

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

Системне програмування

Лабораторна робота №1

«Знайомство із засобами розробки програм на асемблері»

Виконав:
студент групи ІО-24
Довгань Максим

Перевірив:
Порєв В. М.

Київ - 2024

Тема: Знайомство із засобами розробки програм на асемблері.

Мета: навчитися створювати проекти програм на асемблері у середовищах розробки програмного забезпечення та отримати перші навички налагодження програм.

Завдання:

1. Інсталювати програмний пакет MASM32. Написати вихідний текст найпростішої програми **Lab1** на асемблері. Скомпілювати вихідний текст і отримати виконуваний файл програми. Перевірити роботу програми.

2. Інсталювати Microsoft Visual Studio. Створити у середовищі MS Visual Studio проект з ім'ям **Lab1_cpuid**. Встановити необхідні параметри проекту – опції середовища розробки програм. Написати вихідний текст програми згідно з варіантом завдання. Скомпілювати вихідний текст і отримати виконуваний файл програми. Перевірити та налагодити програму.

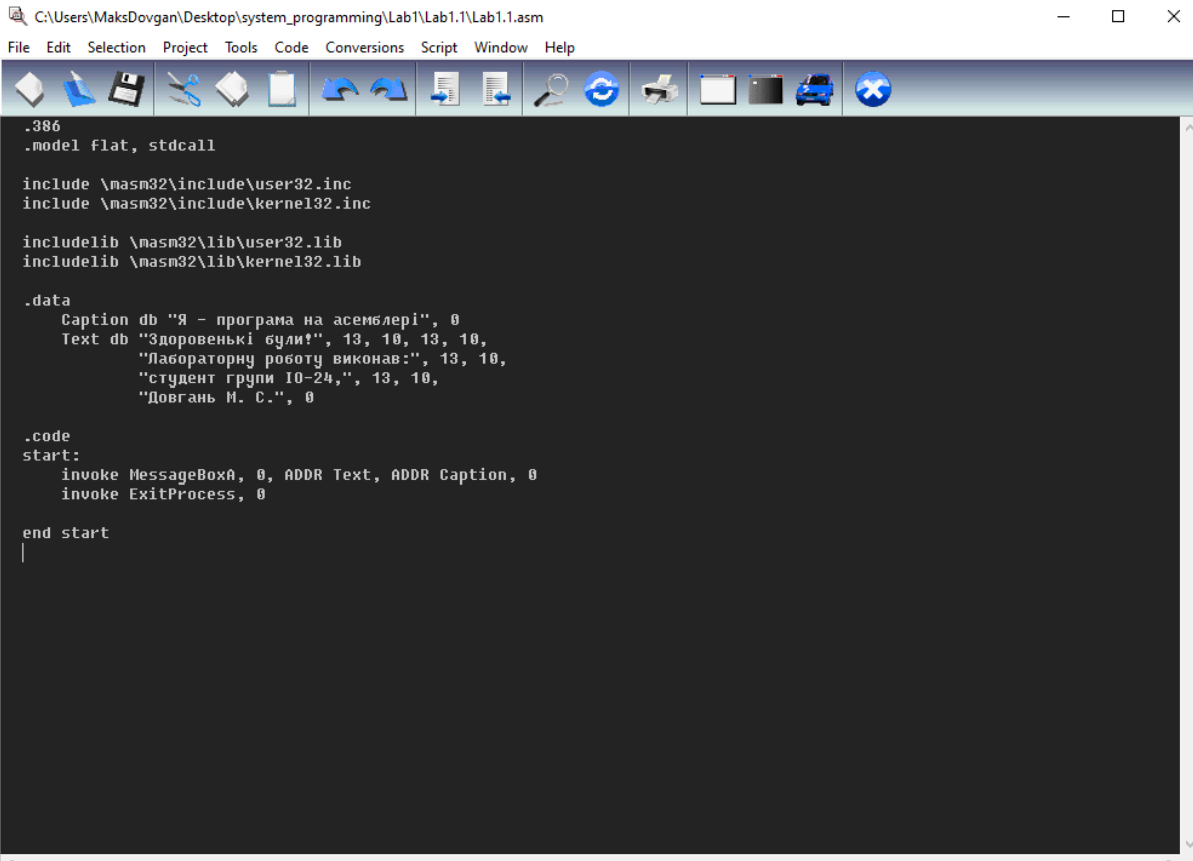
Отримати дизасемблерний текст машинного коду і проаналізувати його.

3. У звіті по лабораторній роботі надати опис програм **Lab1** та **Lab1_cpuid**.

Виконання завдання:

Завдання 1:

Робота у MASM32:



```
.386
.model flat, stdcall

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

includelib \masm32\lib\user32.lib
includelib \masm32\lib\kernel32.lib

.data
    Caption db "Я - програма на асемблері", 0
    Text db "Здоровенькі були!", 13, 10, 13, 10,
        "Лабораторну роботу виконав:", 13, 10,
        "студент групи ІО-24,", 13, 10,
        "Довгань М. С.", 0

.code
start:
    invoke MessageBoxA, 0, ADDR Text, ADDR Caption, 0
    invoke ExitProcess, 0

end start
|
```

In 22 col 10 F9 Indent ON Press F12 to repeat operation

Роздруківка коду програми:

Lab1.asm:

```
.386
.model flat, stdcall

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

includelib \masm32\lib\user32.lib
includelib \masm32\lib\kernel32.lib
```

```

.data
    Caption db "Я - програма на асемблері", 0
    Text db "Здоровенькі були!", 13, 10, 13, 10,
           "Лабораторну роботу виконав:", 13, 10,
           "студент групи ІО-24,", 13, 10,
           "Довгань М. С.", 0

.code
start:
    invoke MessageBoxA, 0, ADDR Text, ADDR Caption, 0
    invoke ExitProcess, 0

end start

