UNIUNEA EUROPEANĂ
GUVERNUL ROMÂNIEI
Instrumente Structurale 2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Programare în limbaj de asamblare

40. Instalarea, activarea şi dezinstalarea acestor programe.

Instalarea unui nou vector de întrerupere

Partea de instalare a unui program TSR trebuie să realizeze următoarele operații:

- să verifice, în primul rând, dacă nu cumva a fost deja instalat în memorie codul său rezident; dacă programul a fost deja instalat, nu îl va mai instala şi programul se va termina cu funcția *4C00h, INT 21h*.
- să captureze un vector de întrerupere pe care îl înlocuieşte cu unul nou. Înainte de a modifica vectorul de întrerupere, programul rezident (cel ce se instalează), salvează în partea sa de inițializare vectorul curent şi apoi îl înscrie pe cel nou, care reprezintă adresa, de tip FAR a procedurii, pentru tratarea întreruperii definite în partea rezidentă;
- să calculeze dimensiunea memoriei necesare pentru codul rezident şi să apeleze funcția DOS *31h*, care lasă codul părții rezidente în memorie şi termină programul.

Prin capturarea unui vector de întrerupere se înțelege înlocuirea vectorului existent, cu unul nou, care conține adresa noii rutine de tratare a întreruperii, definită în programul TSR. De obicei, noua rutină apelează vechea rutină de tratare a întreruperii.

Înlocuirea unui vector de întrerupere, care constă în salvarea celui vechi şi înscrierea unuia nou, se poate implementa direct de utilizator, utilizând instrucțiuni de transfer (mov), dar este mult mai comodă utilizarea funcțiilor DOS (*INT 21h*) *35h* şi *25h*.

Funcția DOS *35h* returnează în ES:BX vechiul vector de întrerupere, cel care se doreşte a fi înlocuit, al cărui număr a fost transmis în registrul AL, astfel:

```
mov ah, 35h         ; funcția DOS de salvare a vechiului vector
mov al, nr_int      ; nr. Într. pe nivelul căreia se pune TSR-ul
int 21h
mov Vechea_Int, bx          ; salv. vechiului vector de întrerupere
mov Vechea_Int[2], es ; din ES:BX, în zona TSR-rezident
```

Funcția *25h* scrie noul vector de întrerupere, transmis în registrele DS:DX, în tabela vectorilor de întrerupere, în locul vectorului al cărui număr (index din tabelă) este specificat în registrul AL:

```
mov ah, 25h         ; funcția DOS de instalare a noului vector
mov al, nr_int      ; nr. Într. pe nivelul căreia se pune TSR-ul
lea dx, Noua_Int; offsetul RTI, adr. seg. este în DS, întrucât
                    ; toate reg. seg., la programele .COM
                    ; sunt inițializate cu segmentul PSP
int 21h             ; deci nu mai trebuie reinițializat cu
                    ;'seg Noua_Int'
```

Ca efect, la apariția unei întreruperi pe nivelul specificat în AL, se va executa noua procedură rezidentă (TSR), în locul celei vechi.

Salvarea vechiului vector şi instalarea celui nou se poate face şi fără funcțiile DOS, direct în tabela vectorilor de întrerupere (TVI); secvența aceasta trebuie însă executată cu întreruperile dezactivate.

Pentru a evita reinstalarea aceluiaşi program TSR, se verifică dacă există deja o copie în memorie a codului rezident, deci dacă acesta a fost deja instalat, prin compararea offset-ului noii rutine cu offset-ul vectorului de întrerupere utilizat. În cazul în care cele două offset-uri sunt diferite, înseamnă că programul nu este rezident şi, deci, poate fi instalat. Dacă, însă, cele două

offseturi sunt identice, întrucât se poate întâmpla ca programe diferite să utilizeze acelaşi vector de întrerupere şi să aibă acelaşi offset, se vor compara identificatorii celor două programe TSR. Două TSR-uri diferite pot avea acelaşi offset al adresei de lansare dacă zona de date ce precede începutul codului TSR are aceeaşi dimensiune. Din acest motiv trebuie înscris în codul rezident un şir de caractere care să identifice în mod unic acel program; acest identificator de program TSR se plasează înaintea programului TSR propriu-zis.

