

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: Изучение организации ветвлений в программах на языке
Assembler.

Студент гр. 9382

Сорокумов С.В.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Изучение организации ветвлений и управляющих структур на языке Assembler.

Постановка задачи.

Разработать на языке Ассемблера IBM PC программу, которая по заданным целым значениям a, b, i, k , размером 1 слово, вычисляет:

Значения $i1=f1(a,b,i)$ и $i2=f2(a,b,i)$,

Значения $res=f3(i1,i2,k)$,

где функции $f1$ и $f2$ определяются из таблицы 1, а $f3$ - из таблицы 2 по цифрам шифра индивидуального задания.

Значения a, b, i, k являются исходными данными, которые должны быть выбраны самостоятельно и задаваться в процессе исполнения программы в режиме отладки. При этом следует рассмотреть все возможные комбинации параметров a, b, i, k , позволяющие проверить различные маршруты выполнения программы.

Вариант 10:

1. $\{-(4i+3), \text{ при } a>b \text{ } 6i-10, \text{ при } a\leq b$
2. $\{20-4i, \text{ при } a>b \text{ } -(6i-6), \text{ при } a\leq b$
3. $\{i1-|i2|, \text{ при } k<0 \text{ } \max(7,i2), \text{ при } k\geq 0$

Выполнение работы.

1. Используемые операции.

Для выполнения вычислений использовались операции:

- a. **ADD** – для сложения двух данных и записи в регистр
- b. **SUB** – для вычитания двух данных и записи в регистр
- c. **NEG** – для получения противоположного значения данных в регистре

Для выполнения условных переходов использовались следующие операции:

- e. **CMR** - сравнение двух чисел

2. Тестирование программы.

№ теста	Исходные данные	Ожидаемый результат	Полученный результат	Корректность работы программы
1	$a=0002_{16} = 2_{10}$ $b=0004_{16} = 4_{10}$ $i=0001_{16} = 1_{10}$ $k=0003_{16} = 3_{10}$	$i1=0009_{16} = 9_{10}$ $i2=FFF4_{16} = -12_{10}$ $res=0015_{16} = 21_{10}$	$i1=0009_{16} = 9_{10}$ $i2=FFF4_{16} = -12_{10}$ $res=0015_{16} = 21_{10}$	+

2	$a=0002_{16} = 4_{10}$ $b=0004_{16} = 2_{10}$ $i=0005_{16} = 1_{10}$ $k=0002_{16} = 3_{10}$	$i1=FFFE_{16} = -2_{10}$ $i2=0010_{16} = 16_{10}$ $res=0006_{16} = 6_{10}$	$i1=FFFE_{16} = -2_{10}$ $i2=0010_{16} = 16_{10}$ $res=0006_{16} = 6_{10}$	+
3	$a=0002_{16} = 4_{10}$ $b=0004_{16} = 2_{10}$ $i=0005_{16} = 1_{10}$ $k=FFFF_{16} = -1_{10}$	$i1=FFFE_{16} = -2_{10}$ $i2=0010_{16} = 16_{10}$ $res=0012_{16} = 18_{10}$	$i1=FFFE_{16} = -2_{10}$ $i2=0010_{16} = 16_{10}$ $res=0012_{16} = 18_{10}$	+

Выводы.

В ходе данной лабораторной работы была изучена организация ветвления, а также операция сравнения, реализация меток и переход по данным меткам на языке Ассемблера. В ходе разработки программы была применена минимизация кода, результаты вычислений контролировались в режиме отладки.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

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

```
DATA SEGMENT
A      DW 4
B      DW 2
I      DW 1
K      DW -1
I1     DW ?
I2     DW ?
RES DW ?
DATA ENDS
```