```

Дизасемблерний код:

```

--- C:\Users\MaksDovgan\Desktop\system_programming\Lab1\Lab1.1\Lab1.asm
-----

.386
.model flat, stdcall

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

includelib \masm32\lib\user32.lib
includelib \masm32\lib\kernel32.lib

.data
    Caption db "Я - програма на асемблері", 0
    Text db "Здоровенькі були!", 13, 10, 13, 10,
           "Автор: Довгань М. С.", 0

.code

start:
    invoke MessageBoxA, 0, ADDR Text, ADDR Caption, 0
00261000 push        0
00261002 push        offset Caption (0264000h)

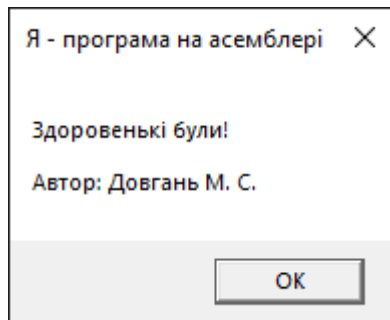
```

```

00261007 push      offset Text (026401Ah)
0026100C push      0
0026100E call       _MessageBoxA@16 (0261026h)
        invoke ExitProcess, 0
00261013 push      0
00261015 call       _ExitProcess@4 (0261020h)

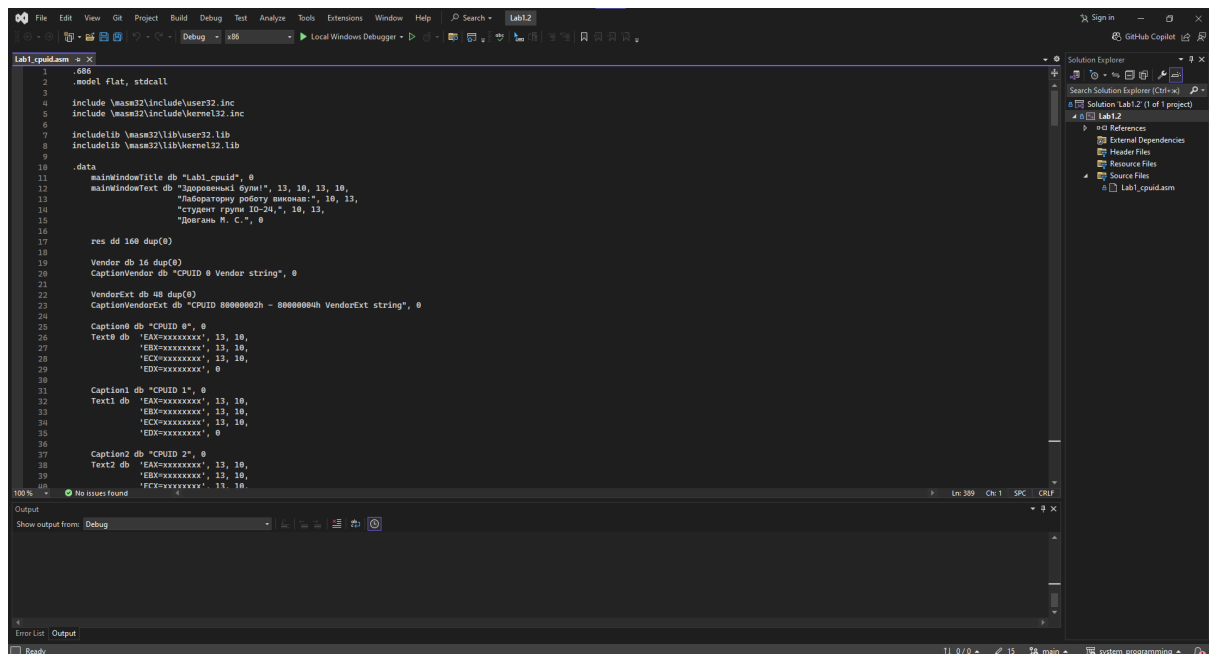
```

Результати виконання програми:



Завдання 2:

Робота у Microsoft Visual Studio:



Роздруківка коду програми:

Lab1_cpuid.asm:

```
.686
.model flat, stdcall

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

includelib \masm32\lib\user32.lib
includelib \masm32\lib\kernel32.lib

.data
    mainWindowTitle db "Lab1_cpuid", 0
    mainWindowText db "Здоровенькі були!", 13, 10, 13, 10,
        "Лабораторну роботу виконав:", 10, 13,
        "студент групи ІО-24,", 10, 13,
        "Довгань М. С.", 0

    res dd 160 dup(0)

    Vendor db 16 dup(0)
    CaptionVendor db "CPUID 0 Vendor string", 0

    VendorExt db 48 dup(0)
    CaptionVendorExt db "CPUID 80000002h - 80000004h VendorExt string", 0

    Caption0 db "CPUID 0", 0
    Text0 db 'EAX=xxxxxxxx', 13, 10,
        'EBX=xxxxxxxx', 13, 10,
        'ECX=xxxxxxxx', 13, 10,
        'EDX=xxxxxxxx', 0

    Caption1 db "CPUID 1", 0
    Text1 db 'EAX=xxxxxxxx', 13, 10,
        'EBX=xxxxxxxx', 13, 10,
        'ECX=xxxxxxxx', 13, 10,
        'EDX=xxxxxxxx', 0
```

```
Caption2 db "CPUID 2", 0
Text2 db  'EAX=xxxxxxxx', 13, 10,
          'EBX=xxxxxxxx', 13, 10,
          'ECX=xxxxxxxx', 13, 10,
          'EDX=xxxxxxxx', 0

Caption00h db "CPUID 80000000h", 0
Text00h db  'EAX=xxxxxxxx', 13, 10,
          'EBX=xxxxxxxx', 13, 10,
          'ECX=xxxxxxxx', 13, 10,
          'EDX=xxxxxxxx', 0

Caption01h db "CPUID 80000001h", 0
Text01h db  'EAX=xxxxxxxx', 13, 10,
          'EBX=xxxxxxxx', 13, 10,
          'ECX=xxxxxxxx', 13, 10,
          'EDX=xxxxxxxx', 0

Caption02h db "CPUID 80000002h", 0
Text02h db  'EAX=xxxxxxxx', 13, 10,
          'EBX=xxxxxxxx', 13, 10,
          'ECX=xxxxxxxx', 13, 10,
          'EDX=xxxxxxxx', 0

Caption03h db "CPUID 80000003h", 0
Text03h db  'EAX=xxxxxxxx', 13, 10,
          'EBX=xxxxxxxx', 13, 10,
          'ECX=xxxxxxxx', 13, 10,
          'EDX=xxxxxxxx', 0

Caption04h db "CPUID 80000004h", 0
Text04h db  'EAX=xxxxxxxx', 13, 10,
          'EBX=xxxxxxxx', 13, 10,
          'ECX=xxxxxxxx', 13, 10,
          'EDX=xxxxxxxx', 0

Caption05h db "CPUID 80000005h", 0
Text05h db  'EAX=xxxxxxxx', 13, 10,
          'EBX=xxxxxxxx', 13, 10,
```

```
'ECX=xxxxxxxx', 13, 10,  
'EDX=xxxxxxxx', 0
```

```
Caption08h db "CPUID 80000008h", 0  
Text08h db 'EAX=xxxxxxxx', 13, 10,  
        'EBX=xxxxxxxx', 13, 10,  
        'ECX=xxxxxxxx', 13, 10,  
        'EDX=xxxxxxxx', 0
```

.code

DwordToStrHex proc

```
    push ebp  
    mov ebp, esp  
    mov ebx, [ebp+8]  
    mov edx, [ebp+12]  
    xor eax, eax  
    mov edi, 7
```

@next:

```
    mov al, dl  
    and al, 0Fh  
    add ax, 48  
    cmp ax, 58  
    jl @store  
    add ax, 7
```

@store:

```
    mov [ebx+edi], al  
    shr edx, 4  
    dec edi  
    cmp edi, 0  
    jge @next  
    pop ebp  
    ret 8
```

