VI. Sistemul de întreruperi

Ce este o întrerupere?

- procesorul poate suspenda execuţia programului curent
- scop - tratarea unor situaţii neprevăzute
- după tratare se reia programul întrerupt
- scopul iniţial - comunicarea cu perifericele
- procesorul nu "aşteaptă" perifericele
  - acestea îl solicită când este necesar

Întreruperi hardware

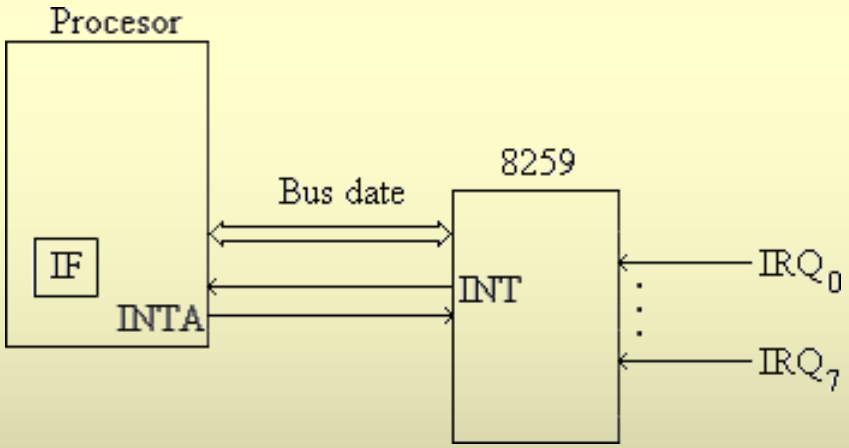
- dezactivabile (*maskable*)
  - procesorul poate refuza tratarea lor
  - depinde de valoarea bistabilului IF (*Interrupt Flag*): 1 - acceptare, 0 - refuz
  - IF poate fi modificat prin program
- nedezactivabile (*non-maskable*)
  - procesorul le tratează întotdeauna

Controllerul de întreruperi (1)

- circuit specializat
- preia cererile de întrerupere de la periferice
- le trimite procesorului
- rezolvă conflictele (mai multe cereri simultane) - arbitrare
  - fiecare periferic are o anumită prioritate

Controllerul de întreruperi (2)

- iniţial - Intel 8259
  - se pot folosi mai multe (cascadare)



- astăzi - integrat în chipset

Tratarea întreruperilor - etape (1)

- dispozitivul periferic generează o cerere de întrerupere pe linia sa IRQ_i
- controllerul trimite un semnal pe linia INT
- procesorul verifică valoarea bistabilului IF
  - numai pentru întreruperi dezactivabile
  - dacă este 0 - refuză cererea; stop
  - dacă este 1 - răspunde cu un semnal INTA

Tratarea întreruperilor - etape (2)

- se întrerupe execuţia programului curent
- se salvează în stivă regiştrii (inclusiv PC)
- se şterge bistabilul IF
  - se blochează execuţia altei întreruperi în timpul execuţiei programului pentru întreruperea în curs
  - poate fi repus pe 1 prin program

Tratarea întreruperilor - etape (3)

- identificare periferic (sursa cererii)
  - controllerul depune pe magistrala de date un octet *type*
  - indică perifericul care a făcut cererea
  - maximum 2^8=256 surse de întrerupere
  - fiecare sursă are rutina proprie de tratare
  - periferice diferite - tratări diferite

Tratarea întreruperilor - etape (4)

- determinarea adresei rutinei de tratare
  - adresa fizică 0 - tabelul vectorilor de întreruperi
  - conţine adresele tuturor rutinelor de tratare
  - dimensiune: 256 adrese × 4 octeţi = 1 Ko
  - octet type = n → adresa rutinei de tratare: la adresa n × 4

Tratarea întreruperilor - etape (5)

- salt la rutina de tratare a întreruperii
- execuţia rutinei de tratare
- revenire în programul întrerupt
  - restabilire valoare bistabil IF
  - restabilire valoare regiştri (din stivă)
  - reluarea execuţiei programului de unde a fost întrerupt

Extindere

- sistem puternic şi flexibil
- poate fi extins - utilizare mai largă
- programele trebuie întrerupte şi în alte situaţii (nu doar pentru comunicarea cu perifericele)
- util mai ales pentru sistemul de operare

Tipuri de întreruperi

- *hardware* - generate de periferice
- *excepţii (traps)* - generate de procesor
  - semnalează o situaţie anormală
  - exemplu: împărţire la 0
- *software* - generate de programe
  - folosite pentru a solicita anumite servicii sistemului de operare

VII. Sistemul de operare

Rolul sistemului de operare (1)

- program cu rol de gestiune a sistemului de calcul
- face legătura între hardware şi aplicaţii
- pune la dispoziţia aplicaţiilor diferite servicii
- supraveghează buna funcţionare a aplicaţiilor
  - poate interveni în cazurile de eroare

Rolul sistemului de operare (2)

