

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний Технічний Університет України
«Київський Політехнічний Інститут»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №4
з дисципліни «Системне програмування – 1»
на тему: «Програмування арифметичних операцій підвищеної розрядності»

Виконав:
студент 2-го курсу ФІОТ
групи ІВ-71
Мазан Я. В.
Перевірив:
Старший викладач
Порєв В. М.

Мета: Навчитися програмувати на асемблері основні арифметичні операції підвищеної розрядності, а також отримати перші навички програмування власних процедур у модульному проекті.

Завдання:

1. Створити у середовищі MS Visual Studio проект з ім'ям **Lab4**.
2. Написати вихідний текст програми згідно варіанту завдання. У проекті мають бути три модуля на асемблері:
 - головний модуль: файл **main4.asm**. Цей модуль створити та написати заново, частково використавши текст модуля main3.asm попередньої роботи №3;
 - другий модуль: використати **module** попередньої роботи №3;
 - третій модуль: створити новий з ім'ям **longop**.
3. У цьому проекті кожний модуль може окремо компілюватися.
4. Скомпілювати вихідний текст і отримати виконуємий файл програми.
5. Перевірити роботу програми. Налогодити програму.
6. Отримати результати – кодовані значення чисел згідно варіанту завдання.
7. Проаналізувати та прокоментувати результати, вихідний текст та дизасемблерний машинний код програми.

Індивідуальний варіант:

№ варіанту	Розрядність додавання (біт)	Розрядність віднімання (біт)
9	352	768

Код програми:

main.asm:

```
%include "io.inc"
```

```
%include "add_subtract.asm"
```

```
%macro PRINT_LONG 2 ; length, dd long_number
mov ebp, %1
mov ecx, 0
%%print:
dec ebp
PRINT_HEX 4, [%2 + ecx*4]
PRINT_STRING " "
inc ecx
cmp ebp, 0
jnz %%print
%endmacro
```

```
section .bss
addA1 : resd 11
addB1 : resd 11
res1 : resd 12
addA2 : resd 11
addB2 : resd 11
res2 : resd 12
subA : resd 24
subB : resd 24
res3 : resd 24
```

```
section .data
inscription db "Лабораторну роботу виконав Мазан Ян, ІВ-71", 13, 13, 10, 0
```

```
section .text
global CMAIN
global testProc
global add352Bit
global subtract768Bit
```

```
CMAIN:
nop
mov ebp, esp ; for correct debugging
mov edx, 0 ; pointer
mov eax, 80000001h
@initAdd1:
add eax, 10000h
mov dword [addA1 + 4*edx], eax
mov dword [addB1 + 4*edx], 80000001h
inc edx
cmp edx, 11
jl @initAdd1
mov eax, 9h
mov edx, 0 ; pointer
@initAdd2:
mov dword [addA2 + 4*edx], eax
mov dword [addB2 + 4*edx], 1h
inc eax
inc edx
cmp edx, 11
jl @initAdd2
mov eax, 9h
mov edx, 0
mov dword [subB], eax
@initSub:
mov dword [subA + 4*edx], 0
mov dword [subB + 4*edx], eax
inc eax
inc edx
cmp edx, 24
jl @initSub
PRINT_STRING inscription
PRINT_STRING "A1: "
PRINT_LONG 11, addA1
NEWLINE
PRINT_STRING "B1: "
PRINT_LONG 11, addB1
NEWLINE
NEWLINE
PRINT_STRING "A2: "
PRINT_LONG 11, addA2
NEWLINE
PRINT_STRING "B2: "
PRINT_LONG 11, addB2
NEWLINE
NEWLINE
PRINT_STRING "subA: "
PRINT_LONG 24, subA
NEWLINE
PRINT_STRING "subB: "
PRINT_LONG 24, subB
NEWLINE
NEWLINE
;PRINT_LONG 12, res2
push 11
push addA1
push addB1
```

```

push res1
call addAnyDwordLength
add esp, 16
PRINT_STRING "A1+B1: "
PRINT_LONG 12, res1
NEWLINE
NEWLINE
push 11
push addA2
push addB2
push res2
call addAnyDwordLength
add esp, 16
PRINT_STRING "A2+B2: "
PRINT_LONG 12, res2
NEWLINE
NEWLINE
push 24
push subA
push subB
push res3
call subAnyDwordLength
add esp, 16
PRINT_STRING "A-B: "
PRINT_LONG 24, res3
xor eax, eax
ret

```

add_subtract.asm

```

section .text
global addAnyDwordLength
global subAnyDwordLength

```

```

addAnyDwordLength:
    push ebp
    mov ebp, esp
    mov esi, dword [ebp+20]          ; length
    mov edx, dword [ebp+16]         ; A
    mov ebx, dword [ebp+12]         ; B
    mov ecx, dword [ebp+8]          ; C

    clc
    pushf
    dec esi
    @addCycle:
        popf
        mov eax, dword [edx + esi*4]
        adc eax, dword [ebx + esi*4]
        pushf
        mov dword [ecx+esi*4+4], eax
        dec esi
        cmp esi, 0
        jge @addCycle
        xor eax, eax
        popf                ; забираємо наш прапор переносу зі стеку для того, щоби при наступному додаванні його використати
        adc eax, 0           ; запис прапору переносу в eax
        mov dword [ecx], eax
    leave
    ret

```

```

subAnyDwordLength:
    push ebp
    mov ebp, esp
    mov esi, dword [ebp+20]          ; length
    mov edx, dword [ebp+16]         ; A
    mov ebx, dword [ebp+12]         ; B

```

```
mov ecx, dword [ebp+8] ; C
```

```
clc
pushf
dec esi
@subCycle:
popf
mov eax, dword [edx + esi*4]
sbb eax, dword [ebx + esi*4]
pushf
mov dword [ecx+esi*4], eax
dec esi
cmp esi, 0
jge @subCycle
popf
leave
ret
```

Результати виконання програми:

Output

Лабораторну роботу виконав Мазан Ян, ІВ-71

A1: 80010001 80020001 80030001 80040001 80050001 80060001 80070001 80080001 80090001 800a0001 800b0001
B1: 80000001 80000001 80000001 80000001 80000001 80000001 80000001 80000001 80000001 80000001 80000001

A2: 9 a b c d e f 10 11 12 13
B2: 1 1 1 1 1 1 1 1 1 1 1

subA: 0
subB: 9 a b c d e f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20

A1+B1: 1 10003 20003 30003 40003 50003 60003 70003 80003 90003 a0003 b0002

A2+B2: 0 a b c d e f 10 11 12 13 14

A-B: ffffffff6 ffffffff5 ffffffff4 ffffffff3 ffffffff2 ffffffff1 ffffffff0 fffffffef fffffffee fffffffed fffffffec fffffffeb fffffffea fffffffe9 fffffffe8
fffffffe7 fffffffe6 fffffffe5 fffffffe4 fffffffe3 fffffffe2 fffffffe1 fffffffe0 fffffffe0

Висновок:

Під час виконання даної лабораторної роботи було створено програму, що додає та віднімає деякі числа з заданою розрядністю з підвищеною точністю.

Лабораторну роботу було виконано в середовищі SASM на асемблері NASM.