Instrucțiuni pentru șiruri. Probleme „complexe” pe șiruri

Caracteristici:

* Au operanzi impliciti
* Prelucreaza elementul curent din sir si se muta la urmatorul element din sir

Pentru a putea utiliza aceste instructiuni TREBUIE:

* Copiat offset-ul sirului sursa in registrul ESI
* Copiat offset-ul sirului destinatie in registrul EDI
* Configurata directia de parcurgere a sirului
  + De la stanga la dreapta: DF = 0; CLD (clear direction flag)
  + De la dreapta la stanga: DF = 1; STD (set direction flag)

Clasificare:

* **Instructiuni pentru transferul datelor**
  + Din memorie in registrii (LODS – LOAD String OF … )

1. **LODSB**

* Incarca in **AL** octetul aflat in memorie la adresa **DS:ESI**
* Daca DF = 0 atunci ESI = ESI + 1
* Daca DF = 1 atunci ESI = ESI - 1

1. **LODSW**

* Incarca in **AX** cuvantul aflat in memorie la adresa **DS:ESI**
* Daca DF = 0 atunci ESI = ESI + 2
* Daca DF = 1 atunci ESI = ESI - 2

1. **LODSD**

* Incarca in **EAX** dublucuvantul aflat in memorie la adresa **DS:ESI**
* Daca DF = 0 atunci ESI = ESI + 4
* Daca DF = 1 atunci ESI = ESI - 4
  + Din registrii in memorie (STOS – STORE String OF ...)

1. **STOSB**

* Incarca in octetul aflat in memorie la adresa **ES:EDI** valoarea din **AL**
* Daca DF = 0 atunci EDI = EDI + 1
* Daca DF = 1 atunci EDI = EDI - 1

1. **STOSW**

* Incarca in cuvantul aflat in memorie la adresa **ES:EDI** valoarea din **AX**
* Daca DF = 0 atunci EDI = EDI + 2
* Daca DF = 1 atunci EDI = EDI - 2

1. **STOSD**

* Incarca in dublucuvantul aflat in memorie la adresa **ES:EDI** valoarea din **EAX**
* Daca DF = 0 atunci EDI = EDI + 4
* Daca DF = 1 atunci EDI = EDI - 4
  + Din memorie in alta zona de memorie (MOVS – MOVE String OF ...)

1. **MOVSB**

* Incarca in octetul aflat in memorie la adresa **ES:EDI** octetul aflat in memorie la adresa **DS:ESI**
* Daca DF = 0 atunci EDI = EDI + 1; ESI = ESI + 1
* Daca DF = 1 atunci EDI = EDI – 1; ESI = ESI - 1

1. **MOVSW**

* Incarca in cuvantul aflat in memorie la adresa **ES:EDI** cuvantul aflat in memorie la adresa **DS:ESI**
* Daca DF = 0 atunci EDI = EDI + 2; ESI = ESI + 2
* Daca DF = 1 atunci EDI = EDI – 2; ESI = ESI - 2

1. **MOVSD**

* Incarca in dublucuvantul aflat in memorie la adresa **ES:EDI** dublucuvantul aflat in memorie la adresa **DS:ESI**
* Daca DF = 0 atunci EDI = EDI + 4; ESI = ESI + 4
* Daca DF = 1 atunci EDI = EDI – 4; ESI = ESI - 4
* **Instructiuni pentru compararea datelor**
  + Compara valoarea dintr-un registru cu valoarea din memorie (SCAS – SCAN String OF ...)

1. **SCASB**

* Compara valoarea din AL cu octetul de la adresa ES:EDI (CMP AL, [ES:EDI])
* Daca DF = 0 atunci EDI = EDI + 1
* Daca DF = 1 atunci EDI = EDI - 1

1. **SCASW**

* Compara valoarea din AX cu cuvantul de la adresa ES:EDI (CMP AX, [ES:EDI])
* Daca DF = 0 atunci EDI = EDI + 2
* Daca DF = 1 atunci EDI = EDI - 2

1. **SCASD**

* Compara valoarea din EAX cu dublucucvantul de la adresa ES:EDI (CMP EAX, [ES:EDI])
* Daca DF = 0 atunci EDI = EDI + 4
* Daca DF = 1 atunci EDI = EDI - 4
  + Compara valoarea din memorie cu o alta valoare din memorie (CMPS - COMPARE STRING OF ...)

1. **CMPSB**

* Compara octetul de la adresa DS:ESI cu octetul de la adresa ES:EDI (CMP [DS:ESI], [ES:EDI])
* Daca DF = 0 atunci EDI = EDI + 1; ESI = ESI + 1
* Daca DF = 1 atunci EDI = EDI – 1; ESI = ESI - 1

1. **CMPSW**

* Compara cuvantul de la adresa DS:ESI cu cuvantul de la adresa ES:EDI (CMP [DS:ESI], [ES:EDI])
* Daca DF = 0 atunci EDI = EDI + 2; ESI = ESI + 2
* Daca DF = 1 atunci EDI = EDI - 2; ESI = ESI - 2

1. **CMPSD**

* Compara dublucuvantul de la adresa DS:ESI cu dublucuvantul de la adresa ES:EDI (CMP [DS:ESI], [ES:EDI])
* Daca DF = 0 atunci EDI = EDI + 4; ESI = ESI + 4
* Daca DF = 1 atunci EDI = EDI - 4; ESI = ESI + 4

