

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:

а) значения функций $i1 = f1(a,b,i)$ и $i2 = f2(a,b,i)$;

б) 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

$/ 15-2*i$, при $a>b$

$f1 = <$

$\backslash 3*i+4$, при $a\leq b$

$/ 2*(i+1) -4$, при $a>b$

$f2 = <$

$\backslash 5 - 3*(i+1)$, при $a\leq b$

$/ \min(|i1|, 6)$, при $k=0$

$f3 = <$

$$\backslash |i1|+|i2|, \text{ при } k \neq 0$$

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 $a \leq b$.

Далее выполняется сравнение значения $i1$ с нулем. Если $i1$ – отрицательное число, то значение берется по модулю для дальнейших вычислений (метка *AbsI*). 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	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
DS:0000	FD	FF	06	00	05	00	00	00	13	00	F3	FF	06	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	20	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	83	3E	08	00	00	7C	2F	EB	35	90	01	0E	08	00

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

2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
DS:0000	FD	FF	06	00	05	00	02	00	13	00	F3	FF	20	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	20	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	83	3E	08	00	00	7C	2F	EB	35	90	01	0E	08	00

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 -3
b    DW 6
i    DW 5
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
sal CX,1 ;i = i * 2
cmp AX,b
JLE Less

```

```

Above:           ;a > b
add i1,15 ;i1 = 15
sub i2,2
add i2,CX ;тк различие на 2
sub i1,CX ;i = 15 - 2i
mov CX, i1
CMP i1,0
JGE F3
neg CX ;берем i1 по модулю
JMP F3

```

```

Less:           ;a <= b
add i1,CX ;i1 = i
add CX,i1 ;i = 3*i
mov i1,CX
add i1,4
add i2,2
sub i2,CX
mov CX, i1
CMP i1,0
JGE F3
neg CX ;берем i1 по модулю

```

```

F3:
CMP k,0
JZ Zero
add res, CX
mov CX, i2
CMP CX,0
JGE Ans

```



```

        neg CX
        JMP Ans

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

Ans:
        add res, CX

Finish:
        ret

Main ENDP
CODE ENDS

        END MAIN

```

Название файла: *lab3.LST*

Microsoft (R) Macro Assembler Version 5.10
02:23:2

11/25/21

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  FFFD                a      DW -3
0002  0006                b      DW 6
0004  0005                i      DW 5

```

```

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 D1 E1                      sal CX,1 ;i = i * 2
0018 3B 06 0002 R              cmp AX,b
001C 7E 22                      JLE Less

001E                      Above: ;a > b
001E 83 06 0008 R 0F          add i1,15 ;i1 = 15
0023 83 2E 000A R 02          sub i2,2
0028 01 0E 000A R              add i2,CX ;тк различие на
                                2
002C 29 0E 0008 R              sub i1,CX ;i = 15 - 2i
0030 8B 0E 0008 R              mov CX, i1
0034 83 3E 0008 R 00          CMP i1,0
0039 7D 2C                      JGE F3
003B F7 D9                      neg CX ;берем i1 по модулю
                                □
003D EB 28 90                      JMP F3

```

```
0040          Less:          ;a <= b
0040 01 0E 0008 R          add i1,CX      ;i1 = i
0044 03 0E 0008 R          add CX,i1      ;i = 3*i
0048 89 0E 0008 R          mov i1,CX
004C 83 06 0008 R 04          add i1,4
0051 83 06 000A R 02          add i2,2
0056 29 0E 000A R          sub i2,CX
005A 8B 0E 0008 R          mov CX, i1
005E 83 3E 0008 R 00          CMP i1,0
0063 7D 02          JGE F3
0065 F7 D9          neg CX      ;берем i1 по модулю
```

□

```
0067          F3:
0067 83 3E 0006 R 00          CMP k,0
006C 74 12          JZ Zero
006E 01 0E 000C R          add res, CX
0072 8B 0E 000A R          mov CX, i2
0076 83 F9 00          CMP CX,0
0079 7D 13          JGE Ans
007B F7 D9          neg CX
007D EB 0F 90          JMP Ans
```

```
0080          Zero:
0080 83 F9 06          CMP CX,6
0083 7C 09          JL Ans
0085 C7 06 000C R 0006      mov res, 6
008B EB 05 90          JMP Finish
```

```
008E          Ans:
008E 01 0E 000C R          add res, CX
```

```
0092          Finish:
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
ABOVE	L NEAR	001E	CODE
ANS	L NEAR	008E	CODE
B	L WORD	0002	DATA
F3	L NEAR	0067	CODE
FINISH	L NEAR	0092	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 = 0093
RES	L WORD	000C	DATA
ZERO	L NEAR	0080	CODE
@CPU	TEXT	0101h	
@FILENAME	TEXT	lab3	
@VERSION	TEXT	510	

86 Source Lines

86 Total Lines

22 Symbols

48016 + 461291 Bytes symbol space free

0 Warning Errors

0 Severe Errors