НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №4

з дісципліни **«**Системне програмування**»**

Виконав:

студент 2 курсу

ФІОТ гр. ІО-21

Кузьменко Володимир

Перевірив:

Порєв В.М.

Київ 2014 р.

Додавання - 736 біт  
Віднімання - 384 біти

**Код програми:**

.586

.model flat, stdcall

option casemap:none

include \masm32\include\kernel32.inc

include \masm32\include\user32.inc

include \masm32\include\windows.inc

include \masm32\modules\module.inc

include \masm32\modules\longop.inc

includelib \masm32\lib\kernel32.lib

includelib \masm32\lib\user32.lib

.data

Valueb1 db 31

Captionb1 db "A+B (1 variant)",0

Captionb2 db "A+B (2 variant)",0

Captionb3 db "A-B ",0

TextBuf db 92 dup(?)

TextBuf1 db 48 dup(?)

ValueA dd 92 dup(?)

ValueB dd 92 dup(?)

ValueA1 dd 48 dup(?)

ValueB1 dd 48 dup(?)

Result dd 92 dup(0)

Result1 dd 48 dup(0)

.code

start:

;А+B 1

mov eax , 80010001h

mov esi ,0

cycleAB1:

mov DWord ptr[ValueA+esi], eax

mov DWord ptr[ValueB+esi], 80000001h

add eax , 10000h

add esi,4

cmp esi,92

jb cycleAB1

push offset ValueA

push offset ValueB

push offset Result

call Add\_128\_LONGOP

push offset TextBuf

push offset Result

push 736

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuf, ADDR Captionb1,0

;А+B 2

mov eax , 15h

mov esi ,0

cycleAB2:

mov DWord ptr[ValueA+esi], eax

mov DWord ptr[ValueB+esi], 00000001h

add eax , 1h

add esi,4

cmp esi,92

jb cycleAB2

push offset ValueA

push offset ValueB

push offset Result

call Add\_128\_LONGOP

push offset TextBuf

push offset Result

push 736

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuf, ADDR Captionb2,0

;А-B

mov eax , 15h

mov esi ,0

cycleAB:

mov DWord ptr[ValueA1+esi], 0

mov DWord ptr[ValueB1+esi], eax

add eax , 1h

add esi,4

cmp esi,48

jb cycleAB

push offset ValueA1

push offset ValueB1

push offset Result1

call Sub\_128\_LONGOP

push offset TextBuf1

push offset Result1

push 384

call StrHex\_MY

invoke MessageBoxA, 0, ADDR TextBuf1, ADDR Captionb3,0

invoke ExitProcess,0

end start

**Модуль longop.asm**

.586

.model flat, c

.code

Add\_128\_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, 92 ; 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\_128\_LONGOP endp

Sub\_128\_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, 48 ; 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

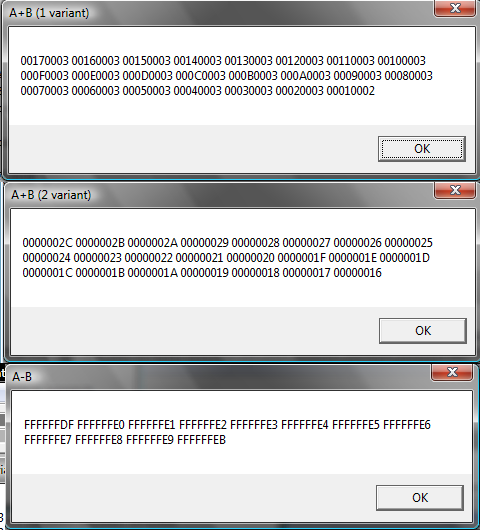
pop ebp

ret 12

Sub\_128\_LONGOP endp

end

**Результати роботи програми:**

****

**Висновок:** Під час виконання лабораторної роботи я закріпив навички програмування на Асемблері. А саме, створення процедур, підключення модулів, механізм циклів. Також я зрозумів що асемблер дозволяє доволі легко запрограмувати арифметичні операції над дуже великими числами. Усе що потрібно це лише додавати(віднімати) окремі групи бітів.

Операції ADC та SBB дозволяють нам це робити 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 він автоматично записується у біт CF регістру EFLAGS.