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. 1303 _____ Бутыло Е.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Разработать на языке Ассемблера программу, которая по заданным целочисленным значениям параметров вычисляет значения функций.

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),

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.

3 | 1.4.3

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

$$f4 = \begin{cases} -(6*i - 4), & \text{при } a>b \\ 3*(i+2), & \text{при } a<=b \end{cases}$$

$$f3 = \begin{cases} |i1 + i2|, & \text{при } k=0 \\ \min(i1,i2), & \text{при } k/=0 \end{cases}$$

Выполнение работы

1. Из таблицы получен вариант набора функций, которые необходимо реализовать, приведенного в каталоге Задания.

2. Программа протранслирована с различными значениями переменных, результат выполнения набора функций зафиксирован в таблице;

Для выполнения данного задания были использованы такие команды общего назначения как:

Команды передачи данных.

1) Mov – присваивание

Двоичные арифметические команды.

1) Add - сложение
2) Sub - вычитание
3) Inc – инкремент
4) Cmp – сравнение
5) Neg – смена знака

Команды побитового сдвига.

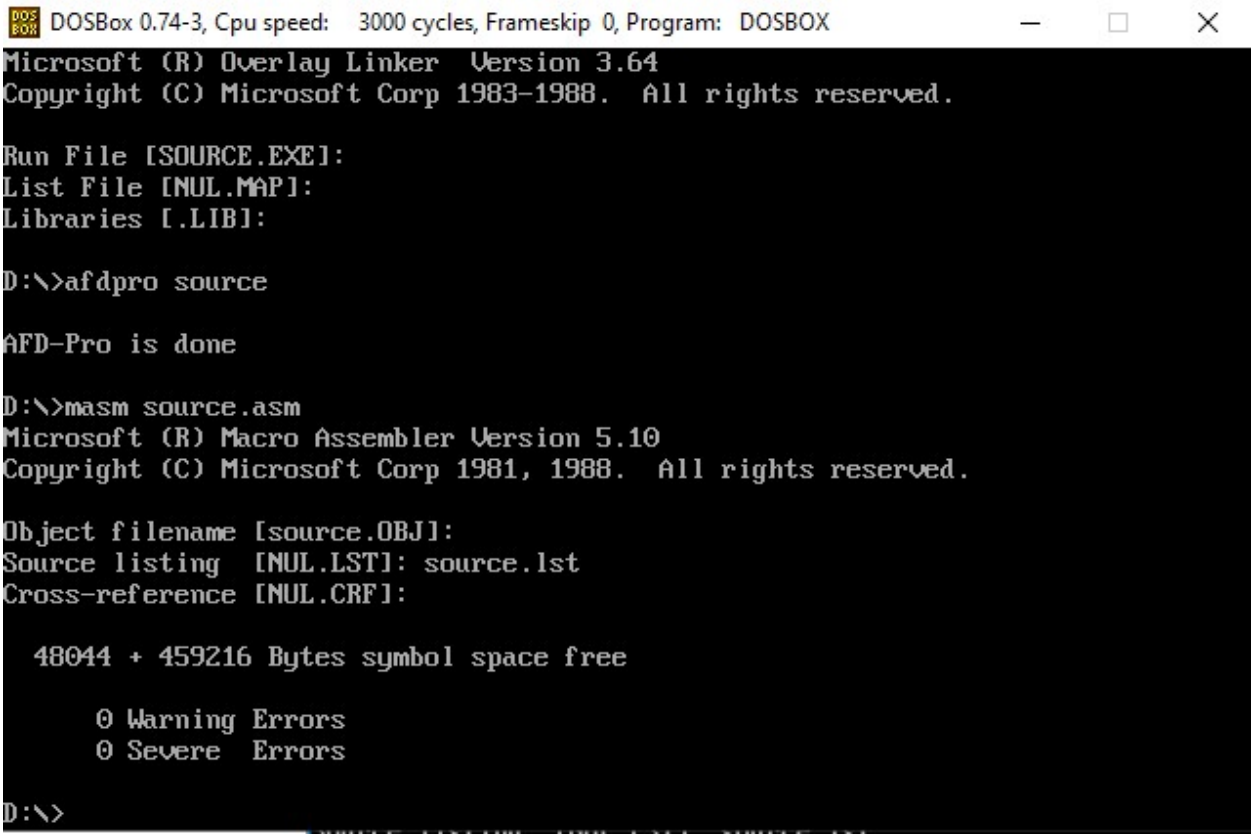
1) Sal - арифметический сдвиг влево

Команды передачи управления.

