Universitatea “Politehnica” Bucureşti
Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei

Tema SO
Gestiunea de intrări şi ieşiri

Student: Paraschiv Radu
Grupa: 433A

2011-2012

CIRCUITE DE INTERFAŢĂ

Dispozitivele de I/O intercaţionează frecvent cu sistemele de operare. Acestea sunt alcătuite, în principal din doulă blocuri: un controler şi dispozitivul propiu-zis. Controlerul este un chip sau un set de chip-uri, montate pe o placă ce poate fi montată în calculator şi care comandă la nivel fizic dispozitivul. Controlerul acceptă coemnzi de la sistemul de operare, cum ar fi să citească date de la un dispozitiv şi să le prelucreze.

Prin intermediul acestui subsistem, utilizatorul introduce programele şi datele dorite în memoria calculatorului, pentru prelucrare, şi tot cu ajutorul său rezultatele se înregistrează, sau se afişează în exterior.

Legătura între UCP şi dispozitivele de I/O , care sunt in general dispozitivele periferice, se face prin intermediul unor circuite de interfaţă de I/O care asigură, din punct de vedere hardware, schimbul corect de date. Aceste circuite de interfaţă se cuplează la magistralele calculatorului şi ele sunt adresabile la nivel de registru-port de intrare-ieşire. Prin port înţelegem aici un loc ,în general un registru, cu adresă specifică, adresă care constituie o "poartă" prin care calculatorul realizează schimb de informaţie cu exteriorul. Registrele port pot fi adresate în spaţiul de adrese al memoriei sau pot fi adresate în spaţiu separat de cel de memorie . Fiecare port de intrare sau ieşire are o adresă specifică. Ca urmare adresele porturilor de intrare-ieşire pot fi tratate în două moduri:

1. ca porturi cu adrese distincte faţă de adresele de memorie
2. adresele porturilor sunt înglobate în spaţiul de adrese al memoriei principale.

Această organizare a adreselor face ca porturile să fie selectate prin aceleaşi semnale de adresă şi control ca şi memoria. În lucrul cu porturile de I/O se vor utiliza instrucţiunile de transfer specifice memoriei.

Este important de observat că, exceptând procesorul şi memoria principală, toate circuitele conectate la magistrala de date a sistemului, deci care partajează această resursă a sistemului, sunt privite de procesorul central ca porturi de intrare / ieşire, chiar dacă funcţiile acestora nu se limitează strict la aceste operaţii. Este o manieră unitară de lucru a UCP cu orice circuit de interfaţă, circuit care poate include mai multe porturi de intrare / ieşire , indiferent de funcţiile specifice ale acestuia. Aceasta face ca din punctul de vedere al UCP să se lucreze cu porturile adresabile într-un mod asemănător cu cel folosit la adresarea locaţiilor de memorie.

Unele circuite de interfaţă sunt incluse în circuitele specializate de control ale periferiicelor. De exemplu, controllerul unităţii de disc, controlează operaţiile fizice efectuate de discul magnetic. Pentru ca să se poată face

cuplarea la calculator controlerul trebuie să respecte nişte specificaţii standardizate de interfaţare.

Funcţia principală a circuitelor interfaţă de I/O este de a rezolva diferenţele funcţionale, electrice şi informaţionale dintre calculator şi periferic. Circuitele controler au în plus şi sarcina specifică de control a unui anumit periferic. Principalele diferenţe între UCP şi periferie, care impun folosirea circuitelor de interfaţă, constau în următoarele:

· perifericele sunt dispozitive a căror funcţionare se bazează pe diferite tehnologii. De aceea trebuie să existe dispozitive de conversie a valorilor semnalului, pentru o adaptare din punct de vedere electric cu calculatorul;

· ritmul de transfer al datelor este mult mai scăzut la periferice faţă de UCP. Pentru transferul de date între periferice şi UCP sau memorie trebuie deci să existe mecanisme de sincronizare.

· codurile şi formatele datelor în echipamentele periferice pot fi diferite faţă de codurile şi formatele folosite în UCP şi memorie.

· există o varietate de periferice, cu moduri de funcţionare diferite şi de aceea acestea trebuie controlate adecvat, pentru a nu perturba celelalte periferice conectate la UCP.

