МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Лабораторна робота №10

з предмету “Системне програмування”

Виконав:

студент 2-го курсу ФІОТ

групи ІО-31

Долинний О.В.

Київ 2015

Лабораторна робота №10

Використання у проекті C++ модулів на асемблері

Мета: Навчитися створювати програми на С++ з використанням модулів на

асемблері.

**Завдання:**

1. Створити у середовищі MS Visual Studio проект C++ з ім’ям Lab10.

2. Написати вихідний текст програми згідно варіанту завдання. У проекті мають бути такі файли вихідного тексту:

- головний файл: lab10.cpp

- файли двох модулів на асемблері: module.asm та longop.asm.

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 машинний код програми.

**Варіант завдання:** 11

|  |  |  |
| --- | --- | --- |
| № варіанту | Вираження | Розрядність |
| 11 | A (B + 1) (C + 2) | 832 |

Роздруківка тексту програми

void MyWork1(HWND hWnd)

{

long oA[26] = { 0x80000001, 0x80000000, 0x80000000, 0x80000000,

0x80000000, 0x80000000, 0x80000000, 0x80000000,

0x80000000, 0x00000000, 0x00040000, 0x00050000,

0x00060000, 0x00070000, 0x00000000, 0x00000000,

0x80000000, 0x80000000, 0x80000000, 0x00000000,

0x80000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x80000000 };

long oB[26] = { 0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x80000000 };

long oC[52] = { 0x00000001, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x80000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x80000000 };

long result1[52] = { 0x80000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000 };

long result2[104] = { 0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000,

0x00000000, 0x00000000, 0x00000000, 0x00000000 };

oB[0] ++;

oC[0] += 2;

char TextBuf[3328];

Mul\_LONGOP(104, result1, oB, oA);

Mul\_LONGOP(208, result2, result1, oC);

StrHex\_MY(3328, result2, TextBuf);

MessageBox(hWnd, TextBuf, "Результат A(B+1)(C+2)", MB\_OK);

}

Mul\_LONGOP proc bytes:DWORD, dest:DWORD, pB:DWORD, pA:DWORD

mov esi, pA ;ESI = адреса множеного

mov ebx, pB ;EBX = адреса множника

mov edi, dest ;EDI = адреса результату

mov ecx, bytes ; кількість байтів

shr ecx, 2

mov cote, ecx

xor ecx, ecx

cycle:

push ebx

push ecx

push edi

shl ecx, 2

add ebx, ecx

add edi, ecx

shr ecx, 2

xor ecx, ecx

cycle\_p: ;для парних доданків

mov eax, [esi+4\*ecx]

mul dword ptr [ebx]

add [edi+4\*ecx], eax

adc [edi+4\*ecx+4], edx

adc byte ptr [edi+4\*ecx+8], 0

inc ecx

inc ecx

cmp ecx, cote

jl cycle\_p

xor ecx, ecx

inc ecx

cycle\_np: ;для непарних доданків

mov eax, [esi+4\*ecx]

mul dword ptr [ebx]

add [edi+4\*ecx], eax

adc [edi+4\*ecx+4], edx

adc byte ptr [edi+4\*ecx+8], 0

inc ecx

inc ecx

cmp ecx, cote

jl cycle\_np

pop edi

pop ecx

pop ebx

inc ecx

cmp ecx, cote

jl cycle

ret

Mul\_LONGOP endp

Add\_LONGOP proc bytes:DWORD, dest:DWORD, pB:DWORD, pA:DWORD

mov esi, pA ;ESI = адреса A

mov ebx, pB ;EBX = адреса B

mov edi, dest ;EDI = адреса результату

mov ecx, bytes ; ECX = потрібна кількість повторень

shr ecx, 2

xor edx, edx

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

cmp edx, ecx

jl cycle

ret

Add\_LONGOP endp

Sub\_LONGOP proc bytes:DWORD, dest:DWORD, pB:DWORD, pA:DWORD

mov esi, pA ;ESI = адреса A

mov ebx, pB ;EBX = адреса B

mov edi, dest ;EDI = адреса результату

mov ecx, bytes ; ECX = потрібна кількість повторень

shr ecx, 2

xor edx, edx

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

cmp edx, ecx

jl cycle

ret

Sub\_LONGOP endp

StrHex\_MY proc bits:DWORD, src:DWORD, dest:DWORD

mov ecx, bits ;кількість бітів числа

cmp ecx, 0

jle @exitp

shr ecx, 3 ;кількість байтів числа

mov esi, src ;адреса числа

mov ebx, dest ;адреса буфера результату

@cycle:

mov dl, byte ptr[esi+ecx-1] ;байт числа - це дві hex-цифри

mov al, dl

shr al, 4

call HexSymbol\_MY

mov byte ptr[ebx], al

mov al, dl

call HexSymbol\_MY

mov byte ptr[ebx+1], al

mov eax, ecx

cmp eax, 4

jle @next

dec eax

and eax, 3

cmp al, 0

jne @next

mov byte ptr[ebx+2], 32

inc ebx

@next:

add ebx, 2

dec ecx

jnz @cycle

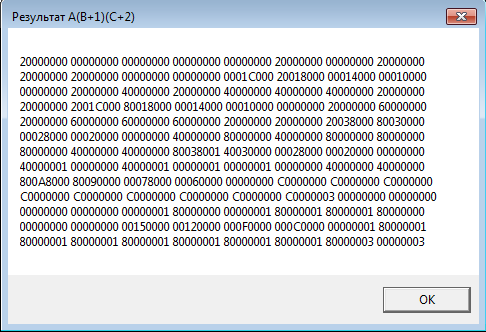
mov byte ptr[ebx], 0

@exitp:

ret

StrHex\_MY endp

**Результати:**



**Висновок:**

Під час виконання лабораторної роботи були покращені навички написання власних модулів, роботи з циклами, а також були закріпленні основні навички в використанні у проекті C++ модулів на асемблері.