**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

|  |  |  |
| --- | --- | --- |
| Студент гр. 0383 |  | Сабанов П.А. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2021

**Цель работы.**

Написать программу, реализующую ветвление в зависимости от условий и работу с числами.

**Текст задания.**

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

а) 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 работы.**

Были написаны необходимые функции f1, f2 и f3. Помимо них были реализованы функции abs и max. abs вычисляет абсолютное значение ax и кладёт его в ax, max вычисляет максимальное значение из ax и dx и кладёт его в ax.

Пусть a = 1, b = 2, c = 3, k = 4.

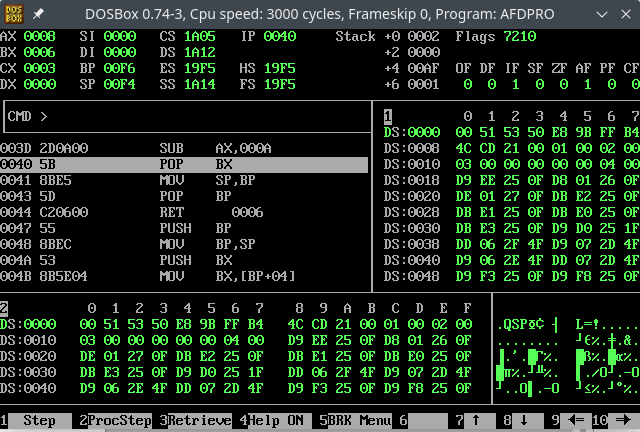
Произведём вычисления:

1) a < b => f1 = 6\*i - 10 = 8;

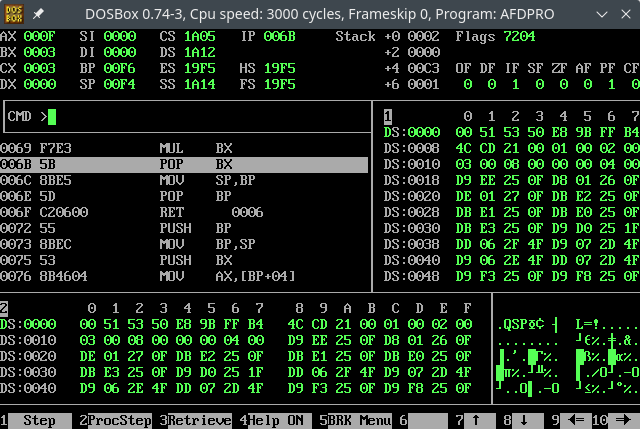
2) a < b => f2 = 3\*(i + 2) = 15;

3) k > 0 => f3 = max(6, |8|) = 8.

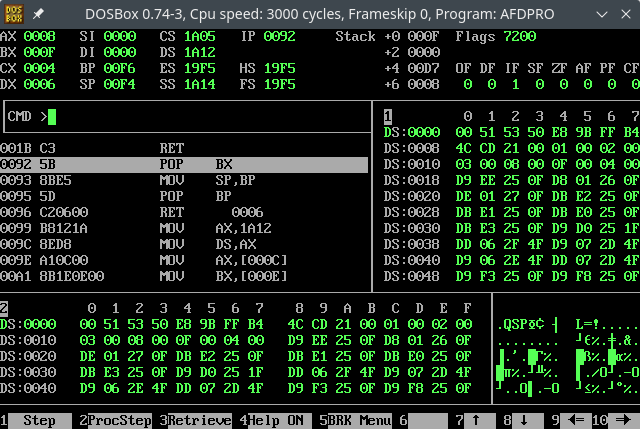
Запустим программу с помощью отладчика afdpro:



Как видно, после работы первой функции в регистре ax находится значение 8.



После работы второй функции в регистре ax находится значение 15.



После работы третьей функции в переменной ax находится значение 8.

**Выводы.**

Была написана программа, реализующая ветвление в зависимости от условий и работу с числами.

