

INTRODUCERE

Obiectivele cursului:

- evidențierea rolului central al sistemului de operare în cadrul componentei software a unui sistem de calcul
- prezentarea evoluției sistemelor calcul și a celor de operare;
- prezentarea conceptelor care stau la baza sistemelor de operare, care vor fi dezvoltate în capitolele următoare;
- prezentarea structurii sistemelor de operare și a funcțiile lor;
- prezentarea componentelor hardware care realizează interfața cu sistemul de operare.

Locul sistemului de operare în cadrul unui sistem de calcul

- Componentele principale ale unui sistem de calcul (SC) sunt cea fizică(hardware) și cea logică(software). La rândul ei, componenta logică este formată din software-ul (programele) de aplicații și software-ul de sistem, așa cum este redat în figura următoare.

Aplicații financiar-bancare	Aplicații pentru rezervarea biletelor de călătorie	Jocuri pe calculator	}	Aplicatii
Compilatoare	Editoare	Interpreter de comenzi		
Sistemul de operare			}	Software de sistem
Limbajul cod mașină				
Mediul microprogramat			}	Hardware
Unități fizice				

- Programele de aplicații au scopul de a rezolva cu ajutorul calculatorului o problemă specifică dintr-un anumit domeniu de activitate (financiar-bancar, științific etc.). Programele de sistem oferă contextul (mediul) în care programatorii pot crea propriile programe de aplicații, care nu sunt disponibile la nivel fizic. Sistemul de operare (SO) face parte din componenta soft de sistem.
- Este dificil să se dea o definiție completă a ceea ce este un SO, în schimb, este mult mai ușor să se vadă ce face un SO. Astfel:
 - SO oferă facilitățile necesare unui programator pentru a-și construi propriile aplicații.
 - SO gestionează resursele fizice(memorie, discuri, imprimante etc) și cele logice (programe de sistem, fișiere, baze de date etc) ale sistemului de calcul, oferind posibilitatea ca utilizatorii să poată folosi în comun aceste resurse, pe baza unor anumite reguli, ceea ce conduce la scăderea cheltuielilor de prelucrare și la creșterea performanțelor sistemului de calcul.
 - SO oferă o interfață prin care aplicațiile utilizator și cele de sistem au acces la componenta hardware.

Evoluția sistemelor de operare

- **generația I:** dispun numai de echipamentul hard.
- **generația a II-a :**
 - componente hardware noi (cititorul de cartele, imprimanta, banda magnetică)
 - componente software (compilator pentru limbajul FORTRAN, programe specializate destinate asamblării, încărcării și înlănțuirii programelor, biblioteci software de funcții uzuale, rutine de interfață cu componentele fizice numite **drivere** etc.).
 - conceptul de **monitor**, ce reprezintă o formă rudimentară de SO. Lucrările (job-urile) erau o succesiune a fazelor (**editarea** textului sursă al programului, **compilarea**, **editarea legăturilor**, **depanarea**, **execuția** programului).

- **Monitorul rezident.** Intervenția umană între faze presupune o mare pierdere de timp și o utilizare ineficientă a echipamentului. Pentru a evita aceste neajunsuri, a apărut conceptul de **încărcare automată a job-urilor și a fazelor**.
- Pentru aceasta a fost creat un mic program, denumit **monitor rezident**, care realizează acest lucru. Programatorul trebuie să insereze, printre cartelele programului și ale datelor sale, unele cartele speciale numite **cartele de comandă** care se adresează acestui monitor rezident. Prin această regulă de diferențiere, s-a definit de fapt un **limbaj de control** al job-urilor. Prin intermediul lui, se comandă trecerea de la o fază la alta sau de la un job la altul.
- Monitorul rezident este programul permanent activ. Pentru desfășurarea înlanțuirilor de faze, este necesar ca operațiile de I/O să fie făcute de către monitor și nu de către programul utilizator, pentru a depista cartelele care îi sunt adresate, adică cele de comandă.
- Tipul de SO prezentat se numește **sistem serial**, deoarece job-urile se execută unul după altul(**prelucrare pe loturi - Batch processing**).
- O altă caracteristică a acestui sistem este **monoprogramarea**, adică CPU nu se ocupă de alt job până când nu-l termină pe cel curent. De asemenea, programului utilizator nu-i este permis să modifice zona de memorie a monitorului rezident și nici să comande oprirea întregului sistem.