Circuitul de interfaţă rezolvă toate aceste diferenţe, fiind inclus între UCP şi periferice, pentru a superviza şi sincroniza toate transferurile de intrare / ieşire. În plus fiecare periferic poate avea propriul controller care supervizeaza funcţionarea corectă, specifică a respectivului periferic. Interfaţarea a două dispozitive fizice constă în a proiecta circuitele de interconectare fizică dintre aceste două dispozitive.

Structura unui sistem de operare

Un sistem de operare reprezintă un set de proceduri (subprograme sau rutine) manuale şi automate, care joacă rolul de interfaţă între programele utilizator şi calculator, având ca obiectiv optimizarea utilizării resurselor fizice şi logice ale sistemului de calcul. Resursele fizice sunt reprezentate de unitatea centrală de prelucrare, memorii şi dispozitivele periferice. Cele logice sunt reprezentate de sistemul de gestiune al fişierelor pe suporţii de memorii externe şi de gestiunea şi execuţia proceselor.

Componenta sistemului de operare care defineşte modul de interacţiune dintre sistemul de operare şi utilizator poartă numele de interfaţă.

Nucleul unui sistem de operare conţine acele module software care efectuează operaţiile primare necesare gestiunii resurselor fizice şi logice. În general, aceste module sunt:

· administratorul de fişiere - stochează informaţii referitoare la toate fişierele memorate pe un suport de stocare de masă (poziţiile fişierelor, spaţiul ocupat, utilizatorii cu drept de acces la ele) şi ce porţiune din memoria de masă este disponibilă pentru stocarea de noi fişiere sau extinderea celor existente. Acest modul administrează toată componenta de organizare logica a informaţiei pe suporţii de memorie externă ;
· administratorul de memorie - are sarcina de a coordona utilizarea memoriei interne a unui sistem de calcul ;
· administratorul de procese - cu cele doua componente ale sale:
  · secvenţiatorul -are rolul de gestiune a proceselor active;
  · executorul - are rolul de coordonare a derulării proceselor active;
· driver-ele pentru dispozitivele periferice - comunică cu controlerele pentru efectuarea operaţiilor de către dispozitivele periferice ale sistemului de calcul.

Fiecare driver este proiectat in mod specific, pentru un anumit tip de placă de extensie şi traduce cererile formulate în termeni generali, de alte componente ale sistemului de operare, într-o secvenţă de instrucţiuni specifice dispozitivului periferic ataşat. De exemplu, pentru un disc, un driver poate converti o cerere generală de scriere a unei portiuni dintr-un fişier, în instrucţiuni care se referă la pistele şi sectoarele discului, transmiţându-le apoi controler-ului corespunzător discului;

Sistemele de operare sunt rezidente pe un suport de memorie externe, atunci când calculatorul nu este alimentat cu tensiune. În exploatare curentă,

majoritatea componentelor sale trebuie să fie în memoria internă. Procesul de aducerea în memoria internă a componentelor unui sistem de operare poartă numele de încarcare , sau booting, a sistemului de operare şi se realizează imediat dupa ce calculatorul este alimentat cu tensiune.

Unitatea Centrală de Prelucrare este astfel realizată încât, la pornirea calculatorului, registrul PC conţine întotdeauna adresa unui program de încarcare care se află inscripţionat într-o zonă de memorie ROM, din memoria internă a calculatorului. Acest program, prin lansarea sa în execuţie, citeşte din blocul de încarcare de pe discul sistem adresa de pe disc la care se găsesc componentele sistemului de operare, acestea urmând să fie încărcate în memoria internă.

