Arhitectura Calculatoarelor

T.1 – NOŢIUNI INTRODUCTIVE

Noţiuni Introductive. Obiectivele disciplinei. Schema de bază a unui calculator. Rolurile componentelor de bază, Principiul de funcţionare a lor, Schema nivelelor conceptuale a calculatoarelor.

Scopul Lecţiei: De a face cunoştinţă cu noţiunile de bază din Arhitectura Calculatoarelor, de a înţelege noţiunea de nivele conceptuale, de a înţelege structura unui sistem de calcul şi funcţiile componentelor lui.

Studentul trebuie ***să cunoască:***

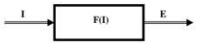
- *Schema de bază a calculatoarelor digitale;*
- *Rolurile componentelor de bază a calculatorului*
- *Schema nivelelor conceptuale a unui calculator*

Conf. Univ. Dr. Creţu Vasilii

Noţiunea de sistem

Sistem (definiţie) = Un ansamblu de elemente inter-relaţionate ce compun un întreg.
Termenul de „system" in latină şi greacă înseamnă „a pune împreună, a combina".
Un **subsistem** este o parte a unui sistem.
În mod tipic un sistem este alcătuit din componente (elemente) care sunt interconectate şi interacţionează pentru a facilita fluxul de informaţie.
În funcţie de tipul sistemului acesta se poate diferenţia de elemente, maşini, procese.
Tipuri de sisteme:
• **Sisteme deschise** – Sisteme care pot fi influenţate de evenimentele din afara graniţelor lor.
• **Sisteme închise** - Sisteme care nu sunt influenţate de evenimentele din afara graniţelor lor.
• **Sisteme dinamice** – Sisteme cu componente sau fluxuri care se **schimbă în timp.**
O distincţie care trebuie specificată este între sistemele **fizice** şi cele **conceptuale**.
Cele conceptuale sunt abstracte şi au la bază idei, ajutând la modelarea sistemelor fizice.
Arhitectura sistem = dispunerea si interconectarea componentelor pentru a obtine functionalitatea dorita a sistemului.

Conceptul de cutie neagră (black box)
Definiţie. Un sistem cu intrari (I), iesiri(E) si transformari (F(x)) cunoscute, dar cu conţinut necunoscut se numeste **black-box**.
Proprietatea cea mai importanta a cutiei negre este *utilizabilitatea*. I.e. utilizare fără a cunoaste detalii de implementare.



Conceptul de cutie neagra este utilizat in proiectarea si realizarea sistemelor de calcul.
Sistemele sunt proiectate modular folosind cutii negre dupa urmatoarea regula: "ori de cite ori este necesara o functie intr-un sistem se foloseste o cutie neagra care realizeaza aceasta functie". Modul in care este implementata respectiva functie nu interseaza pe utilizator, ci doar functionalitatea cutiei negre si modalitatea de folosire a ei.
Pentru a facilita construcţia sistemelor din module cu funcţionalitate cunoscută (black box) acestea au fost **standardizate**.
Un **standard** cuprinde o descriere a modului de utilizare a unui modul (specificaţii de utilizare).
Organizaţiile internationale de standarde: **ISO** (International Standard Organization), **IEEE** (Institute of Electrical and Electronics Engineers), **IETF** (Internet Engineering Task Force) au elaborat o serie de standarde, respectate de producatori, în realizarea modulelor respective.

Un **sistem de calcul** este un sistem care execută programe stocate în memorie în interacţiune cu mediul extern.
Componentele sistemului de calcul sunt
- hardware – echipamente
- software - programe

Arhitectura sistemelor de calcul
Arhitectura sistemelor de calcul sau arhitectura calculatoarelor este **teoria** din spatele construcției unui calculator. În același mod în care un arhitect stabilește principiile și obiectivele construirii unui proiect ca baze ale unor planuri de construcție, în același mod un arhitect de calculatoare stabilește arhitectura sistemului de calcul ca bază a specificațiilor de proiectare.
Obiectivul principal în arhitectura unui sistem de calcul îl reprezintă un **raport cost/performanță** cât mai bun.
Componenta sistem = cutie neagra.
Arhitectura sistem = dispunerea si interconectarea componentelor pentru a obtine functionalitatea dorita a sistemului.

