Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**ЛАБОРАТОРНА РОБОТА № 11**

з дисципліни «Системне програмування-1» на тему

«Програмування команд SEE у модулях на асемблері»

ВИКОНАЛА:

студентка ІІ курсу ФІОТ

групи ІО-64

Бровченко Анастасія

Залікова - 6403

ПЕРЕВІРИВ:

ст.вик. Порєв В. М.

Київ – 2018

**Мета:** Навчитися програмувати модулі на асемблері, у яких містяться команди SSE, команди x87 FPU, а також використовувати такі модулі у проектах C++.

**Завдання:**

1. Створити проект Visual C++ Win32 з ім'ям Lab11.

2. Написати на асемблері процедуру обчислення скалярного добутку двох векторів із використанням команд SSE. Ім'я процедури: **MyDotProduct\_SSE**. Процедуру оформити у окремому модулі і записати файли vectsse.asm, vectsse.h. Додати файл vectsse.asm у проект.

3. Запрограмувати на асемблері процедуру обчислення скалярного добутку двох векторів на основі команд x87 FPU без використання команд SSE. Ім'я процедури: **MyDotProduct\_FPU**. Процедуру оформити у окремому модулі і записати файли vectfpu.asm, vectfpu.h. Додати файл vectfpu.asm у проект.

4. Запрограмувати на C++ обчислення скалярного добутку тих самих векторів як звичайну функцію C++ з ім'ям **MyDotProduct**, яка приймає значення двох масивів і записує результат у числову перемінну (будь-яка оптимізація при компіляції повинна бути відсутня).

5. Зробити меню для вікна програми так, щоб користувач програми мав можливість викликати процедури на асемблері MyDotProduct\_SSE, MyDotProduct\_FPU з модулів vectsse, vectfpu, а також функцію MyDotProduct.

6. Запрограмувати вивід результатів обчислень та виміри часу виконання скалярного добутку для трьох варіантів реалізації.

7. Отримати дизасемблерний текст функції C++ MyDotProduct. Проаналізувати код дизасемблеру, порівняти з кодом на асемблері процедури MyDotProduct\_FPU.

8. Зробити висновки щодо використання модулів на асемблері у програмах на мові C++ .

**Варіант завдання:**

Кількість елементів векторів A та B має бути N=1000

**Роздруківка тексту програми:**

**Lab11.cpp**

// lab11.cpp: определяет точку входа для приложения.

//

#include "stdafx.h"

#include "lab11.h"

#include "vectsse.h"

#include "vectfpu.h"

#define MAX\_LOADSTRING 100

// Глобальные переменные:

HINSTANCE hInst; // текущий экземпляр

TCHAR szTitle[MAX\_LOADSTRING]; // Текст строки заголовка

TCHAR szWindowClass[MAX\_LOADSTRING]; // имя класса главного окна

// Отправить объявления функций, включенных в этот модуль кода:

ATOM MyRegisterClass(HINSTANCE hInstance);

BOOL InitInstance(HINSTANCE, int);

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

INT\_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

int APIENTRY \_tWinMain(HINSTANCE hInstance,

HINSTANCE hPrevInstance,

LPTSTR lpCmdLine,

int nCmdShow)

{

UNREFERENCED\_PARAMETER(hPrevInstance);

UNREFERENCED\_PARAMETER(lpCmdLine);

// TODO: разместите код здесь.

MSG msg;

HACCEL hAccelTable;

// Инициализация глобальных строк

LoadString(hInstance, IDS\_APP\_TITLE, szTitle, MAX\_LOADSTRING);

LoadString(hInstance, IDC\_LAB11, szWindowClass, MAX\_LOADSTRING);

MyRegisterClass(hInstance);

// Выполнить инициализацию приложения:

if (!InitInstance (hInstance, nCmdShow))

{

return FALSE;

}

hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC\_LAB11));

// Цикл основного сообщения:

while (GetMessage(&msg, NULL, 0, 0))

{

if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

}

return (int) msg.wParam;

}

//

// ФУНКЦИЯ: MyRegisterClass()

//

// НАЗНАЧЕНИЕ: регистрирует класс окна.

//

// КОММЕНТАРИИ:

//

// Эта функция и ее использование необходимы только в случае, если нужно, чтобы данный код

// был совместим с системами Win32, не имеющими функции RegisterClassEx'

// которая была добавлена в Windows 95. Вызов этой функции важен для того,

// чтобы приложение получило "качественные" мелкие значки и установило связь

// с ними.

//

ATOM MyRegisterClass(HINSTANCE hInstance)

{

WNDCLASSEX wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDI\_LAB11));

wcex.hCursor = LoadCursor(NULL, IDC\_ARROW);

wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW+1);

wcex.lpszMenuName = MAKEINTRESOURCE(IDC\_LAB11);

wcex.lpszClassName = szWindowClass;

wcex.hIconSm = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI\_SMALL));

return RegisterClassEx(&wcex);

}

//

// ФУНКЦИЯ: InitInstance(HINSTANCE, int)

//

// НАЗНАЧЕНИЕ: сохраняет обработку экземпляра и создает главное окно.

//

// КОММЕНТАРИИ:

//

// В данной функции дескриптор экземпляра сохраняется в глобальной переменной, а также

// создается и выводится на экран главное окно программы.