Procese gestionate de sistemul de operare

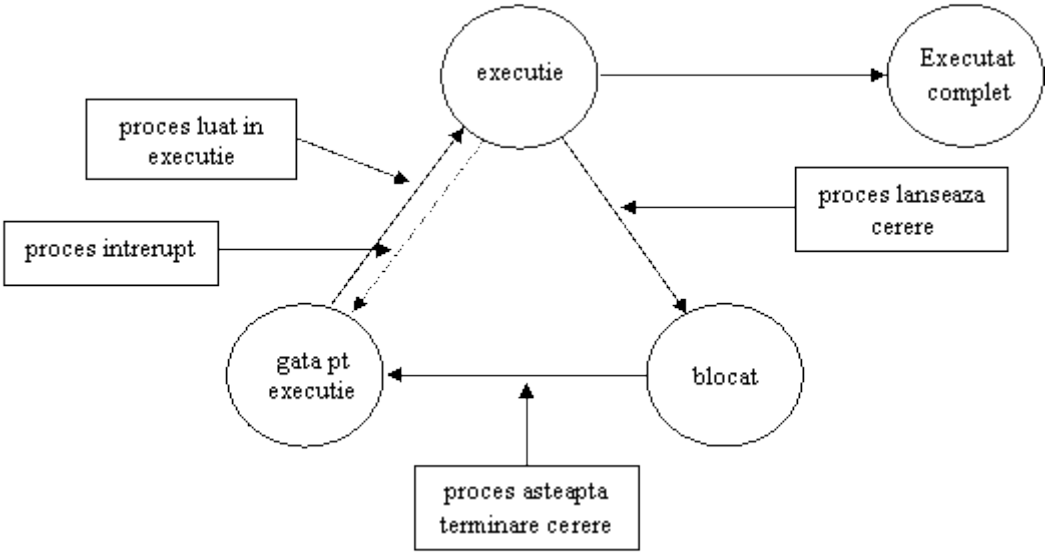
Pentru înţelegerea funcţionarii unui sistem de operare este importantă înţelegrea diferenţei între un program şi un proces. În timp ce un program reprezintă o secvenţă statică de instrucţiuni, un proces reprezintă starea la un moment dat a unui program aflat in execuţie. De exemplu, într-un sistem multiutilizator cu partajarea timpului, doi utilizatori pot să editeze simultan documente diferite. Ambele activităţi pot utiliza acelaşi program, dar fiecare dintre ele va constitui un proces distinct, având propriile seturi de date.

Atributele unui proces sunt constituite de coordonatele ce caracterizează starea unui program la un moment dat. Acestea sunt reprezentate de:

- registrul special PC - caracterizează starea de evoluţie a unui proces ;
- conţinutul registrilor generali ;
- adresa zonei de date specifice procesului, in memoria internă ;
- adresa zonei de cod specifice procesului, in memoria internă ;
- prioritatea procesului ;
- starea curentă a procesului, etc.

Procesele sunt gestionate de administratorul de procese prin cele două componente ale sale secvenţiatorul şi executorul. Secvenţiatorul gestionează lista cu toate procesele, în această listă fiind evidenţiate şi atributele specifice fiecărui proces activ. Un proces activ este declansat de lansarea in execuţie a unui program. Executorul asigură execuţia proceselor active şi asigură trecerea de la un proces activ la un altul, prin salvarea contextului specific procesului întrerupt / blocat, şi respectiv restaurarea contextului specific procesului reluat în execuţie din întrerupere / blocare. Contextul unui proces este reprezentat de atributele sale care nu sunt constante .

Un proces poate avea patru stări distincte:
· în execuţie - procesul se execută;
· blocat - procesul lansează o cerere, de exemplu o operaţie de intrare / ieşire şi se blochează pană la terminarea ei;
· gata de execuţie - procesul este gata pentru a fi luat sau reluat în execuţie. Reluarea survine dupa o blocare sau întrerupere a sa;
· executat complet - procesul este executat în întregime, urmând a fi scos din evidenţele administratorului de procese;



În figura de mai sus sunt prezentate cele patru stări distincte ale unui proces şi acţiunile ce determină trecerea unui proces dintr-o stare în alta.

Există două modalităţi de planificare a proceselor:
· planificare secvenţială - acest mod determină de procese secvenţiale. În această situaţie, în momentul blocării unui proces planificat, CPU-ul care-l realizează este nefolosit până la deblocarea procesului, ceea ce implică o utilizare ineficientă a CPU. Utilizarea planificării secvenţiale a programelor la nivelul CPU s-a realizat în monoprogramare;
· planificare concurentă - acest mod determină apariţia de procese paralele , având ca efect creşterea substanţială a coeficientului de utilizare al CPU. Utilizarea proceselor concurente la nivelul CPU a determinat apariţia multiprogramării.

Având ca obiect principal optimizarea randamentului CPU, multiprogramarea nu ameliorează în orice situaţie serviciile aduse utilizatorilor.