DwordToStrHex endp

main:


```
; ----- Main Window -----
```

```
invoke MessageBoxA, 0, ADDR mainWindowText, ADDR mainWindowTitle, 0
```

```
; ----- CPUID 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
```

```
mov dword ptr[Vendor], ebx
```

```
mov dword ptr[Vendor+4], edx
```

```
mov dword ptr[Vendor+8], ecx
```

```
push [res]
```

```
push offset [Text0+4]
```

```
call DwordToStrHex
```

```
push [res+4]
```

```
push offset [Text0+18]
```

```
call DwordToStrHex
```

```
push [res+8]
```

```
push offset [Text0+32]
```

```
call DwordToStrHex
```

```
push [res+12]
```

```
push offset [Text0+46]
```

```
call DwordToStrHex
```

```
invoke MessageBoxA, 0, ADDR Text0, ADDR Caption0, 0
```

```
; ----- Vendor -----
```

```
invoke MessageBoxA, 0, ADDR Vendor, ADDR CaptionVendor, 0
```

```
; ----- CPUID 1 -----
```

```
mov eax, 1
```

```

cpuid
mov dword ptr[res+16], eax
mov dword ptr[res+20], ebx
mov dword ptr[res+24], ecx
mov dword ptr[res+28], edx

push [res+16]
push offset [Text1+4]
call DwordToStrHex
push [res+20]
push offset [Text1+18]
call DwordToStrHex
push [res+24]
push offset [Text1+32]
call DwordToStrHex
push [res+28]
push offset [Text1+46]
call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text1, ADDR Caption1, 0

; ----- CPUID 2 -----

mov eax, 2
cpuid
mov dword ptr[res+32], eax
mov dword ptr[res+36], ebx
mov dword ptr[res+40], ecx
mov dword ptr[res+44], edx

push [res+32]
push offset [Text2+4]
call DwordToStrHex
push [res+36]
push offset [Text2+18]
call DwordToStrHex
push [res+40]
push offset [Text2+32]
call DwordToStrHex

```

```

push [res+44]
push offset [Text2+46]
call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text2, ADDR Caption2, 0

; ----- CPUID 80000000h -----

mov eax, 80000000h
cpuid
mov dword ptr[res+48], eax
mov dword ptr[res+52], ebx
mov dword ptr[res+56], ecx
mov dword ptr[res+60], edx

push [res+48]
push offset [Text00h+4]
call DwordToStrHex
push [res+52]
push offset [Text00h+18]
call DwordToStrHex
push [res+56]
push offset [Text00h+32]
call DwordToStrHex
push [res+60]
push offset [Text00h+46]
call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text00h, ADDR Caption00h, 0

; ----- CPUID 80000001h -----

mov eax, 80000001h
cpuid
mov dword ptr[res+64], eax
mov dword ptr[res+68], ebx
mov dword ptr[res+72], ecx
mov dword ptr[res+76], edx

```

```

push [res+64]
push offset [Text01h+4]
call DwordToStrHex
push [res+68]
push offset [Text01h+18]
call DwordToStrHex
push [res+72]
push offset [Text01h+32]
call DwordToStrHex
push [res+76]
push offset [Text01h+46]
call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text01h, ADDR Caption01h, 0

; ----- CPUID 80000002h -----

mov eax, 80000002h
cpuid
mov dword ptr[res+80], eax
mov dword ptr[res+84], ebx
mov dword ptr[res+88], ecx
mov dword ptr[res+92], edx

mov dword ptr[VendorExt], eax
mov dword ptr[VendorExt+4], ebx
mov dword ptr[VendorExt+8], ecx
mov dword ptr[VendorExt+12], edx

push [res+80]
push offset [Text02h+4]
call DwordToStrHex
push [res+84]
push offset [Text02h+18]
call DwordToStrHex
push [res+88]
push offset [Text02h+32]
call DwordToStrHex
push [res+92]

```

```

push offset [Text02h+46]
call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text02h, ADDR Caption02h, 0

; ----- CPUID 80000003h -----

mov eax, 80000003h
cpuid
mov dword ptr[res+96], eax
mov dword ptr[res+100], ebx
mov dword ptr[res+104], ecx
mov dword ptr[res+108], edx

mov dword ptr[VendorExt+16], eax
mov dword ptr[VendorExt+20], ebx
mov dword ptr[VendorExt+24], ecx
mov dword ptr[VendorExt+28], edx

push [res+96]
push offset [Text03h+4]
call DwordToStrHex
push [res+100]
push offset [Text03h+18]
call DwordToStrHex
push [res+104]
push offset [Text03h+32]
call DwordToStrHex
push [res+108]
push offset [Text03h+46]
call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text03h, ADDR Caption03h, 0

; ----- CPUID 80000004h -----

mov eax, 80000004h
cpuid
mov dword ptr[res+112], eax

```

```
mov dword ptr[res+116], ebx
mov dword ptr[res+120], ecx
mov dword ptr[res+124], edx
```

```
mov dword ptr[VendorExt+32], eax
mov dword ptr[VendorExt+36], ebx
mov dword ptr[VendorExt+40], ecx
mov dword ptr[VendorExt+44], edx
```

```
push [res+112]
push offset [Text04h+4]
call DwordToStrHex
push [res+116]
push offset [Text04h+18]
call DwordToStrHex
push [res+120]
push offset [Text04h+32]
call DwordToStrHex
push [res+124]
push offset [Text04h+46]
call DwordToStrHex
```

```
invoke MessageBoxA, 0, ADDR Text04h, ADDR Caption04h, 0
```

```
; ----- VendorExt -----
```

```
invoke MessageBoxA, 0, ADDR VendorExt, ADDR CaptionVendorExt, 0
```

```
; ----- CPUID 80000005h -----
```

```
mov eax, 80000005h
cpuid
mov dword ptr[res+128], eax
mov dword ptr[res+132], ebx
mov dword ptr[res+136], ecx
mov dword ptr[res+140], edx
```

```
push [res+128]
push offset [Text05h+4]
```

```

call DwordToStrHex
push [res+132]
push offset [Text05h+18]
call DwordToStrHex
push [res+136]
push offset [Text05h+32]
call DwordToStrHex
push [res+140]
push offset [Text05h+46]
call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text05h, ADDR Caption05h, 0

; ----- CPUID 80000008h -----

mov eax, 80000008h
cpuid
mov dword ptr[res+144], eax
mov dword ptr[res+148], ebx
mov dword ptr[res+152], ecx
mov dword ptr[res+156], edx

push [res+144]
push offset [Text08h+4]
call DwordToStrHex
push [res+148]
push offset [Text08h+18]
call DwordToStrHex
push [res+152]
push offset [Text08h+32]
call DwordToStrHex
push [res+156]
push offset [Text08h+46]
call DwordToStrHex

invoke MessageBoxA, 0, ADDR Text08h, ADDR Caption08h, 0

invoke ExitProcess, 0
end main

```

Дизасемблерный код:

C:\Users\MaksDovgan\Desktop\system_programming\Lab1\Lab1.2\Lab1_cpuid.asm

--

```
        push ebp
001B1010 push        ebp
        mov ebp, esp
001B1011 mov         ebp, esp
        mov ebx, [ebp+8]
001B1013 mov         ebx, dword ptr [ebp+8]
        mov edx, [ebp+12]
001B1016 mov         edx, dword ptr [ebp+0Ch]
        xor eax, eax
001B1019 xor         eax, eax
        mov edi, 7
001B101B mov         edi, 7

        @next:
        mov al, dl
001B1020 mov         al, dl
        and al, 0Fh
001B1022 and         al, 0Fh
        add ax, 48
001B1024 add         ax, 30h
        cmp ax, 58
001B1028 cmp         ax, 3Ah
        jl @store
001B102C jl         @next+12h (01B1032h)
        add ax, 7
001B102E add         ax, 7

        @store:
        mov [ebx+edi], al
001B1032 mov         byte ptr [ebx+edi], al
        shr edx, 4
001B1035 shr         edx, 4
        dec edi
001B1038 dec         edi
```



```

        cmp edi, 0
001B1039  cmp          edi,0
        jge @next
001B103C  jge          @next (01B1020h)
        pop ebp
001B103E  pop          ebp
        ret 8
001B103F  ret          8

