МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Организация ЭВМ и систем»

Тема: Представление и обработка целых чисел. Организация ветвящихся процессов

Вариант 17

Студентка гр.1381 ____ Новак П.И.

Преподаватель ____ Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Изучить представление и обработку целых чисел на языке Ассемблер. Научиться организовывать ветвящиеся процессы.

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.

Вариант №17

/ 7-4*i, при a>b f3 = < \ 8-6*i, при a<=b	/ -(4*i-5), при a>b f7 = < \ 10-3*i, при a<=b	/ min(|i1|, 6), при k=0 f5 = < \ |i1|+|i2|, при k/=0

Выполнение работы.

1. Были созданы три сегмента: сегмент стека (AStack), сегмент данных (DATA) и сегмент кода (CODE). Метки сегментов были записаны в соответствующие регистры с помощью директивы ASSUME. Исходный код программы см. в приложении А.

2. В сегменте DATA были объявлены переменные a, b, i, k, i1, i2, res.

3. В сегменте CODE была создана процедура Main, в которой сначала адрес сегмента данных помещается в регистр ds, а дальше происходит работа с функциями. Были задействованы следующие регистры: ax, cx, dx. Для

выполнения задания при реализации функций использовались следующие команды:

1) JMP (JUMP) – команда безусловного перехода, то есть прыжок может быть как дальним, так и ближним.

2) JG (Jump if greater) – команда, выполняющая короткий переход, если первый операнд больше второго операнда при выполнении операции сравнения с помощью команды cmp.

3) JNE (Jump if not equal) – команда, выполняющая короткий переход, если первый операнд не равен второму при выполнении сравнения с помощью команды cmp.

4) NEG – команда, инвертирующая все биты числа.

5) SAL - команда, осуществляющая сдвиг всех битов операнда влево.

Тестирование.

Чтобы проверить корректность работы программы, было проведено три 1. Результаты работы программы при a=1; b=3; i=2; k=0 представлены в табл.1.

i1	i2	res	Правильность результата
FFFC (-4)	0004(4)	0004 (4)	Верно

Таблица 1 – Результаты первого теста

2. Результаты работы программы при a=1; b=-1; i=-1; k=0 представлены в табл.2.

i1	i2	res	Правильность результата
000B (11)	0009(9)	0014 (21)	Верно

Таблица 2 – Результаты второго теста

2. Результаты работы программы при a=2; b=4; i=-6; k=5 представлены в табл.3.

i1	i2	res	Правильность результата
002C (44)	001C(28)	0048 (72)	Верно

Таблица 3 – Результаты третьего теста

Выводы.

В ходе выполнения лабораторной работы было изучено представление и обработка целых чисел, и организация ветвящихся процессов. Для выполнения задания была написана программа, которая вычисляет значения функций согласно заданным условиям.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММ

Название файла: *lab3.asm*

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

AStack    SEGMENT  STACK
          DW 32 DUP(0)
AStack    ENDS

DATA      SEGMENT

i     DW    0
a     DW    0
b     DW    0
k     DW    0

i1    DW    0          ;f1
i2    DW    0          ;f2
res   DW    0          ;f3

DATA      ENDS

CODE SEGMENT

Main      PROC  FAR
mov   AX,DATA
mov   DS,AX

;Вычисление f1 и f2
mov ax,a   ;ax = a
mov cx,i   ;cx = i
mov dx,b   ;dx = b
cmp ax,dx  ;Сравнение значений a и b
jg PART1   ;если a>b то на PART1

;если a<=b:
mov ax,8   ;ax = 8
mov dx,i   ;dx = i
sal dx,1   ;dx = 2i
sal cx,1   ;cx = 2i
sal cx,1   ;cx = 4i
add cx,dx  ;cx = 2i + 4i = 6i
```

```
sub ax,cx  ;ax = 8 - 6i
mov i1,ax  ;i1(f1) = ax = 8 - 6i

mov cx,i   ;cx = i
mov dx, i  ;dx = i
sal dx,1   ;dx = 2i
add cx,dx  ;cx = i + 2i = 3i
mov ax, 10 ;ax = 10
sub ax, cx ;ax = ax - cx = 10 - 3i
mov i2,ax  ;i2(f2) = ax = 10 - 3i
jmp PART2  ;идем на PART2