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

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

```
f:
mov ax, A
mov bx, B
cmp ax, bx
jg f1
jmp f1_1
```

```
f1:
mov al, I; al = i
mov bl, 2; bl = 2
mul bl; ax = al*ab
mov si, ax; bx = ax
mov al, I; al = i
mov bl, 8; bl = 8
mul bl; ax = lb*al
sub si, ax; bx = bx-ax
add si, 4; bx += 4
mov I1, si
jmp f_2
```

```
f1_1:
mov al, I
mov bl, 3
mul bl
add ax, 6
mov I1, ax
jmp f_2
```

```
f_2:
mov ax, A
mov bx, B
cmp ax, bx
jg f2
jmp f2_2
```

```
f2:
mov al, I; al = i
mov bl, 4; bl = 2
mul bl; ax = al*ab
mov si, ax; bx = ax
mov al, I; al = i
mov bl, 8; bl = 8
mul bl; ax = lb*al
sub si, ax; bx = bx-ax
add si, 20
mov I2, si
jmp f_3
```

```
f2_2:
mov al, I; al = i
mov bl, 2; bl = 2
mul bl; ax = al*ab
mov si, ax; bx = ax
mov al, I
mov bl, 8; bl = 8
mul bl; ax = lb*al
sub si, ax; bx = bx-ax
sub si, 6
mov I2, si
jmp f_3
```

```
f_3:
mov ax, I1
mov bx, I2
mov si, 0
cmp si, K
jg f3
jmp f3_2
```

```
f3:
mov cx, 0
cmp cx, ax
jg NEG_AX
cmp cx, bx
jg NEG_BX
add ax, bx
mov RES, ax
jmp f_end
```

```
f3_2:
cmp ax, 6
jg SET_RES_I
mov si, 6
cmp si, ax
jg SET_RES_6
```

```
NEG_AX:
neg ax
jmp f3
```

```
NEG_BX:
neg bx
jmp f3
```

```
SET_RES_6:
mov res, 6
jmp f_end
```

```
SET_RES_I:
mov ax, I1
mov res, ax
```

```
f_end:
mov ah, 4ch          ;завершаем программу
int 21h
```

```
Main      ENDP
CODE      ENDS
END Main          ;ENDS CODE
```

ПРИЛОЖЕНИЕ Б ФАЙЛ ЛИСТИНГА

Microsoft (R) Macro Assembler Version 5.10
23:03:4

10/14/20

Page

1-1

```
0000          AStack SEGMENT STACK
0000 0020[          DW 32 DUP(?)
    ????
    ]

0040          AStack ENDS

0000          DATA SEGMENT
0000 0004          A      DW 4
0002 0002          B      DW 2
0004 0001          I      DW 1
0006 FFFF          K      DW -1
0008 0000          I1     DW ?
000A 0000          I2     DW ?
000C 0000          RES DW ?
000E          DATA ENDS

0000          CODE      SEGMENT
          ASSUME CS:CODE, DS:DATA, SS:AStack

0000          Main PROC FAR
0000 B8 ---- R      mov ax, DATA
0003 8E D8          mov ds, ax
0005          f:
0005 A1 0000 R      mov ax, A
0008 8B 1E 0002 R   mov bx, B
000C 3B C3          cmp ax, bx
000E 7F 03          jg f1
0010 EB 1D 90       jmp f1_1

0013          f1:
0013 A0 0004 R      mov al, I; al = i
LAB3.ASM(29): warning A4031: Operand types must match
0016 B3 02          mov bl, 2; bl = 2
0018 F6 E3          mul bl; ax = al*ab
001A 8B F0          mov si, ax; bx = ax
001C A0 0004 R      mov al, I; al = i
LAB3.ASM(33): warning A4031: Operand types must match
001F B3 08          mov bl, 8; bl = 8
0021 F6 E3          mul bl; ax = lb*al
0023 2B F0          sub si, ax; bx = bx-ax
0025 83 C6 04       add si, 4; bx += 4
0028 89 36 0008 R   mov I1, si
```

```

002C EB 11 90                jmp f_2

002F                        f1_1:
002F A0 0004 R              mov al, I
LAB3.ASM(42): warning A4031: Operand types must match
0032 B3 03                mov bl, 3
0034 F6 E3                mul bl
0036 05 0006              add ax, 6
0039 A3 0008 R            mov I1, ax
003C EB 01 90                jmp f_2

```

```

003F                        f_2:
003F A1 0000 R            mov ax, A
0042 8B 1E 0002 R        mov bx, B
Microsoft (R) Macro Assembler Version 5.10
23:03:4