- pentru a-şi îndeplini sarcinile, are nevoie de suport hardware
  - cel mai important - sistemul de întreruperi
- componente principale
  - nucleu (*kernel*)
  - drivere

VII.1. Nucleul sistemului de operare

Nucleul sistemului de operare

- în mare măsură independent de structura hardware pe care rulează
- "creierul" sistemului de operare
- gestiunea resurselor calculatorului - hardware şi software
  - funcţionare corectă
  - alocare echitabilă între aplicaţii

Moduri de lucru ale procesorului

- utilizator (*user mode*)
  - restricţionat
  - accesul la memorie - numai anumite zone
  - accesul la periferice - interzis
- nucleu (*kernel mode*)
  - fără restricţii

Modul de rulare al programelor

- sistemul de operare - în mod nucleu
  - poate efectua orice operaţie
- aplicaţiile - în mod utilizator
  - nu pot realiza anumite acţiuni
  - apelează la nucleu

Trecerea între cele două moduri

- prin sistemul de întreruperi
- utilizator → nucleu
  - apel întrerupere software
  - generare excepţie (eroare)
- nucleu → utilizator
  - revenire din rutina de tratare a întreruperii

Consecinţe

- codul unei aplicaţii nu poate rula în modul nucleu
- avantaj: erorile unei aplicaţii nu afectează alte programe
  - aplicaţii
  - sistemul de operare
- dezavantaj: pierdere de performanţă

Structura nucleului

- nu este un program unitar
- colecţie de rutine care cooperează
- funcţii principale
  - gestiunea proceselor
  - gestiunea memoriei
  - gestiunea sistemelor de fişiere
  - comunicarea între procese

VII.2. Apeluri sistem

Apeluri sistem (*system calls*)

- cereri adresate nucleului de către aplicaţii
- acţiuni pe care aplicaţiile nu le pot executa singure
  - numai în modul nucleu al procesorului
  - motiv - siguranţa sistemului
- realizate prin întreruperi software
- similar apelurilor de funcţii

Etapele unui apel sistem (1)

1. programul depune parametrii apelului sistem într-o anumită zonă de memorie
2. se generează o întrerupere software
   - procesorul trece în modul nucleu
3. se identifică serviciul cerut şi se apelează rutina corespunzătoare

Etapele unui apel sistem (2)

4. rutina preia parametrii apelului şi îi verifică
   – dacă sunt erori - apelul eşuează
5. dacă nu sunt erori - realizează acţiunea cerută
6. terminarea rutinei - rezultatele obţinute sunt depuse într-o zonă de memorie accesibilă aplicaţiei care a făcut apelul

Etapele unui apel sistem (3)

7. procesorul revine în mod utilizator
8. se reia execuţia programului din punctul în care a fost întrerupt
   - se utilizează informaţiile memorate în acest scop la apariţia întreruperii
9. programul poate prelua rezultatele apelului din zona în care au fost depuse

Apeluri sistem - concluzii

- comunicare între aplicaţie şi nucleu
  - depunere parametri
  - preluare rezultate
- acţiunile critice sunt realizate într-un mod sigur
- mari consumatoare de timp
- apeluri cât mai rare - lucru cu buffere

Cum folosim bufferele

Exemplu - funcţia *printf*

- formatează textul, apoi îl trimite spre ecran
- nu are acces direct la hardware
  - se foloseşte de un apel sistem
  - *write* (în Linux)
- de fapt, *printf* depune textul formatat într-o zonă proprie de memorie (buffer)
  - doar când bufferul e plin se face un apel sistem

VII.3. Drivere

Ce sunt driverele?

- module de program care gestionează comunicarea cu perifericele
  - specializate - câte un driver pentru fiecare periferic
- parte a sistemului de operare
  - acces direct la hardware
  - se execută în modul nucleu al procesorului

Utilizare

- nu fac parte din nucleu
  - dar se află sub comanda nucleului
- folosite de rutinele de tratare ale întreruperilor hardware
- înlocuire periferic → înlocuire driver
  - organizare modulară
  - nu trebuie reinstalat tot sistemul de operare

VII.4. Gestiunea proceselor

Procese (1)

- se pot lansa în execuţie mai multe programe în acelaşi timp (*multitasking*)
- paralelismul nu este real
  - doar dacă sistemul are mai multe procesoare
  - altfel - concurenţă
- un program se poate împărţi în mai multe secvenţe de instrucţiuni - *procese*
  - se pot executa paralel sau concurent