Astfel, dacă un program necesită un număr redus de operaţii de intrare / ieşire, din momentul în care acesta caştigă CPU , procesul intră în starea de execuţie, celelalte procese active intră în starea de blocare până la terminarea procesului aflat în stare activă. La nivelul utilizatorilor acest mod de lucru lasă impresia că numai unul dintre ei este "servit" de calculator. Acest neajuns se înlătură prin tehnica partajării timpului (time-sharing): fiecare proces poate fi în stare de "execuţie" cel mult o cuantă de timp, după care este întrerupt şi este trecut în stare de "gata de execuţie", alt proces din listă de procese active trecând în starea de "execuţie". Cuantele de timp acordate proceselor pentru starea de "execuţie" sunt egale, in condiţiile în care procesele au aceeaşi prioritate.

Procese concurente. Gestiunea întreruperilor

În cazul planificării concurente a proceselor, este esenţial de înţeles modul de funcţionare în întreruperi al unui calculator şi implicit al CPU, care coordonează întraga activitate.

În funcţionarea unui calculator pot apăre evenimente de genul terminarea cuantei de timp pentru un proces care a fost în starea de "execuţie", lansarea unei cereri de către un proces şi intrarea sa în "blocare" sau terminarea executării cererii lansate de un proces, care trebuie să se materializeze în trecerea procesului din starea de "blocare", în cea de "gata de execuţie". Pe lângă aceste evenimente mai sunt evident şi altele , cum ar fi : trecerea unui proces in stare "execuţie completă", apariţia unui nou proces, etc. În concluzie, se poate spune că întreruperile sunt evenimente din activitatea unui calculator, care trebuie cunoscute de unitatea de comandă din CPU.

Întreruperile sunt administrate de CPU care conţine, pe lângă cele două componente prezentate mai sus şi alte componente printre care unitatea de control întreruperi (UCI). Această componentă conţine circuitele necesare pentru gestiunea şi memorarea temporară a semnalelor de întreruperi generate la producerea unor evenimente de genul celor menţionate anterior. Fiecărui tip de întrerupere, îi corespunde o rutină de tratare. Existând mai multe tipuri de întreruperi, rezultă ca un calculator conţine într-o zonă a memoriei sale interne un setul de rutine de tratare a întreruperilor.

În raport cu procesul aflat in stare de "execuţie", întreruperile se clasifică în modul următor:

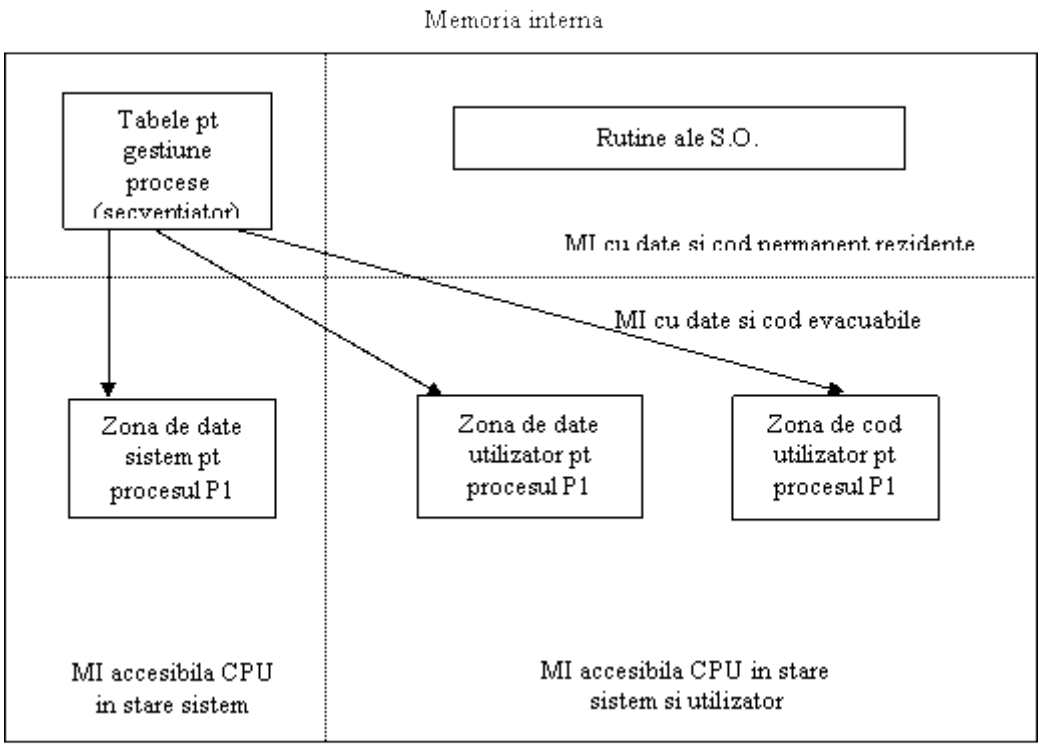
· întreruperi externe - apariţia lor este independentă de procesul aflat în stare de "execuţie" şi sunt luate in evidenţa de UCI. Aceste întreruperi pot fi