- **Conversiile off-line** constau în transferarea conținutului cartelelor perforate pe benzi magnetice. Pentru aceasta se foloseau niște dispozitive de conversie relativ simple. Conținutul benzilor era utilizat de către SC în locul cartelelor perforate, unitatea de bandă fiind intrarea standard. Analog, rezultatele, sub forma unor linii de imprimantă, sunt depuse mai întâi într-un fișier pe un suport magnetic, iar la terminarea lucrului, conținutul fișierului este listat la imprimantă. De asemenea, prin această metodă, se reduce timpul de lenevire al CPU datorat vitezei de lucru mai mici a unor periferice.
- Odată cu operarea off-line a apărut și noțiunea de **independență față de dispozitiv** a programelor, adică aceeași operație de I/O să poată fi realizată de pe diferite dispozitive fizice. Acest lucru se realizează prin așa zisele **dispozitive logice de I/O**, ce reprezintă niște identificatori utilizați de programe, care sunt asociați prin intermediul SO dispozitivelor fizice.
- Utilizarea zonelor tampon (buffere) este un alt concept utilizat de SO în scopul utilizării eficiente a CPU. Informațiile de pe discuri care vor fi prelucrate de CPU sunt aduse în zone de memorie internă numite buffere.

- **Generația a III-a de calculatoare.**
- Conceptul de **multiprogramare** reprezintă modul de exploatare a unui SC cu un singur procesor central, care presupune existența simultană în memoria internă a mai multor programe, care se execută concurent.
- CPU este componenta hardware a calculatorului formată îndeosebi din componente electronice, pe când unitățile de I/O sunt formate din componente mecanice. Deci, pe de o parte CPU este mai scumpă, iar pe de altă parte ea lucrează mult mai rapid decât unitățile de I/O. Astfel, s-a pus problema utilizării cât mai eficientă a unității centrale a calculatorului.
- Multiprogramarea a rezolvat aceste probleme. Pe scurt, lucrul în multiprogramare se desfășoară astfel:
 - în fiecare moment CPU execută o instrucțiune a unui program(starea RUN);
 - restul programelor, fie că așteaptă apariția unui eveniment extern, de exemplu terminarea unui I/O etc. (starea WAIT), fie că sunt pregătite pentru a fi servite în orice moment de către CPU (starea READY).
 - Trecerea unui program din starea RUN în starea WAIT este realizată de către program, în momentul când trebuie să execute o instrucțiune de I/O.
 - Trecerea programelor din starea RUN în starea READY și invers este realizată de către SO pe baza unui algoritm de planificare, concept pe care îl vom detalia într-o secțiune următoare.

- **Canalul de intrare – ieșire.**
- Pentru a crește gradul de exploatare al CPU prin eliminarea timpilor de așteptare al acesteia, datorati diferenței între vitezele de execuție a echipamentele electronice ale CPU și celor mecanice ale dispozitivelor de I/O, a apărut, la nivelul tehnologiei hard existente la vremea respectivă **canalul de intrare – ieșire.**
- Acesta este un procesor specializat pe operații de I/O, care poate funcționa în paralel cu CPU. Pentru a fi lansat în execuție, canalul primește de la CPU o comandă de efectuare a unei operații de I/O. După lansare, cele două procesoare își continuă activitatea în paralel.
- Sincronizarea între CPU și canalul de intrare – ieșire se poate realiza prin **testarea periodica (pooling) a perifericelor de către CPU** sau **printr-o întrerupere lansată de către periferic.**
- O **întrerupere** este o rutină aflată la o adresă fixă de memorie lansată în urma apariției unui semnal hard, care perturbă execuția firească a instrucțiunilor programului în curs. Modul de lucru al întreruperilor va fi prezentat mai târziu. Printre altele, prin intermediul lor se realizează comunicarea dintre SC și dispozitivele lui periferice (discuri, imprimante etc.). Rutina de întrerupere determină perifericul care a emis întreruperea și, eventual dă perifericului o nouă comandă de I/O.⁸

- Relația dintre CPU și un dispozitiv periferic se realizează astfel:
- fiecare periferic conține o zonă tampon proprie, capabilă să păstreze o înregistrare (o linie de imprimantă, imaginea unei cartele).
- CPU, printr-o rutină de I/O, acționează dual, în funcție de operația efectuată. În cazul scrierii, pune din memorie informații în această zonă tampon, iar în cazul citirii preia informațiile și le depune în memorie.
- Dispozitivul periferic acționează și el dual, în funcție de operația care îi este comandată.
- În cazul scrierii, ia informațiile din zona tampon proprie și le depune pe suport. În cazul citirii, ia informațiile de pe suport și le depune în zona tampon proprie.
- În prezent, SC folosesc două tipuri de canale: **selector** și **multiplexor**. Canalul selector este destinat să realizeze schimbul dintre memorie și perifericele rapide (discuri), care lucrează, la un moment dat cu un singur periferic. Canalul multiplexor este capabil să lucreze simultan cu mai multe dispozitive periferice.

