**Laborator 6 - Suport teoretic**

**Operații pe şiruri de bytes/words/doublewords/quadwords**

**Operații pe şiruri**

* Instrucţiunile pe şiruri au operanzi impliciți.

**Tipuri de instrucţiuni pe şiruri:**

* care folosesc un şir sursa şi un şir destinație (MOVSB, MOVSW, MOVSD, CMPSB, CMPSW, CMPSD)
* care folosesc numai un şir sursa (LODSB, LODSW, LODSD)
* care folosesc numai un şir destinație (STOSB, STOSW, STOSD, SCASB, SCASW, SCASD)

**Pentru a fi reținute mai ușor**

* Move String = MOVS
* Compare String = CMPS
* Load String = LODS
* Store String = STOS
* Scan String = SCAS

**Un şir este caracterizat de:**

* *tipul elementelor (octeţi, cuvinte sau dublucuvinte)* => este indicat de ultima litera a instrucţiunii folosite (B=byte, W=word, D=doubleword), ambele şiruri având același tip
* *adresa primului element* => este o adresa FAR memorata astfel:
  + in DS:ESI - pentru şirul sursa
  + in ES:EDI - pentru şirul destinație
* *direcția de parcurgere* => este indicata de valoarea din flagul DF (0 - de la adrese mai mici la adrese mai mari, 1 - de la adrese mai mari la adrese mai mici.)
* *numărul de elemente* => când este nevoie de el, se pune în CX sau ECX

## Instrucţiuni pentru transfer de date

|  |  |
| --- | --- |
| LODSB | În AL se încarcă octetul de la adresa <DS:ESI> Dacă DF=0 atunci inc(ESI), altfel dec(ESI) |
| LODSW | In AX se încarcă cuvântul de la adresa <DS:ESI> Dacă DF=0 atunci ESI:=ESI+2, altfel ESI:=ESI-2 |
| LODSD | In EAX se încarcă dublucuvântul de la adresa <DS:ESI> Dacă DF=0 atunci ESI:=ESI+4, altfel ESI:=ESI-4 |
| STOSB | La adresa <ES:EDI> se încarcă octetul din AL Dacă DF=0 atunci inc(EDI), altfel dec(EDI) |
| STOSW | La adresa <ES:EDI> se încarcă cuvântul din AX Dacă DF=0 atunci EDI:= EDI+2, altfel EDI:= EDI-2 |
| STOSD | La adresa <ES:EDI> se încarcă dublucuvântul din EAX Dacă DF=0 atunci EDI:= EDI+4, altfel EDI:= EDI-4 |
| MOVSB | La adresa <ES:EDI> se încarcă octetul de la adresa <DS:ESI> Dacă DF=0 atunci inc(SI), inc(DI), altfel dec(SI), dec(DI) |
| MOVSW | La adresa <ES:EDI> se încarcă cuvântul de la adresa <DS:ESI> Dacă DF=0 atunci ESI:= ESI+2, EDI:= EDI+2, altfel ESI:= ESI-2, EDI:= EDI-2 |
| MOVSD | La adresa <ES:EDI> se încarcă dublucuvântul de la adresa <DS:ESI> Dacă DF=0 atunci ESI:= ESI+4, EDI:= EDI+4, altfel ESI:= ESI-4, EDI:= EDI-4 |

* **Obs. Având în vedere utilizarea modelului de memorie flat, la orice început de execuție a programului, SO va inițializa cu aceeași valoare regiştrii segment DS = ES, programatorul neavând nici o responsabilitate de încărcare/actualizare/modificare a acestor valori. In cadrul codului sursa ce utilizează instrucţiuni pe şiruri programatorul va trebui sa gestioneze doar offset-urile acestor şiruri.**

### *Exemplu:*

;Avem un şir sursa (cuvinte). Sa se copieze intr-un şir destinație. Știm câte elemente avem.

mov ECX, dim\_şir ; nr de elemente din şir

mov ESI, şir\_sursa ; încărcare offset şir\_sursa in ESI

mov EDI, şir\_dest ; încărcare offset şir\_dest in EDI

CLD

Again:

LODSW

STOSW

LOOP Again

* Având în vedere ca LODS + STOS = MOVS bucla de mai sus se mai poate scrie:

Again:

MOVSW

LOOP Again

* sau (a se vedea secțiunea prefixelor de instrucţiune de mai jos)

rep MOVSW

## Instrucţiuni pentru consultarea şi compararea datelor

|  |  |
| --- | --- |
| SCASB | CMP AL, <ES:EDI>  Dacă DF=0 atunci inc(EDI), altfel dec(EDI) |
| SCASW | CMP AX, <ES:EDI>  Dacă DF=0 atunci EDI:= EDI+2, altfel EDI:= EDI-2 |
| SCASD | CMP EAX, <ES:EDI>  Dacă DF=0 atunci EDI:= EDI+4, altfel EDI:= EDI-4 |
| CMPSB | CMP <DS:ESI>, <ES:EDI>  Dacă DF=0 atunci inc(ESI), inc(EDI), altfel dec(ESI), dec(EDI) |
| CMPSW | CMP <DS:ESI>, <ES:EDI>  Dacă DF=0 atunci ESI:= ESI+2, EDI:= EDI+2, altfel ESI:= ESI-2, EDI:= EDI-2 |
| CMPSD | CMP <DS:ESI>, <ES:EDI>  Dacă DF=0 atunci ESI:= ESI+4, EDI:= EDI+4, altfel ESI:= ESI-4, EDI:= EDI-4 |

### *Exemplu:*

;Se da un şir de octeţi. Sa se găsească ultimul caracter "0".

;... se încarcă toate datele despre şirul "destinație"

MOV AL, '0'

MOV ECX, lung\_şir

STD

Cont\_caut: ;continui căutarea...

SCASB

JE Găsit

LOOP Cont\_caut

;...

Găsit:

INC EDI;ma întorc la caracterul găsit înainte să se fi făcut decrementarea lui EDI

**Prefixe de instrucţiune pentru execuția repetata a unei instrucţiuni**

prefix\_de\_instrucţiune instrucţiune\_pe\_şir

* echivalentă cu

Again:

instrucţiune\_pe\_şir

LOOP Again

* unde *prefix\_de\_instrucţiune* poate fi REP, echivalent cu REPE (*Repeat While Equal*), REPZ (*Repeat While Zero*) - care provoacă execuția repetată a instrucţiunilor SCAS sau CMPS până când ECX devine 0 sau până când apare o nepotrivire ( => ZF=0)
* sau poate fi REPNE (*Repeat While Not Equal*) sau REPNZ (*Repeat While Not Zero*) - care provoacă execuţia repetată a instrucţiunii SCAS sau CMPS până când ECX devine 0 sau până când apare o potrivire ( => ZF=1)

**Observații:**

* instrucţiunile pe şiruri nu afectează flagurile în urma acțiunii asupra regiştrilor ESI, EDI sau ECX
* LODS, STOS, MOVS – nu afectează nici un flag, în timp ce SCAS şi CMPS modifică flagurile doar ca rezultat al comparațiilor efectuate

# Laborator 6 - Exemple

## Operații pe şiruri de bytes/words/doublewords/quadwords

## Exemplu

;Problema. Se da un şir de valori numerice întregi reprezentate pe quadworduri.

;Sa se determine suma cifrelor numărului multiplilor de 8 din şirul octeţilor

;inferiori ai cuvintelor superioare ai dublucuvintelor superioare din elementele şirului de quadworduri.

;Soluție: Parcurgând şirul de quadworduri vom obține întâi numărul ;multiplilor de 8 din

;şirul octeţilor inferiori ai cuvintelor superioare ai dublucuvintelor

;superioare din elementele şirului. Apoi vom obține

;cifrele acestui număr prin împărţiri succesive la 10 şi vom calcula suma ;lor.

bits 32

global start

extern exit; tell nasm that exit exists even if we won't be defining it

import exit msvcrt.dll; exit is a function that ends the calling process. It is defined in msvcrt.dll

; our data is declared here (the variables needed by our program)

segment data use32 class=data

şir dq 123110110abcb0h,1116adcb5a051ad2h,4120ca11d730cbb0h

len equ ($-şir)/8;lungimea şirului (in dublucuvinte)

opt db 8;variabila folosita pentru testarea divizibilității cu 8

zece dw 10;variabila folosita pentru determinarea cifrelor unui număr prin împărțiri succesive la 10

suma dd 0;variabila in care reținem suma cifrelor

; our code starts here

segment code use32 class=code

start:

mov esi, şir

cld;parcurgem şirul de la stânga la dreapta (DF=0).