mascabile sau nemascabile, în cazul unei întreruperi mascabile, se poate forţa funcţie de context, să nu se ţină cont de ea. Întreruperi nemascabile sunt de obicei cele care semnalează probleme in hardware-ul sistemului de calcul;
· întreruperi interne - apariţia lor este determinată de procesul aflat in stare de "execuţie" şi nu sunt luate in evidenţă de UCI, ele aplicandu-se direct de către unitatea de control din CPU. Ele pot fi generate automat de UC în condiţii predefinite sau pot fi generate de programator prin instrucţiuni speciale de întreruperi.

Prelucrarea unei întreruperi inseamnă suspendarea execuţiei procesului aflat in stare de "execuţie" şi salvarea contextului acestuia, în vederea reluării execuţiei sale.

Trebuie avut in vedere că întreruperea procesului nu înseamnă întreruperea ciclului maşină în care se afla procesul la momentul respectiv. Întreruperea se va produce numai după terminarea ciclului extragere-decodificare-execuţie aflat în derulare. Dupa realizarea întreruperii şi salvarea contextului procesului, se execută rutina corespunzătoare întreruperii respective şi în final, daca este cazul se reia execuţia procesului întrerupt, reluarea fiind precedată de restaurarea contextului coreszător acestuia. În prelucrarea întreruperilor este implicat administratorul de procese prin executor. Activitatea acestuia determină şi actualizarea tabelelor de gestiune procese ale secvenţiatorului.

Urmatorul desen prezintă componentele unui proces aflate la un moment dat in memoria internă a calculatorului. În acest scop, mai sunt necesare cateva noţiuni. CPU din punct de vedere software poate avea două stări distincte:
· starea sistem - caz în care procesul aflat în execuţie are acces la bază de date a sistemului de operare pentru consultare şi modificare;
· starea utilizator - caz în care accesul anterior menţionat nu este permis.



Probleme cauzate întreruperi

Problemele sunt cauzate de faptul că procesele folosesc aceleaşi resurse fizice şi logice, pentru care concurează în pentru şi le aloca.

O astfel de problemă poate fi întalnită când o resursă este solicitată de mai multe procese. De exemplu, mai multe procese doresc să scrie în acelaşi fişier sau mai multe procese solicită scrierea la imprimantă. O solutie pentru rezolvarea problemei este excludere mutuală între procese - atât timp cat o resursa este alocata unui proces, aceasta nu mai poate fi alocata altuia. Pentru implementarea soluţiei se poate utiliza un indicator (un bit sau o variabilă booleană) care se menţine setat cât resursa e ocupată. Implementarea unui astfel de indicator trebuie facuta cu atentie, intrucat se poate ajunge usor in situaţii în care excluderea mutuală să nu funcţioneze. De exemplu, în cazul unei implementari extrem de simpliste a acestei soluţii, se poate ajunge la situaţia în care un proces P1 solicitant al resursei, să fie intrerupt exact dupa momentul în care acesta a constatat că resursa este disponibilă. Intră în execuţie procesul P2 care solicită şi el aceaşi resursă, o gaseşte nealocată şi şi-o alocă. În momentul în care procesul P1 reintră în execuţia îşi alocă şi el resursa, intrucât tocmai

înainte de întrerupere constatatse că aceasta nu este alocată. Rezultă, că P1 şi P2 nu s-au exclus reciproc.