```

DwordToStrHex endp

main:

; ----- Main Window -----

```

        invoke MessageBoxA, 0, ADDR mainWindowText, ADDR mainWindowTitle, 0
001B1042  push          0
001B1044  push          offset mainWindowTitle (01B4000h)
001B1049  push          offset mainWindowText (01B400Bh)
001B104E  push          0
001B1050  call          _MessageBoxA@16 (01B1695h)

```

; ----- CPUID 0 -----

```

        mov eax, 0
001B1055  mov          eax,0
        cpuid
001B105A  cpuid
        mov dword ptr[res], eax
001B105C  mov          dword ptr [res (01B4061h)],eax
        mov dword ptr[res+4], ebx
001B1061  mov          dword ptr ds:[1B4065h],ebx
        mov dword ptr[res+8], ecx
001B1067  mov          dword ptr ds:[1B4069h],ecx
        mov dword ptr[res+12], edx
001B106D  mov          dword ptr ds:[1B406Dh],edx

        mov dword ptr[Vendor], ebx
001B1073  mov          dword ptr [Vendor (01B42E1h)],ebx

```

```

        mov dword ptr [Vendor+4], edx
001B1079  mov          dword ptr ds:[1B42E5h],edx
        mov dword ptr [Vendor+8], ecx
001B107F  mov          dword ptr ds:[1B42E9h],ecx

        push [res]
001B1085  push          dword ptr [res (01B4061h)]
        push offset [Text0+4]
001B108B  push          1B4370h
        call DwordToStrHex
001B1090  call          DwordToStrHex (01B1010h)
        push [res+4]
001B1095  push          dword ptr ds:[1B4065h]
        push offset [Text0+18]
001B109B  push          1B437Eh
        call DwordToStrHex
001B10A0  call          DwordToStrHex (01B1010h)
        push [res+8]
001B10A5  push          dword ptr ds:[1B4069h]
        push offset [Text0+32]
001B10AB  push          1B438Ch
        call DwordToStrHex
001B10B0  call          DwordToStrHex (01B1010h)
        push [res+12]
001B10B5  push          dword ptr ds:[1B406Dh]
        push offset [Text0+46]
001B10BB  push          1B439Ah
        call DwordToStrHex
001B10C0  call          DwordToStrHex (01B1010h)

        invoke MessageBoxA, 0, ADDR Text0, ADDR Caption0, 0
001B10C5  push          0
001B10C7  push          offset Caption0 (01B4364h)
001B10CC  push          offset Text0 (01B436Ch)
001B10D1  push          0
001B10D3  call          _MessageBoxA@16 (01B1695h)

; ----- Vendor -----

```

```

        invoke MessageBoxA, 0, ADDR Vendor, ADDR CaptionVendor, 0
001B10D8  push          0
001B10DA  push          offset CaptionVendor (01B42F1h)
001B10DF  push          offset Vendor (01B42E1h)
001B10E4  push          0
001B10E6  call          _MessageBoxA@16 (01B1695h)

; ----- CPUID 1 -----

        mov eax, 1
001B10EB  mov            eax, 1
        cpuid
001B10F0  cpuid
        mov dword ptr[res+16], eax
001B10F2  mov            dword ptr ds:[001B4071h], eax
        mov dword ptr[res+20], ebx
001B10F7  mov            dword ptr ds:[1B4075h], ebx
        mov dword ptr[res+24], ecx
001B10FD  mov            dword ptr ds:[1B4079h], ecx
        mov dword ptr[res+28], edx
001B1103  mov            dword ptr ds:[1B407Dh], edx

        push [res+16]
001B1109  push          dword ptr ds:[1B4071h]
        push offset [Text1+4]
001B110F  push          1B43AFh
        call DwordToStrHex
001B1114  call          DwordToStrHex (01B1010h)
        push [res+20]
001B1119  push          dword ptr ds:[1B4075h]
        push offset [Text1+18]
001B111F  push          1B43BDh
        call DwordToStrHex
001B1124  call          DwordToStrHex (01B1010h)
        push [res+24]
001B1129  push          dword ptr ds:[1B4079h]
        push offset [Text1+32]
001B112F  push          1B43CBh
        call DwordToStrHex

```

```

001B1134  call          DwordToStrHex (01B1010h)
          push [res+28]
001B1139  push           dword ptr ds:[1B407Dh]
          push offset [Text1+46]
001B113F  push           1B43D9h
          call DwordToStrHex
001B1144  call          DwordToStrHex (01B1010h)

          invoke MessageBoxA, 0, ADDR Text1, ADDR Caption1, 0
001B1149  push           0
001B114B  push           offset Caption1 (01B43A3h)
001B1150  push           offset Text1 (01B43ABh)
001B1155  push           0
001B1157  call          _MessageBoxA@16 (01B1695h)

          ; ----- CPUID 2 -----

          mov eax, 2
001B115C  mov           eax, 2
          cpuid
001B1161  cpuid
          mov dword ptr[res+32], eax
001B1163  mov           dword ptr ds:[001B4081h], eax
          mov dword ptr[res+36], ebx
001B1168  mov           dword ptr ds:[1B4085h], ebx
          mov dword ptr[res+40], ecx
001B116E  mov           dword ptr ds:[1B4089h], ecx
          mov dword ptr[res+44], edx
001B1174  mov           dword ptr ds:[1B408Dh], edx

          push [res+32]
001B117A  push           dword ptr ds:[1B4081h]
          push offset [Text2+4]
001B1180  push           1B43EEh
          call DwordToStrHex
001B1185  call          DwordToStrHex (01B1010h)
          push [res+36]
001B118A  push           dword ptr ds:[1B4085h]
          push offset [Text2+18]

```

```

001B1190  push          1B43FCh
          call DwordToStrHex
001B1195  call             DwordToStrHex (01B1010h)
          push [res+40]
001B119A  push          dword ptr ds:[1B4089h]
          push offset [Text2+32]
001B11A0  push          1B440Ah
          call DwordToStrHex
001B11A5  call             DwordToStrHex (01B1010h)
          push [res+44]
001B11AA  push          dword ptr ds:[1B408Dh]
          push offset [Text2+46]
001B11B0  push          1B4418h
          call DwordToStrHex
001B11B5  call             DwordToStrHex (01B1010h)

          invoke MessageBoxA, 0, ADDR Text2, ADDR Caption2, 0
001B11BA  push          0
001B11BC  push          offset Caption2 (01B43E2h)
001B11C1  push          offset Text2 (01B43EAh)
001B11C6  push          0
001B11C8  call          _MessageBoxA@16 (01B1695h)

          ; ----- CPUID 80000000h -----

          mov eax, 80000000h
001B11CD  mov          eax,80000000h
          cpuid
001B11D2  cpuid
          mov dword ptr[res+48], eax
001B11D4  mov          dword ptr ds:[001B4091h],eax
          mov dword ptr[res+52], ebx
001B11D9  mov          dword ptr ds:[1B4095h],ebx
          mov dword ptr[res+56], ecx
001B11DF  mov          dword ptr ds:[1B4099h],ecx
          mov dword ptr[res+60], edx
001B11E5  mov          dword ptr ds:[1B409Dh],edx

          push [res+48]

