Отчет

по лабораторной работе №7

по дисциплине “Архитектура компьютера”

Тема: “Преобразование целых чисел. Использование процедур в Ассемблере.”

Вариант №2

(2.1.2.2.В)

Студент гр. 5383 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Допира В. Е.

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кирьянчиков В.А.

**Цель работы**

Разработать на языке Ассемблер IBM PC две процедуры: одна - прямого и другая - обратного преобразования целого числа, заданного в регистре АХ (или в паре регистров DХ:АX) в строку, представляющую его символьное изображение в заданной системе счисления (с учетом или без учета знака в зависимости от варианта задания). Строка должна храниться в памяти, а также выводиться на экран для индикации. Отрицательные числа при представлении с учетом знака должны в памяти храниться в дополнительном коде, а на экране изображаться в прямом коде с явным указанием знака или в символьном виде со знаком.

Пример для однобайтовых чисел:

Десятичное число в символьном виде Двоично-десят. упаков.число в ДК в ПК

+ 35 00110101 00110101

- 35 11001011 10110101

Вариант №2: 2.1.2.2.В.

- 1-я цифра задает длину целого числа: 2- 32 бита;

- 2-я цифра задает вид представления числа: 1- с учетом знака;

- 3-я цифра задает систему счисления для символьного изображения числа: 2- восьмеричная

Написать простейшую головную программу для иллюстрации корректности выполнения заданных преобразований. При этом вызываемые процедуры будут типа: 2 - far.

Связь по данным между основной программой и подпрограммами осуществляется B - через кадр стека.

**Файл листинга**

0000AStack SEGMENT STACK  
0000 0400[ DB 1024 DUP(?)  
??  
]  
0400AStack ENDS  
  
0000DATA SEGMENT  
0000 2B 2E 2E 2E 2E 2E StartStr DB "+...........","$"  
2E 2E 2E 2E 2E 2E  
24  
000D 2B 30 30 30 30 30 FinalStr DB "+00000000000","$"  
30 30 30 30 30 30  
24  
001ADATA ENDS  
  
0000CODE SEGMENT  
ASSUME SS:AStack, DS:DATA, CS:CODE  
  
  
0000NumStr PROC FAR ;число из стека 4 байта, начиная с bp+8, кодирует в строку в 8ричном формате

0000 55 push bp  
0001 8B EC mov bp, sp  
0003 57 push di  
0004 50 push ax  
0005 53 push bx  
0006 51 push cx  
0007 52 push dx  
  
0008 8B 7E 06 mov di, [bp+6];адрес строки  
000B 8B 56 08 mov dx, [bp+8] ;число, которое нужно закодировать  
000E 8B 46 0A mov ax, [bp+10]  
  
0011 83 FA 00 cmp dx, 0   
0014 7D 0C jge sign  
0016 B3 2D mov bl, '-'  
0018 F7 D8 neg ax  
001A 83 D2 00 adc dx, 0 ;число "нормально перешло " в дополнительный код

001D F7 DA neg dx  
  
001F EB 03 90 jmp aftersign  
  
0022sign: ;если >=0  
0022 B3 2B mov bl, '+'  
  
0024aftersign:   
0024 88 1D mov [di], bl ;к строке приписываем "+" или "-"

0026 83 C7 0B add di, ;записываем в строку с конца

0029 B9 000B mov cx, 11  
002Cfollowingnumber:  
002C 8A D8 mov bl, al ;записываем начало числа  
002E 80 E3 07 and bl, 0111b ;берем 3 младших бита числа  
0031 80 C3 30 add bl, 48 ;переходим к закодированным числовым символам

0034 88 1D mov [di], bl ;запишем получившуюся цифру в строку результата

0036 4F dec di ;подвинули указатель на строку  
0037 D1 E8 shr ax, 1 ;сдвигаем число вправо на 3  
0039 D1 E8 shr ax, 1  
003B D1 E8 shr ax, 1  
003D 8B DA mov bx, dx  
003F 83 E3 07 and bx, 0111b  
0042 D1 CB ror bx, 1  
0044 D1 CB ror bx, 1  
0046 D1 CB ror bx, 1  
0048 D1 EA shr dx, 1  
004A D1 EA shr dx, 1  
004C D1 EA shr dx, 1  
004E 03 C3 add ax, bx  
0050 E2 DA loop followingnumber  
  
0052numend:  
0052 5A pop dx  
0053 59 pop cx  
0054 5B pop bx  
0055 58 pop ax  
0056 5F pop di  
0057 5D pop bp  
0058 CA 0006 ret 6  
005BNumStr ENDP  
  
005BStrNum PROC FAR ;процедура декодирования строки обратно в число

005B 55 push bp  
005C 8B EC mov bp, sp  
005E 56 push si  
005F 50 push ax  
0060 53 push bx  
0061 51 push cx  
0062 52 push dx  
  
0063 B8 0000 mov ax, 0  
0066 BA 0000 mov dx, 0  
0069 8B 76 06 mov si, [bp+6]  
006C B9 000B mov cx, 11   
  
006Ffollowingchar:  
; shld ax, dx, 3  
006F D1 E2 shl dx, 1  
0071 D1 E2 shl dx, 1  
0073 D1 E2 shl dx, 1  
0075 8B D8 mov bx, ax  
0077 81 E3 E000 and bx, 01110000000000000b  
007B D1 C3 rol bx, 1  
007D D1 C3 rol bx, 1  
007F D1 C3 rol bx, 1  
0081 D1 E0 shl ax, 1  
0083 D1 E0 shl ax, 1  
0085 D1 E0 shl ax, 1  
0087 03 D3 add dx, bx  
  