1) Jmp - безусловный переход
2) Int - вызов программного прерывания
3) Jge(jump greater equal) - выполняет короткий переход, если первый операнд больше второго операнда или равен ему при выполнении операции сравнения с помощью команды cmp
4) Jg(jump greater) - выполняет короткий переход, если первый операнд больше второго операнда при выполнении операции сравнения с помощью команды cmp.
5) Jne(jump negative equal) - выполняет короткий переход, если первый операнд не равен второму операнду при выполнении операции сравнения с помощью команды cmp.

Также были использованы метки (для примера B2), для перехода между некоторыми командами. Метка - это символьное имя, обозначающее ячейку памяти, которая содержит некоторую команду.

Трансляция программы



4. Программа выполнена в пошаговом режиме под управлением отладчика с фиксацией значений используемых переменных.

source.exe

№ теста	Тестируемый случай	Функции для данного случая	Данные	
			входные	выходные
1	a > b k = 0	f1 = 15 – 2*i f2 = -(6*i – 4) f3 = abs(f1 + f2)	a = 7, b = 3 k = 0 i = 2	f1 = 11 = 000B f2 = -8 = FFF8 f3 = 3 = 0003
2	a > b k != 0	f1 = 15 – 2*i f2 = -(6*i – 4) f3 = min(f1, f2)	a = 7, b = 3 k = 1 i = 3	f1 = 9 = 0009 f2 = -14 = FFF2 f3 = -14 = FFF2
3	a <= b k = 0	f1 = 3*i + 4 f2 = 3*(i + 2) f3 = abs(f1 + f2)	a = 5, b = 5 k = 0 i = 2	f1 = 10 = 000A f2 = 12 = 000C f3 = 22 = 0016
4	a <= b k != 0	f1 = 3*i + 4 f2 = 3*(i + 2) f3 = min(f1, f2)	a = 3, b = 5 k = 1 i = 3	f1 = 13 = 000D f2 = 15 = 000F f3 = 13 = 000D

Выводы

В ходе выполнения лабораторной работы были получены навыки разработки программы с заданными целочисленными значениями на языке программирования Ассемблер.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: source.asm

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

AStack    SEGMENT  STACK
          DW 32 DUP(0)
AStack    ENDS

DATA      SEGMENT

i     DW    ?
a     DW    ?
b     DW    ?
k     DW    ?

i1    DW    ?          ;f1
i2    DW    ?          ;f4
res   DW    ?          ;f3

DATA      ENDS

CODE SEGMENT

Main      PROC  FAR
      mov   AX,DATA
      mov   DS,AX

;Вычисление f1 и f2
      mov ax,a   ;заносим значение a в ax
      mov dx,b   ;заносим значение b в dx
      mov cx,i   ;заносим i в cx
      cmp ax,dx  ;Сравнение значений a и b
      jg A1      ;если a>b то на A1


      mov ax,i   ;если a<=b
      sal cx,1   ;умножение i на 2 cx = i*2
```

```
      add cx,ax  ;cx = 2*i + i = 3*i
      mov ax,4   ;ax = 4
      add cx,ax  ;cx = 3*i + 4
      mov i1,cx  ;сохранение результата в f1

      mov cx,i   ;восстанавливаем значени i в cx
      inc cx           ;cx = i + 1
      inc cx           ;cx = i + 2
      mov ax,cx  ;ax = i + 2
      sal cx,1   ;cx = 2*(i + 2)
      add cx,ax  ;cx = 2*(i + 2) + (i + 2) = 3*(i + 2)
      mov i2,cx  ;сохраняем рез-т в f2
      jmp A2           ;Пропускаем следующие шаги