mov ecx, len;vom parcurge elementele şirului intr-o bucla loop cu len iterații.

mov ebx, 0;in registrul ebx vom retine numărul multiplilor lui 8.

repeta:

lodsd;in eax vom avea dublucuvântul mai puțin semnificativ al quadword-ului curent din şir

lodsd;in eax vom avea dublucuvântul cel mai semnificativ al quadword-ului curent din şir

shr eax, 16

mov ah, 0;ne interesează doar octetul mai puțin semnificativ din acest cuvânt (AL)

div byte[opt];vedem dacă al este divizibil cu 8

cmp ah, 0;dacă restul nu este 0, reluam ciclul repeta.

;Altfel incrementam numărul multiplilor de 8 din registrul ebx.

jnz nonmultiplu

inc ebx

nonmultiplu:

loop repeta;dacă mai sunt elemente de parcurs(ecx>0) reia ciclul.

;mai departe, obținem cifrele numărului ebx in baza 10 prin împărțiri succesive la 10 şi calculam suma acestor cifre.

mov eax, ebx

mov edx, 0

transf:

div dword[zece];împărţim la 10 numărul din registrul ca sa aflam ultima cifra; aceasta cifra se afla in EDX

add dword[suma], edx;adunam cifra la suma.

cmp eax, 0

jz sfarsit;dacă catul este 0 înseamnă ca am obținut toate cifrele şi putem parasi bucla transf

;Altfel, îl pregătim pentru o noua iterație

mov edx, 0

jmp transf;reluam bucla pentru obținerea unei noi cifre.

sfarsit:;încheiem programul.

push dword 0; push the parameter for exit onto the stack

call [exit]; call exit to terminate the program

# Laborator 6 - Probleme propuse

## Operații pe şiruri de bytes/words/doublewords/quadwords

## Exerciții

Problemele din acest laborator trebuie rezolvate folosind instrucţiuni specifice lucrului cu şiruri: LODSB, STOSB, MOVSB, SCASB, CMPSB, LODSW, STOSW, MOVSW, SCASW, CMPSW, LODSD, STOSD, MOVSD, SCASD, CMPSD.

1. Se da un şir de dublucuvinte conținând date împachetate (4 octeţi scrişi ca un singur dublucuvânt). Sa se obțină un nou şir de dublucuvinte, in care fiecare dublucuvânt se va obține după regula: suma octeţilor de ordin impar va forma cuvântul de ordin impar, iar suma octeţilor de ordin par va forma cuvântul de ordin par. Octeţii se considera numere cu semn, astfel ca extensiile pe cuvânt se vor realiza corespunzător aritmeticii cu semn.

### *Exemplu:*

pentru şirul inițial:

127F5678h, 0ABCDABCDh, ...

Se va obține:

006800F7h, 0FF56FF9Ah

1. Se da un şir de cuvinte. Sa se obţină din acesta un şir de dublucuvinte, in care fiecare dublucuvânt va conţine nibble-urile despachetate pe octet (fiecare cifra hexa va fi precedata de un 0), aranjate crescător in interiorul dublucuvântului.

### *Exemplu:*

pentru şirul inițial:

1432h, 8675h, 0ADBCh, ...

Se va obține:

01020304h, 05060708h, 0A0B0C0Dh, ...

1. Se da un şir de 3 dublucuvinte, fiecare dublucuvânt conținând 2 valori pe cuvânt (despachetate, deci fiecare cifra hexa e precedata de un 0). Sa se creeze un şir de octeţi care să conţină acele valori (împachetate deci pe un singur octet), ordonate crescător în memorie, acestea fiind considerate numere cu semn.

### *Exemplu:*

pentru şirul inițial:

0702090Ah, 0B0C0304h, 05060108h

se va obține:

72h, 9Ah, 0BCh, 34h, 56h, 18h

care se va ordona crescător:

9Ah, 0BCh, 18h, 34h, 56h, 72h

1. Se da un şir de octeţi s. Să se construiască şirul de octeţi d, care conţine pe fiecare poziție numărul de biţi 1 ai octetului de pe poziția corespunzătoare din s.

