НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Звіт

з комп’ютерного практикуму №1
з дисципліни «Системне програмне забезпечення»
Тема: «Створення програм на асемблері»

Прийняв
доцент кафедри ІПІ
Лісовиченко О.І.
“22” червня 2023 р.

Виконав
Студент групи ІП-13
Дойчев К.М.

Київ – 2023

Комп’ютерний практикум №1

Тема: Створення програм на асемблері

Завдання:

1. Для програми створити файл типу .asm. Ця програма не має засобів виводу даних, тому правильність її виконання треба перевірити за допомогою td.exe.
2. Скомпілювати програму, включивши потрібні опції для налагоджувача та створення файлу лістингу типу .lst.
3. Ознайомитись зі структурою файлу .lst. За вказівкою викладача, для певної команди асемблера розглянути структуру машинної команди і навести її у звіті.
4. Скомпонувати .obj-файл програми. Включити опції для налагодження та створення .map-файлу.
5. Занести до звіту адреси початку та кінця всіх сегментів з .map-файлу.
6. Завантажити до налагоджувача td.exe одержаний .exe-файл програми.
7. У вікні CPU у полі DUMP знайти початкову адресу сегмента даних та записати його до звіту. Знайти масиви SOURCE та DEST. Дані у масиві SOURCE подаються у шістнадцятковій системі.
8. У покроковому режимі за допомогою клавіші F7 виконати програму. Одержані результати у масиві DEST показати викладачеві.

Текст програми

```
section .data
    SOURCE   db 10, 20, 30, 40
    DEST     db 4 dup('?') ; duplicate '?' 4 times

section .text
global _start

_start:
    ; Zero out the DEST array
    mov byte [DEST], 0
    mov byte [DEST+1], 0
    mov byte [DEST+2], 0
    mov byte [DEST+3], 0

    ; Copy the SOURCE array to DEST
    mov al, byte [SOURCE]
    mov byte [DEST+3], al
    mov al, byte [SOURCE+1]
    mov byte [DEST+2], al
    mov al, byte [SOURCE+2]
    mov byte [DEST+1], al
    mov al, byte [SOURCE+3]
    mov byte [DEST], al

    ; Exit the program
    mov eax, 1
```

