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		Демчук П. Д.
Преподаватель		Ефремов М. А.

Санкт-Петербург

2022

Цель работы.

Изучить представление и обработку целых чисел. Научиться организации ветвящихся процессов. Применить полученные знания на практике.

Задание.

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);

Вариант 5:

```
       / 15-2*i, при a>b
f1 = <
       \ 3*i+4, при a<=b

       / 2*(i+1)-4, при a>b
f2 = <
       \ 5-3*(i+1), при a<=b

       / min(|i1|, 6), при k=0
f3 = <
       \ |i1 - i2|, при k!=0
```

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.

Замечания:

1) при разработке программы нельзя использовать фрагменты, представленные на ЯВУ, в частности, для ввода-вывода данных. Исходные данные должны вводиться, а результаты контролироваться в режиме отладки;

2) при вычислении функций f1 и f2 вместо операции умножения следует использовать арифметический сдвиг и, возможно, сложение;

3) при вычислении функций f1 и f2 нельзя использовать процедуры;

4) при разработке программы следует минимизировать длину кода, для чего, если надо, следует преобразовать исходные выражения для вычисления функций.

Выполнение работы.

Созданы три сегмента AStack, DATA, CODE - сегмент стека, сегмент кода и сегмент данных соответственно. В сегменте данных объявлены переменные a, b, i, k, i1, i2, res.

В сегменте кода находится процедура Main, в которой вычисляются значения данных в условии функций. Сначала в регистр ax записывается значение 2i. Далее с помощью функции cmp происходит сравнение значений переменных a и b. Команда jle проверяет условие a<=b, при его выполнении производится переход по указанному адресу.

Для вычисления функции f3 сначала вычисляется модуль i1, далее производится сравнение k с нулем. При помощи команды jne (выполнение условия k!=0), производится переход по указанному адресу.

Для совершения безусловных переходов в программе используется команда jmp.

В переменную res записывается значение регистра ax, который содержит значение функции f3.

Тестирование.

№	a	b	i	k	i1	i2	res
1	5	-2	2	1	000B	0002	000D
2	2	2	1	2	0007	FFFF	0008
3	-3	-2	-1	0	0001	0005	0001

Выводы.

В ходе лабораторной работы были изучены представление и обработка целых чисел, организация ветвящихся процессов на языке Ассемблер.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lr3.asm

```
AStack  SEGMENT  STACK
    DW 12 DUP(?)
AStack    ENDS

DATA    SEGMENT
    a DW -3
    b DW -2
    i DW -1
    k DW 0
    i1 DW ?
    i2 DW ?
    res DW ?
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
f12:
    mov ax,i
```

```
    shl ax, 1  ;ax=2i
         mov cx, a
         cmp cx, b
         jle f12_case2   ;a<=b
f12_case1:
    ;f1=15-2i
    mov i1,15
    sub i1,ax
    mov dx, i1

    ;f2=2(i+1)-4=2i-2
    mov i2,ax
    sub i2,2
    mov dx, i2
    jmp result
f12_case2:
        ;f1=3i+4
    add ax,i   ;ax=3i
    mov i1,ax
    add i1,4
    mov dx, i1

        ;f2=5-3(i+1)=2-3i
        mov i2,2
        sub i2,ax
        mov dx, i2
result:
    cmp i1,0
    jge f3_case1    ;i1>=0
```

```
    neg i1
f3_case1:
    cmp k,0
    jne abs_i2 ;k!=0
    cmp i1,6
    jge min_6  ;i1>=6
    mov ax,i1  ;min=i1
    jmp f3_end
min_6:
    mov ax,6   ;min=6
    jmp f3_end
abs_i2:
    cmp i2,0
    jge f3_case2    ;i2>=0
    neg i2
f3_case2:
    mov ax,i1
    add ax,i2
f3_end:
    mov res,ax
    ret

Main ENDP
CODE    ENDS
END Main
```