- **SPOOLING - Simultaneous Peripheral Operation On-Line.**
- Au apărut discurile magnetice, care au permis operarea on-line, simultan, cu mai multe periferice (SPOOLING). El s-a obținut prin îmbinarea utilizării zonelor tampon multiple cu conversiile off-line și cu multiprogramarea.
- SPOOLING funcționează astfel:
 - Se citesc de la un cititor, cartelele care compun un job.
 - Când jobul a fost citit complet, imaginile cartelelor sunt depuse într-un buffer pe disc. Un astfel de job spunem că se află în starea HOLD.
 - Când CPU este liber, el alege unul dintre joburile aflate în stare HOLD și-l lansează în execuție.
 - Liniile “tipărite” de jobul în execuție sunt depuse într-un buffer pe disc.
 - Atunci când jobul a fost executat complet, bufferul lui de ieșire pe disc devine disponibil pentru listare. Spunem că jobul se află în starea FINISH.
 - Conversia de ieșire se lansează automat, listând pe o imprimanta bufferul unui job aflat în starea FINISH.

- Operațiile de conversie se fac în paralel cu execuția în regim de multiprogramare. Dacă pentru execuție sunt disponibile mai multe zone ale memoriei interne(partiții), atunci toate au acces la cozile HOLD si FINISH, executandu-se în paralel mai multe joburi. De remarcat la tehnica SPOOLING este **simultaneitatea**. La un moment dat sistemul are în lucru trei categorii de joburi:
 - joburi în **curs de citire** de la unul sau mai multe cititoare; aceste joburi sunt trecute în coada HOLD.
 - joburi în **curs de execuție**; numărul lor poate fi cel mult egal cu numărul de partiții disponibile sp lucreze în multiprogramare;
 - joburi în curs de listare; numărul lor poate fi cel mult egal cu numărul de imprimante active conectate la sistem; aceste joburi sunt preluate din coada FINISH.
- Această tehnică se folosește și la calculatoarele actuale pentru listările la imprimantă.
- Dintre sistemele de operare specifice acestei generații de calculatoare, cele mai reprezentative sunt sistemul SİRIS pentru calculatoarele IRIS 50, care s-au produs și în țara noastră sub denumirea FELIX 256/512/1024 și sistemul OS/360 pentru calculatoarele IBM 360.

- **Generației a-IV-a de calculatoare.**
- **Sistemele interactive:** Un sistem interactiv permite comunicarea on-line dintre utilizator și sistem.
 - De regulă, utilizatorul are la dispoziție un terminal cu tastatură și ecran, prin care comunică cu sistemul.
 - În astfel de sisteme, utilizatorul dă comanda, așteaptă răspunsul și, în funcție de rezultatul furnizat de comanda precedentă, decide asupra noii comenzi.
 - El poate, astfel să experimenteze ușor și să vadă rezultatele imediat. SO pentru SC interactive conțin (cel puțin) câte un editor de texte pentru corectarea programelor sursă și (cel puțin) câte un depanator interactiv care poate asista alte programe în execuție.
 - Spre deosebire de sistemele seriale, sistemele interactive au un timp de răspuns rezonabil de ordinul secundelor, eventual al minutelor.
- **Time-sharing** (timp partajat) este o variantă a multiprogramării.
 - A apărut la generația a III-a de calculatoare (sistemul CTSS dezvoltat de MIT, sistemul MULTICS (MULTiplexed Information and Computing Service și mai ales sistemul UNIX).
 - A fost dezvoltat pe scară largă de generația a IV-a de calculatoare. Ele au fost proiectate pentru a permite mai multor utilizatori să fie conectați simultan la același sistem de calcul. Acest sisteme îmbină **interactivitatea** și **multiprogramarea**.

Terminal virtual.

- Fiecare utilizator stabilește o sesiune cu sistemul, prin intermediul unui **terminal virtual**, care este o simulare a hardware-ului calculatorului și este implementat de către SO.
- Sistemul comută rapid de la un program la altul, înregistrând comenzile solicitate de fiecare utilizator prin terminalul său.
- Deoarece o tranzacție a utilizatorului cu sistemul necesită un timp de lucru mic al CPU, rezultă că într-un timp scurt, fiecare utilizator este servit cel puțin o dată. În acest fel, fiecare utilizator are impresia că lucrează singur cu sistemul.
- Dacă sistemele seriale încearcă să optimizeze numărul de job-uri prelucrate pe unitatea de timp, sistemele timesharing realizează o servire echitabilă a mai multor utilizatori, aflați la diverse terminale.

