МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА(ЛЕНИНА)

КАФЕДРА МО ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Организация ЭВМ и систем»

Тема: Представление и обработка целых чисел. Организация ветвящихся процессов

Студентка гр. 1381 Деркачева Д.Я.

Преподаватель Ефремов М. А.

Санкт-Петербург

2022

Цель работы

Изучить представление и обработку целочисленных значений на языке Ассемблера. Разобраться в организации ветвящихся процессов.

Текст задания

Разработать на языке Ассемблера программу, которая по заданным целочисленным значениям параметров a, b, i, k вычисляет:

- a) значения функций i1 = f1(a,b,i) и i2 = f2(a,b,i);
- b) значения результирующей функции res = f3(i1,i2,k), где вид функций f1 и f2 определяется из табл. 2, а функции f3 - из табл.3 по цифрам шифра индивидуального задания (n1,n2,n3), приведенным в табл.4.
- Значения a, b, i, k являются исходными данными, которые должны выбираться студентом самостоятельно и задаваться в процессе исполнения программы в режиме отладки. При этом следует рассмотреть всевозможные комбинации параметров a, b и k, позволяющие проверить различные маршруты выполнения программы, а также различные знаки параметров a и b.

Ход выполнения работы

Вариант 7 - функции 1.8.7

$$f1 = \begin{cases} 15-2*i, & \text{при } a>b \\ 3*i+4, & \text{при } a<=b \end{cases} \qquad f8 = \begin{cases} -(6*i+8), & \text{при } a>b \\ 9-3*(i-1), & \text{при } a<=b \end{cases}$$

$$f7 = \begin{cases} |i1| + |i2|, & \text{при } k<0 \\ \max(6, |i1|), & \text{при } k>=0 \end{cases}$$

В процессе выполнения задания была разработана программа, которая состоит из несколько частей:

1. Описание сегментов программы. В их число входят сегмент стека AStack , сегмент данных DATA в котором была выделена память и проинициализированы переменные a, b, i, k, i1, i2, res.

2. Потом был создан сегмент кода CODE, в котором прописана сама программа. При помощи директивы ASSUME были указаны сегментный регистр CS, который указывает на сегмент кода, DS - сегмент данных, и SS - сегмент стека.

3.В мейне прописываются необходимые вещи для нормальной работы любой программы, такие как сохранение адреса начала PSP в стеке, загрузка сегментного регистра данных и т.д.

4. Затем с метки f1_f2 анализируются значения a и b. Если a>b то выполняется блок программы ответственный за функции f1 и f2 для a>b, который вызывается за счет jg a_more_b - сравнения знаковых чисел и уход по указанной метке. Иначе выполняется блок a_less_b для функций f1 и f2 при a<=b, в этих частях программы происходят арифметические расчеты по схеме, данной для моего варианта. Завершается этот блок на метке f1_f2_end, которая присваивает i1 значение f1(i), для i2 аналогично f2(i).

5. В блоке с меткой f3 анализируется значение k и выполняется функция f3 в соответствии со значением k.

6. В блоках k_more_0 и k_less_0 вычисляется значение функции f3 для значений k больше и меньше 0 соответственно

7. f3_end выполняет выход из программы

8. В конце указывается директива END Main, которая говорит компилятору с какого модуля программы начинать работу

Текст исходного файла программы lb3.asm

Текст исходной программы lb3.asm см. в приложении А.

Текст файла диагностического lb3.lst

Текст диагностического файла lb3.lst см. в приложении В.

Тестирование

Табл.1. Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарий

1	a = 5; b = -2; i = 2; k = 0	i1 = 000B (11);
		i2 = FFEC (-20);
		res = 000B (11);
			Успешное завершение программы

2	a = 5; b = 2; i = -2; k = 1	i1 = 0013 (19);
		i2 = 0004 (4);
		res =0013 (19);
			Успешное завершение программы

3	a = -5; b = 2; i = -2; k = -1	i1 = FFFE (-2);
		i2 = 0012 (18);
		res = 0014 (20);
			Успешное завершение программы

Выводы по работе

В ходе выполнения лабораторной работы был получен навык работы с механизмом ветвления на ассемблере, изучена обработка целочисленных значений.

