate eventual rezultatele intermediare sau finale.

Registrul este o unitate de memorie capabilă să memoreze un număr de biţi egal cu dimensiunea (capacitatea) sa. Poate fi considerat ca fiind format dintr-un număr de bistabile, procedurile de înscriere şi citire realizându-se simultan la nivel fizic pentru toate bistabilele componente.

Memoria în ansamblul ei poate fi privită ca o alăturare de regiştri, fiecare registru având o adresă proprie.

În funcţie de tipul de informaţie pe care conţine memoria poate fi clasificată în:

a) Memoria program

aici se memorează succesiunea de instrucţiuni corespunzătoare aplicaţiei date;
este o memorie care poate fi **numai** citită de microprocesor pentru că trebuie să păstreze informaţii necesare tot timpul cât durează aplicaţia respectivă, fără a fi alterate fără voie.
se realizează cu dispozitive integrate de tip ROM (Read Only Memory) şi PROM (Programable ROM), EPROM.

b) Memorie de date

necesită facilitaţi de înscriere şi citire a informaţiei fără ca aceasta să fie păstrată permanent, deci în ea se reţin variabilele programului.
este o memorie de tip volatil;
se realizează cu dispozitive integrate de tip RAM (Random Acces Memory), SRAM (Static RAM), SDRAM.

Din punct de vedere conceptual, împărţirea în memorie program şi memorie de date nu este necesară. Din considerente practice, este util să existe o memorie nucleu care să păstreze programele necesare funcţionării corecte a sistemului şi al cărui conţinut să nu se piardă odată cu dispariţia tensiunii de alimentare.

Memoria sistemului se organizează liniar, de obicei în cuvinte cu dimensiunea identică cu a cuvântului microprocesorului, deci $l_{CMEM} = d$, unde l_{CMEM} reprezintă lungimea cuvântului stocat în memorie. Ca urmare blocul de memorie operativă poate fi privit ca un spaţiu m, denumit *spaţiul memoriei*, în care se pot memora cuvinte binare de lungime d.

Numărul maxim de cuvinte al spaţiului m care pot fi adresate *direct* de către microprocesor este determinat de dimensiunea *a* a magistralei de adrese. Rezultă deci că spaţiul adreselor, A, conţine 2^a adrese cu care pot fi selectate cuvinte în spaţiul memoriei.

Aşadar, se poate spune că *operaţia de adresare* alocă unei adrese din A un element din M prin funcţia de translare:

$f_T: A \rightarrow M$.

Dacă A = M atunci f_T este *funcţia identitate.* La multe sisteme pentru conducere automată, dimensiunea memoriei, , M, este mai redusă decât capacitatea A de adresare a microprocesorului.

În Fig. 5 de mai jos se dă structura unui sistem de calcul cu microprocesorul Intel 8080, iar în Fig. 6 se dă arhitectura internă a microprocesorului I8080.