A1:                    ;если a>b
      mov cx,i   ;восстановление значения i в cx
      sal cx,1   ;cx = i*2
      mov ax,15  ;ax = 15
      sub ax,cx  ;ax = ax - cx
      mov i1,ax  ;сохраняем результат в i1

      mov ax,cx  ;ax = 2*i
      sal cx,1   ;cx:=2*i*2
      add cx,ax  ;cx = 4*i + 2*i = 6*i
      mov ax,4   ;ax = 4
      sub ax,cx  ;ax = ax - cx = 4 - 6*i
      mov i2,ax  ;сохраняем результат в f2

;Вычисление f3
A2:
      mov ax,k
      mov bx,0

      cmp ax,bx  ;сравниваем k и 0
      JNe B1           ;если k не равно 0 то перйти на B1

                       ;решение при к = 0
      mov dx,i1  ;dx = i1
      add dx,i2  ;dx = i1 + i2
```

```
      cmp dx,bx
      JGe C1           ;если i1+ i2 >= 0 то перейти на C1

      neg dx           ;если i1 + i2 < 0 то меняем знак на
противоположный
      mov res,dx ;res = dx
      jmp B2

C1:
      mov res,dx
      jmp B2

B1:
                       ;если k не равно 0
      mov ax,i1
      mov bx,i2
      cmp ax,bx
      JGe C2           ;если i1 >= i2 то перейти на C2

