Міністерство освіти і науки, молоді та спорту України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №5

З дисципліни «Системне програмування»

Тема: «Програмування множення чисел підвищеної розрядності»

Виконав:
Студент групи ІО-32
Попенко Р.Л.

Перевірив:
ст. викладач
Порєв В. М.

Київ 2015

Мета: Навчитися програмувати на асемблері множення чисел підвищеної розрядності, а також закріпити навички програмування власних процедур у модульному проекті.

Завдання:

1. Створити у середовищі MS Visual Studio проект з ім’ям Lab5.
2. Написати вихідний текст програми згідно варіанту завдання. У проекті мають бути три модуля на асемблері:
- головний модуль: файл main5.asm. Цей модуль створити та написати заново, частково використавши текст модуля main4.asm попередньої роботи №4;
- другий модуль: використати module попередніх робіт №3, 4;
- третій модуль: модуль longop попередньої роботи №4 доповнити новим кодом відповідно завданню.
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.

Текст програми:

```
.586
.model flat, stdcall
include \masm32\include\kernel32.inc
include \masm32\include\user32.inc
include module.inc
include longop.inc
includelib \masm32\lib\kernel32.lib
includelib \masm32\lib\user32.lib

option casemap :none

.data
	Caption db "n!" ,0
	Caption1 db "n! * n!" ,0
	textBuf dd 30 dup(?)
	textBuf1 dd 18 dup(?)

	val dd 12 dup(?)
	val2 dd 1h,0h,0h,0h,0h,0h,0h,0h,0h,0h,0h,0h
	x dd 1h
	n dd 68

	result dd 18 dup(?)


.code
	main:

	@cycle:
```

```
		push offset val
		push offset val2
		push x
		call Mul_N_x_32_LONGOP

	inc x

		mov ecx, 0
		@cycleIn:
			mov eax, dword ptr[val + 4 * ecx]
			mov dword ptr[val2 + 4 * ecx], eax
			inc ecx
			cmp ecx, 10
		jl @cycleIn

	dec n
	jnz @cycle

	push offset textBuf
	push offset val
	push 352
	call StrHex_MY

	invoke MessageBoxA, 0, ADDR textBuf, ADDR Caption, 40h

	push offset val
	push offset val
	push offset result
	call Mul_N_x_N_LONGOP

	push offset textBuf1
	push offset result
	push 704
	call StrHex_MY

	invoke MessageBoxA, 0, ADDR textBuf1, ADDR Caption1, 40h
	invoke ExitProcess, 0

end main
```

Longop.asm

```
.586
.model flat, c

.data
	x dd 1h
	bitNumber dd ?

	a dd 0
	b dd 0
	r dd 0

.code

Mul_N_x_32_LONGOP proc


	push ebp
	mov ebp, esp

	mov esi, [ebp + 16]
	mov edi, [ebp + 12]
	mov ebx, [ebp + 8]
	mov x, ebx

	mov ecx, 6
	xor ebx, ebx
	@cycle1:

		mov eax, dword ptr[edi + 8 * ebx]
```

```
		mul x
		mov dword ptr[esi + 8 * ebx], eax
		mov dword ptr[esi + 8 * ebx + 4], edx

		inc ebx
		dec ecx

		jnz @cycle1


	mov ecx, 6
	xor ebx, ebx

	@cycle2:

		mov eax, dword ptr[edi + 8 * ebx + 4]
		mul x

		clc
		adc eax, dword ptr[esi + 8 * ebx + 4]
		mov dword ptr[esi + 8 * ebx + 4], eax
		clc
		adc edx, dword ptr[esi + 8 * ebx + 8]
		mov dword ptr[esi + 8 * ebx + 8], edx

		inc ebx
		dec ecx

		jnz @cycle2


	pop ebp
	ret 12


Mul_N_x_32_LONGOP endp


Mul_N_x_N_LONGOP proc

push ebp
	mov ebp, esp

	mov esi, dword ptr[ebp + 16]
	mov edi, dword ptr[ebp + 12]
	mov ebx, dword ptr[ebp + 8]

	mov ecx, 12
	@cycle:

		push ecx

		mov ecx, 12
		@cycleInner:
			push ecx

			mov ecx, a
			mov eax, dword ptr[esi + 4 * ecx]

			;clc
			mov ecx, b
			mul dword ptr[edi + 4 * ecx]

			;mov ecx, a
			;add ecx, b
			;mov r, ecx

			mov ecx, r

			clc
			adc eax, dword ptr[ebx + 4 * ecx]
			mov dword ptr[ebx + 4 * ecx], eax
```

```
                    mov eax, dword ptr[ebx + 4 * ecx]

                    ;clc
                    adc edx, dword ptr[ebx + 4 * ecx + 4]
                    mov dword ptr[ebx + 4 * ecx + 4], edx
                    mov eax, dword ptr[ebx + 4 * ecx + 4]

                    inc a
                    inc r
                    pop ecx
                    dec ecx
                    jnz @cycleInner

                inc b

                xor eax, eax
                mov a, eax
                mov eax, b
                mov r, eax
                pop ecx
                dec ecx
                jnz @cycle

        pop ebp
        ret 8

Mul_N_x_N_LONGOP endp

end
```

Аналіз результатів:

Дана програма виконує операції множення і знаходження факторіалу з числами підвищеної точності.

Висновок:

Під час виконання лабораторної роботи були покращені навички написання власних модулів, а також були закріпленні основні навички в операціях множення і знаходження факторіалу чисел з підвищеною точністю.