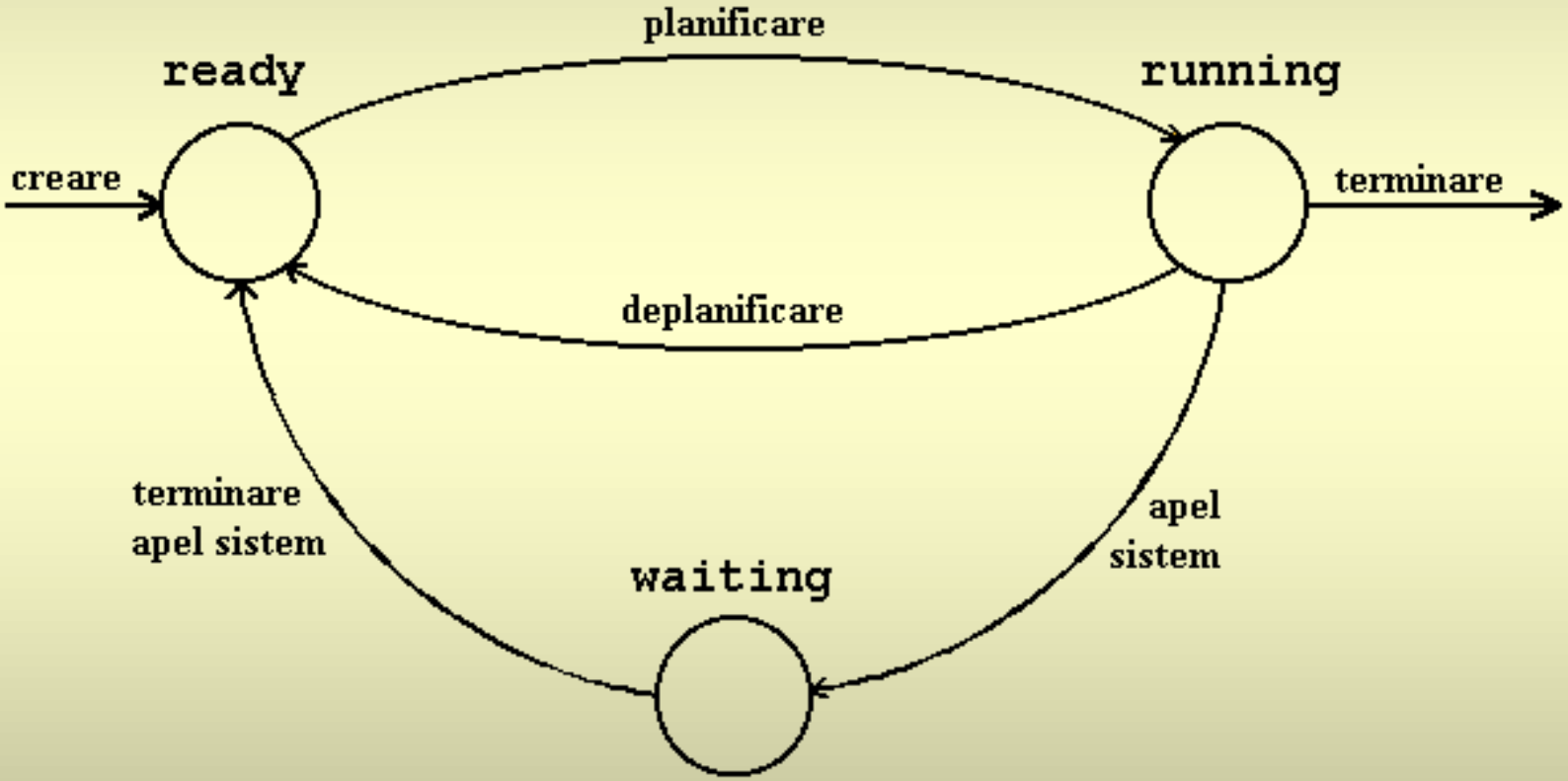
Procese (2)

- sistemul de operare lucrează cu procese
  - nu cu programe
- la lansare, un program constă dintr-un singur proces
  - poate crea alte procese
  - care pot crea alte procese ş.a.m.d.
- un procesor poate executa la un moment dat instrucţiunile unui singur proces

Stările unui proces (1)

- în execuţie (*running*)
  - instrucţiunile sale sunt executate de procesor
- gata de execuţie (*ready*)
  - aşteaptă să fie executat de procesor
- în aşteptare (*waiting*)
  - aşteaptă terminarea unui apel sistem
  - nu concurează momentan pentru planificarea la procesor

Stările unui proces (2)



Stările unui proces (3)

- procesul aflat în execuţie părăseşte această stare
  - la terminarea sa
    - normală sau în urma unei erori
  - la efectuarea unui apel sistem (→ *waiting*)
  - când instrucţiunile sale au fost executate un timp suficient de lung şi este rândul altui proces să fie executat (deplanificare)

Forme de multitasking

- non-preemptiv
  - nu permite deplanificarea unui proces
  - un proces poate fi scos din execuţie doar în celelalte situaţii
  - dezavantaj - erorile de programare pot bloca procesele (ex. buclă infinită)
- preemptiv

Deplanificarea

- cum ştie sistemul de operare cât timp s-a executat un proces?
- este necesară o formă de măsurare a timpului
- ceasul de timp real
  - dispozitiv periferic
  - generează cereri de întrerupere la intervale regulate de timp