Отчёт по лабораторной работе №4

Дисциплина: архитектура компьютера

Адмиральская Александра Андреевна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

2 Задание

1.Создание программы Hello world! 2.Работа с транслятором NASM 3.Работа с расширенным синтаксисом командной строки NASM 4.Работа с компоновщиком LD 5.Запуск исполняемого файла 6.Выполнение заданий для самостоятельной работы.

3 Выполнение лабораторной работы

Для начала создаем каталог для работы с программами на языке ассемблера NASM и переходим в него (рис. 3.1).

Рис. 3.1: Создание каталога и переход в него.

Далее создаем текстовый файл с именем hello.asm и открываем файл с помощью текстового редактора gedit (рис. 3.2).

Рис. 3.2: Создание файла hello.asm и открытие его с помощью gedit.

Вводим в файле текст (рис. 3.3).

```
; hello.asm

SECTION .data ; Начало секции данных

hello: DB 'Hello world!',10 ; 'Hello world!' плюс

; символ перевода строки

helloLen: EQU $-hello ; Длина строки hello

SECTION .text ; Начало секции кода

GLOBAL _start

_start: ; Точка входа в программу

mov eax,4 ; Системный вызов для записи (sys_write)

mov ebx,1 ; Описатель файла '1' - стандартный вывод

mov ecx,hello ; Адрес строки hello в ecx

mov edx,helloLen ; Размер строки hello

int 80h ; Вызов ядра

mov eax,1 ; Системный вызов для выхода (sys_exit)

mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)

int 80h ; Вызов ядра

```

Рис. 3.3: Ввод текста в файл.

Превращаем текст программы для вывода “Hello world!” в объектный код с помощью транслятора NASM, используя команду nasm -f elf hello.asm (рис. 3.4).

```
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ nasm -f elf  hello.asm
```

Рис. 3.4: Превращаем текст программы в объектный код.

Затем передаем объектный файл на обработку компоновщику и запускаем на выполнение созданный исполняемый файл (рис. 3.5).

```
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ ./hello
Hello world!
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $
```

Рис. 3.5: Передача файла на обработку и его запуск на выполнение.

Приступим к выполнению заданий для самостоятельной работы. С помощью

команды cp создаем копию файла hello.asm с именем lab4.asm (рис. 3.6).

```
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4.asm  lab4.asm.save
```

Рис. 3.6: Создание копии файла hello.asm.

Используя текстовый редактор gedit, вносим изменения в текст программы так, чтобы вместо Hello world! на экран выводилась строка с моими фамилией и именем (рис. 3.7).

```
; hello.asm

SECTION .data ; Начало секции данных

hello: DB 'Адмиральская Александра',10 ; 'Hello world!' плюс

; символ перевода строки

helloLen: EQU $-hello ; Длина строки hello

SECTION .text ; Начало секции кода

GLOBAL _start

_start: ; Точка входа в программу

mov eax,4 ; Системный вызов для записи (sys_write)

mov ebx,1 ; Описатель файла '1' - стандартный вывод

mov ecx,hello ; Адрес строки hello в ecx

mov edx,helloLen ; Размер строки hello

int 80h ; Вызов ядра

mov eax,1 ; Системный вызов для выхода (sys_exit)

mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)

int 80h ; Вызов ядра

```

Рис. 3.7: Внесение изменений в текст программы.

Затем транслируем полученный текст программы lab4.asm в объектный файл (рис. 3.8).

```
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4.asm  lab4.asm.save  lab4.o
```

Рис. 3.8: Трансляция текста программы в объектный файл.

Выполняем компоновку объектного файла и запускаем получившийся исполняемый файл (рис. 3.9).

```
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ ld -m elf_i386 lab4.o -o lab4
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4  lab4.asm  lab4.asm.save  lab4.o
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ ./lab4
Адмиральская Александра
```

Рис. 3.9: Компоновка и запуск файла.

Далее копируем файлы hello.asm и lab4.asm в локальный репозиторий в каталог ~/work/study/2024-2025/“Архитектура компьютера”/arch-pc/labs/lab04/ (рис. 3.10).

```
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ cp hello.asm ~/work/study/2024-2025/"Архитектура компьютер
a"/arch-pc/labs/lab04/
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ cp lab4.asm ~/work/study/2024-2025/"Архитектура компьютера
"/arch-pc/labs/lab04/
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4  lab4.asm  lab4.asm.save  lab4.o
aaadmiraljskaya@dk3n61 ~/work/arch-pc/lab04 $ cd ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc
/labs/lab04/
aaadmiraljskaya@dk3n61 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm  lab4.asm  presentation  report
```

Рис. 3.10: Копирование файлов.

Последним шагом загружаем файлы на Github (рис. 3.11).

```
aaadmiraljskaya@dk3n61 ~/work/study/2024-2025/Архитектура компьютера/arch-pc $ git add .
aaadmiraljskaya@dk3n61 ~/work/study/2024-2025/Архитектура компьютера/arch-pc $ git commit -m "Add fales
for lab04"
Текущая ветка: master
Ваша ветка опережает «origin/master» на 3 коммита.
  (используйте «git push», чтобы опубликовать ваши локальные коммиты)

нечего коммитить, нет изменений в рабочем каталоге
aaadmiraljskaya@dk3n61 ~/work/study/2024-2025/Архитектура компьютера/arch-pc $ git push
Перечисление объектов: 41, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (24/24), готово.
Запись объектов: 100% (24/24), 15.90 КиБ | 7.95 МиБ/с, готово.
Total 24 (delta 13), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (13/13), completed with 7 local objects.
To github.com:aadmiralskaya/study_2024-2025_arhpc.git
   d86b430..e862c43  master -> master
aaadmiraljskaya@dk3n61 ~/work/study/2024-2025/Архитектура компьютера/arch-pc $
```

Рис. 3.11: Загрузка файлов на Github.

4 Выводы

При выполнении данной лабораторной работы я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.