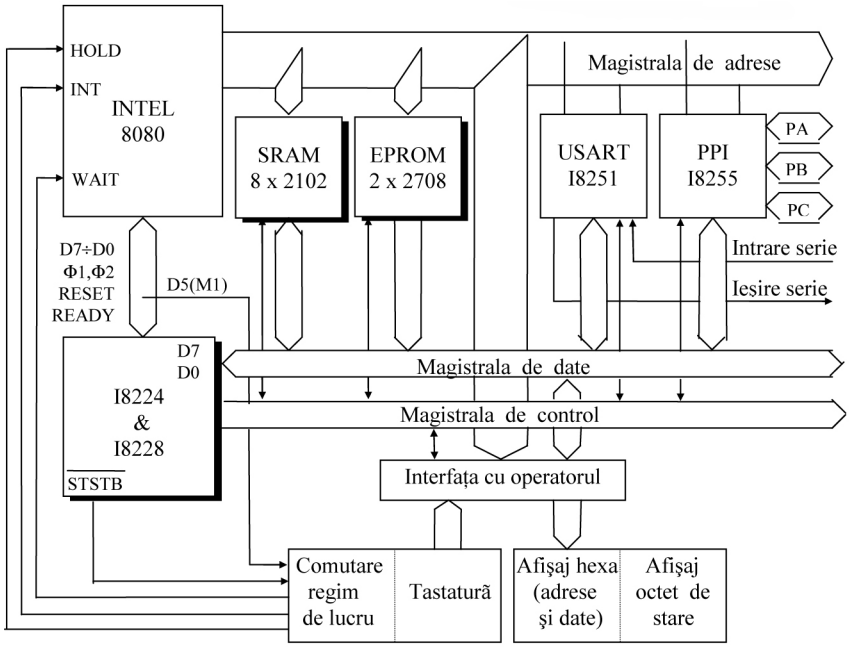

Fig. 5

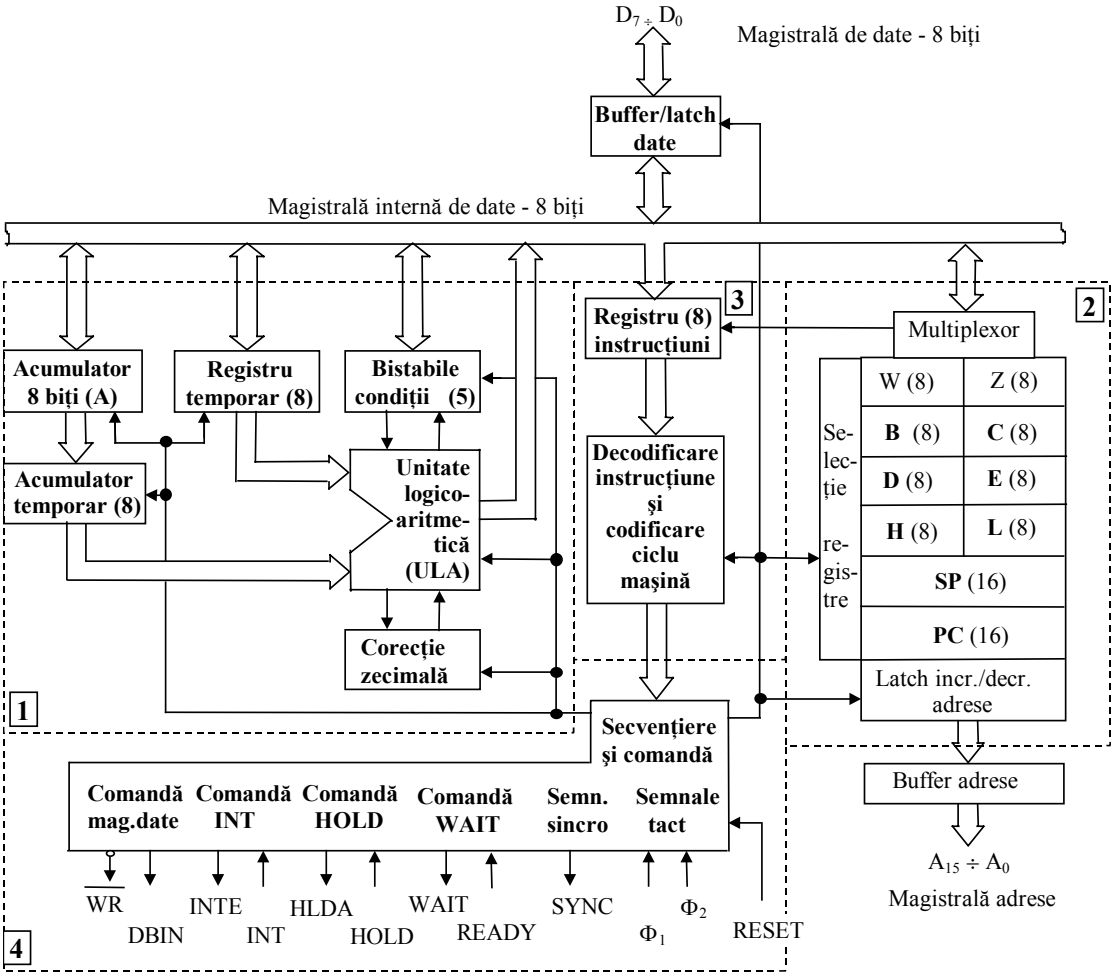

Fig. 6 Schema bloc a microprocesorului I8080

Datorită complexităţii crescute a sistemului de calcul (uneori nejustificată pentru aplicaţii simple) , au apărut microcontrolere, care includ in arhitectura internă majoritatea blocurilor externe dintr-un sistem cu microprocesor; astfel, structura sistemului de calcul din Fig. 5 se regăseşte (în mare parte) în arhitectura internă a microcontrolerului MCS8051, prezentată în Fig. 7.

