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. 0382 _____ Здобнова К.Д.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Изучить обработку целых чисел на ассемблере. Написать программу с использованием меток перехода.

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 5 – 1.6.5

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

$$f2 = \begin{cases} 2*(i+1)-4, & \text{при } a>b \\ 5-3*(i+1), & \text{при } a<=b \end{cases}$$

$$f3 = \begin{cases} \min(|i1|, 6), & \text{при } k=0 \end{cases}$$

\ |i1|+|i2|, при k/=0

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.

Сегмент данных:

a, b, i, k – исходные данные, значения меняются в ходе тестирования; i1 – результат выполнения функции f1; i2 – результат выполнения функции f2 ; res – результат выполнения функции f3. Все слова, кроме исходных данных, на начало выполнения программы инициализированы 0.

Сегмент кода:

Выполняется директива *assume*, которая соотносит сегментные регистры и сегменты.

Данные переменной a записываются в регистр AX, b в BX, I в CX, далее происходит сравнение сегментов AX и BX, если a > b то происходит переход к метке *Above*, в противном случае к метке *Less*.

В метке *Above* происходит выполнение функций f1 и f2 при условии a > b, в метке *Less* происходит выполнение функций f1 и f2 при условии a <= b.

Далее выполняется сравнение значения i1 с нулем. Если i1 – отрицательное число, то значение берется по модулю для дальнейших вычислений (метка *Abs1*). Функция f3 выполняется в метке *F3*, где происходит сравнение значения переменной k с нулем, если флаг zero равен 1, то программа переходит в метку

Zero, в противном случае выполняет программу последовательно и выполняет сумму модулей i1 и i2 (при отрицательном значении i2 программа переходит по метке Abs2). После вычисления значения переменной res осуществляется безусловный переход к метке Finish.

Команда ret в конце процедуры возвращает в DOS.

Тестирование

1) a = -3, b = 6, i = 5, k = 0.

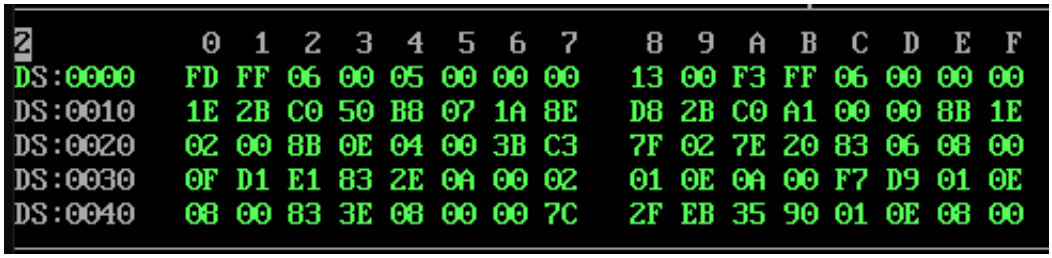
a = -3 = FFDF (FD FF)

b = 6 = 0006 (06 00)

i = 5 = 0005 (05 00)

k = 0 = 0000 (00 00)

res = 06 00 = 0006 = 6



2) a = -3, b = 6, i = 5, k = 2.

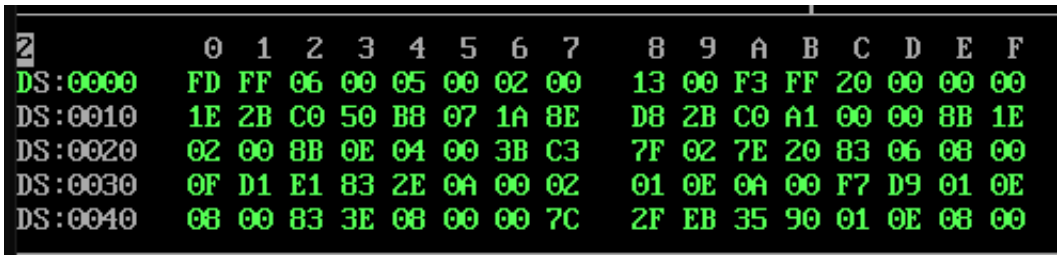
a = -3 = FFDF (FD FF)

b = 6 = 0006 (06 00)

i = 5 = 0005 (05 00)

k = 2 = 0002 (02 00)

res = 20 00 = 0020 = 32



3) a = 10, b = 2, i = 5, k = 0.

