Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Системне програмування

Лабораторна робота №7

«Виконання операцій з плаваючою
точкою та вивчення команд x87 FPU»

Виконав:
студент групи ІО-24
Довгань М. С.

Перевірив:
Порєв В. М.

Київ - 2024

Тема: Виконання операцій з плаваючою точкою та вивчення команд x87 FPU.

Мета: навчитися програмувати операції з плаваючою точкою на Асемблері.

Завдання:

1. Створити у середовищі MS Visual Studio проект з ім’ям **Lab7**.
2. Написати вихідний текст програми згідно варіанту завдання. У проекті мають бути головний файл **main7.asm** та інші модулі (за необхідності).
3. У цьому проекті кожний модуль може окремо компілюватися.
4. Скомпілювати вихідний текст і отримати виконуваний файл програми.
5. Перевірити роботу програми. Налагодити програму.
6. Отримати результати – числові значення згідно варіанту завдання.
7. Проаналізувати та прокоментувати результати та вихідний текст.

Індивідуальний варіант завдання:

Запрограмувати обчислення математичного вираження на Асемблері на основі команд x87 FPU. Потрібно обчислити:

$Res = \log_2(A_0B_0 + A_1B_1 + \ldots + A_{n-1}B_{n-1})$ із 32-бітовим форматом даних.

Результат надати у вікні MessageBox у вигляді десяткового дробу з цілої частини, точки та 7 розрядів дробової частини, отриманих із 32-бітового формату з плаваючою точкою. Перетворення з двійкового формату з плаваючою точкою у рядок десяткових цифр оформити у вигляді процедури з ім’ям **FloatToDec**.

Виконання завдання:

Роздруківка коду програми:

module.inc:

```
EXTERN FloatToDec : proc
```

module.asm:

```
.586
.model flat, c

.data
      points dw 8
      min dd ?
      result_local dd ?

.code
FloatToDec proc
    push ebp
    mov ebp, esp
    mov edi, [ebp+12]
    mov esi, [ebp+8]

    mov eax, esi
    and eax, 80000000h
    cmp eax, 0

    je @no_sign
        mov byte ptr [edi], 45
        inc edi

    @no_sign:
    mov ecx, edi
    mov eax, esi
    and eax, 7F800000h
    shr eax, 23
    cmp eax, 0
```

```
jne @zero_exp
    mov byte ptr [edi], 48
    jmp @endproc

@zero_exp:
cmp eax, 0FFh
jne @inf_nan
    mov word ptr [edi], 6E69h
    mov word ptr [edi+2], 7974h
    jmp @endproc

@inf_nan:
sub eax, 7Fh
cmp eax, 0
jge @denormalized

mov byte ptr [edi], 48
inc ecx
mov ebx, esi
and ebx, 7FFFFFh
add ebx, 800000h
mov edx, 0FFFFFFFFh
imul edx
mov edx, ecx
mov ecx, eax
shr ebx, cl
mov ecx, edx
jmp @fraction

@denormalized:
jg @greater_than_one
    mov byte ptr [edi], 49
    inc ecx
    mov ebx, esi
    and ebx, 7FFFFFh
    jmp @fraction

@greater_than_one:
push ecx
```

```
mov ecx, 23
sub ecx, eax
push ecx
mov eax, esi
and eax, 7FFFFFh
add eax, 800000h
xor ebx, ebx
mov ebx, 1
shl ebx, cl
mov edx, ebx

@calculate_fraction_mask:
    inc cl
    shl ebx, 1
    add ebx, edx
    cmp cl, 24
jne @calculate_fraction_mask

mov edx, eax
and edx, ebx
mov ebx, eax
sub ebx, edx
pop ecx
shr edx, cl
mov eax, 23
sub eax, ecx
mov ecx, eax
shl ebx, cl

mov eax, edx
pop ecx
push ebx
mov ebx, 10

@integer_part:
    xor edx, edx
    div ebx
    add edx, 48
    mov byte ptr [ecx], dl
```

```
    inc ecx
    cmp eax, 0
jne @integer_part

mov eax, ecx
dec eax

@swap:
    xor edx, edx
    mov dh, byte ptr [eax]
    mov dl, byte ptr [edi]
    mov byte ptr [eax], dl
    mov byte ptr [edi], dh
    inc edi
    dec eax
    cmp edi, eax
jl @swap
pop ebx

@fraction:
    mov byte ptr [ecx], 44
    inc ecx
    mov ax, points

@fraction_loop:
    shl ebx, 1
    mov edx, ebx
    shl edx, 2
    add ebx, edx
    mov edx, ebx
    and edx, 0FF800000h
    shr edx, 23
    add dl, 48
    mov [ecx], dl
    and ebx, 7FFFFFh
    inc ecx
    dec ax
    cmp ax, 0
    jne @fraction_loop
```