Diferențiere între noțiunile de job și proces (program în execuție).

- În sistemele timesharing, la același moment un job poate executa două sau mai multe procese, pe când în sistemele seriale un job presupune un singur proces.
- Într-un sistem timesharing multiprogramat procesele se mai numesc și **task-uri** iar un astfel de calculator se mai numește și sistem **multitasking**.

Redirectarea si legarea in pipe. A apărut la calculatoarele din generația a III-a, dar a fost dezvoltat și utilizat îndeosebi de calculatoarele din generația a IV-a. Aplicarea lor presupune că fiecare program lansat de la un terminal are un **fișier standard de intrare** și un **fișier standard de ieșire**. De cele mai multe ori acestea coincid cu tastatura, respectiv cu terminalul de la care se fac lansările. Redirectarea intrărilor standard (intrare sau/și ieșire) permite utilizatorului să înlocuiască intrarea standard cu orice fișier, respective să se scrie rezultatele, afișate de obicei pe ecran într-un fișier oarecare, nou creat sau să fie adăugate la un fișier deja existent. Informațiile de redirectare sunt valabile din momentul lansării programului pentru care se cere acest lucru și până la terminarea lui. După terminarea programului se revine la fișierele standard implicite.

Dintre SO cele mai cunoscute ale generației a IV-a de calculatoare, amintim sistemele DOS, Windows, Unix pentru calculatoare personale și pentru rețele de calculatoare, sistemele RSX pentru minicalculatoare (calculatoare mainframe).