Arhitecturi generale
1. **Arhitectura multistrat** – niveluri ierarhice. Un nivel inferior ofera suport nivelului superior pentru executia functiilor sale
2. **Decompozitia functionala** – descompunere a componentelor dupa functiile realizate
3. **Decompozitia conceptuala** – descompunere a sistemului dupa entitatile identificate (ce inglobeaza toata functionalitatea obiectului).

Descrierea sistemului de calcul
Definiție. Masina de calcul care executa secvential programe scrise in limbajul masinii respective, stocate in memorie, in interactiune cu mediul extern.
Un program = solutie **algoritmica** a unei probleme scrisa intr-un limbaj, numit limbaj de programare.
Un **algoritm**=solutie secventiala a unei probleme.
Limbajul de programare=limbajul masina.
Limbajul masina este limbajul executat de masina.
Limbajul de programare este translatat in limbaj masina pentru executie.
Sunt doua forme de executie:
- **compilare si executie**
- **interpretare** (masina virtuala care interpreteaza si executa programul)

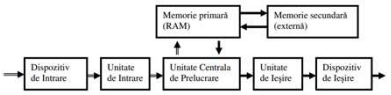
Elemente componente:
- de **procesare** (prelucrare)
- de **memorare**
- de **comunicatie**
  - **cu mediul extern**
  - **suport de comunicație**

Program = succesiune de instructiuni ce implementeaza un algoritm.

Din punct de vedere perceptiei, sistemul de calcul este impartit in doua mari parti:
- partea **hardware** – reprezentata de circuitele electronice, placi, cabluri, memorii, etc. care reprezinta echipamentul propriu-zis de calcul si care sunt tangibile.
- partea **software** – reprezentata prin programe care implementeaza algoritmi si reprezinta idei abstracte.

Diferenta dintre hardware si software pina nu demult a fost evidenta, cu timpul insa ele au devenit logic echivalente. Ambele putind realiza aceleasi functii, iar alegerea implementarii facindu-se dupa criterii pret/performanta

Partile componente ale sistemului de calcul.
Fluxul informației
Pentru a intelege functionarea calculatorului vom introduce notiunea de **informatie** care, furnizata de utilizator sau mediu, este convertita în format binar, intern, prelucrat de sistemul de calcul (***date***).
Adoptarea reprezentarii binare a fost impusa de utilizarea in constructia calculatoarelor a dispozitivelor cu doua stari stabile, notate conventional cu 0 si 1. Unitatea de masura pentru numerele binare este **bit-ul** (**bi**nary digit).
Prezentare succinta a componentelor sistemelor de calcul cu referire directa la calculatoare. Numeroasele componente ale unui sistem de calcul pot fi grupate in unitati avind functii mai complexe bine precizate. In figura urmatoare denumirea fiecarei unitati indica functia ei, iar sagetile de legatura arata modul de transmitere a informatiei de la una la alta.