**ПРИЛОЖЕНИЕ А**

**Исходный код программы**

DOSSEG

.MODEL SMALL

.STACK 100h

.DATA

a DW 1

b DW 2

i DW 3

i1 DW 0

i2 DW 0

k DW 4

.CODE

abs PROC NEAR

@abs:

neg ax

js @abs

ret

abs ENDP

max PROC NEAR

cmp ax, dx

jnl max\_exit

mov ax, dx

max\_exit:

ret

max ENDP

Main PROC FAR

mov ax, @data

mov ds, ax

;f1(int a, int b, int i) {

; if (a > b)

; return -(4\*i + 3);

; return 6\*i - 10;

;}

;return in ax

mov ax, i

mov bx, a

mov cx, b

cmp bx, cx

jng f1\_2

mov bx, 4

mul bx

add ax, 3

neg ax

jmp f1\_exit

f1\_2:

mov bx, 6

mul bx

sub ax, 10

f1\_exit:

mov i1, ax

;f2(int a, int b, int i) {

; int c = 3\*(i+2);

; if (a > b)

; return -2\*c;

; return c;

;}

;return in ax

mov ax, i

add ax, 2

mov bx, 3

mul bx

mov bx, a

mov cx, b

cmp bx, cx

jg f2\_exit

mov bx, -2

imul bx

f2\_exit:

mov i2, ax

;f3(int i1, int i2, int k) {

; if (k < 0)

; return |i1| + |i2|;

; return max(|i1|, 6);

;}

;return in ax

mov ax, i1

call abs

cmp word ptr k, 0

jnl f3\_2

mov bx, ax

mov ax, i2

call abs

add ax, bx

jmp f3\_exit

f3\_2:

mov bx, 6

cmp ax, bx

jnl f3\_exit

; |i1| < 6

mov ax, bx

f3\_exit:

; f3(i1,i2,k) in ax

mov ah, 4ch

int 21h

Main ENDP

END Main

**ПРИЛОЖЕНИЕ Б**

**Листинг компиляции программы**

#Microsoft (R) Macro Assembler Version 5.10 11/24/21 20:48:1

Page 1-1

DOSSEG

.MODEL SMALL

.STACK 100h

.DATA

0000 0001 a DW 1

0002 0002 b DW 2

0004 0003 i DW 3

0006 0000 i1 DW 0

0008 0000 i2 DW 0

000A 0004 k DW 4

.CODE

0000 abs PROC NEAR

0000 @abs:

0000 F7 D8 neg ax

0002 78 FC js @abs

0004 C3 ret

0005 abs ENDP

0005 max PROC NEAR

0005 3B C2 cmp ax, dx

0007 7D 02 jnl max\_exit

0009 8B C2 mov ax, dx

000B max\_exit:

000B C3 ret

000C max ENDP

000C Main PROC FAR

000C B8 ---- R mov ax, @data

000F 8E D8 mov ds, ax

;f1(int a, int b, int i) {

; if (a > b)

; return -(4\*i + 3);

; return 6\*i - 10;

;}

;return in ax

0011 A1 0004 R mov ax, i

0014 8B 1E 0000 R mov bx, a

0018 8B 0E 0002 R mov cx, b

001C 3B D9 cmp bx, cx

001E 7E 0D jng f1\_2

0020 BB 0004 mov bx, 4

0023 F7 E3 mul bx

0025 05 0003 add ax, 3

0028 F7 D8 neg ax

#Microsoft (R) Macro Assembler Version 5.10 11/24/21 20:48:1

Page 1-2

002A EB 09 90 jmp f1\_exit

002D f1\_2:

002D BB 0006 mov bx, 6

0030 F7 E3 mul bx

0032 2D 000A sub ax, 10

0035 f1\_exit:

0035 A3 0006 R mov i1, ax

;f2(int a, int b, int i) {

; int c = 3\*(i+2);

; if (a > b)

; return -2\*c;

; return c;

;}

;return in ax

0038 A1 0004 R mov ax, i

003B 05 0002 add ax, 2

003E BB 0003 mov bx, 3

0041 F7 E3 mul bx

0043 8B 1E 0000 R mov bx, a

0047 8B 0E 0002 R mov cx, b

004B 3B D9 cmp bx, cx

004D 7F 05 jg f2\_exit

004F BB FFFE mov bx, -2

0052 F7 EB imul bx

0054 f2\_exit:

0054 A3 0008 R mov i2, ax

;f3(int i1, int i2, int k) {

; if (k < 0)

; return |i1| + |i2|;

; return max(|i1|, 6);

;}

;return in ax

0057 A1 0006 R mov ax, i1

005A E8 0000 R call abs

005D 83 3E 000A R 00 cmp word ptr k, 0

0062 7D 0D jnl f3\_2

0064 8B D8 mov bx, ax

0066 A1 0008 R mov ax, i2

0069 E8 0000 R call abs

006C 03 C3 add ax, bx

006E EB 0A 90 jmp f3\_exit

0071 f3\_2:

#Microsoft (R) Macro Assembler Version 5.10 11/24/21 20:48:1

Page 1-3

0071 BB 0006 mov bx, 6

0074 3B C3 cmp ax, bx

0076 7D 02 jnl f3\_exit

; |i1| < 6

0078 8B C3 mov ax, bx

007A f3\_exit:

; f3(i1,i2,k) in ax

007A B4 4C mov ah, 4ch

007C CD 21 int 21h

007E Main ENDP

END Main

#Microsoft (R) Macro Assembler Version 5.10 11/24/21 20:48:1

Symbols-1

Segments and Groups:

N a m e Length Align Combine Class

DGROUP . . . . . . . . . . . . . GROUP

\_DATA . . . . . . . . . . . . 000C WORD PUBLIC 'DATA'

STACK . . . . . . . . . . . . 0100 PARA STACK 'STACK'

\_TEXT . . . . . . . . . . . . . 007E WORD PUBLIC 'CODE'

Symbols:

N a m e Type Value Attr

A . . . . . . . . . . . . . . . L WORD 0000 \_DATA

ABS . . . . . . . . . . . . . . N PROC 0000 \_TEXT Length = 0005

B . . . . . . . . . . . . . . . L WORD 0002 \_DATA

F1\_2 . . . . . . . . . . . . . . L NEAR 002D \_TEXT

F1\_EXIT . . . . . . . . . . . . L NEAR 0035 \_TEXT

F2\_EXIT . . . . . . . . . . . . L NEAR 0054 \_TEXT

F3\_2 . . . . . . . . . . . . . . L NEAR 0071 \_TEXT

F3\_EXIT . . . . . . . . . . . . L NEAR 007A \_TEXT

I . . . . . . . . . . . . . . . L WORD 0004 \_DATA

I1 . . . . . . . . . . . . . . . L WORD 0006 \_DATA

I2 . . . . . . . . . . . . . . . L WORD 0008 \_DATA

K . . . . . . . . . . . . . . . L WORD 000A \_DATA

MAIN . . . . . . . . . . . . . . F PROC 000C \_TEXT Length = 0072

MAX . . . . . . . . . . . . . . N PROC 0005 \_TEXT Length = 0007

MAX\_EXIT . . . . . . . . . . . . L NEAR 000B \_TEXT

@ABS . . . . . . . . . . . . . . L NEAR 0000 \_TEXT

@CODE . . . . . . . . . . . . . TEXT \_TEXT

@CODESIZE . . . . . . . . . . . TEXT 0

@CPU . . . . . . . . . . . . . . TEXT 0101h

@DATASIZE . . . . . . . . . . . TEXT 0

@FILENAME . . . . . . . . . . . TEXT lab3

@VERSION . . . . . . . . . . . . TEXT 510

#Microsoft (R) Macro Assembler Version 5.10 11/24/21 20:48:1

Symbols-2

123 Source Lines

123 Total Lines

33 Symbols

47914 + 457296 Bytes symbol space free

0 Warning Errors

0 Severe Errors

**ПРИЛОЖЕНИЕ В**

**Карта памяти программы**

Start Stop Length Name Class

00000H 0008DH 0008EH \_TEXT CODE

0008EH 00099H 0000CH \_DATA DATA

000A0H 0019FH 00100H STACK STACK

Origin Group

0008:0 DGROUP

Program entry point at 0000:001C