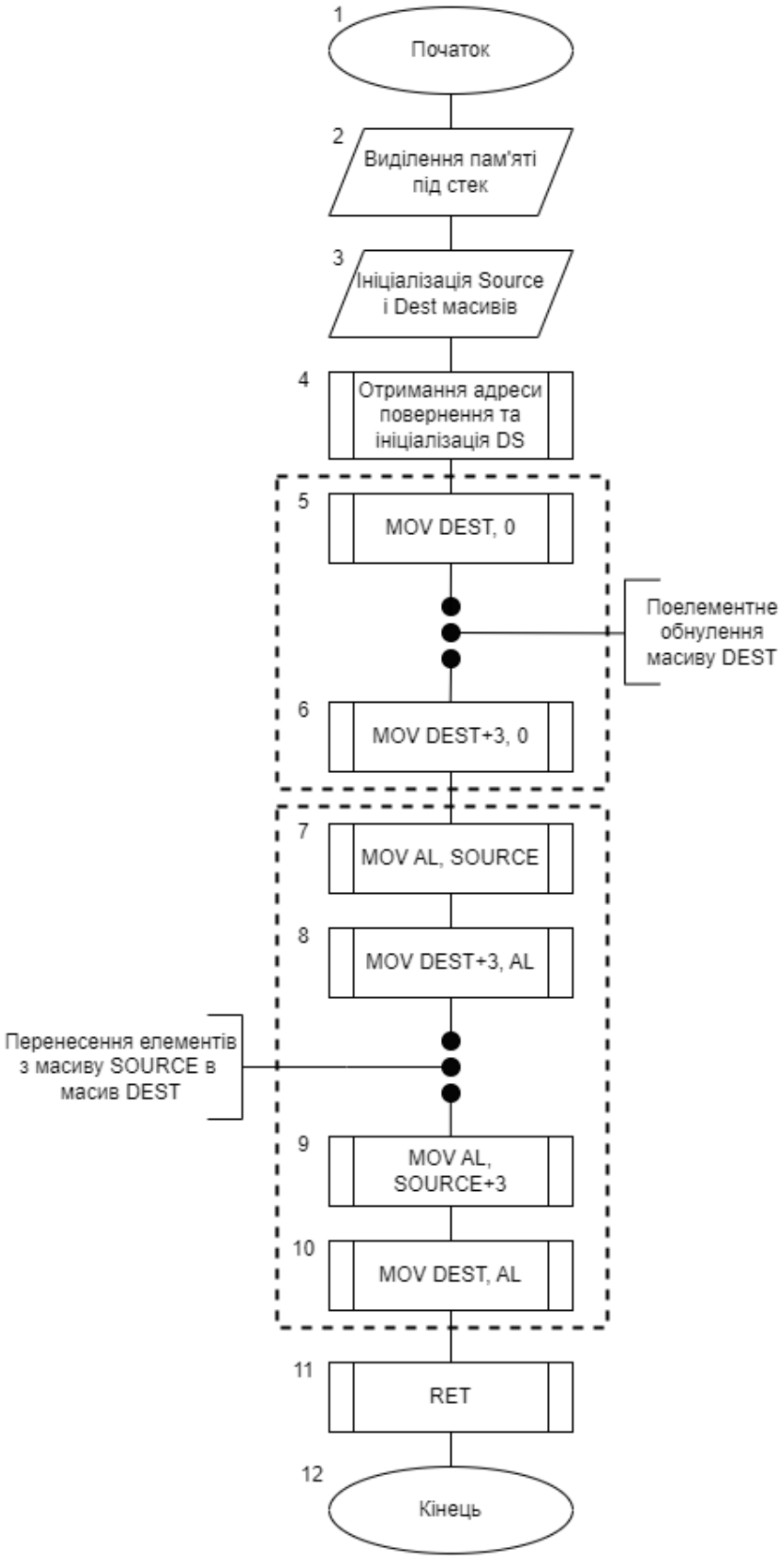
```
    xor ebx, ebx
    int 0x80
```

Введені та отримані результати

Вміст .lst файлу:

```
1                                   section .data
    2 00000000 0A141E28                 SOURCE   db 10, 20, 30, 40
    3 00000004 3F3F3F3F                 DEST     db 4 dup('?')
    4
    5                                   section .text
    6                                   global _start
    7
    8                                   _start:
    9                                       ; Zero out the DEST array
   10 00000000 C60425[04000000]00           mov byte [DEST], 0
   11 00000008 C60425[05000000]00           mov byte [DEST+1], 0
   12 00000010 C60425[06000000]00           mov byte [DEST+2], 0
   13 00000018 C60425[07000000]00           mov byte [DEST+3], 0
   14
   15                                       ; Copy the SOURCE array to DEST
   16 00000020 8A0425[00000000]             mov al, byte [SOURCE]
   17 00000027 880425[07000000]             mov byte [DEST+3], al
   18 0000002E 8A0425[01000000]             mov al, byte [SOURCE+1]
   19 00000035 880425[06000000]             mov byte [DEST+2], al
   20 0000003C 8A0425[02000000]             mov al, byte [SOURCE+2]
   21 00000043 880425[05000000]             mov byte [DEST+1], al
   22 0000004A 8A0425[03000000]             mov al, byte [SOURCE+3]
   23 00000051 880425[04000000]             mov byte [DEST], al
   24
   25                                       ; Exit the program
   26 00000058 B801000000                   mov eax, 1
   27 0000005D 31DB                         xor ebx, ebx
   28 0000005F CD80                         int 0x80
```

Схема функціонування програми


1
Початок
2
Виділення пам'яті під стек
3
Ініціалізація Source і Dest масивів
4
Отримання адреси повернення та ініціалізація DS
5
MOV DEST, 0
Поелементне обнулення масиву DEST
6
MOV DEST+3, 0
7
MOV AL, SOURCE
8
MOV DEST+3, AL
Перенесення елементів з масиву SOURCE в масив DEST
9
MOV AL, SOURCE+3
10
MOV DEST, AL
11
RET
12
Кінець

Вікно DUMP

До виконання програми:

```
Breakpoint 1, _start () at main.asm:10
(gdb) x/4xb &SOURCE
0x402000 <SOURCE>:      0x0a    0x14    0x1e    0x28
(gdb) x/4xb &DEST
0x402004 <DEST>:        0x3f    0x3f    0x3f    0x3f
(gdb)
```

Після виконання програми:
Масив DEST

```
(gdb) next
(gdb) x/4xb &DEST
0x402004 <DEST>:        0x28    0x1e    0x14    0x0a
(gdb)
```

ВИСНОВОК

Під час виконання даного комп’ютерного практикуму, я використав текстовий редактор для створення файлу з розширенням .asm. Після цього, я скомпілював програму з необхідними опціями для налагоджувача та створення файлу лістингу (.lst). За допомогою цього лістингу, я дослідив структуру файлу та проаналізував машинні команди. Далі, я завантажив отриманий .elf-файл програми разом з налагоджувачем GNU debugger (gbg). За допомогою команд x/4xb &SOURCE і x/4xb &DEST я міг перевірити значення у сегментах SOURCE і DEST