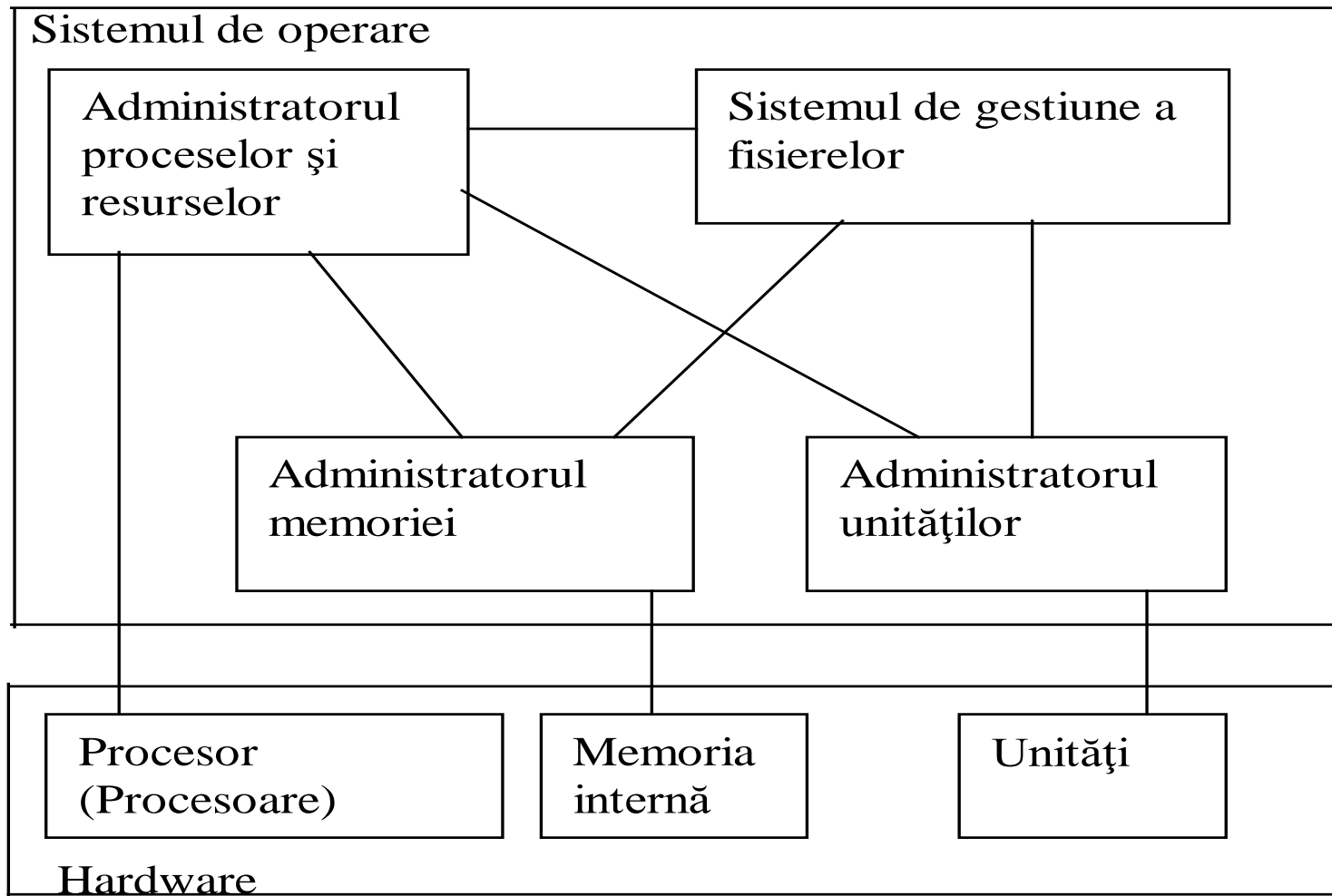
Conceptul general de proces

- Dacă sistemele de operare cu prelucrare în loturi executau lucrări(job-uri), sistemele de operare moderne bazate pe divizarea timpului execută task-uri. Aceste două concepte corespund termenului de proces. În cadrul sistemelor de operare, procesul reprezintă o entitate activă, un program în execuție ale cărei instrucțiuni sunt parcurse secvențial și executate de către unitatea centrală a calculatorului. Dacă prin program înțelegem o entitate statică, o codificare într-un anumit limbaj de programare a unui algoritm, sub forma unui fișier stocat pe un suport extern de informație, prin proces definim o entitate dinamică, încărcat în memoria internă a sistemului de calcul. Unul sau mai multe procese pot fi asociate unui program, dar ele sunt considerate entități distincte.
- Orice proces folosește mai multe resurse ale sistemului de calcul: **contorul de program** (PC – **P**rogram **C**ounter) este un registru al UC care conține adresa de memorie a următoarei instrucțiuni care urmează să fie executată; **stiva de program** conține date utilizate în execuție, parametri ai subprogramelor, adrese de retur și alte variabile locale; **secțiunea de date**, conține variabile globale; **timpul de lucru** al UC; **fișiere**; **dispozitive de I/O** etc. Aceste resurse sunt alocate procesului fie în momentul creerii lui, fie în timpul execuției lui

- În multe sisteme, procesul este unitatea de lucru. Astfel de sisteme constau dintr-o colecție de procese. Sistemul de operare, ca și componentă soft, este format din mai multe procese, care execută codul de sistem, pe când procesele utilizator execută programe scrise de aceștia. Toate aceste procese se execută concurrent la nivelul unui sistem de calcul, sau al unei rețele de calculatoare. În mod tradițional un proces conținea un singur fir de execuție; sistemele de operare moderne, privesc un proces ca fiind format din unul sau mai multe fire de execuție.
- Execuția proceselor este un mecanism combinat hard și soft. Componenta software care este implicată în administrarea proceselor și firelor de execuție este sistemul de operare, ale cărui sarcini sunt:
 - crearea și distrugerea proceselor și firelor de execuție;
 - planificarea proceselor;
 - sincronizarea proceselor;
 - comunicarea între procese;
 - manipularea interblocării proceselor.

Funcțiile de bază ale sistemelor de operare

- Aceste funcții cooperează între ele pentru a satisface cerințele utilizatorilor. În figura urm. sunt prezentate interacțiunile între modulele care realizează funcțiile SO, precum și între aceste module și componentele hard ale sistemului de calcul.



Administrarea proceselor și a resurselor

- Procesul reprezintă unitatea de bază a calculului, definită de un programator, iar resursele sunt elemente ale mediului de calcul necesare unui proces pentru a fi executat. Crearea, execuția și distrugerea proceselor, comunicarea între ele, împreună cu alocarea resurselor după anumite politici, sunt aspecte deosebit de importante care vor fi discutate în detaliu în cursurile următoare.

Gestiunea memoriei.

- SO alocă necesarul de memorie internă solicitat de procese și asigură protecția memoriei între procese. O parte este realizată prin hard, iar o parte prin soft. Această problemă va fi obiectul cursurilor următoare.