Pentru a lăsa rezidentă în memorie numai partea rezidentă a programului TSR se va utiliza funcţia *31h*, care are ca parametru de intrare dimensiunea de memorie ocupată de partea rezidentă, exprimată în paragrafe (1 paragraf = 16 octeţi), transmisă în registrul DX (adresa de segment este cea conţinută în DS, care are acelaşi conţinut cu toate celelalte registre segment în cazul programelor .COM). Funcţia va lăsa rezident în memorie codul de lungime specificată şi realizează ieşirea din program, în DOS. Memoria ocupată de partea de iniţializare a programului va fi eliberată de DOS.

Activarea unui program TSR

Partea rezidentă defineşte o zonă de date şi prelucrările respective, reprezentate de una sau mai multe proceduri.

Procedurile dintr-un program rezident devin active ca răspuns la un apel direct al unui alt program sau la apariţia unei întreruperi hardware. După lansare un lucru nu poate fi aşteptat pentru codul apelant: să iniţializeze corespunzător registrele segment. Singurul registru segment cu semnificaţie pentru codul rezident este registrul *CS*. Pentru acces la variabilele locale -> rutinele vor iniţializa *DS* sau alte registre segment. De ex. o rutină care numără de câte ori alte programe o apelează, după ce a devenit rezidentă. Ea ar conţine o singură instrucţiune *inc contor*, dar s-ar incrementa variabila de la offsetul *contor* în segmentul referit de *DS*, şi deci nu va incrementa variabila *contor*.

Două soluţii la această problemă: 1. toate variabilele în segmentul de cod, şi să se utilizeze prefixul segment (*inc cs:contor*); utilizată doar când avem puţine variabile. 2. salvare registrul segment de date, DS, la intrarea în TSR, iniţializarea lui cu noua valoare, iar înainte de ieşire din TSR să fie refăcut la valoarea avută.

Activarea unui program TSR presupune întreruperea celui curent executat. În cazul în care codul rezident foloseşte funcţii de sistem, el trebuie să testeze, la activarea sa, dacă programul întrerupt executa o funcţie *DOS/BIOS*. Codul rezident nu poate reactiva o funcţie de sistem, în timp ce se execută o altă funcţie *DOS/BIOS*, deoarece DOS-ul nu este un sistem de operare reentrant (accesul la disc: *13h, 25h, 26h,* nu poate fi întrerupt).

Această problemă este doar pentru TSR-uri active, întrucât cele pasive sunt apelate şi se vor executa în cadrul programului apelant, care nu poate executa simultan două apeluri.

Pentru a evita astfel de operaţii sistemul de operare utilizează două variabile: '*eroare critică*' şi '*secţiune'* sau *'stare critică*', pe care le activează corespunzător stării în care se află. Cele două variabile pot fi testate de codul rezident, în momentul în care se utilizează funcţii DOS. Adresele celor două variabile, şi deci starea lor, sunt returnate de funcţiile *34h* (numită indicator *secţiune critică* sau indicator *inDOS*) şi respectiv, *50D6h* (*eroare critică*), în registrele ES:BX. Cele două adrese pot fi memorate în zona de date a părţii rezidente, de către partea de iniţializare a programului TSR, pentru a le testa direct.

Întreruperea multiplexată, INT 2FH

Când se instalează un program TSR care are şi componente pasive, trebuie ales un vector de întrerupere pe care să fie instalat, astfel ca alte programe să poată comunica cu rutinele sale, pasive. Se poate alege un vector aproape la întâmplare dar aceasta poate conduce la anumite probleme de compatibilitate. Ce se întâmplă dacă altcineva utilizează deja acest vector de întrerupere ?

