**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Національний Технічний Університет України**

**«Київський Політехнічний Інститут»**

*Факультет інформатики та обчислювальної техніки*

*Кафедра обчислювальної техніки*

**Лабораторна робота №2**

*з дисципліни «Системне програмування – 1»*

*на тему: «Знайомство із середовищем розробки програм Microsoft Visual Studio»*

**Виконав:**

студент 2-го курсу ФІОТ

групи ІО-44

*Барабаш Т.А.*

**Перевірив:**

Старший викладач

*Порєв В. М.*

**Київ – 2016**

**ЛАБОРАТОРНА РОБОТА №2**

*Знайомство із середовищем розробки програм Microsoft Visual Studio*

**Мета:** Отримати перші навички роботи з Microsoft Visual Studio для створення програм, написаних мовою асемблера, а також вивчити команди MOV та CPUID.

**I. Завдання**

1. Створити у середовищі MS Visual Studio проект з ім’ям Lab2. Встановити необхідні параметри проекту – опції середовища розробки програм.
2. Написати вихідний текст програми на асемблері, додати файл вихідного тексту у проект. Зміст вихідного тексту згідно з варіантом завдання.
3. Скомпілювати вихідний текст і отримати виконуємий файл програми.
4. Перевірити роботу програми. Налагодити програму.
5. Отримати дизасембльований текст машинного коду і проаналізувати його.

**ІІ. Код програми**

.586

.model flat, stdcall

include \masm32\include\kernel32.inc

include \masm32\include\user32.inc

includelib \masm32\lib\kernel32.lib

includelib \masm32\lib\user32.lib

.data

Caption db "Лабораторна робота №2",0

TextName db "Виконав: Барабаш Т.А.",0

res dd 256 dup(0)

Text db "EAX=xxxxxxxx", 13, 10,

"EBX=xxxxxxxx", 13, 10,

"ECX=xxxxxxxx", 13, 10,

"EDX=xxxxxxxx", 0

Caption0 db "Cpuid 0", 0

Caption1 db "Cpuid 1", 0

Caption2 db "Cpuid 2", 0

Caption00 db "Cpuid 80000000h", 0

Caption01 db "Cpuid 80000001h", 0

Caption02 db "Cpuid 80000002h", 0

Caption03 db "Cpuid 80000003h", 0

Caption04 db "Cpuid 80000004h", 0

Caption05 db "Cpuid 80000005h", 0

Caption08 db "Cpuid 80000008h", 0

Model db 32 dup(0)

CaptionModel db "CPUID 0 Model", 0

.code

;ця процедура записує 8 символів HEX коду числа

;перший параметр - 32-бітове число

;другий параметр - адреса буфера тексту

DwordToStrHex proc

push ebp

mov ebp,esp

mov ebx,[ebp+8] ;другий параметр

mov edx,[ebp+12] ;перший параметр

xor eax,eax

mov edi,7

@next:

mov al,dl

and al,0Fh ;виділяємо одну шістнадцяткову цифру

add ax,48 ;так можна тільки для цифр 0-9

cmp ax,58

jl @store

add ax,7 ;для цифр A,B,C,D,E,F

@store:

mov [ebx+edi],al

shr edx,4

dec edi

cmp edi,0

jge @next

pop ebp

ret 8

DwordToStrHex endp

main:

invoke MessageBoxA, 0, ADDR TextName, ADDR Caption, 0

mov eax, 0

cpuid

mov dword ptr[Model], ebx

mov dword ptr[Model+4], edx

mov dword ptr[Model+8], ecx

invoke MessageBoxA, 0, ADDR Model, ADDR CaptionModel, 0

mov eax, 0

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption0, 0

mov eax, 1

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption1, 0

mov eax, 2

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption2, 0

mov eax, 80000000h

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption00, 0

mov eax, 80000001h

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption01, 0

mov eax, 80000002h

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption02, 0

mov eax, 80000003h

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption03, 0

mov eax, 80000004h

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption04, 0

mov eax, 80000005h

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption05, 0

mov eax, 80000008h

cpuid

mov dword ptr[res], eax

mov dword ptr[res+4], ebx

mov dword ptr[res+8], ecx

mov dword ptr[res+12], edx

push [res]

push offset [Text+4]

call DwordToStrHex

push offset [Text+18]

call DwordToStrHex

push offset [Text+32]

call DwordToStrHex

push offset [Text+46]

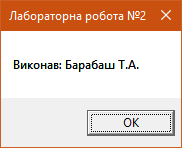
call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text, ADDR Caption08, 0

invoke ExitProcess, 0

end main;

**III. Результат**



**ІV. Висновок**

У ході виконання лабораторної роботи було закріплено на практиці навички використання Microsoft Visual Studio 2015, а також було написано просту програму, яка відобразила інформацію про процесор комп’ютера.