### *Exemplu:*

s: 5, 25, 55, 127

in binary:

101, 11001,10111,1111111

d: 2, 3, 5, 7

1. Se dau doua şiruri de octeţi s1 şi s2. Sa se construiască şirul de octeţi d, care conţine pentru fiecare octet din s2 poziția sa in s1, sau 0 in caz contrar.

### *Exemplu:*

pos: 1, 2, 3, 4, 5

s1: 7, 33, 55, 19, 46

s2: 33, 21, 7, 13, 27, 19, 55, 1, 46

d: 2, 0, 1, 0, 0, 4, 3, 0, 5

1. Se da un şir de cuvinte s. Sa se construiască şirul de octeţi d, astfel încât d sa conțină pentru fiecare poziție din s:

- numărul de biţi de 0, dacă numărul este negativ  
- numărul de biţi de 1, dacă numărul este pozitiv

### *Exemplu:*

s: -22, 145, -48, 127

in binary:

1111111111101010, 10010001, 1111111111010000, 1111111

d: 3, 3, 5, 7

1. Se da un şir de dublucuvinte. Să se obţină şirul format din octeţii superiori ai   
   cuvintelor superioare din elementele şirului de dublucuvinte care sunt divizibili cu 3.

### *Exemplu:*

Se da şirul de dublucuvinte:

s DD 12345678h, 1A2B3C4Dh, FE98DC76h

Să se obţină şirul

d DB 12h.

1. Se da un şir de dublucuvinte. Să se obţină şirul format din octeţii inferiori ai   
   cuvintelor superioare din elementele şirului de dublucuvinte care sunt palindrom in scrierea in baza 10.

### *Exemplu:*

Se da şirul de dublucuvinte:

s DD 12345678h, 1A2C3C4Dh, 98FCDC76h

Să se obţină şirul

d DB 2Ch, FCh.

1. Se da un şir de dublucuvinte. Să se obţină, începând cu partea inferioara a dublucuvântului, dublucuvântul format din octeţii superiori pari ai cuvintelor inferioare din elementele şirului de dublucuvinte. Dacă nu sunt îndeajuns octeţi se va completa cu octetul FFh.

### *Exemplu:*

Se da şirul de dublucuvinte:

s DD 12345678h, 1A2C3C4Dh, 98FCDD76h, 12783A2Bh

Să se obţină dublucuvânt

d DD FF3A3C56h.

1. Se da un şir A de cuvinte. Construiți două şiruri de octeţi    
    - B1: conţine ca elemente partea superioară a cuvintelor din A

 - B2: conţine ca elemente partea inferioară a cuvintelor din A

1. Se dă un şir A de dublucuvinte. Construiți doua şiruri de octeţi

 - B1: conţine ca elemente partea superioara a cuvintelor superioare din A

 - B2: conţine ca elemente partea inferioara a cuvintelor inferioare din A

1. Se da un şir A de dublucuvinte. Construiți doua şiruri de octeţi

- B1: conţine ca elemente partea inferioară a cuvintelor inferioare din A

- B2: conţine ca elemente partea superioară a cuvintelor superioare din A

1. Se da un şir S de dublucuvinte.

Să se obţină şirul D format din octeţii inferiori ai cuvintelor inferioare din elementele şirului de dublucuvinte, care sunt multiplii de 7.

### *Exemplu:*

s DD 12345607h, 1A2B3C15h, 13A33412h

d DB 07h, 15h

1. Se da un şir S de dublucuvinte.

Să se obţină şirul D format din octeţii dublucuvintelor din şirul D sortati in ordine crescatoare în interpretarea fără semn.

### *Exemplu:*

s DD 12345607h, 1A2B3C15h

d DB 07h, 12h, 15h, 1Ah, 2Bh, 34h, 3Ch, 56h

1. Se da un şir S de dublucuvinte. Să se obţină şirul D format din octeţii dublucuvintelor din şirul D sortaţi în ordine descrescătoare in interpretarea fără semn.

### *Exemplu:*

s DD 12345607h, 1A2B3C15h

d DB 56h, 3Ch, 34h, 2Bh, 1Ah, 15h, 12h, 07h

1. Se dau două şiruri de caractere ordonate alfabetic s1 şi s2. Să se construiască prin interclasare şirul ordonat s3 care să conțină toate elementele din s1 şi s2.
2. Se dă un şir de dublucuvinte. Să se ordoneze descrescător şirul cuvintelor inferioare ale acestor dublucuvinte. Cuvintele superioare rămân neschimbate.

