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. 0383 _____ Рудакова Ю.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.

Выполнение работы.

Вариант 12:

$$i1=f2 = \begin{cases} -(4*i+3), & \text{при } a>b \\ 6*i-10, & \text{при } a<=b \end{cases}$$

$$i2=f7 = \begin{cases} -(4*i-5), & \text{при } a>b \\ 10-3*i, & \text{при } a<=b \end{cases}$$

$$res=f4 = \begin{cases} \min(|i1-i2|, 2), & \text{при } k<0 \\ \max(-6, -i2), & \text{при } k>=0 \end{cases}$$

Числа для работы программы вводятся сразу в asm файл. Для реализации алгоритмов использовались команда сравнения cmp и различные условные переходы. Для функций f2 и f7 условия одинаковы, поэтому их вычисление проходит в одном блоке. Сначала командой cmp сверяются значения a и b. С помощью команды jle проверяется, что a <= b, и в зависимости от результата программа переходит к блоку, где рассчитываются соответствующие значения f2 и f7. Для операций умножения использовался битовый сдвиг влево(команда shl) и сложение (команда add).

Результаты тестирования представлены в табл. 1.

Тексты исходных файлов программ см. в приложении А.

Тексты файлов диагностических сообщений см. в приложении Б.

Тестирование.

Таблица 1. Проверка работы программы.

№	Входные данные	Значение i1	Значение i2	Значение res	Комментарий
1	a = 2, b = 3, i = 1, k = 4	-4	7	-6	Программа работает корректно
2	a = 1, b = 4, i = 2, k = 0	2	4	-4	Программа работает корректно
3	a = 6, b = 4, i = 3, k = -2	-15	-7	2	Программа работает корректно
4	a = 10, b = -3, i = 0, k = -10	-3	5	2	Программа работает корректно

Выводы.

В ходе выполнения данной лабораторной работы была изучена работа с целыми числами и условными переходами на языке Ассемблер.

ПРИЛОЖЕНИЕ А

ТЕКСТЫ ИСХОДНЫХ ФАЙЛОВ ПРОГРАММ

Название файла: **lab3.asm**

```
; Стек  программы
AStack SEGMENT  STACK
        DW 12 DUP(?)
AStack  ENDS
;Данные программы
DATA        SEGMENT
;Директивы описания данных
a       DW    2
b       DW    1
i       DW    -2
k       DW    3
i1      DW    0
i2      DW    0

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 ax, cx
   shl cx, 1
   shl cx, 1 ; C = 4i
mov bx, b
   cmp a, bx

; a>b
jle f27
    mov ax, cx
    add cx, 3
    neg cx
   mov i1, cx
    sub ax, 5
   neg ax
    mov cx, ax
   mov i2, cx
    jmp final

; a<b
f27:
    add cx, ax
    add cx, ax ; 6i
   sub cx, 10
    mov i1, cx
```

```
 mov cx, ax
add ax, cx
 add ax, cx
mov cx, 10
sub cx, ax
mov i2, cx

;рассчет f4
final:
mov bx, k
cmp bx, 0
jge next
mov cx, i1
sub cx, i2
cmp cx, 0
jge skip
neg cx
skip:
cmp cx, 2
jge min
jmp Fin        ; | | < 2
min:
mov cx, 2     ; | | >= 2
jmp Fin
next:
 mov ax, i2
 neg ax
cmp ax, -6 ;???
jl max
```

```
        mov cx, ax   ; |-i2| >= -6
        jmp Fin
        max:
        mov cx, -6           ; |-i2| < -6
        jmp Fin
        Fin:
        ret
Main    ENDP
CODE    ENDS
END Main
```

ПРИЛОЖЕНИЕ Б

ТЕКСТЫ ФАЙЛОВ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ

Название файла: **lab.lst**

```
Microsoft (R) Macro Assembler Version 5.10                11/18/21 01:26:3
                                                          Page   1-1


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

0018                    AStack  ENDS
                        ;Данные программы
0000                    DATA        SEGMENT
                        ;Директивы описания данны
                        x
0000  0002              a       DW   2
0002  0001              b       DW   1
0004  FFFE              i       DW   -2
0006  0003              k       DW   3
0008  0000              i1      DW   0
000A  0000              i2      DW   0

000C                    DATA        ENDS

                        ; Код программы
0000                    CODE        SEGMENT
```