```

```

001B11EB  push          dword ptr ds:[1B4091h]
          push offset [Text00h+4]
001B11F1  push          1B4435h
          call DwordToStrHex
001B11F6  call          DwordToStrHex (01B1010h)
          push [res+52]
001B11FB  push          dword ptr ds:[1B4095h]
          push offset [Text00h+18]
001B1201  push          1B4443h
          call DwordToStrHex
001B1206  call          DwordToStrHex (01B1010h)
          push [res+56]
001B120B  push          dword ptr ds:[1B4099h]
          push offset [Text00h+32]
001B1211  push          1B4451h
          call DwordToStrHex
001B1216  call          DwordToStrHex (01B1010h)
          push [res+60]
001B121B  push          dword ptr ds:[1B409Dh]
          push offset [Text00h+46]
001B1221  push          1B445Fh
          call DwordToStrHex
001B1226  call          DwordToStrHex (01B1010h)

          invoke MessageBoxA, 0, ADDR Text00h, ADDR Caption00h, 0
001B122B  push          0
001B122D  push          offset Caption00h (01B4421h)
001B1232  push          offset Text00h (01B4431h)
001B1237  push          0
001B1239  call          _MessageBoxA@16 (01B1695h)

          ; ----- CPUID 80000001h -----

          mov eax, 80000001h
001B123E  mov          eax, 80000001h
          cpuid
001B1243  cpuid
          mov dword ptr[res+64], eax
001B1245  mov          dword ptr ds:[001B40A1h], eax

```

```

        mov dword ptr[res+68], ebx
001B124A  mov          dword ptr ds:[1B40A5h], ebx
        mov dword ptr[res+72], ecx
001B1250  mov          dword ptr ds:[1B40A9h], ecx
        mov dword ptr[res+76], edx
001B1256  mov          dword ptr ds:[1B40ADh], edx

        push [res+64]
001B125C  push          dword ptr ds:[1B40A1h]
        push offset [Text01h+4]
001B1262  push          1B447Ch
        call DwordToStrHex
001B1267  call          DwordToStrHex (01B1010h)
        push [res+68]
001B126C  push          dword ptr ds:[1B40A5h]
        push offset [Text01h+18]
001B1272  push          1B448Ah
        call DwordToStrHex
001B1277  call          DwordToStrHex (01B1010h)
        push [res+72]
001B127C  push          dword ptr ds:[1B40A9h]
        push offset [Text01h+32]
001B1282  push          1B4498h
        call DwordToStrHex
001B1287  call          DwordToStrHex (01B1010h)
        push [res+76]
001B128C  push          dword ptr ds:[1B40ADh]
        push offset [Text01h+46]
001B1292  push          1B44A6h
        call DwordToStrHex
001B1297  call          DwordToStrHex (01B1010h)

        invoke MessageBoxA, 0, ADDR Text01h, ADDR Caption01h, 0
001B129C  push          0
001B129E  push          offset Caption01h (01B4468h)
001B12A3  push          offset Text01h (01B4478h)
001B12A8  push          0
001B12AA  call          _MessageBoxA@16 (01B1695h)

```

```

; ----- CPUID 80000002h -----

mov eax, 80000002h
001B12AF mov     eax,80000002h
        cpuid
001B12B4 cpuid
        mov dword ptr[res+80], eax
001B12B6 mov     dword ptr ds:[001B40B1h],eax
        mov dword ptr[res+84], ebx
001B12BB mov     dword ptr ds:[1B40B5h],ebx
        mov dword ptr[res+88], ecx
001B12C1 mov     dword ptr ds:[1B40B9h],ecx
        mov dword ptr[res+92], edx
001B12C7 mov     dword ptr ds:[1B40BDh],edx

        mov dword ptr[VendorExt], eax
001B12CD mov     dword ptr [VendorExt (01B4307h)],eax
        mov dword ptr[VendorExt+4], ebx
001B12D2 mov     dword ptr ds:[1B430Bh],ebx
        mov dword ptr[VendorExt+8], ecx
001B12D8 mov     dword ptr ds:[1B430Fh],ecx
        mov dword ptr[VendorExt+12], edx
001B12DE mov     dword ptr ds:[1B4313h],edx

        push [res+80]
001B12E4 push     dword ptr ds:[1B40B1h]
        push offset [Text02h+4]
001B12EA push     1B44C3h
        call DwordToStrHex
001B12EF call     DwordToStrHex (01B1010h)
        push [res+84]
001B12F4 push     dword ptr ds:[1B40B5h]
        push offset [Text02h+18]
001B12FA push     1B44D1h
        call DwordToStrHex
001B12FF call     DwordToStrHex (01B1010h)
        push [res+88]
001B1304 push     dword ptr ds:[1B40B9h]
        push offset [Text02h+32]

```



```

001B130A  push          1B44DFh
          call DwordToStrHex
001B130F  call            DwordToStrHex (01B1010h)
          push [res+92]
001B1314  push          dword ptr ds:[1B40BDh]
          push offset [Text02h+46]
001B131A  push          1B44EDh
          call DwordToStrHex
001B131F  call            DwordToStrHex (01B1010h)

          invoke MessageBoxA, 0, ADDR Text02h, ADDR Caption02h, 0
001B1324  push          0
001B1326  push          offset Caption02h (01B44AFh)
001B132B  push          offset Text02h (01B44BFh)
001B1330  push          0
001B1332  call          _MessageBoxA@16 (01B1695h)

          ; ----- CPUID 80000003h -----

          mov eax, 80000003h
001B1337  mov          eax, 80000003h
          cpuid
001B133C  cpuid
          mov dword ptr[res+96], eax
001B133E  mov          dword ptr ds:[001B40C1h], eax
          mov dword ptr[res+100], ebx
001B1343  mov          dword ptr ds:[1B40C5h], ebx
          mov dword ptr[res+104], ecx
001B1349  mov          dword ptr ds:[1B40C9h], ecx
          mov dword ptr[res+108], edx
001B134F  mov          dword ptr ds:[1B40CDh], edx

          mov dword ptr[VendorExt+16], eax
001B1355  mov          dword ptr ds:[001B4317h], eax
          mov dword ptr[VendorExt+20], ebx
001B135A  mov          dword ptr ds:[1B431Bh], ebx
          mov dword ptr[VendorExt+24], ecx
001B1360  mov          dword ptr ds:[1B431Fh], ecx
          mov dword ptr[VendorExt+28], edx

```

```

001B1366  mov             dword ptr ds:[1B4323h],edx

        push [res+96]
001B136C  push             dword ptr ds:[1B40C1h]
        push offset [Text03h+4]
001B1372  push             1B450Ah
        call DwordToStrHex
001B1377  call             DwordToStrHex (01B1010h)
        push [res+100]
001B137C  push             dword ptr ds:[1B40C5h]
        push offset [Text03h+18]
001B1382  push             1B4518h
        call DwordToStrHex
001B1387  call             DwordToStrHex (01B1010h)
        push [res+104]
001B138C  push             dword ptr ds:[1B40C9h]
        push offset [Text03h+32]
001B1392  push             1B4526h
        call DwordToStrHex
001B1397  call             DwordToStrHex (01B1010h)
        push [res+108]
001B139C  push             dword ptr ds:[1B40CDh]
        push offset [Text03h+46]
001B13A2  push             1B4534h
        call DwordToStrHex
001B13A7  call             DwordToStrHex (01B1010h)

        invoke MessageBoxA, 0, ADDR Text03h, ADDR Caption03h, 0
001B13AC  push             0
001B13AE  push             offset Caption03h (01B44F6h)
001B13B3  push             offset Text03h (01B4506h)
001B13B8  push             0
001B13BA  call             _MessageBoxA@16 (01B1695h)

        ; ----- CPUID 80000004h -----

        mov eax, 80000004h
001B13BF  mov             eax,80000004h
        cpuid