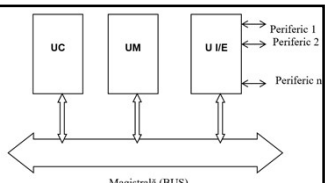
Informatia, furnizata de mediul extern (utilizator), este preluată de **dispozitivul de intrare**, codificată (convertită în format binar) și transmisă **unității de intrare** care realizează interfața cu unitatea de procesare (unitatea centrală de procesare = UCP) .
Exemple de dispozitive de intrare: tastatura, mouse, scanner, MODEM, etc. Astfel, la tastatura, apasarea unei taste produce codului binar corespunzator al tastei apăsate.
Scannerul preia o imagine și o transformă într-o succesiune de coduri binare. MODEM-ul preia datele transmise de la distanță.
Unitatea de intrare realizează interfața cu UCP a.î. dispozitivele de intrare pot fi diferite. Informatia este inregistrata si pastrata în **memoria primară**. De aici ea poate fi transmisa ulterior altor unitati functionale. Informatia este supusa prelucrarii in **UCP**. UCP este formată din **unitatea de calcul** și **unitatea de comandă**. Unitatea de calcul efectueaza operatii simple, aritmetice si logice, asupra unor operanzi din memorie, inregistrind rezultatele tot in memorie. **Unitatea de comanda** are ca rol coordonarea functionarii celorlalte unitati, pe baza unor instructiuni sau comenzi.

Informatia care nu este prelucrata la un moment dat poate fi pastrata in unitati de **memorie externa** (de obicei discuri magnetice), mai lente decit **memoria interna** (operativa) dar cu o capacitate mai mare. Informatia poate fi transmisa, daca este cazul,de la o memorie la alta.
Din punct de vedere al localizării, memoria sistemului de calcul este împărțită în:
- memorie internă
- memorie externă
Memoria internă este formată din:
- **RAM (Read Only Memory) –** volatila (își pierde conținutul dacă nu este alimentată cu curent electric) = memoria primară a sistemului de calcul (indispensabilă)
- **PROM (Programmable Read Only Memory)** - nevolatila = memorie care poate fi doar citită și care cuprinde programe ale fabricanților sistemului de calcul
- **CMOS (Complementar Metal Oxid Siliciu)** – memorie asemănătoare memoriei RAM, de capacitate foarte mică, alimentată permanent de la o baterie, care păstrează setările de configurare a sistemului de calcul.

Memoria externa - **discuri magnetice** și **optice.**
Rezultatele prelucrarilor sint transmise utilizatorului prin **unitatea de iesire** către **dispozitivul de ieșire**. Dispozitivul de ieșire realizeaza conversia datelor din format binar în formatul necesar reprezentarii informatiei. Exemple de dispozitive de ieșire : monitor, imprimnată, MODEM, plotter, etc. De exemplu, o imprimanta converteste codurile binare ale caracterelor in formatul tiparit. Similar, un monitor (display) transforma reprezentarile binare ale informatiei in formatul afișat.

SCHEMA DE BAZĂ A UNUI CALCULATOR

Orice calculator are în componență patru mari unități fundamentale:
-Unitatea centrală (UC).
-Unitatea de memorie (UM).
-Unitatea de Intrare/Ieșire (U I/E).
-Magistrale de interconectare (BUS-uri).

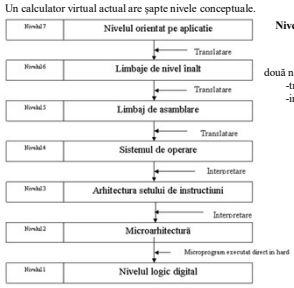


Rolurile acestor componente sunt:
-Unitatea centrală (UC) controlează toate componentele, executând instrucțiunile unui program; efectuează calcule aritmetice și logice.
-Memoria (UM) păstrează programele în curs de execuție și datele asociate lor.
-Unitatea de Intrare/Ieșire (U I/E) leagă sistemul cu lumea externă prin intermediul unităților periferice: ecran, tastatură, discuri, benzi magnetice, rețele etc.
-Magistralele sunt de trei feluri:
- magistrale de adresă, care vehiculează adresa memorie sau a unității I/E generată de UC (sau, în unele cazuri de alte unități de control);
- magistrale de date, care vehiculează informația (instrucțiuni, date) între UC, memorie și unitățile I/E;
- magistrale de control, care vehiculează semnalele utilizate de UC pentru controlul sistemului (adresă, memorie validă, adresă I/E validă, citire/scriere, așteptare, întrerupere etc.).