Există mai multe tehnici utilizate la implementarea corectă a indicatorului pentru excluderea mutuală a alocarii unei resurse:

· Existenţa unei instructiuni complexe de testare-setare indicator în setul de instrucţiuni cod maşină;

· Existenţa unor instructiuni de invalidare/validare întreruperi care să fie utilizate cât durează zona critică a codului (zona de cod critică este în acest caz de la testarea alocarii resursei si până la instrucţiunea de alocare a acesteia);

· O implementare mai complexă a acestui indicator este mecanismul de semafor. Un semafor este un ansamblu format dintr-o variabilă "s", cu valori întregi, o lista de procese Q(s) si două operaţii primitive P si V. V(s) măreste valoarea argumentului "s" cu o unitate, iar P(s) micşorează valoarea argumentului "s" cu o unitate, numai dacă valoarea argumentului rămâne pozitivă, altfel operaţia fiind intarziată până când condiţia este îndeplinită:

P(s) => s = s - 1

s < 0 => procesul se blochează şi este introdus în Q(s);

s >= 0 => procesul se execută;

V(s) => s = s + 1

s < 0 => alege un proces din Q(s) şi îl activează;

s >= 0 => nu face nimic

Pentru a înţelege mai bine titulatura mecanismului, primitivă P(s) poate fi echivalată cu sosirea la semafor. Dacă acesta este roşu (s < 0), se aşteaptă culoarea verde în coada Q(s), daca se poate trece. Primitiva V(s) este echivalentă cu punerea pe culoarea verde a semaforului. Daca există procese în coada Q(s), acestea pot trecem însă numai câte unul.

Chiar daca excluderea mutuală a resurselor este corect implementată, în condiţiile în care mai multe resurse sunt solicitate de mai multe procese la momente diferite de timp, pot apăre interblocări fara ieşire (aşa-numitul deadlock). De exemplu, procesul P1 are alocată resursa R1 de la momentul T1 şi are nevoie de ea până la momentul T5. Similar, procesul P2 are alocată resursa R2 de la momentul T2 şi are nevoie de ea până la momentul T6. Pe parcursul de rularii proceselor, presupunem că procesul P1 are nevoie de resursa R2 la momentul T3, iar procesul P2 are nevoie de resursa R1 la momentul T4. Rezultă o situaţie fără ieşire, ambele procese aşteptând eliberarea resursei necesare, care este alocată celuilalt proces.

Situaţia de interblocare a proceselor poate apăre în una din urmatoarele condiţii:

· a - competiţia este pentru resurse ce nu pot fi partajate;

· b - resursele sunt solicitate la momente diferite de timp de către procese;

· c - o resursă odată alocată nu se poate forţa dezalocarea sa;

Situaţia de interblocare se evită prin eliminarea celor trei condiţii anterior mentionate:

· Eliminarea condiţiei c: nu se fac eforturi pentru a se evita deadlock-ul, ci doar pentru a se constata. La constatare se omoară procesele cel mai puţin evoluate;

· Eliminare condiţii a si b - evitare interblocare:

· o modalitate este reprezentata de alocarea odată a tuturor resurselor de către un proces (această metodă poate crea uneori complicaţii in programarea procesului sau chiar nu se poate folosi);

· se caută o cale de transformare a resurselor nepartajabile în resurse partajabile. Un exemplu in acest sens îl constituie tehnica spooling folosită la scrierea la imprimantă. De exemplu, când imprimanta este alocată unui proces care o foloseşte efectiv, celelalte procese care cer ataşarea resursei în acest interval, vor lista pe o imprimantă virtuală, reprezentată practic de un fişier pe disc. În momentul în care imprimanta devine disponibilă fişerele conţinând liste, sunt scoase la imprimantă.

Bibliografie:

http://tldp.org/LDP/khg/HyperNews/get/devices/whatis.html
http://luky.xhost.ro/sistem_i_e.htm
http://www.didactic.ro/materiale-didactice/fisaciorchine-sistemul-de-intrareiesire

Gorgan D., Sebestyen-Pal Gh., *Computer Design*, Editura albastra, Cluj-Napoca,. ISBN 973-650-123-X, 2005.

Pop V. – *Analiza şi sinteza dispozitivelor numerice*. Curs litografiat, Institutul Politehnic Timişoara, 1986, vol. I şi II.