```

```

001B13C4  cpuid
          mov dword ptr[res+112], eax
001B13C6  mov          dword ptr ds:[001B40D1h],eax
          mov dword ptr[res+116], ebx
001B13CB  mov          dword ptr ds:[1B40D5h],ebx
          mov dword ptr[res+120], ecx
001B13D1  mov          dword ptr ds:[1B40D9h],ecx
          mov dword ptr[res+124], edx
001B13D7  mov          dword ptr ds:[1B40DDh],edx

          mov dword ptr[VendorExt+32], eax
001B13DD  mov          dword ptr ds:[001B4327h],eax
          mov dword ptr[VendorExt+36], ebx
001B13E2  mov          dword ptr ds:[1B432Bh],ebx
          mov dword ptr[VendorExt+40], ecx
001B13E8  mov          dword ptr ds:[1B432Fh],ecx
          mov dword ptr[VendorExt+44], edx
001B13EE  mov          dword ptr ds:[1B4333h],edx

          push [res+112]
001B13F4  push          dword ptr ds:[1B40D1h]
          push offset [Text04h+4]
001B13FA  push          1B4551h
          call DwordToStrHex
001B13FF  call          DwordToStrHex (01B1010h)
          push [res+116]
001B1404  push          dword ptr ds:[1B40D5h]
          push offset [Text04h+18]
001B140A  push          1B455Fh
          call DwordToStrHex
001B140F  call          DwordToStrHex (01B1010h)
          push [res+120]
001B1414  push          dword ptr ds:[1B40D9h]
          push offset [Text04h+32]
001B141A  push          1B456Dh
          call DwordToStrHex
001B141F  call          DwordToStrHex (01B1010h)
          push [res+124]
001B1424  push          dword ptr ds:[1B40DDh]

```

```

        push offset [Text04h+46]
001B142A  push          1B457Bh
        call DwordToStrHex
001B142F  call          DwordToStrHex (01B1010h)

        invoke MessageBoxA, 0, ADDR Text04h, ADDR Caption04h, 0
001B1434  push          0
001B1436  push          offset Caption04h (01B453Dh)
001B143B  push          offset Text04h (01B454Dh)
001B1440  push          0
001B1442  call          _MessageBoxA@16 (01B1695h)

; ----- VendorExt -----

        invoke MessageBoxA, 0, ADDR VendorExt, ADDR CaptionVendorExt, 0
001B1447  push          0
001B1449  push          offset CaptionVendorExt (01B4337h)
001B144E  push          offset VendorExt (01B4307h)
001B1453  push          0
001B1455  call          _MessageBoxA@16 (01B1695h)

; ----- CPUID 80000005h -----

        mov eax, 80000005h
001B145A  mov          eax,80000005h
        cpuid
001B145F  cpuid
        mov dword ptr[res+128], eax
001B1461  mov          dword ptr ds:[001B40E1h],eax
        mov dword ptr[res+132], ebx
001B1466  mov          dword ptr ds:[1B40E5h],ebx
        mov dword ptr[res+136], ecx
001B146C  mov          dword ptr ds:[1B40E9h],ecx
        mov dword ptr[res+140], edx
001B1472  mov          dword ptr ds:[1B40EDh],edx

        push [res+128]
001B1478  push          dword ptr ds:[1B40E1h]
        push offset [Text05h+4]

```

```

001B147E  push          1B4598h
          call DwordToStrHex
001B1483  call            DwordToStrHex (01B1010h)
          push [res+132]
001B1488  push          dword ptr ds:[1B40E5h]
          push offset [Text05h+18]
001B148E  push          1B45A6h
          call DwordToStrHex
001B1493  call            DwordToStrHex (01B1010h)
          push [res+136]
001B1498  push          dword ptr ds:[1B40E9h]
          push offset [Text05h+32]
001B149E  push          1B45B4h
          call DwordToStrHex
001B14A3  call            DwordToStrHex (01B1010h)
          push [res+140]
001B14A8  push          dword ptr ds:[1B40EDh]
          push offset [Text05h+46]
001B14AE  push          1B45C2h
          call DwordToStrHex
001B14B3  call            DwordToStrHex (01B1010h)

          invoke MessageBoxA, 0, ADDR Text05h, ADDR Caption05h, 0
001B14B8  push          0
001B14BA  push          offset Caption05h (01B4584h)
001B14BF  push          offset Text05h (01B4594h)
001B14C4  push          0
001B14C6  call          _MessageBoxA@16 (01B1695h)

          ; ----- CPUID 80000008h -----

          mov eax, 80000008h
001B14CB  mov          eax, 80000008h
          cpuid
001B14D0  cpuid
          mov dword ptr[res+144], eax
001B14D2  mov          dword ptr ds:[001B40F1h], eax
          mov dword ptr[res+148], ebx
001B14D7  mov          dword ptr ds:[1B40F5h], ebx

