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 _____ Коротков А.В.
Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

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.

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.

Вариант 5:

$$i1 = f1 = \begin{cases} 15 - 2*i, & \text{при } a>b \\ 3*i + 4, & \text{при } a<=b \end{cases}$$

$$i2 = f6 = \begin{cases} 2*(i + 1) - 4, & \text{при } a>b \\ 5 - 3*(i-1), & \text{при } a<=b \end{cases}$$

$$res = f5 = \begin{cases} \min(|i1|, 6), & \text{при } k=0 \\ |i1|+|i2|, & \text{при } k/=0 \end{cases}$$

Выполнение работы.

В ходе выполнения лабораторной работы для реализации ветвящихся процессов были использованы команда cmp и команды условных перемещений к меткам: jle, je, jge. Числа, используемые в формулах заданий задаются напрямую в исходном коде. Для выполнения основных математических операций были использованы команды sub, add и shl (побитовый сдвиг), используемый для умножения на 2.

Результаты тестирования представлены в табл. 1.

Текст исходного файла программы см. в приложении А.

Текст файла листинга см. в приложении Б.

Таблица 1. Проверка работы программы.

№	Входные данные	Значение i1	Значение i2	Значение res	Комментарий
1	a = 10, b = 5, i = 5, k = 4	5	8	13	Результат верен
2	a = 5, b = 10, i = 5, k = 0	19	-13	6	Результат верен
3	a = 4, b = 9, i = 3, k = 0	13	-7	6	Результат верен
4	a = 4, b = 0, i = 0, k = 0	15	-2	6	Результат верен

Выводы.

В ходе выполнения данной лабораторной работы была изучена реализация ветвящихся процессов на языке Ассемблер, включающая в себя манипулирование целыми числами и условные переходы по меткам.

ПРИЛОЖЕНИЕ А

ТЕКСТЫ ИСХОДНЫХ ФАЙЛОВ ПРОГРАММ

Название файла: **lr3.asm**

```
; Стек  программы
AStack SEGMENT  STACK
            DW 12 DUP(?)
AStack  ENDS
;Данные программы
DATA SEGMENT
     ;Директивы описания данных
     a     DW 7
     b     DW 6
     i     DW 2
     k     DW 0
     i1    DW 0
     i2    DW 0
     temp     DW 0     ; temp

DATA       ENDS

; Код программы
CODE SEGMENT
            ASSUME CS:CODE, DS:DATA, SS:AStack

     ; Головная процедура
Main PROC  FAR
            push    DS
            sub     AX,AX
            push    AX
            mov     AX,DATA
            mov     DS,AX
            mov     CX, 0

            mov     cx, i
```

```
        mov     temp, cx
        add     temp, 1     ; temp = i+1
        shl     cx, 1       ; C = 2i
        mov     bx, b
        cmp     a, bx
        jle     fun16

; a > b
        mov     ax, cx
        mov     cx, 15
        sub     cx, ax
        mov     i1, cx
        mov     cx, temp
        add     cx, temp
        sub     cx, 4
        mov     i2, cx
        jmp     fin

; a <= b
fun16:
        add     cx, i
        add     cx, 4
        mov     i1, cx
        mov     ax, temp
        shl     ax, 1
        add     ax, temp
        mov     cx, 5
        sub     cx, ax
        mov     i2, cx

;f5
fin:
        mov     bx, k
        mov     ax, i1
```

```
        cmp     ax, 0
        jge     skip_i1     ; |i1|
        neg     ax
skip_i1:
        cmp     bx, 0
        je      zero
        mov     cx, i2
        cmp     cx, 0
        jge     skip_i2
        neg     cx
skip_i2:
        add     cx, ax
        jmp     result
zero:
        cmp     ax, 6
        jge     min
        mov     cx, ax    ; |i1| < 6
        jmp     result
min:
        mov     cx, 6     ; |i1| >= 6
        jmp     result
result:
        ret
Main ENDP
CODE      ENDS
END Main
```