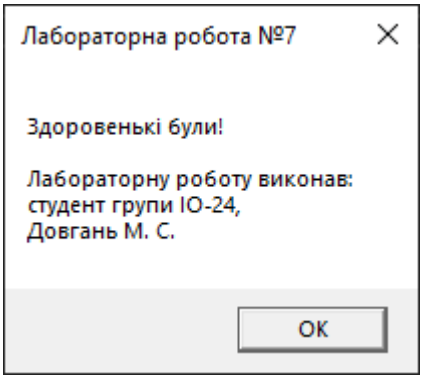
```
    @endproc:
    pop ebp
    ret 8

FloatToDec endp

end
```

main7.asm:

```
.586
.model flat, stdcall

include module.inc

include \masm32\include\user32.inc
include \masm32\include\kernel32.inc

.data
    mainWindowTitle db "Лабораторна робота №7", 0
      mainWindowText db "Здоровенькі були!", 13, 10, 13, 10,
                                "Лабораторну роботу виконав:", 13, 10,
                                "студент групи ІО-24,", 13, 10,
                                "Довгань М. С.", 0

    Text db 100 dup(0), 0
    Caption db "Обчислене математичне вираження", 0

    result dd 0.0
    iterationsNum dd 7
    one dd 1.0

    firstArray dd 3.7, 2.1, -7.5, 5.8, -4.6, -1.8, 9.3
    secondArray dd -5.6, -4.4, 7.3, 3.9, 1.4, -6.4, 6.3

    lastWindowTitle db "Програма завершила роботу", 0
      lastWindowText db "Дякую за увагу!", 0
```

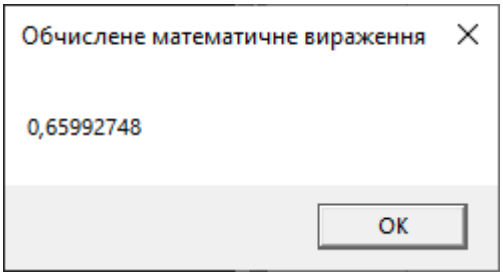
```
.code
main:
    invoke MessageBoxA, 0, ADDR mainWindowText, ADDR mainWindowTitle, 0

    mov edx, 0
    fld one
    fldz

@while:
    fld dword ptr [firstArray + 4 * edx]
    fmul dword ptr [secondArray + 4 * edx]
    faddp st(1), st(0)

    inc edx
    cmp edx, iterationsNum
    jl @while

    fyl2x
    fstp dword ptr [Result]

    push offset Text
    push Result
    call FloatToDec

    invoke MessageBoxA, 0, ADDR Text, ADDR Caption, 0

    invoke MessageBoxA, 0, ADDR lastWindowText, ADDR lastWindowTitle, 0

    invoke ExitProcess, 0

end main
```

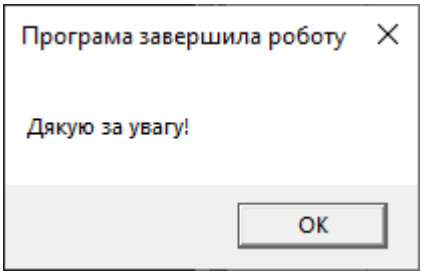
Результати виконання програми:



Перший масив: 3.7, 2.1, -7.5, 5.8, -4.6, -1.8, 9.3;

Другий масив: -5.6, -4.4, 7.3, 3.9, 1.4, -6.4, 6.3:





Аналіз виконання програми:

Створена мною програма виконує завдання лабораторної роботи, відповідно до мого індивідуального варіанту завдання. Значення кількості ітерацій, елементи першого та другого масиву заносяться безпосередньо до програми у файлі main7.asm у розділ .data у вигляді dd - define double word, що значить 4 байти. Одразу при запуску програма видає користувачеві стартове вікно-привітання, в якому міститься привітання, номер лабораторної роботи та інформація про її виконавця. Потім програма обчислює задане індивідуальне математичне вираження. Цей

результат отримується та створюється вікно, в якому безпосередньо міститься результат даного обчислення. Після того, як користувач програми ознайомився зі значенням, для нього виводиться останнє вікно, яке повідомляє його, що вивела всі значення та завершує роботу.

Висновок: під час виконання даної лабораторної роботи я ознайомився, вивчив та навчився програмувати операції з плаваючою точкою на Асемблері.