Principiul de funcționare a unui calculator este relativ simplu. În UM există programe, fiecare program având un număr de instrucțiuni. Ciclurile de executare a unei instrucțiuni sunt următoarele:
-Ciclul extragere instrucțiune (instruction fetch). UC face o citire a memoriei la adresa la care se află instrucțiunea. Instrucțiunea are un număr de biți, în funcție de arhitectura calculatorului, de obicei multiplu de 8. Instrucțiunea citită este adusă pe magistrală și depusă într-un registru al UC-ului.
-Ciclul de aflare a operanzilor. Oricare instrucțiune lucrează cu operanzi. Între operanzi se petrece o operație dată de un câmp al instrucțiunii, numit codul instrucțiunii. În această fază trebuie depistați operanzii, mai precis adresele unde se găsesc operanzii. Aceștia se pot găsi în două tipuri de locații:
-în registrele generale ale UC-ului;
-la o adresă de memorie.
Există mai multe tipuri de adresare pentru determinarea adreselor de operanzi. La sfârșitul acestui ciclu, în UC trebuie să se existe adresele fizice ale operanzilor participanți la instrucțiune.
-Ciclul de aducere a operanzilor în UC. În acest ciclu se aduc operanzii participanți la instrucțiune de la adresele determinate în ciclul anterior. Ei sunt aduși din registrele generale sau de la adresele de memorie în registrele funcționale.
-Ciclul de execuție propriu zisă. În acest ciclu are loc execuția propriu zisă a instrucțiunii, dată de codul instrucțiunii.
-Ciclul de depunere a rezultatului. Orice instrucțiune are ca scop final aflarea unui rezultat care poate fi un operand în cazul instrucțiunilor aritmetice(de exemplu suma pentru cod de adunare, produsul pentru cod de înmulțire) sau poziționarea unor indicatori în cazul instrucțiunilor logice (de exemplu, în cazul unui cod de comparație între doi operanzi, poziționarea indicatorului z=1 pentru identitatea celor doi operanzi).
La sfârșitul acestui ciclu, care înseamnă și sfârșitul executării instrucțiunii, se calculează adresa instrucțiunii următoare și adresa de la care va fi adusă instrucțiunea următoare.
Execuția unui program înseamnă execuția succesivă a instrucțiunilor din care este alcătuit. Programele care compun sistemul de operare asigură gestiunea resurselor (procesor, memorie, I/E) și fac legătura cu programele de aplicație.

SCHEMA NIVELELOR CONCEPTUALE A UNUI CALCULATOR

Un calculator virtual actual are şapte nivele conceptuale.

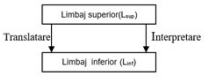


Nivelurile conceptuale ale maşinii virtuale corespunzătoare calculatoarelor actuale.

Maşina virtuală este organizată pe niveluri iar între două niveluri pot exista două operaţii:

-translatarea şi

-interpretarea.



Utilizarea limbajului inferior este greoaie şi de aceea s-a creat un limbaj superior, mult mai aproape de gândirea umană. Se pot scrie programe atât în L_{sup} cât şi în L_{inf} dar calculatorul va executa totdeauna setul de instrucţiuni din L_{inf} pentru care a fost proiectat fizic.

Translatarea înseamnă transformarea programului din L_{sup} în întregime într-un program din L_{inf} . Programul din L_{sup} este abandonat iar noul program din L_{inf} este încărcat în memorie şi executat. Translatarea seamănă cu compilarea.

Interpretarea înseamnă execuţia instrucţiunilor din L_{sup} , pas cu pas, fiecare instrucţiune fiind executată imediat. Este scrierea unui program în L_{inf} care preia programe din L_{sup} ca date de intrare şi le execută examinând fiecare instrucţiune pe rând şi executând secvenţa echivalentă de instrucţiuni direct în L_{inf} , dar fără să genereze un nou program în L_{inf} .