Gestiunea fișerelor

- **SO conține o colecție de module(SGF) prin intermediul cărora se asigură deschiderea, închiderea și accesul utilizatorului la fișierele rezidente pe diferite suporturi de informații. Componentă de bază a SO, este cel mai des invocată de către utilizator și de către operator.**

Administrarea unităților

- **Se referă la modul în care discurile, terminalele, imprimantele, procesoarele, memoria și alte componente hardware sunt alocate, protejate în timpul alocării și partajate după anumite reguli. Accesarea acestor componente se realizează prin intermediul driverelor.**

Arhitectura Von Neumann

- Sistemele de calcul se bazează pe conceptul de arhitectură Von Neumann, conform căruia partea de hardware este formată din:
 - ▶ **Unitatea centrală de calcul** (CPU - Central Processing Unit), compusă din **unitatea aritmetică și logică** (ALU-Arithmetical-Logical Unit) și **unitatea de control**.
 - ▶ **Unitatea de memorie primară sau executabilă sau internă.**
 - ▶ **Unități de I/O.**
- Toate unitățile sunt conectate folosind o **magistrală** (bus), care se împarte într-o **magistrală de date** și una **de adrese**.
- Pentru memorie și alte unități se poate adăuga o magistrală de I/O, care nu este folosită de CPU. O astfel de organizare, permite ca o unitate să poată citi/scrie informații din memorie, fără alocarea CPU. Fiecare magistrală poate fi gândită ca fiind formată din mai multe linii (fire) paralele, care pot păstra o cifră binară.

Unitatea aritmetică și logică

- Conține, pe lângă unitatea funcțională un număr de **registri generali** și de **stare**. Registrii generali sunt folosiți în efectuarea operațiilor aritmetice și logice, atât pentru memorarea operanzilor încărcăți din memorie, cât și a rezultatului operației, care apoi va fi salvat într-o locație de memorie.
- CPU extrage și execută instrucțiunile cod mașină ale procesului încărcat în memoria internă. În acest sens, CPU conține:
- ► o componentă care extrage o instrucțiune memorată într-o locație de memorie;
- ► o componentă care decodifică instrucțiunea;
- ► o componentă care se ocupă de execuția instrucțiunii, împreună cu alte componente ale SC.
- **Registrii** contor de program **PC (Program Counter)**, respectiv registrul instrucțiune **IR (Instruction Register)**, conțin adresa de memorie, respectiv o copie a instrucțiunii în curs de prelucrare.

Unitățile de I/O

- Unitățile de I/O sunt folosite pentru a plasa date în memoria primară și pentru a stoca cantități mari de date pentru o perioadă lungă de timp.
- Astfel, ele pot fi unități de stocare (unități bloc), cum ar fi discurile, respectiv unități caracter cum ar fi tastatura, mouse-ul, display-ul terminalului precum și unități de comunicație, cum ar fi portul serial conectat la un modem sau o interfață la rețea.
- Fiecare unitate folosește un controller de unitate pentru a o conecta la adresele calculatorului și la magistrala de date. Controller-ul oferă un set de componente fizice pe care instrucțiunile CPU le pot manipula pentru a efectua operații de I/O. Ca și construcție, controller-ele diferă, dar fiecare oferă aceeași interfață de bază.
- SO ascunde aceste detalii de funcționare ale controller-ilor, oferind programatorilor funcții abstracte pentru accesul la o unitate, scrierea/citirea de informații etc.
- Controller-ul de unitate furnizează o interfață folosită de către mediul microprogramat de la cel mai înalt nivel. Componenta SO care manipulează dispozitivele de I/O este formată din driverele de unitate.

Moduri de lucru ale procesorului

- **Setarea modului de lucru.** Procesoarele contemporane conțin un bit care definește modul de lucru al procesorului. Acest bit poate fi setat în modul **utilizator** sau **supervizor**. În modul supervizor, procesorul poate executa orice instrucțiune cod mașină, pe când în modul utilizator el poate executa numai o parte dintre aceste instrucțiuni. Instrucțiunile care pot fi executate numai în modul supervizor se numesc instrucțiuni privilegiate.
- De exemplu, astfel de instrucțiuni sunt cele de I/O.
- Un proces, dacă este executat în mod utilizator, el nu poate să-și execute propriile instrucțiuni de I/O. De aceea, aceste instrucțiuni sunt executate prin intermediul SO. Când un program de aplicație face o cerere către sistem, o instrucțiune cod mașină specială este apelată pentru a comuta procesorul în modul supervizor și începe să execute driverul unității respectiv.
- Corespunzător celor două moduri de lucru, memoria internă este împărțită în zona de memorie utilizator și zona de memorie supervizor. Dacă bitul mod de lucru este setat pe utilizator, atunci procesul respectiv are acces numai la zona cu același nume. Astfel, se realizează și protecția zonei de memorie supervizor.

Nucleul **SO**

- Este acea parte a SO care este executată în modul supervizor.
- Alte procese, legate de diverse aplicații ale utilizatorilor sau chiar aplicații soft de sistem sunt executate în mod utilizator.
- Execuția acestor instrucțiuni nu afectează securitatea sistemului.
- Când un proces dorește să execute anumite operații în mod supervizor, atunci se va face o **comutare** din modul utilizator, în cel supervizor.
- Acest lucru se realizează prin intermediul unei instrucțiuni **trap**, numită instrucțiune de apel al supervizorului, care setează bitul de mod de lucru și face un salt la o locație de memorie, care se află în spațiul de memorie protejat, locație care conține începutul unei proceduri sistem care va rezolva cererea procesului.
- Când execuția acestei rutine supervizor s-a terminat, SO va reseta bitul de mod de lucru din supervizor în utilizator.