      mov res,ax
      jmp B2

C2:
      mov res,bx ;если i1 >= i2

B2:
      int 20h

Main      ENDP
CODE      ENDS
          END Main
```

Название файла: source.lst

```
      Microsoft (R) Macro Assembler Version 5.10
10/16/22 15:13:3
                                                              Page
1-1
```

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

0000                AStack    SEGMENT  STACK
0000  0020[                        DW 32 DUP(0)
  0000
          ]

0040                AStack    ENDS

0000                DATA      SEGMENT

0000  0000                i     DW    ?
0002  0000                a     DW    ?
0004  0000                b     DW    ?
0006  0000                k     DW    ?

0008  0000                i1    DW    ?          ;f1
000A  0000                i2    DW    ?          ;f4
000C  0000                res   DW    ?          ;f3

000E                DATA      ENDS

0000                CODE SEGMENT

0000                Main      PROC  FAR
0000  B8 ---- R            mov   AX,DATA
0003  8E D8                 mov   DS,AX

              ;Вычисление f1 и f2
0005  A1 0002 R             mov ax,a   ;заносим знаэ
               ение a в ax
0008  8B 16 0004 R           mov dx,b   ;заносим знаэ
               ение b в dx
000C  8B 0E 0000 R           mov cx,i   ;заносим i в cx
0010  3B C2                  cmp ax,dx  ;Сравнение зЙ
              ½ачений a и b
0012  7F 23                  jg A1      ;если a>b то на
               A1
```

```
0014  A1 0000 R           mov ax,i   ;если a<=b
0017  D1 E1                sal cx,1   ;умножение i Й
                ½a 2 cx = i*2
0019  03 C8                add cx,ax  ;cx = 2*i + i = 3*i

001B  B8 0004              mov ax,4   ;ax = 4
001E  03 C8                add cx,ax  ;cx = 3*i + 4
0020  89 0E 0008 R         mov i1,cx  ;сохранение э
                езультата в f1

0024  8B 0E 0000 R         mov cx,i   ;восстанавли
                ваем значени i в cx
0028  41                   inc cx          ;cx = i + 1
0029  41                   inc cx          ;cx = i + 2

     Microsoft (R) Macro Assembler Version 5.10
10/16/22 15:13:3
                                                        Page
1-2


002A  8B C1                mov ax,cx  ;ax = i + 2
002C  D1 E1                sal cx,1   ;cx = 2*(i + 2)
002E  03 C8                add cx,ax  ;cx = 2*(i + 2) + (i
+
                2) = 3*(i + 2)
0030  89 0E 000A R         mov i2,cx  ;сохраняем рЙ
                µз-т в f2
0034  EB 1D 90             jmp A2          ;Пропускаем э
                ледующие шаги

0037                A1:                    ;если a>b
0037  8B 0E 0000 R         mov cx,i   ;восстановле
                ние значения i в cx
003B  D1 E1                sal cx,1   ;cx = i*2
003D  B8 000F              mov ax,15  ;ax = 15
0040  2B C1                sub ax,cx  ;ax = ax - cx
0042  A3 0008 R           mov i1,ax  ;сохраняем рЙ
```

```
                µзультат в i1

0045  8B C1                    mov ax,cx  ;ax = 2*i
0047  D1 E1                    sal cx,1   ;cx:=2*i*2
0049  03 C8                    add cx,ax  ;cx = 4*i + 2*i = 6*i
004B  B8 0004                  mov ax,4   ;ax = 4
004E  2B C1                    sub ax,cx  ;ax = ax - cx = 4 -
6*i

0050  A3 000A R           mov i2,ax  ;сохраняем рЙ
                µзультат в f2

                ;Вычисление f3
0053                A2:
0053  A1 0006 R           mov ax,k
0056  BB 0000                  mov bx,0

0059  3B C3                    cmp ax,bx  ;сравниваем k
                и 0
005B  75 1C                    JNe B1          ;если k не рав
                но 0 то перйти на B1

                                        ;решение
                при к = 0
005D  8B 16 0008 R             mov dx,i1  ;dx = i1
0061  03 16 000A R             add dx,i2  ;dx = i1 + i2
0065  3B D3                    cmp dx,bx
0067  7D 09                    JGe C1          ;если i1+ i2 >=
0 э
                о перейти на C1

0069  F7 DA                    neg dx          ;если i1 + i2 <
0 э
                о меняем знак на противоп
                оложный
006B  89 16 000C R             mov res,dx ;res = dx
006F  EB 1D 90                 jmp B2

0072                C1:
```

```
0072  89 16 000C R              mov res,dx
0076  EB 16 90                   jmp B2

     Microsoft (R) Macro Assembler Version 5.10
10/16/22 15:13:3
                                                                    Page
1-3


0079                 B1:
                                          ;если k нЙ
                µ равно 0
0079  A1 0008 R           mov ax,i1
007C  8B 1E 000A R              mov bx,i2
0080  3B C3                   cmp ax,bx
0082  7D 06                   JGe C2          ;если i1 >= i2
то
                  перейти на C2

0084  A3 000C R           mov res,ax
0087  EB 05 90                   jmp B2

008A                 C2:
008A  89 1E 000C R              mov res,bx ;если i1 >= i2

008E                 B2:
008E  CD 20                   int 20h

0090                 Main      ENDP
0090                 CODE      ENDS
                          END Main

     Microsoft (R) Macro Assembler Version 5.10
10/16/22 15:13:3

Symbols-1
```

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0040	PARA	STACK	
CODE	0090	PARA	NONE	
DATA	000E	PARA	NONE	

Symbols:

N a m e	Type	Value	Attr
A	L WORD	0002	DATA
A1	L NEAR	0037	CODE
A2	L NEAR	0053	CODE
B	L WORD	0004	DATA
B1	L NEAR	0079	CODE
B2	L NEAR	008E	CODE
C1	L NEAR	0072	CODE
C2	L NEAR	008A	CODE
I	L WORD	0000	DATA
I1	L WORD	0008	DATA
I2	L WORD	000A	DATA
K	L WORD	0006	DATA
MAIN	F PROC	0000	CODE
Length = 0090			
RES	L WORD	000C	DATA
@CPU	TEXT	0101h	
@FILENAME	TEXT	source	
@VERSION	TEXT	510	

```
  104 Source  Lines
  104 Total   Lines
   22 Symbols

48044 + 459216 Bytes symbol space free

    0 Warning Errors
    0 Severe  Errors
```