0089 83 C6 01 add si, 1  
008C 8A 1C mov bl, [si]  
008E 80 EB 30 sub bl, 48  
0091 02 C3 add al, bl  
0093 E2 DA loop followingchar  
  
0095 83 EE 0B sub si, 11  
0098 8A 1C mov bl, [si]  
009A 80 FB 2B cmp bl, '+'  
009D 74 07 je charend  
009F F7 D8 neg ax  
00A1 83 D2 00 adc dx, 0  
00A4 F7 DA neg dx  
  
00A6charend:  
00A6 89 56 08 mov [bp+8], dx  
00A9 89 46 0A mov [bp+10], ax  
00AC 5A pop dx  
00AD 59 pop cx  
00AE 5B pop bx  
00AF 58 pop ax  
00B0 5E pop si  
00B1 5D pop bp  
00B2 CA 0002 ret 2  
00B5StrNum ENDP  
  
00B5Main PROC FAR  
00B5 1E push ds  
00B6 2B C0 sub ax, ax  
00B8 50 push ax  
00B9 B8 ---- R mov ax, data  
00BC 8E D8 mov ds, ax  
  
00BE 50 push ax ;кодируем число из ax,dx в строку  
00BF 52 push dx  
00C0 BB 0000 R mov bx, OFFSET StartStr  
00C3 53 push bx  
00C4 9A 0000 ---- R call NumStr  
  
00C9 83 EC 04 sub sp, 4 ;выделяем 4 байта из стека  
00CC BB 000D R mov bx, OFFSET FinalStr  
00CF 53 push bx  
00D0 9A 005B ---- R call StrNum  
00D5 5A pop dx  
00D6 58 pop ax  
  
00D7 BA 0000 R mov dx, OFFSET StartStr  
00DA B4 09 mov ah, 9  
00DC CD 21 int 21h  
  
00DE CB ret  
00DFMain ENDP  
00DFCODE ENDS  
END Main  
Microsoft (R) Macro Assembler Version 5.10 12/14/16 01:33:5  
Symbols-1  
  
Segments and Groups:  
N a m e Length AlignCombine Class  
ASTACK . . . . . . . . . . . . . 0400PARASTACK  
CODE . . . . . . . . . . . . . . 00DFPARANONE  
DATA . . . . . . . . . . . . . . 001APARANONE  
  
Symbols:   
N a m e Type Value Attr  
AFTERSIGN . . . . . . . . . . . L NEAR0024CODE  
CHAREND . . . . . . . . . . . . L NEAR00A6CODE  
  
FINALSTR . . . . . . . . . . . . L BYTE000DDATA  
FOLLOWINGCHAR . . . . . . . . . L NEAR006FCODE  
FOLLOWINGNUMBER . . . . . . . . L NEAR002CCODE  
  
MAIN . . . . . . . . . . . . . . F PROC00B5CODELength = 002A  
  
NUMEND . . . . . . . . . . . . . L NEAR0052CODE  
NUMSTR . . . . . . . . . . . . . F PROC0000CODELength = 005B  
  
SIGN . . . . . . . . . . . . . . L NEAR0022CODE  
STARTSTR . . . . . . . . . . . . L BYTE0000DATA  
STRNUM . . . . . . . . . . . . . F PROC005BCODELength = 005A  
  
@CPU . . . . . . . . . . . . . . TEXT 0101h  
@FILENAME . . . . . . . . . . . TEXT lr7  
@VERSION . . . . . . . . . . . . TEXT 510  
  
159 Source Lines  
159 Total Lines  
19 Symbols  
  
47910 + 457284 Bytes symbol space free  
  
0 Warning Errors  
0 Severe Errors

**Карта памяти**

Start Stop Length Name Class

00000H 003FFH 00400H ASTACK

00400H 00419H 0001AH DATA

00420H 004FEH 000DFH CODE

Program entry point at 0042:00B5

**Тестирование**

Преобразование числа в строку:

|  |  |
| --- | --- |
| **Исходная строка** | **Число** |
| 0000 0000 | +00000000000 |
| 0000 0001 | +00000000001 |
| 7FFF FFFF | +17777777777 |
| 8000 0000 | -20000000000 |
| FFFF FFFE | -00000000002 |
| FFFF FFFF | -00000000001 |

Преобразование строки в число:

|  |  |
| --- | --- |
| **Исходное число** | **Строка** |
| -20000000000 | 8000 0000 |
| -17777777777 | 8000 0001 |
| -10000000000 | C000 0000 |
| 00000000000 | 0000 0000 |
| 10000000000 | 4000 0000 |
| 17777777777 | 7FFF FFFF |

**Вывод**

Программа состоит из 2-х ассемблерных процедур, которые преобразуют целое число в 32 битах с учетом знака в восьмеричной системе в строку, и обратно. При этом вызываемые процедуры типа far, то есть обращение к процедуре может осуществляться из другого сегмента кода. Связь данных между основной программой и подпрограммами осуществляется через кадр стека.