```

10/14/20

Page

1-2

```

0046 3B C3                cmp ax, bx
0048 7F 03                jg f2
004A EB 1D 90                jmp f2_2

004D                        f2:
004D A0 0004 R            mov al, I; al = i
LAB3.ASM(57): warning A4031: Operand types must match
0050 B3 04                mov bl, 4; bl = 2
0052 F6 E3                mul bl; ax = al*ab
0054 8B F0                mov si, ax; bx = ax
0056 A0 0004 R            mov al, I; al = i
LAB3.ASM(61): warning A4031: Operand types must match
0059 B3 08                mov bl, 8; bl = 8
005B F6 E3                mul bl; ax = lb*al
005D 2B F0                sub si, ax; bx = bx-ax
005F 83 C6 14            add si, 20
0062 89 36 000A R        mov I2, si
0066 EB 1D 90                jmp f_3

0069                        f2_2:
0069 A0 0004 R            mov al, I; al = i
LAB3.ASM(70): warning A4031: Operand types must match
006C B3 02                mov bl, 2; bl = 2
006E F6 E3                mul bl; ax = al*ab
0070 8B F0                mov si, ax; bx = ax
0072 A0 0004 R            mov al, I
LAB3.ASM(74): warning A4031: Operand types must match
0075 B3 08                mov bl, 8; bl = 8
0077 F6 E3                mul bl; ax = lb*al
0079 2B F0                sub si, ax; bx = bx-ax
007B 83 EE 06            sub si, 6

```



```

007E 89 36 000A R      mov I2, si
0082 EB 01 90          jmp f_3

```

```

0085                      f_3:
0085 A1 0008 R          mov ax, I1
0088 8B 1E 000A R      mov bx, I2
008C BE 0000          mov si, 0
008F 3B 36 0006 R      cmp si, K
0093 7F 03            jg f3
0095 EB 14 90          jmp f3_2

```

```

0098                      f3:
0098 B9 0000          mov cx, 0
009B 3B C8          cmp cx, ax
009D 7F 18          jg NEG_AX
009F 3B CB          cmp cx, bx
00A1 7F 18          jg NEG_BX
00A3 03 C3          add ax, bx
00A5 A3 000C R      mov RES, ax
00A8 EB 24 90          jmp f_end

```

```

00AB                      f3_2:
00AB 3D 0006          cmp ax, 6
00AE 7F 18          jg SET_RES_I
00B0 BE 0006          mov si, 6
00B3 3B F0          cmp si, ax

```

Microsoft (R) Macro Assembler Version 5.10
23:03:4

10/14/20

Page

1-3

```

00B5 7F 08          jg SET_RES_6

```

```

00B7                      NEG_AX:
00B7 F7 D8          neg ax
00B9 EB DD          jmp f3

```

```

00BB                      NEG_BX:
00BB F7 DB          neg bx
00BD EB D9          jmp f3

```

```

00BF                      SET_RES_6:
00BF C7 06 000C R 0006      mov res, 6
00C5 EB 07 90          jmp f_end

```

```

00C8                      SET_RES_I:
00C8 A1 0008 R          mov ax, I1
00CB A3 000C R          mov res, ax

```

```

00CE          f_end:
00CE  B4 4C          mov  ah, 4ch          ;завершаем
пор
          pammy
00D0  CD 21          int  21h

00D2          Main      ENDP
00D2          CODE      ENDS
          END Main          ;ENDS CODE
Microsoft (R) Macro Assembler Version 5.10          10/14/20
23:03:4

```

Symbols-1

Segments and Groups:

	N a m e	Length	Align	Combine Class
ASTACK	0040	PARA	STACK
CODE	00D2	PARA	NONE
DATA	000E	PARA	NONE

Symbols:

	N a m e	Type	Value	Attr
A	L WORD	0000	DATA
B	L WORD	0002	DATA
F	L NEAR	0005	CODE
F1	L NEAR	0013	CODE
F1_1	L NEAR	002F	CODE
F2	L NEAR	004D	CODE
F2_2	L NEAR	0069	CODE
F3	L NEAR	0098	CODE
F3_2	L NEAR	00AB	CODE
F_2	L NEAR	003F	CODE
F_3	L NEAR	0085	CODE
F_END	L NEAR	00CE	CODE
I	L WORD	0004	DATA
I1	L WORD	0008	DATA
I2	L WORD	000A	DATA
K	L WORD	0006	DATA

MAIN	F PROC	0000	CODE	Length
= 00D2				
NEG_AX	L NEAR	00B7	CODE	
NEG_BX	L NEAR	00BB	CODE	
RES	L WORD	000C	DATA	
SET_RES_6	L NEAR	00BF	CODE	
SET_RES_I	L NEAR	00C8	CODE	
@CPU	TEXT	0101h		
@FILENAME	TEXT	LAB3		
@VERSION	TEXT	510		

Microsoft (R) Macro Assembler Version 5.10
23:03:4

10/14/20

Symbols-2

132 Source Lines
132 Total Lines
30 Symbols

48016 + 459244 Bytes symbol space free

7 Warning Errors
0 Severe Errors