Apelurile de sistem

- furnizează o interfață între un proces și sistemul de operare. Prin intermediul acestora, un program utilizator comunică cu SO și cere anumite servicii de la acesta.
- Apelurile de sistem pot fi împărțite în cinci categorii importante:
 - ▶ **controlul proceselor**(încărcarea, execuția, sfârșitul, abandonarea, setarea și obținerea atributelor, alocarea și eliberarea de memorie etc);
 - ▶ **manipularea fișierelor**(creere, ștergere, deschidere, închidere, citire, scriere, repoziționare, setarea și obținerea atributelor);
 - ▶ **manipularea unităților**(cerere și eliberare unitate, citire scriere și repoziționare, obținerea și setarea atributelor, atașarea/detașarea logică);
 - ▶ **întreținerea informațiilor**(obținerea și setarea timpului sau datei calendaristice sau a sistemului, obținerea de informații despre componentele fizice și logice ale sistemului și posibilitatea modificării lor);
 - ▶ **comunicații**(crearea și anularea unei conexiuni, transmiterea și primirea de mesaje, transferul informațiilor de stare, atașarea/detașarea logică a unităților la distanță).

Limbajele de asamblare și limbajele evaluate moderne conțin instrucțiuni (comenzi) prin care sunt lansate apeluri de sistem. După transmiterea parametrilor, este declanșată o instrucțiune trap, pentru a oferi controlul SO. Când a terminat de executat rutina respectivă, SO returnează un cod de stare într-un registru, care specifică terminarea normală sau anormală și execută o instrucțiune de revenire din instrucțiunea trap

Generarea, configurarea și lansarea în execuție a sist. de operare

- În cazul generațiilor de calculatoare mai vechi, sistemul de operare este destinat numai unui anumit tip de calculator.
- **Exemplu.** Sistemul SIRIS este destinat familiei de calculatoare FELIX C-256,512,1024, iar sistemul de operare RSX este destinat minicalculatoarelor compatibile PDP, printre care se numără și cele românești CORAL și INDEPENDENT.
- Sist. de op. moderne sunt realizate pentru a lucra pe o clasă de sisteme de calcul, care în multe cazuri diferă prin configurația lor. Sistemul trebuie să fie generat (sau configurat) pentru fiecare sistem de calcul în parte, proces care se numește **generarea sistemului**(SYSGEN). Programul SYSGEN citește dintr-un fișier sau cere informațiile care se referă la configurația hardware a sistemului respectiv. Prin acest proces sunt determinate următoarele informații:
 - ► Ce fel de CPU este folosită? Care dintre opțiunile(set de instrucțiuni cod mașină extins, instrucțiuni aritmetice în virgulă flotantă etc) sunt instalate? Dacă sistemul este multi-procesor, fiecare CPU va fi descrisă.
 - ► Câtă memorie este disponibilă?
 - ► Care unități sunt disponibile? Sistemul trebuie să cunoască adresele acestor unități, adresele întreruperilor corespunzătoare unităților, precum și niște caracteristici tehnice ale unităților.
 - ► Setarea unor parametri ai sistemului de operare, cum ar fi de exemplu, numărul și dimensiunea zonelor tampon utilizate, numărul maxim de procese care pot fi lansate la un moment dat, tipul algoritmului de planificare utilizat de CPU etc.

- După stabilirea acestor informații, sistemul de operare poate fi utilizat în mai multe moduri:
- administratorul de sistem poate modifica codul sursă al sistemului de operare, situație în care toate programele sursă vor fi recompilate și vor fi refăcute legăturile;
- selectarea unor module noi care fac parte dintr-o bibliotecă precompilată, fiind necesară numai refacerea legăturilor;
- selectarea unor opțiuni se face în timpul execuției, situație caracteristică sistemelor de operare moderne.
- Unul dintre serviciile de bază ale unui **SO** este acela de a se putea **autoîncărca** de pe disc în memoria internă și de a se autolansa în execuție. Această acțiune se desfășoară la fiecare punere sub tensiune a unui **SC**, precum și atunci când utilizatorul dorește să reîncarce **SO**, fiind cunoscută sub numele **încărcare sau lansare în execuție** a **SO**. Pentru lansare se utilizează un mecanism combinat **hard** și **soft**, numit **bootstrap**.