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

                        ; 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  8B C1                     mov ax, cx
0012  D1 E1                     shl cx, 1
0014  D1 E1                     shl cx, 1 ; C = 4i
0016  8B 1E 0002 R                mov bx, b
001A  39 1E 0000 R              cmp a, bx

                                ; a>b
001E  7E 19                        jle f27
0020  8B C1                           mov ax, cx
0022  83 C1 03                        add cx, 3
0025  F7 D9                           neg cx
0027  89 0E 0008 R              mov i1, cx
002B  2D 0005                         sub ax, 5
002E  F7 D8                     neg ax
0030  8B C8                           mov cx, ax
0032  89 0E 000A R              mov i2, cx
0036  EB 1B 90                        jmp final
```

```
                                ; a<b
0039                            f27:
Microsoft (R) Macro Assembler Version 5.10                  11/18/21 01:26:3
                                                            Page  1-2


0039  03 C8                              add cx, ax
003B  03 C8                              add cx, ax ; 6i
003D  83 E9 0A                      sub cx, 10
0040  89 0E 0008 R                       mov i1, cx
0044  8B C8                              mov cx, ax
0046  03 C1                         add ax, cx
0048  03 C1                              add ax, cx
004A  B9 000A                       mov cx, 10
004D  2B C8                         sub cx, ax
004F  89 0E 000A R                  mov i2, cx

                                ;рассчет f4
0053                                final:
0053  8B 1E 0006 R                  mov bx, k
0057  83 FB 00                      cmp bx, 0
005A  7D 1D                         jge next
005C  8B 0E 0008 R                       mov cx, i1
0060  2B 0E 000A R                  sub cx, i2
0064  83 F9 00                      cmp cx, 0
0067  7D 02                         jge skip
0069  F7 D9                         neg cx
006B                            skip:
```

```
006B  83 F9 02                    cmp cx, 2
006E  7D 03                       jge min
0070  EB 1C 90                    jmp Fin       ; | | < 2
0073                        min:
0073  B9 0002                     mov cx, 2     ; | | >= 2
0076  EB 16 90                    jmp Fin
0079                        next:
0079  A1 000A R                    mov ax, i2
007C  F7 D8                          neg ax
007E  3D FFFA                      cmp ax, -6 ;???
0081  7C 05                        jl max
0083  8B C8                       mov cx, ax   ; |-i2| >= -6
0085  EB 07 90                    jmp Fin
0088                        max:
0088  B9 FFFA                     mov cx, -6          ; |-i2| < -6
008B  EB 01 90                    jmp Fin
008E                        Fin:
008E  CB                          ret
008F                  Main        ENDP
008F                  CODE        ENDS
                      END Main
Microsoft (R) Macro Assembler Version 5.10                  11/18/21 01:26:3
                                                            Symbols-1


Segments and Groups:

                N a m e                 Length     Align    Combine Class
```

```
ASTACK . . . . . .  0018  PARA  STACK
CODE . . . . . .    008F  PARA  NONE
DATA . . . . . .    000C  PARA  NONE

Symbols:

        N a m e          Type    Value   Attr

A . . . . . .           L WORD  0000  DATA

B . . . . . .           L WORD  0002  DATA

F27 . . . . . .         L NEAR  0039  CODE
FIN . . . . . .         L NEAR  008E  CODE
FINAL . . . . . .       L NEAR  0053  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 = 008F
MAX . . . . . .         L NEAR  0088  CODE
MIN . . . . . .         L NEAR  0073  CODE

NEXT . . . . . .        L NEAR  0079  CODE

SKIP . . . . . .        L NEAR  006B  CODE
```

```
@CPU . . . . . .     TEXT  0101h
@FILENAME  . . . . . .       TEXT  lab3
@VERSION . . . . . .         TEXT  510


  93 Source  Lines
  93 Total   Lines
  22 Symbols

 47976 + 461331 Bytes symbol space free

  0 Warning Errors
  0 Severe  Errors
```