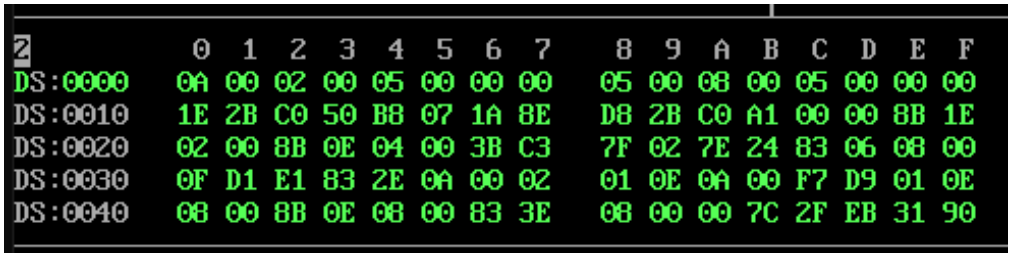
a = 10 = 000A (0A 00)

b = 2 = 0002 (02 00)

i = 5 = 0005 (05 00)

k = 0 = 0000 (00 00)

res = 05 00 = 0005 = 5



```
2          0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
DS:0000   0A 00 02 00 05 00 00 00   05 00 08 00 05 00 00 00
DS:0010   1E 2B C0 50 B8 07 1A 8E   D8 2B C0 A1 00 00 8B 1E
DS:0020   02 00 8B 0E 04 00 3B C3   7F 02 7E 24 83 06 08 00
DS:0030   0F D1 E1 83 2E 0A 00 02   01 0E 0A 00 F7 D9 01 0E
DS:0040   08 00 8B 0E 08 00 83 3E   08 00 00 7C 2F EB 31 90
```

4) a = 10, b = 2, i = 5, k = 2.

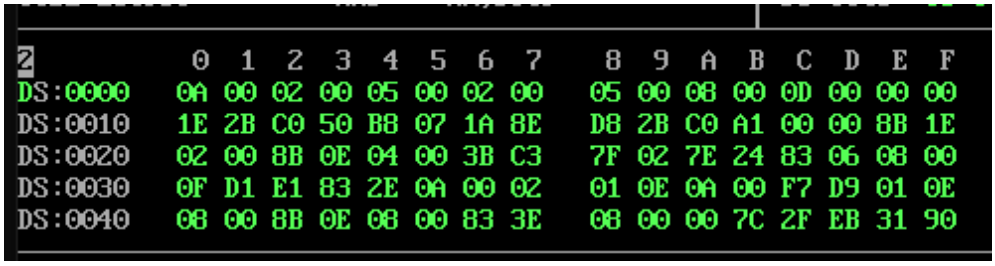
a = 10 = 000A (0A 00)

b = 2 = 0002 (02 00)

i = 5 = 0005 (05 00)

k = 2 = 0002 (02 00)

res = 0D 00 = 000D = 13



```
2          0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
DS:0000   0A 00 02 00 05 00 02 00   05 00 08 00 0D 00 00 00
DS:0010   1E 2B C0 50 B8 07 1A 8E   D8 2B C0 A1 00 00 8B 1E
DS:0020   02 00 8B 0E 04 00 3B C3   7F 02 7E 24 83 06 08 00
DS:0030   0F D1 E1 83 2E 0A 00 02   01 0E 0A 00 F7 D9 01 0E
DS:0040   08 00 8B 0E 08 00 83 3E   08 00 00 7C 2F EB 31 90
```

Выводы.

Была изучена обработка целых чисел на ассемблере. Реализована программа с использованием меток условного перехода.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: *lab3.asm*

```
; Стек программы
AStack SEGMENT STACK
    DW 12 DUP(?)
AStack          ENDS

; Данные программы
DATA            SEGMENT
;68 стр
; Директивы описания данных
a    DW 0
b    DW 0
i    DW 0
k    DW 0
i1   DW 0
i2   DW 0
res  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
    sub AX,AX
    mov AX,a
```

```
    mov BX,b
    mov CX,i
    cmp AX,BX
    JG Above
    JLE Less

Above:          ;a > b
    add i1,15  ;i1 = 15
    sal CX,1   ;i = i * 2
    sub i2,2
    add i2,CX  ;тк различие на 2
    neg CX     ;i = - 2 * i
    add i1,CX  ;i = 15 - 2i
    mov CX, i1
    CMP i1,0
    JL Abs1
    JMP F3

Less:           ;a <= b
    add i1,CX   ;i1 = i
    sal CX,1    ;i = 2*i
    add CX,i1   ;i = 3*i
    mov i1,CX
    add i1,4
    add i2,2
    neg CX
    add i2,CX
    mov CX, i1
    CMP i1,0
    JL Abs1
    JMP F3

Abs1:           ;берем i1 по модулю
    neg i1

F3:
```

```
        CMP k,0
        JZ Zero
        add res, CX
        mov CX, i2
        CMP CX,0
        JL Abs2
        add res,CX
        JMP Finish

    Abs2:
        neg CX
        add res, CX
        JMP Finish

    Zero:
        CMP CX,6
        JL MinI
        mov res, 6
        JMP Finish

    MinI:
        mov res, CX

    Finish:
       ret

    Main ENDP
    CODE ENDS
              END MAIN
```