PART1:                ;если a>b
mov cx,i   ;cx = i
sal cx,1   ;cx = 2i
sal cx,1   ;cx = 4i
mov ax,7   ;ax = 7
sub ax,cx  ;ax = ax - cx = 7 - 4i
mov i1,ax  ;i1(f1) = ax = 7 - 4i

mov ax,-5  ;ax = -5
add ax,cx  ;ax = ax + cx = 4i - 5
neg ax          ;ax = -(4i - 5)
mov i2,ax  ;i2(f2) = cx = -(4i - 5)

;Вычисление f3
PART2:
mov bx,0
cmp bx,i1
jg ABS1         ;если 0 > i1, то на ABS1
jmp CHECK  ;иначе идем на CHECK

ABS1:
neg i1

CHECK:
cmp bx,i2
jg PART21  ;если 0 > i2, то на ABS2
jmp PART21 ;иначе идём на PART21

ABS2:
neg i2

PART21:
mov ax,k
```

```
cmp ax,bx  ;сравниваем k и 0
jne PART4  ;если k не равно 0, то на PART4

;если к = 0
mov ax,i1  ;ax = i1
mov bx,6   ;bx = 6
cmp ax,bx
jg PART3   ;если i1 > 6 то на PART3

mov res,ax ;res = ax = i1
jmp ENDPART
PART3:
mov res,bx ;res = bx = 6
jmp ENDPART

PART4:                ;если k не равно 0
mov ax, i1
add ax,i2  ;ax = ax + i2
mov res,ax ;res = ax = i1 + i2

ENDPART:
int 20h

Main      ENDP
CODE      ENDS
          END Main
```

Название файла: *lab3.lst*

```
Microsoft (R) Macro Assembler Version 5.10                  10/23/22
13:09:4
                                                             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    0
0002  0000              a     DW    0
0004  0000              b     DW    0
0006  0000              k     DW    0

0008  0000              i1    DW    0          ;f1
000A  0000              i2    DW    0          ;f2
000C  0000              res   DW    0          ;f3

000E                    DATA      ENDS

0000                    CODE SEGMENT

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

;P'C‹C‡PëCЃP»PµPSPëPµ f1 Pë f2
0005  A1 0002 R                mov ax,a   ;ax = a
0008  8B 0E 0000 R                mov cx,i   ;cx = i
000C  8B 16 0004 R                mov dx,b   ;dx = b
0010  3B C2                     cmp ax,dx  ;PЎCЂP°PIPSPµPSPëPµ P·P
SP°C‡PµPSPëP№ a Pë b
0012  7F 2B                     jg PART1   ;PµCЃP»Pë a>b C,Ps PSP°
 PART1

;PµCЃP»Pë a<=b:
```

```
0014  B8 0008                   mov ax,8   ;ax = 8
0017  8B 16 0000 R               mov dx,i   ;dx = i
001B  D1 E2                     sal dx,1   ;dx = 2i
001D  D1 E1                     sal cx,1   ;cx = 2i
001F  D1 E1                     sal cx,1   ;cx = 4i
0021  03 CA                     add cx,dx  ;cx = 2i + 4i = 6i
0023  2B C1                     sub ax,cx  ;ax = 8 - 6i
0025  A3 0008 R            mov i1,ax  ;i1(f1) = ax = 8 - 6i

0028  8B 0E 0000 R               mov cx,i   ;cx = i
002C  8B 16 0000 R               mov dx, i  ;dx = i
0030  D1 E2                     sal dx,1   ;dx = 2i
0032  03 CA                     add cx,dx  ;cx = i + 2i = 3i
0034  B8 000A                   mov ax, 10 ;ax = 10
0037  2B C1                     sub ax, cx ;ax = ax - cx = 10 - 3i
0039  A3 000A R            mov i2,ax  ;i2(f2) = ax = 10 - 3i
Microsoft (R) Macro Assembler Version 5.10                  10/23/22
13:09:4
                                                             Page     1-2


003C  EB 1B 90                  jmp PART2  ;PëPrPµPj PSP° PART2

003F                  PART1:                ;PµCŕP»Pë a>b
003F  8B 0E 0000 R               mov cx,i   ;cx = i
0043  D1 E1                     sal cx,1   ;cx = 2i
0045  D1 E1                     sal cx,1   ;cx = 4i
0047  B8 0007                   mov ax,7   ;ax = 7
004A  2B C1                     sub ax,cx  ;ax = ax - cx = 7 - 4i
004C  A3 0008 R            mov i1,ax  ;i1(f1) = ax = 7 - 4i

