Національний технічний університет України
«Київський політехнічний інститут»
Факультет інформатики та обчислювальної техніки
Кафедра технічної кібернетики

Звіти до комп’ютерних практикумів з модуля «Системне програмування»

Прийняв
доцент кафедри ТК
Лісовиченко О.І.
“...” ..... 2018 р.
Виконав
Студент групи ІТ-61
Лесогорський К.С

Київ – 2018

Комп’ютерний практикум No1

Тема: Створення програм на асемблері

Завдання:

1. Для програми, наведеної вище, 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 TARGET. Дані у масиві SOURCE подаються у
шістнадцятковій системі.
8. У покроковому режимі за допомогою клавіші F7 виконати програму. Одержані
результати у масиві TARGET показати викладачеві.

Текст програми:

```
SECTION .data
        SOURCE: db 10,20,30,40; INITIAL ARRAY
SECTION .bss
        TARGET: resb 4; FREE SPACE TO MOVE BYTES FROM SOURCE
SECTION .text
        GLOBAL _start

_start:
        mov al,[SOURCE]
        mov [TARGET+3],al
        mov al,[SOURCE+1]
        mov [TARGET+2],al
        mov al,[SOURCE+2]
        mov [TARGET+1],al
        mov al,[SOURCE+3]
        mov [TARGET],al
        ;exit programm
        mov ax,1
        mov bx,1
        int 0x80;sys_call
```

Введені та отримані результати

Вміст .lst файлу:

```
    1                           SECTION .data
    2 00000000 0A141E28                     SOURCE: db 10,20,30,40; INITIAL ARRAY
    3                           SECTION .bss
    4 00000000 <res 00000004>               TARGET: resb 4; FREE SPACE TO MOVE BYTES
FROM SOURCE
    5                           SECTION .text
    6                                 GLOBAL _start
    7
    8                           _start:
    9 00000000 A0[00000000]                 mov al,[SOURCE]
   10 00000005 A2[03000000]                 mov [TARGET+3],al
   11 0000000A A0[01000000]                 mov al,[SOURCE+1]
   12 0000000F A2[02000000]                 mov [TARGET+2],al
   13 00000014 A0[02000000]                 mov al,[SOURCE+2]
   14 00000019 A2[01000000]                 mov [TARGET+1],al
   15 0000001E A0[03000000]                 mov al,[SOURCE+3]
   16 00000023 A2[00000000]                 mov [TARGET],al
   17                                 ;exit programm
   18 00000028 66B80100                    mov ax,1
   19 0000002C 66BB0100                     mov bx,1
   20 00000030 CD80                        int 0x80;sys_call
```

Вміст .map файлу:

```
- NASM Map file ------------------------------------------------------------

Source file:  lab00.asm
Output file:  lab00.o

-- Program origin ------------------------------------------------------------

00000000

-- Sections (summary) --------------------------------------------------------

Vstart          Start           Stop            Length    Class     Name
          0             0            20  00000020  progbits  .text
         20            20            24  00000004  progbits  .data
         24            24            28  00000004  nobits    .bss

-- Sections (detailed) -------------------------------------------------------

---- Section .text -----------------------------------------------------------

class:     progbits
length:              20
start:               0
align:     not defined
follows:   not defined
vstart:              0
valign:    not defined
vfollows:  not defined
```

---- Section .data --

class: progbits
length: 4
start: 20
align: 4
follows: not defined
vstart: 20
valign: not defined
vfollows: not defined

---- Section .bss ---

class: nobits
length: 4
start: 24
align: not defined
follows: not defined
vstart: 24
valign: 4
vfollows: not defined

Масиви source та target на початку виконання:

```
0x80490b4 <SOURCE>:     0x0a    0x14    0x1e    0x28
(gdb) x/4xb 0x80490b8
0x80490b8 <TARGET>:     0x00    0x00    0x00    0x00
```

Масиви source та target в кінці виконання:

```
0x80490b4 <SOURCE>:     0x0a    0x14    0x1e    0x28
(gdb) x/4xb 0x80490b8
0x80490b8 <TARGET>:     0x28    0x1e    0x14    0x0a
(gdb)
```

Блок-схема:


початок
1
Ініціалізація масивів SOURCE і TARGET
2
Пересилання значення з SOURCE[3] до регістру DX
3
Пересилання значення регістру DX до TARGET[0]
4
Пересилання значення з SOURCE[2] до регістру DX
5
Пересилання значення регістру DX до TARGET[1]
6
Пересилання значення з SOURCE[1] до регістру DX
7
Пересилання значення регістру DX до TARGET[2]
8
Пересилання значення з SOURCE[0] до регістру DX
9
Пересилання значення регістру DX до TARGET[3]
кінець

Висновок:

Було створено файл типу .asm, написано програму та скомпільовано компілятором. Команда: *nasm lab00.asm -f elf -l asm.lst -g -o lab00.o*
Після чого було отримано об'єктний файл та файл лістингу. Розглянуто структуру машинних команд. Для створення .map файлу треба було компілювати в формат bin. В цьому файлі розміщуються відносні адреси початку та кінця секцій пам'яті програми.
Відладка програми велась з допомогою GDB. Для отримання адрес секцій використовувалася команда *maintenance info sections*, після чого можна була виводити значення даних із цих секцій. Використовувалося покрокове виконання програми. В кінці програма виконала поставлене завдання – в масив target записала масив source в реверсивному порядку.