ПРИЛОЖЕНИЕ Б. ЛИСТИНГ ПРОГРАММЫ

Название файла: lr3.lst

```
    Microsoft      (R)      Macro      Assembler      Version      5.10
10/18/22 23:26:4
                                                                    Page
1-1


     0000                    AStack  SEGMENT  STACK
     0000  000C[                    DW 12 DUP(?)
           ????
                  ]

     0018                    AStack    ENDS

     0000                    DATA    SEGMENT
     0000  FFFD                    a DW -3
     0002  FFFE                    b DW -2
     0004  FFFF                    i DW -1
     0006  0000                    k DW 0
     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  1E                push  ds
0001  2B C0                    sub   ax,ax
0003  50                push  ax
0004  B8 ---- R                mov   ax,DATA
0007  8E D8                    mov   ds,ax
0009                    f12:
0009  A1 0004 R                mov ax,i
000C  D1 E0                    shl ax, 1  ;ax=2i
000E  8B 0E 0000 R             mov cx, a
0012  3B 0E 0002 R             cmp cx, b
0016  7E 1D                    jle f12_case2   ;a<=b
0018                    f12_case1:
                        ;f1=15-2i
0018  C7 06 0008 R 000F        mov i1,15
001E  29 06 0008 R             sub i1,ax
0022  8B 16 0008 R             mov dx, i1

                        ;f2=2(i+1)-4=2i-2
0026  A3 000A R                mov i2,ax
0029  83 2E 000A R 02               sub i2,2
002E  8B 16 000A R             mov dx, i2
0032  EB 1F 90                      jmp result
0035                    f12_case2:
                           ;f1=3i+4
0035  03 06 0004 R             add ax,i   ;ax=3i
0039  A3 0008 R                mov i1,ax
003C  83 06 0008 R 04               add i1,4
0041  8B 16 0008 R             mov dx, i1
```

```
                                ;f2=5-3(i+1)=2-3i
0045  C7 06 000A R 0002          mov i2,2
004B  29 06 000A R               sub i2,ax
```



```
004F  8B 16 000A R               mov dx, i2
0053                        result:
0053  83 3E 0008 R 00              cmp i1,0
0058  7D 04                   jge f3_case1    ;i1>=0
005A  F7 1E 0008 R            neg i1
005E                        f3_case1:
005E  83 3E 0006 R 00              cmp k,0
0063  75 13                   jne abs_i2 ;k!=0
0065  83 3E 0008 R 06              cmp i1,6
006A  7D 06                   jge min_6  ;i1>=6
006C  A1 0008 R               mov ax,i1  ;min=i1
006F  EB 19 90                     jmp f3_end
0072                        min_6:
0072  B8 0006                 mov ax,6   ;min=6
0075  EB 13 90                     jmp f3_end
0078                        abs_i2:
0078  83 3E 000A R 00              cmp i2,0
007D  7D 04                   jge f3_case2    ;i2>=0
007F  F7 1E 000A R            neg i2
0083                        f3_case2:
```

```
0083  A1 0008 R               mov ax,i1
0086  03 06 000A R            add ax,i2
008A                    f3_end:
008A  A3 000C R               mov res,ax
008D  CB                ret

008E                    Main  ENDP
008E                    CODE    ENDS
                      END Main
```

Microsoft (R) Macro Assembler Version 5.10 10/18/22 23:26:4

Symbols-1

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA	STACK	
CODE	008E	PARA	NONE	
DATA	000E	PARA	NONE	

Symbols:

N a m e	Type	Value	Attr
A	L WORD	0000	DATA
ABS_I2	L NEAR	0078	CODE

```
B  . . . . . .  L WORD  0002  DATA

F12  . . . . . .  L NEAR  0009  CODE
F12_CASE1  . . . . . .  L NEAR  0018  CODE
F12_CASE2  . . . . . .  L NEAR  0035  CODE
F3_CASE1 . . . . . .  L NEAR  005E  CODE
F3_CASE2 . . . . . .  L NEAR  0083  CODE
F3_END . . . . . .  L NEAR  008A  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
= 008E
MIN_6  . . . . . .  L NEAR  0072  CODE

RES  . . . . . .  L WORD  000C  DATA
RESULT . . . . . .  L NEAR  0053  CODE

@CPU . . . . . .  TEXT  0101h
@FILENAME  . . . . . .  TEXT  lr3_comp
@VERSION . . . . . .  TEXT  510


     80 Source  Lines
     80 Total   Lines
     25 Symbols
```

```
47974 + 461333 Bytes symbol space free

    0 Warning Errors

    0 Severe  Errors
```