Diferenţa fundamentală între cele două tehnici este că în translatare un program mai întâi este transformat în totalitate în alt program şi apoi executat programul translat, în timp ce în interpretare se execută fiecare instrucţiune pas cu pas.

Faţă de interpretare, translatarea are avantajul unei viteze de execuţie mult mai mari.

Nivelurile au următoarele semnificaţii:

Nivelul 1 este hard pur. Este format din circuite electrice şi electronice. cuprinde si **nivelul echipamentelor** (care se afla la cel mai scazut nivel si in care utilizatorul poate vedea tranzistoarele). La nivelul digital se afla obiectele numite **porti(gates)** care sunt construite din tranzistoare, dar au intrari si iesiri digitale (semnale ce reprezinta "0" sau "1"). Portile pot fi combinate pentru a forma o memorie de un bit care poate stoca "0" sau "1", iar memoriile de 1 bit pot fi combinate pentru a forma grupuri de 16, 32 sau 64 biti, numite **registre**.

Nivelul 2 este nivelul microprogram care interpreteaza instructiunile nivelului 3 şi le execută în nivelul 1. Fiecare instrucţiune a nivelului 3 este executată de un microprogram. La **nivelul microarhitecturii** se afla registre, o memorie locala si un circuit numit **UAL (Arithmetic and Logic Unit)** care poate executa operatii simple aritmetice si logice. UAL este o unitate combinationala cu doua intrari si o iesire. Aici se poate distinge o cale de date, prin care circula datele, una de adrese si una de control si stare. La unele calculatoare operatiile caii de date sunt controlate prin *microprogram*, iar la altele direct prin hardware. Microprogramul este un interpretor al instructiunilor nivelului superior.

Nivelul 3 este nivelul setului de instrucţiuni al maşinii, instrucţiuni executate pe nivelul hard. numit si **ISA (Instruction Set Architecture)** reprezinta primul nivel programabil de catre utilizator. Aici gasim instructiunile masina furnizate de producator, care la rindul lor sunt interpretate sau executate prin hardware-ul de la nivelul inferior.

Nivelul 4 . Sistemul de operare este hibrid deoarece cuprinde atât instrucţiuni interpretate de nivelul patru cât şi instrucţiuni interpretate de nivelul trei. **sistemului de operare** este hibrid, adica se intilneso si instructiuni ISA si instructiuni noi ale sistemului de operare care sunt interpretate de sistemul de operare sau direct de microprogram.

Primele trei nivele nu sunt utilizate de programatorul obisnuit, ci de **programatorii de sistem** care realizeaza sau intretin interpretoarele sau translatoarele pentru masina virtuala. De la nivelul 4 in sus masina virtuala este folosita de programatorii de aplicatie. Nivelurile 2 si 3 sunt de obicei interpretate, iar de la 4 in sus translatate, desi exista si exceptii. Alta deosebire este ca limbajele primelor trei niveluri sunt numerice (i.e. Sunt alcatuite din siruri de numere). La nivelul 5 limbajul de programare se numeste **de asamblare** si reprezinta o scriere simbolica pentru nivelul inferior. Programul care interpreteaza limbajul de asmblare se numeste **asamblor**.

Nivelul 5 este nivelul limbajului de asamblare, destinat programatorilor de aplicaţii. Dacă primele niveluri erau interpretate, acesta este translatat de către un program numit *asamblor*.

Nivelul 6 este nivelul de limbaj inalt. Programele scrise în acest nivel sunt translatate către nivelele cinci şi şase de către programe specializate numite *compilatoare.* **limbaje de nivel inalt (HLL = High Level Language)** si sunt realizate pentru programatorul de aplicatii. Aici gasim: C, C++, Pascal, Prolog, Java, LISP, etc. Aceste limbaje sunt traduse prin translatoare numite **compilatoare** (exista si exceptii: Java este interpretat).

Nivelul 7 conţine limbaje destinate unor domenii foarte speciale cum ar fi proiectarea asistată, administraţia, grafica etc.