Приложение А

Текст исходного файла lb3

```
AStack SEGMENT STACK
    DW 32 DUP(?)
AStack ENDS

DATA SEGMENT
    a Dw 5
    b Dw 2
    i Dw -2
    k Dw 1
    i1 Dw 0
    i2 Dw 0
    res Dw 0
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA, SS:AStack

Main PROC FAR
    push ds
    sub ax, ax
    push ax
    mov ax, DATA
    mov ds, ax
```

```
f1_f2:
    mov bx, i
    shl bx, 1; bx = 2i
    mov cx, i
    add cx, bx ; cx = 3i

    mov ax, a
    cmp ax, b
    jg a_more_b

a_less_b:
    mov bx, cx ; bx = cx = 3i
    neg cx ; cx = -3i
    add bx, 4h ; bx = 3i + 4
    add cx, 0Ch ; cx = -3i + 12
    jmp f1_f2_end

a_more_b:
    shl cx, 1; cx = 6i
    add cx, 8h ; cx = 6i + 8
    neg bx ; bx = -2i
    neg cx ; cx = -(6i + 8)
    add bx, 0Fh ; bx = -2i + 15

f1_f2_end:
    mov i1, bx ; i1 = f1(i)
    mov i2, cx ; i2 = f2(i)
```

```
f3:
    mov ax, k
    cmp ax, 0h
    jge k_more_0
k_less_0_1:
    mov ax, i1
    cmp ax, 0h
    jge k_less_0_2
    neg ax
k_less_0_2:
    mov bx, i2
    cmp bx, 0h
    jge k_less_0_3
    neg bx
k_less_0_3:
    add ax, bx
    cmp ax, 0h
    jmp f3_end

k_more_0:
    mov ax, i1
    cmp ax, 0h
    jge abs_i1
    neg ax
abs_i1:
    cmp ax, 6h
```

```
    jge f3_end
    mov ax, 6h

f3_end:
    mov res, ax
    ret
Main ENDP
CODE ENDS
    END Main
```