ПРИЛОЖЕНИЕ Б

ТЕКСТЫ ФАЙЛОВ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ

Название файла: **lr3.lst**

```
Microsoft (R) Macro Assembler Version 5.10                  11/20/21
03:32:1

                                                             Page
1-1


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

 0018                 AStack  ENDS
                ;Данные программы
 0000                 DATA SEGMENT
                      ;Директивы описания д
                anных
 0000  0007                a     DW 7
 0002  0006                b     DW 6
 0004  0002                i     DW 2
 0006  0000                k     DW 0
 0008  0000                i1    DW 0
 000A  0000                i2    DW 0
 000C  0000                temp    DW 0    ; temp

 000E                 DATA      ENDS

                ; Код программы
 0000                 CODE SEGMENT
                             ASSUME CS:CODE, DS:DATA, SS:ASta
                ck
```

```
                        ; Головная процедура
 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  B9 0000                           mov    CX, 0

 000C  8B 0E 0004 R                      mov    cx, i
 0010  89 0E 000C R                      mov    temp, cx
 0014  83 06 000C R 01                   add    temp, 1   ; temp = i+1
 0019  D1 E1                        shl    cx, 1     ; C = 2i
 001B  8B 1E 0002 R                      mov    bx, b
 001F  39 1E 0000 R                      cmp    a, bx
 0023  7E 1D                        jle    fun16

                        ; a > b
 0025  8B C1                        mov    ax, cx
 0027  B9 000F                           mov    cx, 15
 002A  2B C8                        sub    cx, ax
 002C  89 0E 0008 R                      mov    i1, cx
 0030  8B 0E 000C R                      mov    cx, temp
 0034  03 0E 000C R                      add    cx, temp
 0038  83 E9 04                          sub    cx, 4
 003B  89 0E 000A R                      mov    i2, cx
 003F  EB 1E 90                          jmp    fin

Microsoft (R) Macro Assembler Version 5.10                  11/20/21
03:32:1

                                                             Page
1-2


                        ; a <= b
```

```
0042                        fun16:
0042  03 0E 0004 R                      add    cx, i
0046  83 C1 04                          add    cx, 4
0049  89 0E 0008 R                      mov    i1, cx
004D  A1 000C R                   mov    ax, temp
0050  D1 E0                        shl    ax, 1
0052  03 06 000C R                      add    ax, temp
0056  B9 0005                          mov    cx, 5
0059  2B C8                        sub    cx, ax
005B  89 0E 000A R                      mov    i2, cx

                          ;f5
005F                        fin:
005F  8B 1E 0006 R                      mov    bx, k
0063  A1 0008 R                   mov    ax, i1
0066  3D 0000                           cmp    ax, 0
0069  7D 02                        jge    skip_i1     ; |i1|
006B  F7 D8                        neg    ax
006D                        skip_i1:
006D  83 FB 00                          cmp    bx, 0
0070  74 10                        je     zero
0072  8B 0E 000A R                      mov    cx, i2
0076  83 F9 00                          cmp    cx, 0
0079  7D 02                        jge    skip_i2
007B  F7 D9                        neg    cx
007D                        skip_i2:
007D  03 C8                        add    cx, ax
007F  EB 11 90                          jmp    result
0082                        zero:
0082  3D 0006                           cmp    ax, 6
0085  7D 05                        jge    min
0087  8B C8                        mov    cx, ax    ; |i1| < 6
0089  EB 07 90                          jmp    result
008C                        min:
```

```
 008C  B9 0006                          mov    cx, 6     ; |i1| >= 6
 008F  EB 01 90                         jmp    result
 0092                        result:
 0092  CB                             ret
 0093                  Main ENDP
 0093                  CODE      ENDS
                 END Main
```


Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA	STACK	
CODE	0093	PARA	NONE	
DATA	000E	PARA	NONE	

Symbols:

N a m e	Type	Value	Attr
A	L WORD	0000	DATA
B	L WORD	0002	DATA
FIN	L NEAR	005F	CODE
FUN16	L NEAR	0042	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 =
0093
MIN  . . . . . .    L NEAR     008C  CODE

RESULT . . . . . .  L NEAR     0092  CODE

SKIP_I1  . . . . .  L NEAR     006D  CODE
SKIP_I2  . . . . .  L NEAR     007D  CODE

TEMP . . . . . .    L WORD     000C  DATA

ZERO . . . . . .    L NEAR     0082  CODE

@CPU . . . . . .    TEXT  0101h
@FILENAME  . . . .  TEXT  LR3
@VERSION . . . . .  TEXT  510


     91 Source  Lines
     91 Total   Lines
     23 Symbols

  48070 + 461237 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```