МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Лабораторна робота №4

з предмету “Системне програмування”

Виконав:

студент 2-го курсу ФІОТ

групи ІО-31

Долинний О.В.

Київ 2015

Лабораторна робота №4

Програмування арифметичних операцій підвищеної розрядності

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. Налагодити програму.

6. Отримати результати – кодовані значення чисел згідно варіанту завдання.

7. Проаналізувати та прокоментувати результати, вихідний текст та дизасембльований машинний код програми.

**Варіант завдання:** 11

Розрядність додавання: 416 біт

Розрядність віднімання: 704 біт

Роздруківка тексту програми

.586

.model flat, stdcall

option casemap :none

include \masm32\include\kernel32.inc

include \masm32\include\user32.inc

include \masm32\include\windows.inc

include module.inc

include longop.inc

includelib \masm32\lib\kernel32.lib

includelib \masm32\lib\user32.lib

.const

.data

.data

Valueb1 db 31

Caption1 db "A+B (1 variant)",0

Caption2 db "A+B (2 variant)",0

Caption3 db "A-B ",0

TextBuf1 db 52 dup(?)

TextBuf2 db 52 dup(?)

TextBuf3 db 88 dup(?)

ValueA1 dd 52 dup(?)

ValueB1 dd 52 dup(?)

ValueA2 dd 52 dup(?)

ValueB2 dd 52 dup(?)

ValueA3 dd 88 dup(?)

ValueB3 dd 88 dup(?)

Result1 dd 52 dup(0)

Result2 dd 52 dup(0)

Result3 dd 88 dup(0)

.code

main:

;А+B 1

mov eax , 80010001h

mov ecx, 52 ; ECX = потрібна кількість повторень

mov edx,0

cycleAB1:

mov DWord ptr[ValueA1+4\*edx], eax

mov DWord ptr[ValueB1+4\*edx], 80000001h

add eax , 10000h

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycleAB1

push offset ValueA1

push offset ValueB1

push offset Result1

call Add\_416\_LONGOP

push offset TextBuf1

push offset Result1

push 416

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuf1, ADDR Caption1,0

;А+B 2

mov eax , 0Bh

mov ecx, 52 ; ECX = потрібна кількість повторень

mov edx,0

cycleAB2:

mov DWord ptr[ValueA2+4\*edx], eax

mov DWord ptr[ValueB2+4\*edx], 00000001h

add eax , 1h

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycleAB2

push offset ValueA2

push offset ValueB2

push offset Result2

call Add\_416\_LONGOP

push offset TextBuf2

push offset Result2

push 416

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuf2, ADDR Caption2,0

;А-B

mov eax , 0Bh

mov ecx, 88 ; ECX = потрібна кількість повторень

mov edx,0

cycleAB3:

mov DWord ptr[ValueA3+4\*edx], 0

mov DWord ptr[ValueB3+4\*edx], eax

add eax , 1h

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycleAB3

push offset ValueA3

push offset ValueB3

push offset Result3

call Sub\_704\_LONGOP

push offset TextBuf3

push offset Result3

push 704

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuf3, ADDR Caption3,0

invoke ExitProcess, 0

end main

.586

.model flat, c

.code

Add\_416\_LONGOP proc

push ebp

mov ebp,esp

mov esi, [ebp+16] ;ESI = адреса A

mov ebx, [ebp+12] ;EBX = адреса B

mov edi, [ebp+8] ;EDI = адреса результату

mov ecx, 52 ; ECX = потрібна кількість повторень

mov edx,0

clc ; обнулює біт CF регістру EFLAGS

cycle:

mov eax, dword ptr[esi+4\*edx]

adc eax, dword ptr[ebx+4\*edx] ; додавання групи з 32 бітів

mov dword ptr[edi+4\*edx], eax

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycle

pop ebp

ret 12

Add\_416\_LONGOP endp

Sub\_704\_LONGOP proc

push ebp

mov ebp,esp

mov esi, [ebp+16] ;ESI = адреса A

mov ebx, [ebp+12] ;EBX = адреса B

mov edi, [ebp+8] ;EDI = адреса результату

mov ecx, 88 ; ECX = потрібна кількість повторень

mov edx,0

clc ; обнулює біт CF регістру EFLAGS

cycle:

mov eax, dword ptr[esi+4\*edx]

sbb eax, dword ptr[ebx+4\*edx] ; віднімання групи з 32 бітів

mov dword ptr[edi+4\*edx], eax

inc edx

dec ecx ; лічильник зменшуємо на 1

jnz cycle

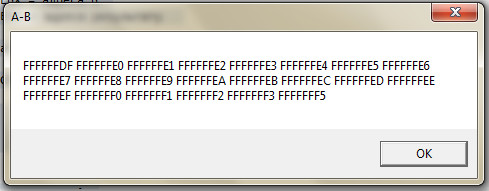
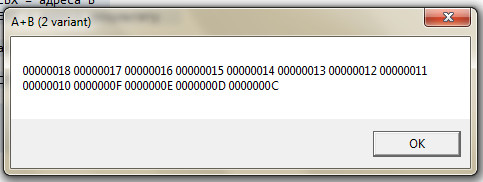
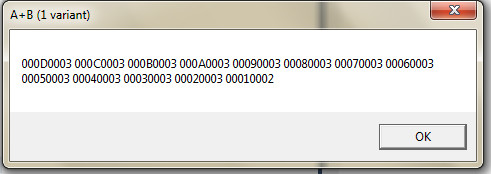
pop ebp

ret 12

Sub\_704\_LONGOP endp

end

**Результати:**



**Аналіз результатів:**

Дана програма виконує операції віднімання і додавання з числами підвищеної розрядності.

Операції ADC та SBB дозволяють нам додавати та віднімати з переносом (ADC) чи позичанням біту (SBB).Наприклад, якщо деяка група бітів містить 32 біти, то результат додавання буде 33 –бітним. 33 біт – біт переносу – автоматично записується у біт CF регістру EFLAGS.

**Висновок:**

Під час виконання лабораторної роботи були покращені навички написання власних модулів, роботи з циклами, а також були закріпленні основні навички в операціях додавання і віднімання чисел з підвищеною розрядністю.