004F  B8 FFFB                   mov ax,-5  ;ax = -5
0052  03 C1                     add ax,cx  ;ax = ax + cx = 4i - 5
0054  F7 D8                     neg ax          ;ax = -(4i - 5)
0056  A3 000A R            mov i2,ax  ;i2(f2) = cx = -(4i - 5
)

;P'C‹C‡PëCŕP»PµPSPëPµ f3
0059                  PART2:
0059  BB 0000                   mov bx,0
005C  3B 1E 0008 R               cmp bx,i1
0060  7F 03                     jg ABS1         ;PµCŕP»Pë 0 > i1, C,Ps
PSP° ABS1
0062  EB 05 90                  jmp CHECK  ;PëPSP°C‡Pµ PëPrPµPj PS
P° CHECK
```

```
0065                    ABS1:
0065  F7 1E 0008 R                 neg i1

0069                    CHECK:
0069  3B 1E 000A R                 cmp bx,i2
006D  7F 07                       jg PART21  ;PµCЃP»Pë 0 > i2, C,Ps
PSP° ABS2
006F  EB 05 90                    jmp PART21 ;PëPSP°C‡Pµ PëPrC‘Pj PS
P° PART21

0072                    ABS2:
0072  F7 1E 000A R                 neg i2

0076                    PART21:
0076  A1 0006 R               mov ax,k

0079  3B C3                       cmp ax,bx  ;CЃCЂP°PIPSPëPIP°PµPj k
 Pë 0
007B  75 17                       JNe PART4  ;PµCЃP»Pë k PSPµ CЂP°PI
PSPs 0, C,Ps PSP° PART4

;PµCЃP»Pë Pє = 0
007D  A1 0008 R               mov ax,i1  ;ax = i1
0080  BB 0006                      mov bx,6   ;bx = 6
0083  3B C3                       cmp ax,bx
0085  7F 06                       jg PART3   ;PµCЃP»Pë i1 > 6 C,Ps P
SP° PART3

0087  A3 000C R               mov res,ax ;res = ax = i1
Microsoft (R) Macro Assembler Version 5.10                  10/23/22
13:09:4
                                                             Page     1-3


008A  EB 12 90                    jmp ENDPART
008D                    PART3:
008D  89 1E 000C R                 mov res,bx ;res = bx = 6
0091  EB 0B 90                    jmp ENDPART

0094                    PART4:                ;PµCЃP»Pë k PSPµ CЂP°PI
PSPs 0
0094  A1 0008 R               mov ax, i1
0097  03 06 000A R                  add ax,i2  ;ax = ax + i2
009B  A3 000C R               mov res,ax ;res = ax = i1 + i2
```

```
009E                    ENDPART:
009E  CD 20                         int 20h

00A0                    Main        ENDP
00A0                    CODE        ENDS
          END Main
Microsoft (R) Macro Assembler Version 5.10                  10/23/22
13:09:4
                                                             Symbols-1


Segments and Groups:

                N a m e         Length      Align     Combine Class

ASTACK . . . . . .              0040  PARA  STACK
CODE . . . . . .                00A0  PARA  NONE
DATA . . . . . .                000E  PARA  NONE

Symbols:

                N a m e         Type   Value      Attr

A  . . . . . .                  L WORD     0002  DATA
ABS1 . . . . . .                L NEAR     0065  CODE
ABS2 . . . . . .                L NEAR     0072  CODE

B  . . . . . .                  L WORD     0004  DATA

CHECK  . . . . . .              L NEAR     0069  CODE

ENDPART  . . . . . .            L NEAR     009E  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 = 00A0

PART1  . . . . . .              L NEAR     003F  CODE
PART2  . . . . . .              L NEAR     0059  CODE
PART21 . . . . . .              L NEAR     0076  CODE
```

```
PART3  . . . . . .      L NEAR     008D  CODE
PART4  . . . . . .      L NEAR     0094  CODE

RES  . . . . . .        L WORD     000C  DATA

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


    111 Source  Lines
    111 Total   Lines
     25 Symbols

  47982 + 459278 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```