- Mecanismul **bootstrap** intră în lucru la apăsarea butonului de pornire <START> și cuprinde următoarele etape:
- Se citește din memoria ROM un număr de locații consecutive. Aceste locații conțin instrucțiunile de copiere ale programului care va încărca nucleul SO. Motivul pentru care acest program este stocat în memoria ROM, este că aceasta nu trebuie să fie inițializată și că procesul respectiv se află la o adresă fixă, de unde poate fi lansat în cazul punerii sub tensiune sau re-setării sistemului. De asemenea, deoarece ROM este o memorie “read-only”, ea nu poate fi virusată.
- Se lansează în execuție programul citit la pasul anterior, care citește de pe un disc un program mai complex, care încarcă nucleul sistemului de operare.
- 3. Se lansează în execuție programul citit la pasul anterior care va încărca nucleul sistemului de operare.
- **Observație.** Se pune firesc întrebarea, de ce programul care încarcă nucleul SO nu este citit din memoria ROM? Răspunsul este că instalarea unui alt sistem de operare, ar însemna modificarea acestui program, deci a conținutului memoriei ROM, care este o problemă destul de costisitoare.

Tipuri de sisteme de operare

- **Sisteme de operare pentru calculatoare mari.** Calculatoarele mari sunt caracterizate de capacitatea de a procesa volume foarte mari de informații. Ele lucrează cu dispozitive de I/O de capacitate mare de stocare și sunt orientate spre execuția mai multor sarcini în același timp. Pe astfel de calculatoare se pot executa: servere de Web, servere pentru site-uri dedicate comerțului electronic și servere care gestionează tranzacții între companii. Aceste sisteme de operare oferă trei tipuri de servicii: procesarea loturilor, procesarea tranzacțiilor și partajarea timpului.
- Un sistem de procesare în loturi prelucrează lucrări obișnuite, care nu impun intervenția utilizatorului.
- **Exemple** de astfel de prelucrări sunt: procesarea operațiilor uzuale din cadrul companiilor de asigurări, raportarea vânzărilor dintr-un lanț de magazine.
- Sistemele de procesare a tranzacțiilor gestionează pachete mari de cereri scurte.
- **Exemplu** de astfel de prelucrare este procesul de căutare/verificare într-o bază de date a unei companii aeriene care conține rezervările de bilete.
- La un astfel de sistem de calcul, sunt legate terminale de la care se introduc un număr mare de tranzacții diverse.
- **Exemplu** de astfel de sistem de operare este OS/370, care a fost realizat de firma IBM..

- **Sisteme de operare pentru calculatoare pe care se execută servere.** Aceste sisteme de calcul fac parte din rețele de calculatoare și rolul lor este de a servi cererile aplicațiilor lansate de diverși utilizatori. În funcție de complexitatea operațiilor pe care le execută, aceste calculatoare pot fi de la calculatoare personale la calculatoare de putere foarte mare.
- **Exemple de servere:** Servere care administrează utilizatorii unei rețele de calculatoare, servere de fișiere, servere de baze de date, servere de poștă electronică, servere care asigură servicii de Web etc.
- **Sisteme de operare pentru calculatoare multiprocesor.** Aceste sisteme de calcul încorporează mai multe unități de prelucrare (procesoare), în vederea creșterii capacității de prelucrare. Aceste calculatoare necesită sisteme de operare speciale, care să rezolve diverse probleme specifice legate de încărcarea echilibrată a procesoarelor, partajarea resurselor etc.

- **Sisteme de operare pentru calculatoare personale.** Aceste sisteme de operare sunt concepute pentru a oferi posibilități de lucru unui singur utilizator. Ele pot funcționa individual sau legate în rețea. În cadrul unei rețele, calculatoarele pe care se execută servere folosesc variante diferite de sisteme de operare față de cele ale sistemelor de calcul personale.
- **Exemplu.** Există patru versiuni ale sistemului Windows 2000: versiunea Profesional este destinată calculatoarelor individuale; celelalte trei versiuni (Server, Advanced Server și DataCenter Server) sunt destinate calculatoarelor server dintr-o rețea. Windows 2000 DataCenter Server este destinat serverilor multi-procesor.
- Versiunea XP propune două variante: Windows XP. Net Server, utilizat de calculatoarele server dintr-o rețea și Windows XP Profesional, destinat calculatoarelor individuale precum și calculatoarelor client legate în rețea.
- **Sisteme de operare pentru calculatoare în timp real.** Sistemele de calcul în timp real consideră timpul de răspuns ca o caracteristică fundamentală a bunei lor funcționări.