Exercitii:

1. Se da un sir de octeti care contine litere mici. Sa se construiasca un nou sir de octeti care sa contina literele mari corespunzatoare literelor mici din sirul sursa.

|  |  |
| --- | --- |
| Data segment | Code segment |
| sir db ‘a’,’d’,’r’,’f’  lun equ $-sir  rez times lun db 0  dif equ ‘A’-’a’ | MOV EDI,rez  MOV ESI,sir  MOV ECX,lun  JECXZ Final  CLD  Repeta:  LODSB  ADD AL,dif  STOSB  LOOP Repeta  Final:  PUSH DWORD 0  CALL [exit] |

‘d’ - ‘a’ + ‘A’ = 3 + ‘A’ = ‘D’

1. Se da un sir de octeti. Sa se obtina sirul oglindit.

Ex. S db 1,2,3,4,5 => D db 5,4,3,2,1

|  |  |
| --- | --- |
| Data segment | Code segment |
| A db 1,2,3,4,5  L equ $-sir  Rez times l db 0 | Mov ecx,L  Mov esi,A+L-1  Mov edi,Rez  Jecxz Final  Repeta:  STD  LODSB  CLD  STOSB  Loop Repeta  Final:  Push dword 0  Call[exit] |

1. Se dau doua siruri de cuvinte. Sa se concateneze sirul octetilor inferiori ai cuvintelor din primul sir cu sirul octetilor superiori ai cuvintelor din cel de-al doilea sir. Sirul rezultat trebuie sortat crescator in interpretarea cu semn

Ex:

S1 dw 2345h, 0a5h, 368h, 3990h

S2 dw 4h, 2655h, 10

S1 S2

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 45h | 23h | a5h | 00h | 68h | 03h | 90h | 39h | 04h | 00h |
| 55h | 26h | 0ah | 00h |  |  |  |  |  |  |

P1: D db 45h, 0a5h, 68h, 90h, 00h, 26h, 00h

P2: D db … , 00h, 00h, …

|  |  |
| --- | --- |
| Data segment | Code segment |
| s1 dw 2345h, 0a5h, 368h,  3990h  l1 equ ($-s1)/2  s2 dw 4h, 2655h, 10  l2 equ ($-s2)/2  d times (l1+l2) db 0 | Mov ecx, l1  Mov esi, s1  Mov edi, d  Jecxz Final1  Repeta1:  Cld  Lodsw  Stosb  LOOP Repeta1  Final1:  Mov ecx, l2  Mov esi, s2  Jecxz Final  Repeta2:  Cld  Lodsw  Mov al,ah  Stosb  LOOP Repeta2  Mov dl,1  Repeta:  Cmp dl,0  Jz Terminat  Mov dl,0  Mov ecx, l1+l2-1  Mov esi,d  Jecxz Terminat  RepetaFor:  Mov al,[esi]  Cmp al,[esi+1]  Jle Next  Mov bl,[esi+1]  Mov [esi+1],al  Mov [esi],bl  Mov dl,1  Next:  Inc esi  LOOP RepetaFor  Jmp Repeta  Terminat:  Push dword 0  Call [exit] |

Ok = 1

While (ok == 1){

Ok = 0;

For (I=1; I<=len-1; I++)

If(d[I+1]<d[I]){

Aux=d[I];

D[I]=d[I+1]

D[I+1]=aux

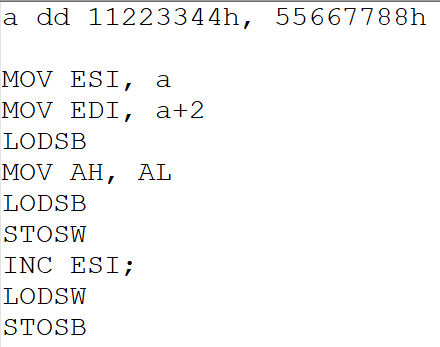
Ok=1

}

}

1. Se da un sir de valori numerice intregi reprezentate pe dublucuvinte. Sa se construiasca sirul corespunzator octetilor din reprezentarea in memorie a dublucuvintelor care contin doar valori negative.

|  |  |
| --- | --- |
| Data segment | Code segment |
| a dd 0FABC2365h,1234578h  l equ ($-a)/4  b times l\*4 db 0 | Mov ecx,l\*4  Mov esi,a  Mov edi,b  Jecxz final1  Cld  Repeta1:  Lodsb;al=a  Scasb  Jge decrementare  Dec edi  Stosb  Jmp exit1  decrementare:  Dec edi  Exit1:  Loop Repeta1  Final1: |



Cum arata memory layout-ul si care sunt registrii implicati la fiecare comanda si ce valoare au?

a dd 11223344h, 55667788h

44h|33h|22h|11h|88h|77h|66h|55h

MOV ESI, a

MOV EDI, a+2

LODSB; AL=44h; ESI=a+1

MOV AH, AL; AH=44h

LODSB; AL=33h; ESI=a+2

STOSW; se pune in memorie la adresa a+2 valoarea 4433h si EDI=a+4

44h|33h|33h|44h|88h|77h|66h|55h

INC ESI; ESI=a+3

LODSW; AX=8844h; ESI=a+4

STOSB; se pune in memorie la adresa a+4 valoarea 44h si EDI=a+5

44h|33h|33h|44h|44h|77h|66h|55h