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

28. Transferul parametrilor pentru proceduri în limbaj de asamblare.

Transferul parametrilor pentru proceduri

Parametrii (datele de intrare) pot fi transmişi în diferite moduri:

- **prin registre**; această modalitate poate avea, însă, următoarele dezavantaje:
  - neconvenabilă, dacă se transmit un număr mare de parametri, caz în care numărul registrelor poate fi insuficient, sau
  - neeconomică sub aspectul timpului de execuție, necesitând frecvente salvări şi refaceri de registre.
- **prin memorie**, care presupune rezervarea de memorie pentru aceşti parametri, deci poate fi neeconomică din punct de vedere al spațiului de memorie ocupat (pentru structuri mari de date), mai ales dacă astfel de date sunt necesare doar temporar;
- **prin stivă**, care este metoda cea mai avantajoasă, întrucât permite accesul uşor la informație (cu instrucțiunile PUSH, respectiv POP), iar după utilizare memoria alocată în acest scop este eliberată, deci zona respectivă de memorie devine disponibilă. În cazul transmiterii parametrilor prin stivă, aceasta trebuie manipulată cu atenție de către procedură, astfel încât, în final, în vârful stivei să se găsească adresa de revenire, pentru a se executa corect ultima instrucțiune din procedură: RET.

În ceea ce priveşte parametrii transmişi prin stivă, aceştia pot fi:

- valoare, se transmite valoarea parametrului;
- adresă, se transmite adresa parametrului, deci operațiile descrise de procedură se vor efectua direct asupra parametrului respectiv; în acest caz, procedura nu mai trebuie să returneze programului apelant parametrul respectiv.

În ceea ce priveşte returnarea parametrilor, această operație se poate realiza utilizând una din metodele folosite pentru transmiterea parametrilor către procedură, descrise anterior (registre, memorie sau stivă).

În general, pentru definirea unei proceduri se parcurg următoarele etape:

- stabilirea prelucrărilor efectuate de procedură;
- definirea datelor de intrare/ieşire, precum şi a modului de transmitere a parametrilor (registre, memorie sau stivă);
- definirea tipului procedurii (NEAR/FAR), mai ales dacă parametrii se transmit prin stivă, situație în care trebuie determinată poziția acestora în stivă;
- pentru directivele simplificate de segmentare nu este necesară precizarea tipului procedurii, deoarece tipul este determinat de asamblor, din directiva MODEL; dacă în procedură se utilizează nume simbolice, pentru a obține în mod corect accesul la acestea trebuie salvat DS în stivă, după care se încarcă noua valoare şi se foloseşte directiva ASSUME pentru a preciza asocierea segmentului la DS, iar la sfârşit se reface registrul DS cu valoarea din stivă.
- înainte de orice prelucrare trebuie salvate în stivă registrele utilizate de procedură, iar înainte de revenirea în programul apelant, acestea trebuie restaurate.

În ceea ce priveşte programele scrise în limbaje de nivel înalt, acestea folosesc anumite convenții pentru transmiterea parametrilor către subprograme. Compilatorul translatează o instrucțiune de apel a unui subprogram astfel: depune în stivă (push) argumentele subprogramului şi apelează apoi subprogramul cu o instrucțiune de apel (call). În funcție de compilator, argumentele sunt puse în stivă într-o anumită ordine. În cazul în care se leagă o procedură scrisă în limbaj de asamblare cu un program scris într-un limbaj de nivel înalt, trebuie cunoscut modul de

transmitere a parametrilor şi convenţiile stabilite de limbajul respectiv pentru apel de subprogram (nume externe, valori returnate).

Parametrii pot fi transmişi la subprograme prin valoare sau prin referinţă:

- la transmiterea prin valoare se va depune în stivă o copie a argumentelor; modificările efectuate asupra acestor valori nu se vor regăsi în parametrii efectivi din procedura apelată. Rezultatele vor fi cunoscute doar dacă vor fi comunicate;
- la transmiterea prin referinţă, se depune în stivă adresa (pointerul) argumentului (de tip NEAR sau FAR, după locul în care se află procedura apelată). În acest caz, modificările se vor efectua direct asupra argumentelor ce aparţin programului apelant.

În Pascal şi în C se cunosc ambele mecanisme de transmitere a parametrilor. Limbajele de nivel înalt folosesc următoarele registre pentru returnarea de valori către programul apelant:

(AL) – valori de un octet;
(AX) – valori de doi octeţi sau adresă de tip NEAR;
(EAX) – valori de 4 octeţi sau adresă de tip NEAR la 386/486;
(DX,AX) – valori de 4 octeţi sau pointer FAR, pentru 286;
(EDX,EAX) – valori de 8 octeţi sau pointer FAR, pentru 386/486;

Pentru valori mai mari de 4 octeţi, limbajul C consideră că valoarea este returnată în zona de memorie alocată pentru segmentul de date şi întoarce în (DX:AX) pointerul către acea zonă. Limbajul PASCAL rezervă o zonă de date în stivă: zona rezervată în stivă şi offsetul către aceasta se realizează de către programul apelant; procedura returnează în (DX:AX) pointerul către această zonă.

Descărcarea stivei se face în programul apelant, procedura terminându-se cu RET; la descărcarea unor argumente, se poate termina cu *RET număr_argumente×dimensiune*.

Deci, în principiu, programul apelant realizează operaţiile:

- depune argumentele în stivă;
- apelează procedura;
- descarcă stiva.

Variabilele locale într-o procedură se folosesc pentru a economisi spaţiu de memorie (cu menţiunea că valorile lor nu se transmit în afara procedurii). Spaţiul de memorie necesar pentru variabilele locale se rezervă în stivă şi este eliberat la terminarea execuţiei procedurii.

Alocarea de spaţiu pentru variabilele locale se realizează cu o secvenţă de forma:

push bp	Salvare registre de lucru
mov bp,sp	Variabile locale
sub sp,n	Salvare (BP)
	Adresa de revenire în program
	Argumente transmise procedurii

unde n = număr de octeţi alocaţi.

Pentru accesul la variabilele locale se vor utiliza adresări de forma [BP – 2], [BP – 6] etc. iar pentru argumente, de forma: [BP + 2], [BP + 4], etc.

Iată un exemplu de adresare la un argument transmis prin stivă şi la o variabilă locală, într-o procedură de tip FAR:

```
argument    equ                 <[bp+6]>
var_loc     equ                 <[bp-2]>
p1          proc    far
            push    bp          ; salvare valoare BP
            mov     bp,sp       ; initializare cu varful curent al stivei
            sub     sp,2        ; rezervare spatiu pentru var_loc, 2 octeti
            .........
            mov     ax,argument ; se ia argumentul transmis prin stiva
            add     var_loc,ax  ; utilizarea variabilei locale
            .........
            mov     sp,bp       ; eliberarea spatiului rezervat var_loc
            pop     bp          ; refacerea valorii lui BP
            ret                 ; sau ret 2, daca se descarca stiva in procedura
p1                      endp
```

Dacă descărcarea stivei nu se face în procedură, atunci în programul apelant trebuie descărcată stiva cu *add sp,2*.