```

```

        mov dword ptr[res+152], ecx
001B14DD  mov          dword ptr ds:[1B40F9h],ecx
        mov dword ptr[res+156], edx
001B14E3  mov          dword ptr ds:[1B40FDh],edx

        push [res+144]
001B14E9  push         dword ptr ds:[1B40F1h]
        push offset [Text08h+4]
001B14EF  push         1B45DFh
        call DwordToStrHex
001B14F4  call         DwordToStrHex (01B1010h)
        push [res+148]
001B14F9  push         dword ptr ds:[1B40F5h]
        push offset [Text08h+18]
001B14FF  push         1B45EDh
        call DwordToStrHex
001B1504  call         DwordToStrHex (01B1010h)
        push [res+152]
001B1509  push         dword ptr ds:[1B40F9h]
        push offset [Text08h+32]
001B150F  push         1B45FBh
        call DwordToStrHex
001B1514  call         DwordToStrHex (01B1010h)
        push [res+156]
001B1519  push         dword ptr ds:[1B40FDh]
        push offset [Text08h+46]
001B151F  push         1B4609h
        call DwordToStrHex
001B1524  call         DwordToStrHex (01B1010h)

        invoke MessageBoxA, 0, ADDR Text08h, ADDR Caption08h, 0
001B1529  push         0
001B152B  push         offset Caption08h (01B45CBh)
001B1530  push         offset Text08h (01B45DBh)
001B1535  push         0
001B1537  call         _MessageBoxA@16 (01B1695h)
        invoke ExitProcess, 0
001B153C  push         0
001B153E  call         _ExitProcess@4 (01B168Fh)

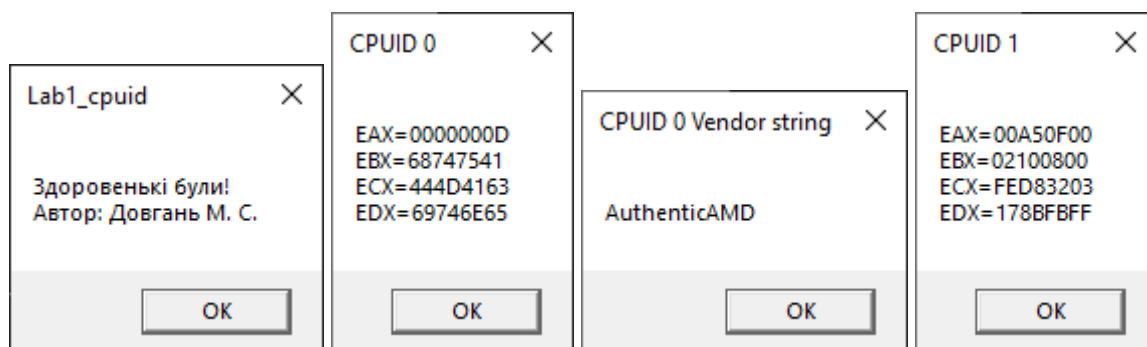
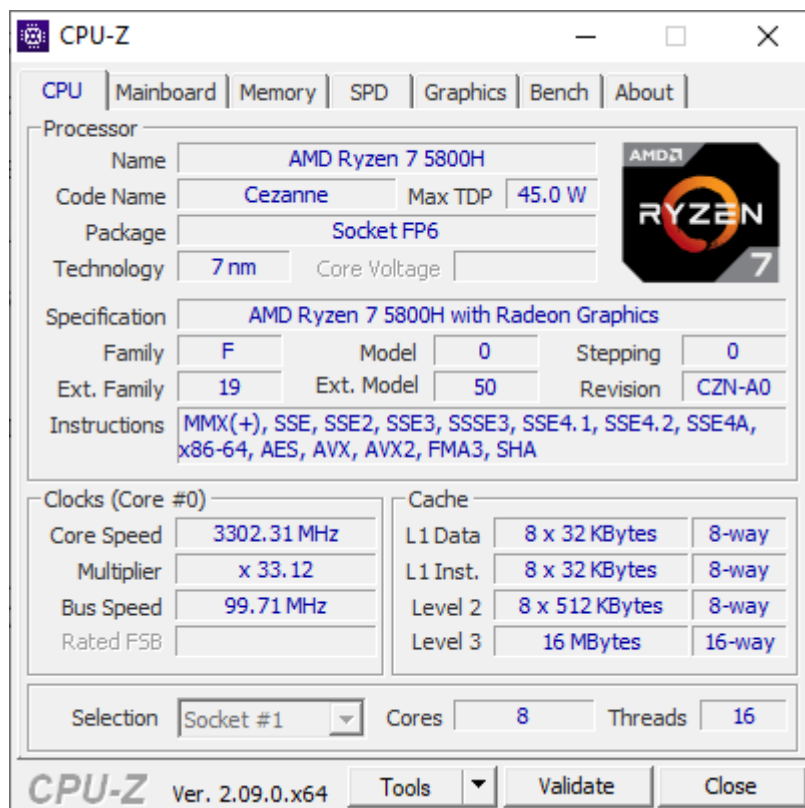
```

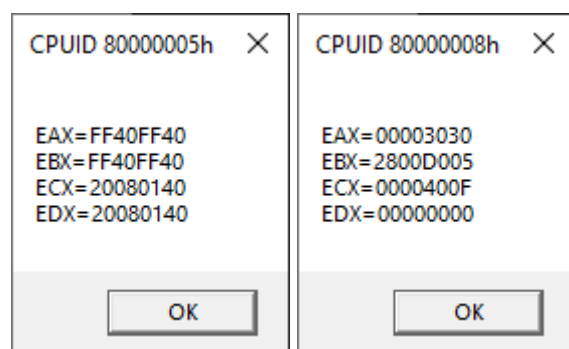
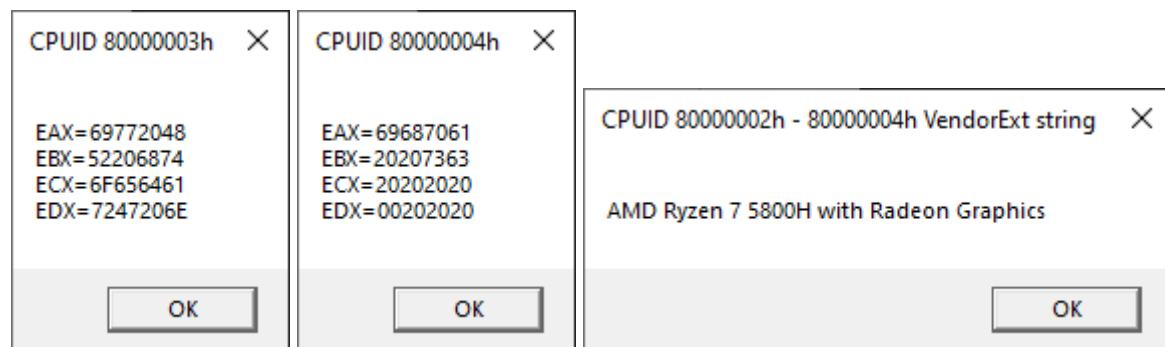
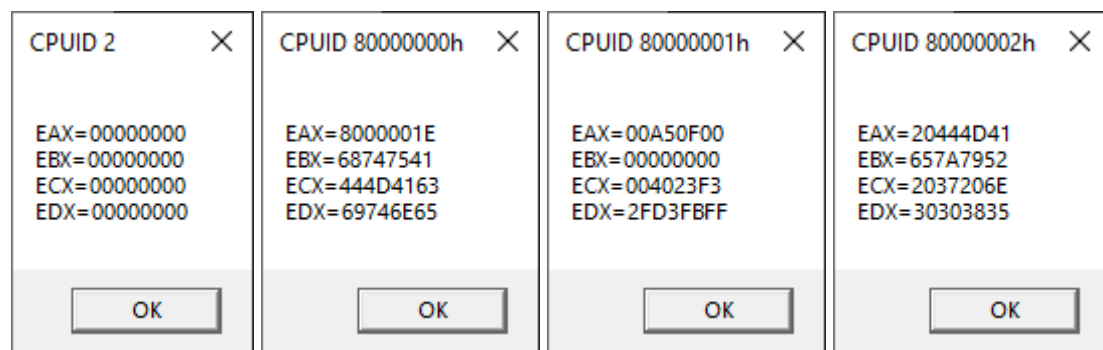
Результати виконання програми:

Для кращої наглядності, я провів тестування створеної програми на двох машинах - основному ноутбучі та допоміжному, які мають процесори від двох різних виробників - у основного це AMD, в допоміжного - INTEL.