### *Exemplu:*

dându-se:

şir DD 12345678h 1256ABCDh, 12AB4344h

rezultatul va fi

1234ABCDh, 12565678h, 12AB4344h.

1. Se da un şir de dublucuvinte. Sa se ordoneze crescător şirul cuvintelor superioare ale acestor dublucuvinte. Cuvintele inferioare rămân neschimbate.

### *Exemplu:*

dându-se:

şir DD 12AB5678h, 1256ABCDh, 12344344h

rezultatul va fi

12345678h, 1256ABCDh, 12AB4344h.

1. Dându-se doua şiruri de octeţi sa se calculeze toate pozițiile unde al doilea şir apare ca subşir in primul şir.
2. Se da un şir de octeţi reprezentând un text (succesiune de cuvinte separate de spatii). Sa se identifice cuvintele de tip palindrom (ale căror oglindiri sunt similare cu cele de plecare): "cojoc", "capac" etc.
3. Dându-se un şir de cuvinte Să se obţină şirul (de octeţi) cifrelor in baza zece ale fiecărui cuvânt din acest şir.

### *Exemplu:*

dacă avem şirul:

şir DW 12345, 20778, 4596

obținem rezultatul

1, 2, 3, 4, 5, 2, 0, 7, 7, 8, 4, 5, 9, 6.

1. Se da un şir de octeţi 'input' şi încă doua şiruri de dimensiune N fiecare, 'src' şi 'dst'. Să se obţină un nou şir 'output' din şirul 'input' in care se vor înlocui toți octeţii cu valoarea src[i] cu dst[i], unde i=1..N.
2. Dându-se un şir de octeţi Să se obţină un şir de cuvinte care sa conțină în octeţii inferiori mulțimea caracterelor din şirul de octeţi, iar octetul superior al unui cuvânt să conțină numărul de apariţii al octetului inferior din acel cuvânt în şirul de octeţi dat.

### *Exemplu:*

se dă şirul

şir DB 2, 4, 2, 5, 2, 2, 4, 4

se va obține şirul

rez DW 0402h, 0304h, 0105h.

1. Dându-se un şir de dublucuvinte, Să se obţină un alt şir de dublucuvinte în care se vor păstra doar dublucuvintele din primul şir care au un număr par de biţi cu valoare 1.
2. Se dă un şir de octeţi. Să se obţină şirul oglindit al reprezentării binare a acestui şir de octeţi.

### *Exemplu:*

Se dă şirul de octeţi:

s DB 01011100b, 10001001b, 11100101b

Să se obţină şirul

d DB 10100111b, 10010001b, 00111010b.

1. Se dă un şir de dublucuvinte. Să se obţină şirul format din octeţii superiori ai cuvintelor inferioare din elementele şirului de dublucuvinte, care sunt multiplii de 10.

### *Exemplu:*

Se da şirul de dublucuvinte:

s DD 12345678h, 1A2B3C4Dh, FE98DC76h

Să se obţină şirul

d DB 3Ch, DCh.

1. Dându-se un şir de cuvinte, să se calculeze cel mai lung subşir de cuvinte ordonate crescător din acest şir.
2. Dându-se un şir de octeţi şi un subşir al sau, să se elimine din primul şir toate aparițiile subşirului.
3. Se dau doua şiruri de octeţi. Să se parcurgă cel mai scurt şir dintre cele doua şiruri şi să se construiască un al treilea şir care va conţine cel mai mare element de același rang din cele două şiruri, iar până la lungimea celui mai lung şir, şirul al treilea se va completa alternativ cu valoarea 1 respectiv 0.
4. Se dă un şir de cuvinte. Să se construiască două şiruri de octeţi, s1 şi s2, astfel: pentru fiecare cuvânt,
   * dacă numărul de biţi 1 din octetul high al cuvântului este mai mare decât numărul de biţi 1 din octetul low, atunci s1 va conţine octetul high, iar s2 octetul low al cuvântului
   * dacă numărul de biţi 1 din cei doi octeţi ai cuvântului sunt egali, atunci s1 va conţine numărul de biţi 1 din octet, iar s2 valoarea 0
   * altfel, s1 va conţine octetul low, iar s2 octetul high al cuvântului.