Приложение В

Текст диагностического файла lb3

```
Microsoft      (R)      Macro      Assembler      Version      5.10
10/30/22 23:40:3
                                                                Page
1-1


       1 0000                  AStack SEGMENT STACK
       2 0000  0020[                   DW 32 DUP(?)
       3     ????
       4             ]
       5
       6 0040                  AStack ENDS
       7
       8 0000                  DATA SEGMENT
       9 0000  0005                a Dw 5
      10 0002  0002                b Dw 2
      11 0004  FFFE                i Dw -2
      12 0006  0001                k Dw 1
      13 0008  0000                i1 Dw 0
      14 000A  0000                i2 Dw 0
      15 000C  0000                res Dw 0
      16 000E                 DATA ENDS
      17
```

```
 18 0000                CODE SEGMENT
 19                    ASSUME CS:CODE, DS:DATA, SS:AStack
 20
 21 0000                Main PROC FAR
 22 0000  1E                push ds
 23 0001  2B C0                  sub ax, ax
 24 0003  50                push ax
 25 0004  B8 ---- R         mov ax, DATA
 26 0007  8E D8                  mov ds, ax
 27
 28 0009                f1_f2:
 29 0009  8B 1E 0004 R           mov bx, i
 30 000D  D1 E3                  shl bx, 1; bx = 2i
 31 000F  8B 0E 0004 R           mov cx, i
 32 0013  03 CB                  add cx, bx ; cx = 3i
 33
 34 0015  A1 0000 R         mov ax, a
 35 0018  3B 06 0002 R           cmp ax, b
 36 001C  7F 0D                  jg a_more_b
 37
 38 001E                a_less_b:
 39 001E  8B D9                  mov bx, cx ; bx = cx = 3i
 40 0020  F7 D9                  neg cx ; cx = -3i
 41 0022  83 C3 04               add bx, 4h ; bx = 3i + 4
 42 0025  83 C1 0C               add cx, 0Ch ; cx = -3i + 12
 43 0028  EB 0D 90               jmp f1_f2_end
 44
 45 002B                a_more_b:
 46 002B  D1 E1                  shl cx, 1; cx = 6i
 47 002D  83 C1 08               add cx, 8h ; cx = 6i + 8
 48 0030  F7 DB                  neg bx ; bx = -2i
 49 0032  F7 D9                  neg cx ; cx = -(6i + 8)
 50 0034  83 C3 0F               add bx, 0Fh ; bx = -2i + 15
 51
 52 0037                f1_f2_end:
 53 0037  89 1E 0008 R           mov i1, bx ; i1 = f1(i)
 54 003B  89 0E 000A R           mov i2, cx ; i2 = f2(i)
Microsoft      (R)      Macro      Assembler      Version      5.10
10/30/22 23:40:3
                                                              Page
1-2


 55
```

```
      56 003F                  f3:
      57 003F  A1 0006 R           mov ax, k
      58 0042  3D 0000                  cmp ax, 0h
      59 0045  7D 1D                    jge k_more_0
      60 0047                  k_less_0_1:
      61 0047  A1 0008 R           mov ax, i1
      62 004A  3D 0000                  cmp ax, 0h
      63 004D  7D 02                    jge k_less_0_2
      64 004F  F7 D8                    neg ax
      65 0051                  k_less_0_2:
      66 0051  8B 1E 000A R             mov bx, i2
      67 0055  83 FB 00                 cmp bx, 0h
      68 0058  7D 02                    jge k_less_0_3
      69 005A  F7 DB                    neg bx
      70 005C                  k_less_0_3:
      71 005C  03 C3                    add ax, bx
      72 005E  3D 0000                  cmp ax, 0h
      73 0061  EB 13 90                 jmp f3_end
      74
      75 0064                  k_more_0:
      76 0064  A1 0008 R           mov ax, i1
      77 0067  3D 0000                  cmp ax, 0h
      78 006A  7D 02                    jge abs_i1
      79 006C  F7 D8                    neg ax
      80 006E                  abs_i1:
      81 006E  3D 0006                  cmp ax, 6h
      82 0071  7D 03                    jge f3_end
      83 0073  B8 0006                  mov ax, 6h
      84
      85 0076                  f3_end:
      86 0076  A3 000C R           mov res, ax
      87 0079  CB                  ret
      88 007A                  Main ENDP
      89 007A                  CODE ENDS
      90                       END Main
Microsoft      (R)      Macro      Assembler      Version      5.10
10/30/22 23:40:3

Symbols-1


Segments and Groups:
```

```
                N a m e                 Length     Align    Combine
Class

ASTACK . . . . . .   0040 PARA STACK
CODE . . . . . .   007A PARA NONE
DATA . . . . . .   000E PARA NONE

Symbols:

                N a m e                 Type  Value     Attr

A  . . . . . .   L WORD    0000 DATA
ABS_I1 . . . . . .   L NEAR    006E CODE
A_LESS_B . . . . . .   L NEAR    001E CODE
A_MORE_B . . . . . .   L NEAR    002B CODE

B  . . . . . .   L WORD    0002 DATA

F1_F2  . . . . . .   L NEAR    0009 CODE
F1_F2_END  . . . . . .   L NEAR    0037 CODE
F3 . . . . . .   L NEAR    003F CODE
F3_END . . . . . .   L NEAR    0076 CODE

I  . . . . . .   L WORD    0004 DATA
I1 . . . . . .   L WORD    0008 DATA
I2 . . . . . .   L WORD    000A DATA

K  . . . . . .   L WORD    0006 DATA
K_LESS_0_1 . . . . . .   L NEAR    0047 CODE
K_LESS_0_2 . . . . . .   L NEAR    0051 CODE
K_LESS_0_3 . . . . . .   L NEAR    005C CODE
K_MORE_0 . . . . . .   L NEAR    0064 CODE

MAIN . . . . . .   F PROC    0000 CODE Length     =
007A

RES  . . . . . .   L WORD    000C DATA

@CPU . . . . . .   TEXT  0101h
@FILENAME  . . . . . .   TEXT  lb3
@VERSION . . . . . .   TEXT  510


     87 Source  Lines
```

```
   87 Total   Lines
   27 Symbols

47490 + 459770 Bytes symbol space free

    0 Warning Errors
    0 Severe  Errors
```