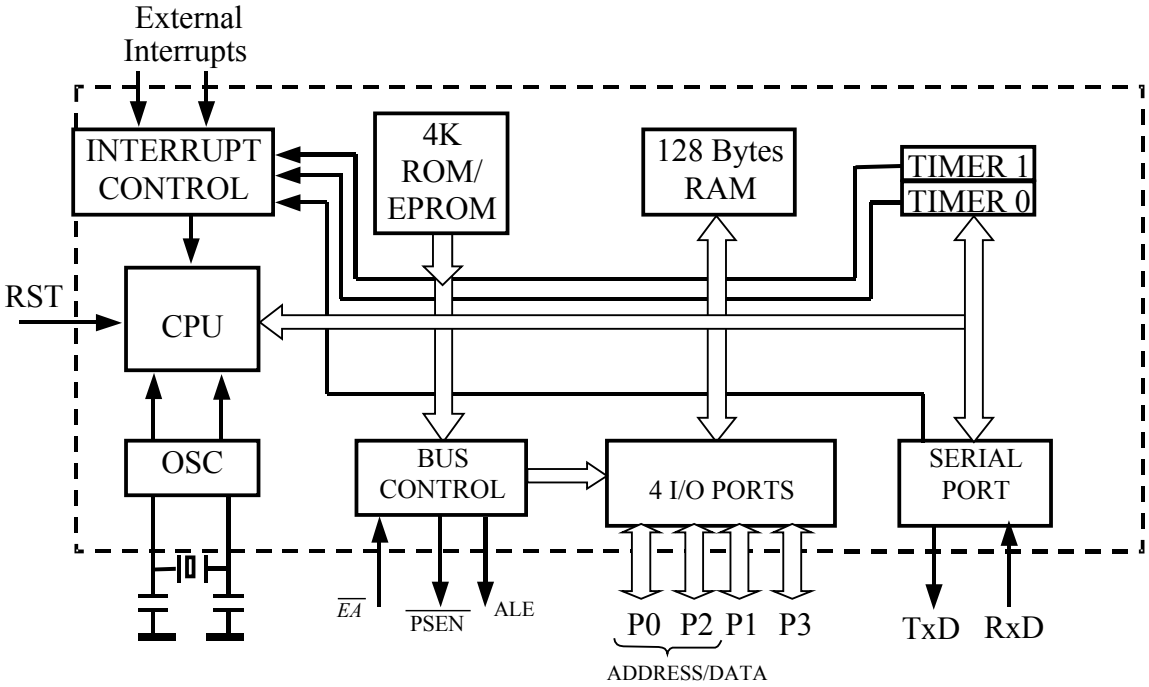

Fig. 7 Arhitectura internă a microcontrolerului MCS8051

Este cât se poate de adevărat că microcontrolerele pot avea arhitectura internă mai mul sau mai puţin complicată faţă de cea a unui microprocesor ; cel mai important lucru, însă, este alegerea optimă a microprocesorului sau microcontrolerului pentru aplicaţia dorită. De exp. ,se doreşte realizarea unei aplicaţii simple de aprindere intermitentă a unui LED, cu timpi de aprindere/stingere variabili în funcţie de stare unor comutatoare. Pentru realizare, am nevoie de CPU, Bus Control, memorie program (ROM / Eprom / EEprom), Memorie de date (RAM), Porturi de Intrare/ieşire şi Oscilator.
Cum toate acestea se găsesc untr-un singur microcontroler, din considerente de economie de bani şi efort (de realizare a aplicaţiei), pentru realizare, se va alege microcontrolerul.
Dar în cazul în care se doreşte o gestionarea rapidă a informaţiilor din memorie, folosind protocolul DMA (Direct Memory Acces) (+regiştrii aferenţi), transferul de informaţie nu mai trece „prin" CPU , ci se face legătura „directă" între memorii ~ caz în care se va alege un microprocesor.

Arhitecturi CISC şi RISC

Aşa cum se poate observa din Fig. 5; există linii de adresă si linii date (bus de adresă / bus de date), separate; iar din Fig. 7 se poate observa că există un singur bus de adresă/date. Astfel, în primul caz spunem că avem arhitectură Harvard, iar în al doilea caz, arhitectură von Newman. Arhitectura Harvard este mai nouă decât von Newman şi determină o viteză de lucru mai mare a microcontrolerului , putându-se găsi în literatura de specialitate şi sub de numirea de arhitectură RISC (Reduced Instruction Set Computer), deoarece pentru obţinerea unei informaţii dintr-o locaţie de memorie, CPU apelează la magistrala de adrese pentru găsirea unei locaţii de memorie, informaţia din acea locaţie nu se mai întoarce la CPU pe acelaşi bus (linii), ci pe bus separat de date. Acest lucru este avantajos

deoarece în timp ce se primeşte informaţia de date, deja pe bus-ul de adresă (CPU) trimite adresa următorului operand utilizat în execuţia unor operaţiuni aritmetice (adunare, scădere, etc.).
Microcontrolerele cu arhitectura von Newman se mai găsesc în literatura de specialitate şi sub denumirea de CISC (Complex Instruction Set Computer). In Fig. 8 sunt prezentate schematic cele două tipuri de arhitecturi.

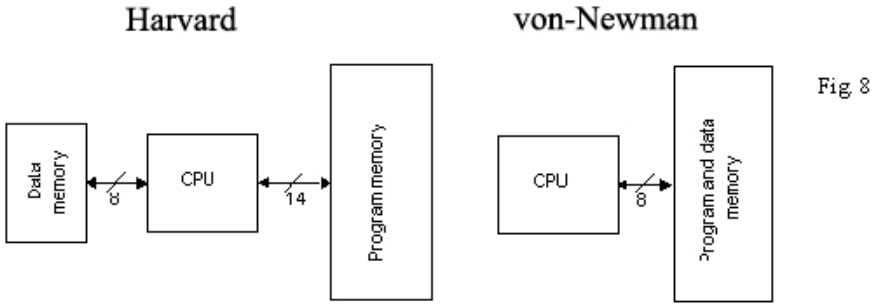

Fig. 8