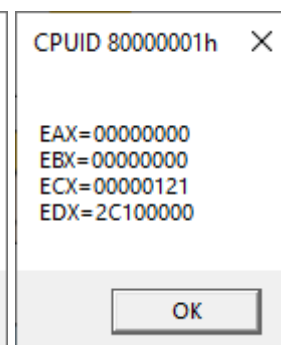
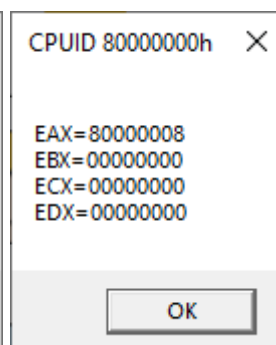
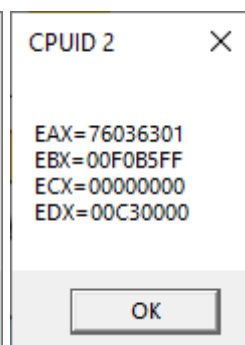
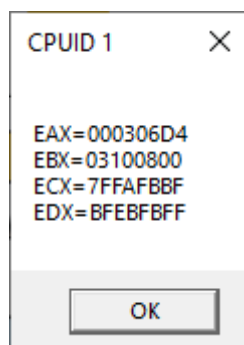
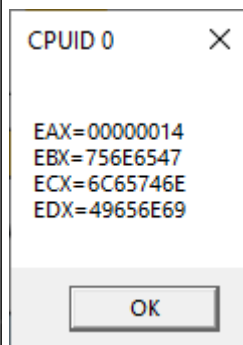
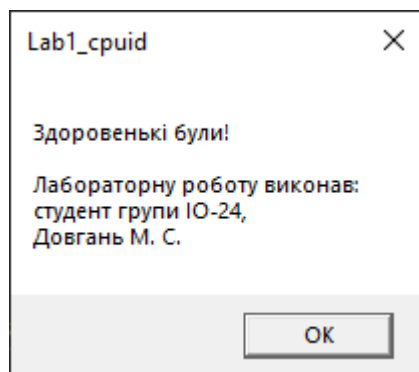
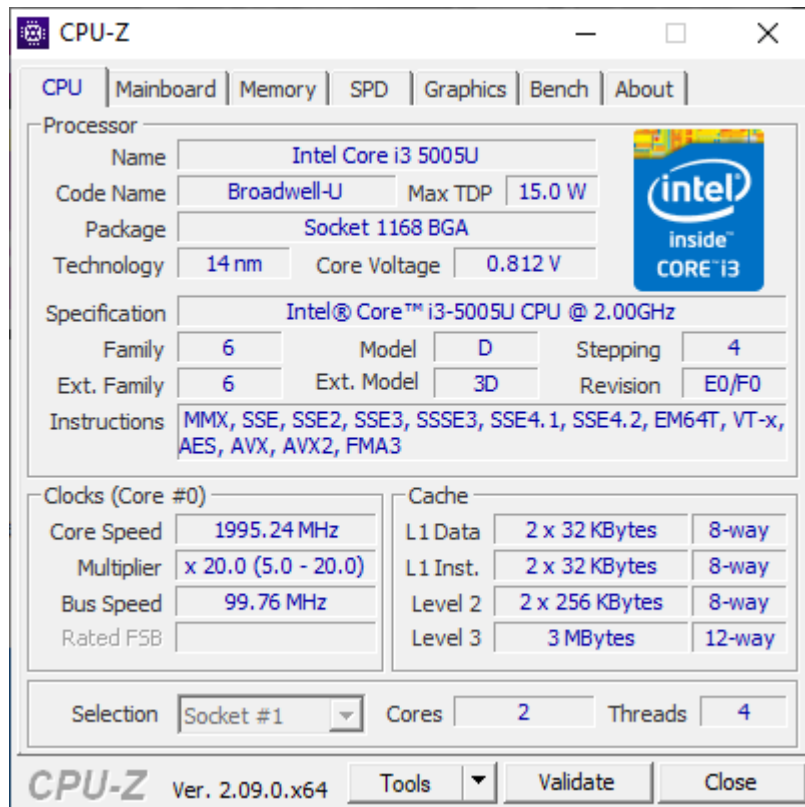
Також я додав скріншоти зі спеціальної утиліти, яка визначає виробника, тип, модель та характеристики процесора - CPUID (CPU-Z), задля переконання у правильній роботоспроможності програми.

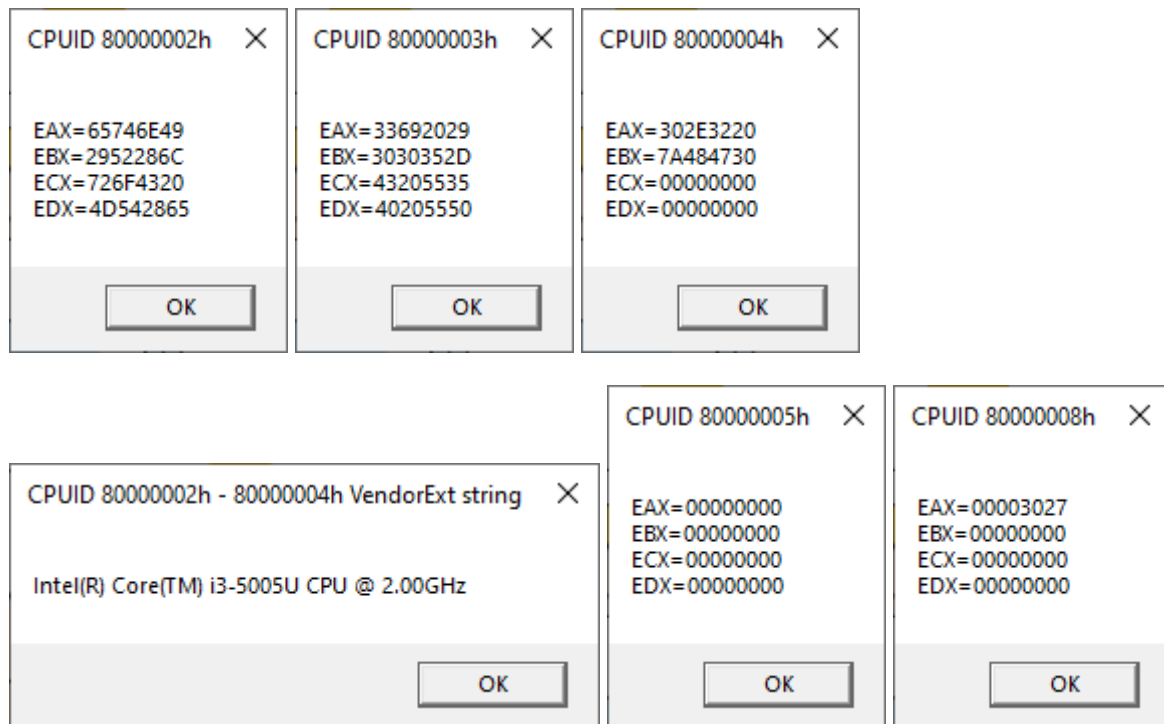
З основного комп'ютера:





З допоміжного комп'ютера:





Аналіз виконання роботи:

Для початку виконання роботи я ознайомився із теоретичними відомостями з комп'ютерного практикуму до виконання лабораторних робіт - визначенням мови Асемблер, різними директивами, точками входу, форматами запису, командами, зокрема, CPUID. Інстальював програмний пакет MASM32 та Microsoft Visual Studio Community, написав, а також скомпілював свою першу найпростішу програму Lab1 на асемблері. Згідно з варіантом завдання написав вихідний текст програми Lab1_cpuid, налагодив та перевінив правильність виконання програми на двох комп'ютерах, для кращої наочності та повної упевненості у її коректній роботі та знайшов і ознайомився з документом "Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 2A: Instruction Set Reference".

Висновок: під час виконання даної лабораторної роботи я ознайомився із засобами розробки програм на асемблері, та, загалом, ознайомився з цією низькорівневою мовою програмування, а також навчився створювати проекти програм на асемблері у середовищах розробки програмного забезпечення, таких як MASM32 та Microsoft Visual Studio і отримав перші навички налагодження програм.