Название файла: *lab3.LST*

```
Microsoft (R) Macro Assembler Version 5.10                  11/10/21
22:48:1
                                                             Page     1-1


                        ; Стек программы
 0000                   AStack SEGMENT STACK
```

```
0000  000C[                     DW 12 DUP(?)
       ????
               ]

0018                  AStack          ENDS

                      ; Данные программы
0000                  DATA        SEGMENT
                      ;68 стр
                      ; Директивы описания даннэ
                      □x
0000  0000                  a     DW 0
0002  0000                  b     DW 0
0004  0000                  i     DW 0
0006  0000                  k     DW 0
0008  0000                  i1    DW 0
000A  0000                  i2    DW 0
000C  0000                  res   DW 0

000E                  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  2B C0                       sub AX,AX
000B  A1 0000 R             mov AX,a
000E  8B 1E 0002 R                mov BX,b
0012  8B 0E 0004 R                mov CX,i
0016  3B C3                       cmp AX,BX
0018  7F 02                       JG Above
001A  7E 24                       JLE Less

001C                  Above:          ;a > b
001C  83 06 0008 R 0F                  add i1,15  ;i1 = 15
0021  D1 E1                       sal CX,1   ;i = i * 2
0023  83 2E 000A R 02                  sub i2,2
```

```
0028  01 0E 000A R              add i2,CX  ;тк различие на
                        2
002C  F7 D9                     neg CX     ;i = - 2 * i
002E  01 0E 0008 R              add i1,CX  ;i = 15 - 2i
0032  8B 0E 0008 R              mov CX, i1
0036  83 3E 0008 R 00                CMP i1,0
003B  7C 2F                     JL Abs1
003D  EB 31 90                  JMP F3
```



```
0040                Less:           ;a <= b
0040  01 0E 0008 R              add i1,CX   ;i1 = i
0044  D1 E1                     sal CX,1    ;i = 2*i
0046  03 0E 0008 R              add CX,i1   ;i = 3*i
004A  89 0E 0008 R              mov i1,CX
004E  83 06 0008 R 04                add i1,4
0053  83 06 000A R 02                add i2,2
0058  F7 D9                     neg CX
005A  01 0E 000A R              add i2,CX
005E  8B 0E 0008 R              mov CX, i1
0062  83 3E 0008 R 00                CMP i1,0
0067  7C 03                     JL Abs1
0069  EB 05 90                  JMP F3

006C                Abs1:           ;берем i1 по модулэ
                    □
006C  F7 1E 0008 R              neg i1

0070                F3:
0070  83 3E 0006 R 00                CMP k,0
0075  74 1D                     JZ Zero
0077  01 0E 000C R              add res, CX
007B  8B 0E 000A R              mov CX, i2
007F  83 F9 00                  CMP CX,0
0082  7C 07                     JL Abs2
0084  01 0E 000C R              add res,CX
0088  EB 1C 90                  JMP Finish

008B                Abs2:
008B  F7 D9                     neg CX
```

```
 008D  01 0E 000C R              add res, CX
 0091  EB 13 90                  JMP Finish

 0094                  Zero:
 0094  83 F9 06                  CMP CX,6
 0097  7C 09                     JL MinI
 0099  C7 06 000C R 0006         mov res, 6
 009F  EB 05 90                  JMP Finish

 00A2                  MinI:
 00A2  89 0E 000C R              mov res, CX

 00A6                  Finish:
 00A6  CB                  ret

 00A7                  Main  ENDP
 00A7                  CODE  ENDS
                               END MAIN
Microsoft (R) Macro Assembler Version 5.10                  11/10/21
22:48:1
                                                             Symbols-1


Segments and Groups:

                N a m e         Length      Align     Combine Class

ASTACK . . . . . .              0018  PARA  STACK
CODE . . . . . .                00A7  PARA  NONE
DATA . . . . . .                000E  PARA  NONE

Symbols:

                N a m e         Type   Value      Attr

A  . . . . . .                  L WORD     0000  DATA
ABOVE  . . . . . .              L NEAR     001C  CODE
ABS1 . . . . . .                L NEAR     006C  CODE
ABS2 . . . . . .                L NEAR     008B  CODE

B  . . . . . .                  L WORD     0002  DATA

F3 . . . . . .                  L NEAR     0070  CODE
FINISH . . . . . .              L NEAR     00A6  CODE
```

```
I  . . . . . .  L WORD   0004  DATA
I1 . . . . . .  L WORD   0008  DATA
I2 . . . . . .  L WORD   000A  DATA

K  . . . . . .  L WORD   0006  DATA

LESS . . . . . .  L NEAR   0040  CODE

MAIN . . . . . .  F PROC   0000  CODE  Length = 00A7
MINI . . . . . .  L NEAR   00A2  CODE

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

ZERO . . . . . .  L NEAR   0094  CODE

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


     97 Source  Lines
     97 Total   Lines
     24 Symbols

  48056 + 461251 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```