În multe situaţii alegerea este clară, cum ar fi în situaţia în care codul pasiv al TSR-ului extinde anumite servicii (tastatură, video). În celelalte situaţii, cum ar fi crearea unui nou driver pentru un nou dispozitiv, care să nu interfere funcţiile furnizate pentru acest dispozitiv, cu alte întreruperi, alegerea la întâmplare a unui vector de întrerupere este riscantă.

Pentru a rezolva aceste situaţii DOS-ul furnizează soluţia utilizării unei întreruperi multiplexate, *INT 2FH*, care permite instalarea, testarea prezenţei şi a comunicării cu un program TSR.

Pentru a utiliza această întrerupere, o aplicaţie plasează o valoare de identificare în *AH* şi un număr de funcţie în *AL*, şi apoi execută *INT 2FH*. Fiecare TSR din lanţul *INT 2FH* va compara valoarea din *AH* cu propria sa valoare unică de identificare. Dacă valoarea sa coincide cu cea din *AH*, atunci TSR-ul respectiv va prelucra comanda specificată de valoarea din registrul *AL*. Dacă nu se identifică, prin valoarea din *AH,* atunci TSR-ul respectiv transmite controlul la următorul manipulator din lanţul *INT 2FH*. Aceasta simplifică cumva problema, dar nu o elimină. De obicei, trebuie ales numărul întreruperii înainte de proiectarea TSR-ului. Cum, însă, o aplicaţie va şti ce valoare să încarce în *AH*, dacă se asignează dinamic această valoare, când TSR-ul este rezident ?

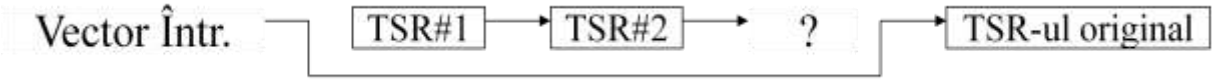
Se permite oricărei aplicaţii interesate să determine identificatorul (ID) TSR-ului. Prin convenţie, funcţia zero (*AL=0*) este un apel ce determină prezenţa TSR-ului. Acest apel se va realiza înainte de solicitarea oricărui serviciu TSR, pentru a determina dacă programul TSR este prezent în memorie.

Dezinstalarea unui program TSR

La dezinstalarea unui TSR trebuie făcute trei lucruri:

- *oprirea/ suspendarea* oricăror activităţi nerezolvate sau în aşteptare (adică, un TSR poate avea anumiţi indicatori iniţializaţi pentru a lansa anumite activităţi ulterioare);
- *refacerea* tuturor vectorilor de întrerupere la valorile lor iniţiale;
- *eliberarea* spaţiului de memorie ocupat, adică cedarea spaţiului de memorie rezervat înapoi către DOS, astfel ca alte aplicaţii să-l poată utiliza;

Dacă dezinstalarea TSR-ului constă doar în refacerea valorilor vectorilor de întrerupere, se va crea o problemă, dacă există alte TSR-uri care utilizează aceeaşi vectori de întrerupere. Să presupunem că avem următorul lanţ de TSR-uri, pe un anumit nivel de întrerupere:



Aceasta va dezactiva efectiv TSR-urile înlănțuite din program, iar aceste TSR-uri vor avea efecte neașteptate, în aceste condiții. O soluție ar fi tipărirea unui mesaj de eroare, care să informeze utilizatorul că nu se poate dezinstala acest TSR, până nu se dezinstalează toate TSR-urile instalate anterior acestuia.

Problema va fi în a determina TSR-urile înlănțuite în întrerupere. Pentru aceasta ştim că vectorii de întrerupere ar trebui încă să refere rutina noastră, dacă acesta este ultimul TSR care se execută. Atunci, trebuie să comparăm vectorii de întrerupere instalați cu adresele rutinelor de servire a întreruperii respective. Dacă toate acestea se potrivesc, atunci se poate elimina în siguranță TSR-ul din memorie. Dacă, însă, doar unul dintre ei nu se potrivește, atunci nu putem elimina TSR-ul din memorie.