//

BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)

{

HWND hWnd;

hInst = hInstance; // Сохранить дескриптор экземпляра в глобальной переменной

hWnd = CreateWindow(szWindowClass, szTitle, WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, NULL, NULL, hInstance, NULL);

if (!hWnd)

{

return FALSE;

}

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

return TRUE;

}

\_\_declspec(align(16)) float oA[1000];

\_\_declspec(align(16)) float oB[1000];

\_\_declspec(align(16)) float res;

\_\_declspec(align(16)) char TextBuf[100];

void prepare() {

for (long i = 0; i<1000; i++)

{

oA[i] = 1.0 + i;

oB[i] = pow(-1.0, i);

}

}

void vectorSSE(HWND hWnd) {

prepare();

SYSTEMTIME st;

long tst, ten;

GetLocalTime(&st);

tst = 60000 \* (long)st.wMinute

+ 1000 \* (long)st.wSecond

+ (long)st.wMilliseconds;

for (long i = 0; i<1000000; i++)

{

MyDotProduct\_SSE(&res, oA, oB, 1000);

}

GetLocalTime(&st);

ten = 60000 \* (long)st.wMinute

+ 1000 \* (long)st.wSecond

+ (long)st.wMilliseconds - tst;

sprintf\_s(TextBuf, "Скалярний добуток = %f\nЧас виконання = %ld мс", res, ten);

MessageBox(hWnd, TextBuf, "SSE", MB\_OK);

}

void vectorFPU(HWND hWnd) {

prepare();

SYSTEMTIME st;

long tst, ten;

GetLocalTime(&st);

tst = 60000 \* (long)st.wMinute

+ 1000 \* (long)st.wSecond

+ (long)st.wMilliseconds;

for (long i = 0; i<1000000; i++)

{

MyDotProduct\_FPU(&res, oA, oB, 1000);

}

GetLocalTime(&st);

ten = 60000 \* (long)st.wMinute

+ 1000 \* (long)st.wSecond

+ (long)st.wMilliseconds - tst;

sprintf\_s(TextBuf, "Скалярний добуток = %f\nЧас виконання = %ld мс", res, ten);

MessageBox(hWnd, TextBuf, "FPU", MB\_OK);

}

float MyDotProduct(float \*A, float \*B, long N) {

float r = 0;

for (long i = 0; i < N; i++) {

r += A[i] \* B[i];

}

return r;

}

void vectorCPP(HWND hWnd) {

prepare();

SYSTEMTIME st;

long tst, ten;

GetLocalTime(&st);

tst = 60000 \* (long)st.wMinute

+ 1000 \* (long)st.wSecond

+ (long)st.wMilliseconds;

for (long i = 0; i<1000000; i++)

{

res = MyDotProduct(oA, oB, 1000);

}

GetLocalTime(&st);

ten = 60000 \* (long)st.wMinute

+ 1000 \* (long)st.wSecond

+ (long)st.wMilliseconds - tst;

sprintf\_s(TextBuf, "Скалярний добуток = %f\nЧас виконання = %ld мс", res, ten);

MessageBox(hWnd, TextBuf, "C++", MB\_OK);

}

//

// ФУНКЦИЯ: WndProc(HWND, UINT, WPARAM, LPARAM)

//

// НАЗНАЧЕНИЕ: обрабатывает сообщения в главном окне.

//

// WM\_COMMAND - обработка меню приложения

// WM\_PAINT -Закрасить главное окно

// WM\_DESTROY - ввести сообщение о выходе и вернуться.

//

//

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

{

int wmId, wmEvent;

PAINTSTRUCT ps;

HDC hdc;

switch (message)

{

case WM\_COMMAND:

wmId = LOWORD(wParam);

wmEvent = HIWORD(wParam);

// Разобрать выбор в меню:

switch (wmId)

{

case ID\_32771:

vectorCPP(hWnd);

break;

case ID\_32772:

vectorSSE(hWnd);

break;

case ID\_32773:

vectorFPU(hWnd);

break;

case IDM\_ABOUT:

DialogBox(hInst, MAKEINTRESOURCE(IDD\_ABOUTBOX), hWnd, About);

break;

case IDM\_EXIT:

DestroyWindow(hWnd);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

break;

case WM\_PAINT:

hdc = BeginPaint(hWnd, &ps);

// TODO: добавьте любой код отрисовки...

EndPaint(hWnd, &ps);

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

return 0;

}

// Обработчик сообщений для окна "О программе".

INT\_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

switch (message)

{

case WM\_INITDIALOG:

return (INT\_PTR)TRUE;

case WM\_COMMAND:

if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)

{

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

}

break;

}

return (INT\_PTR)FALSE;

}

**Vectfpu.asm**

.586

.model flat, c

.data

.code

MyDotProduct\_FPU proc dest:DWORD, A:DWORD, B:DWORD, N:DWORD

mov eax, A

mov ebx, B

mov edx, dest

mov ecx, N

dec ecx

fldz

@cycle:

fld dword ptr[eax+4\*ecx]

fmul dword ptr[ebx+4\*ecx]

faddp st(1), st(0)

dec ecx

jge @cycle

fstp dword ptr[edx]

ret

MyDotProduct\_FPU endp

End

**Vectsse.asm**

.686

.xmm

.model flat, C

.data

.code

MyDotProduct\_SSE proc dest:DWORD, A:DWORD, B:DWORD, N:DWORD

mov eax, A

mov ebx, B

mov edi, dest

mov ecx, N

xorps xmm2, xmm2

@cycle:

movaps xmm0, [eax+4\*ecx]

movaps xmm1, [ebx+4\*ecx]

mulps xmm0, xmm1

haddps xmm0, xmm0

haddps xmm0, xmm0

addps xmm2, xmm0

sub ecx, 4

jge @cycle

movaps [edi], xmm2

ret

MyDotProduct\_SSE endp

end