

Отчет по лабораторной работе №4

Создание и процесс обработки программ на языке ассемблера NASM

Карапетян Мари Рафаеловна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
	Список литературы	13

Список иллюстраций

4.1	Создание каталога	9
4.2	Переход в каталог	9
4.3	Создание текстового файла	9
4.4	Открытие файла	9
4.5	Ввод текста	10
4.6	Компиляция текста	10
4.7	Проверка объектного файла	10
4.8	Создание файлов	10
4.9	Проверка файлов	10
4.10	Передача файла на компоновку	11
4.11	Проверка исполняемого файла hello	11
4.12	Запуск на выполнение созданный исполняемый файл	11
4.13	Создание копию файла hello.asm с именем lab4.asm	11
4.14	Внесем изменения в текст программы	12
4.15	Оттранслирование, компоновка, запуск	12
4.16	Копирование файлов в локальный репозиторий	12

Список таблиц

1 Цель работы

Основные процедуры компиляции и сборки программ, написанных на ассемблере NASM

2 Задание

- Программа Hello world!
- Транслятор NASM
- Расширенный синтаксис командной строки NASM
- Компоновщик LD
- Запуск исполняемого файла

3 Теоретическое введение

Основные принципы работы компьютера Основными функциональными элементами любой электронно-вычислительной машины (ЭВМ) являются центральный процессор, память и периферийные устройства (рис. 4.1). Взаимодействие этих устройств осуществляется через общую шину, к которой они подключены. Физически шина представляет собой большое количество проводников, соединяющих устройства друг с другом. В современных компьютерах проводники выполнены в виде электропроводящих дорожек на материнской (системной) плате. Основной задачей процессора является обработка информации, а также организация координации всех узлов компьютера. В состав центрального процессора (ЦП) входят следующие устройства: • арифметико-логическое устройство (АЛУ) — выполняет логические и арифметические действия, необходимые для обработки информации, хранящейся в памяти; • устройство управления (УУ) — обеспечивает управление и контроль всех устройств компьютера; • регистры — сверхбыстрая оперативная память небольшого объёма, входящая в состав процессора, для временного хранения промежуточных результатов выполнения инструкций; регистры процессора делятся на два типа: регистры общего назначения и специальные регистры. Для того, чтобы писать программы на ассемблере, необходимо знать, какие регистры процессора существуют и как их можно использовать. Большинство команд в программах написанных на ассемблере используют регистры в качестве операндов. Практически все команды представляют собой преобразование данных хранящихся в регистрах процессора, это например пересылка данных между регистрами или между регистрами и

памятью, пре- образование (арифметические или логические операции) данных хранящихся в регистрах.

4 Выполнение лабораторной работы

Создаем каталог для работы с программами на языке ассемблера NASM (Рис.@fig:001)

```
mrkarapetyan@dk2n26 ~ $ mkdir -p ~/work/arch-pc/lab04
```

Рис. 4.1: Создание каталога

Перейдем в созданный каталог (Рис.@fig:002)

```
mrkarapetyan@dk2n26 ~ $ cd ~/work/arch-pc/lab04
```

Рис. 4.2: Переход в каталог

Создайте текстовый файл с именем hello.asm (Рис.@fig:003)

```
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ touch hello.asm
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ ls
hello.asm
```

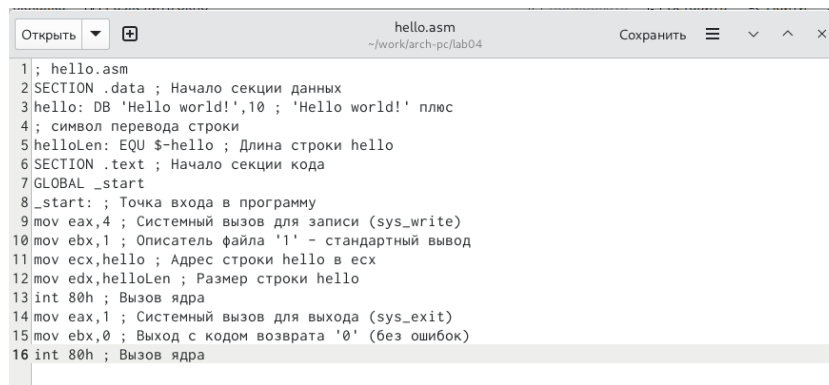
Рис. 4.3: Создание текстового файла

Откроем этот файл с помощью любого текстового редактора (Рис.@fig:004)

```
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ gedit hello.asm
```

Рис. 4.4: Открытие файла

Введем в него текст (Рис.@fig:005)



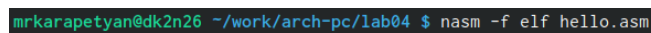
```

1; hello.asm
2SECTION .data ; Начало секции данных
3hello: DB 'Hello world!',10 ; 'Hello world!' плюс
4; символ перевода строки
5helloLen: EQU $-hello ; Длина строки hello
6SECTION .text ; Начало секции кода
7GLOBAL _start
8_start: ; Точка входа в программу
9mov eax,4 ; Системный вызов для записи (sys_write)
10mov ebx,1 ; Описатель файла '1' - стандартный вывод
11mov ecx,hello ; Адрес строки hello в ecx
12mov edx,helloLen ; Размер строки hello
13int 80h ; Вызов ядра
14mov eax,1 ; Системный вызов для выхода (sys_exit)
15mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
16int 80h ; Вызов ядра

```

Рис. 4.5: Ввод текста

Скомпилируем данный текст (Рис.@fig:006)



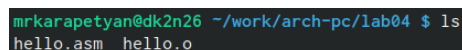
```

mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm

```

Рис. 4.6: Компиляция текста

Проверим, что объектный файл был создан (Рис. 4.7).



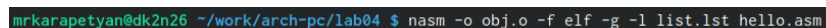
```

mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o

```

Рис. 4.7: Проверка объектного файла

Скомпилируем исходный файл hello.asm в obj.o и создадим файл листинга list.lst(Рис.@fig:008)



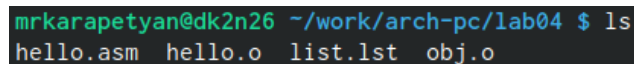
```

mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm

```

Рис. 4.8: Создание файлов

Проверим, что файлы были созданы (Рис.@fig:009)



```

mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o  list.lst  obj.o

```

Рис. 4.9: Проверка файлов

Передадим объектный файл на обработку компоновщику (Рис.@fig:010)

```
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
```

Рис. 4.10: Передача файла на компоновку

Проверим, что исполняемый файл hello был создан (Рис.@fig:011)

```
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  list.lst  obj.o
```

Рис. 4.11: Проверка исполняемого файла hello

Зададим имя создаваемого исполняемого файла (Рис.@fig:012)

[Зададим имя создаваемого исполняемого файла] (image/12.jpg){#fig:012
width=70%}

Запустим на выполнение созданный исполняемый файл, находящийся в текущем каталоге (Рис.@fig:013)

```
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ ./hello
Hello world!
```

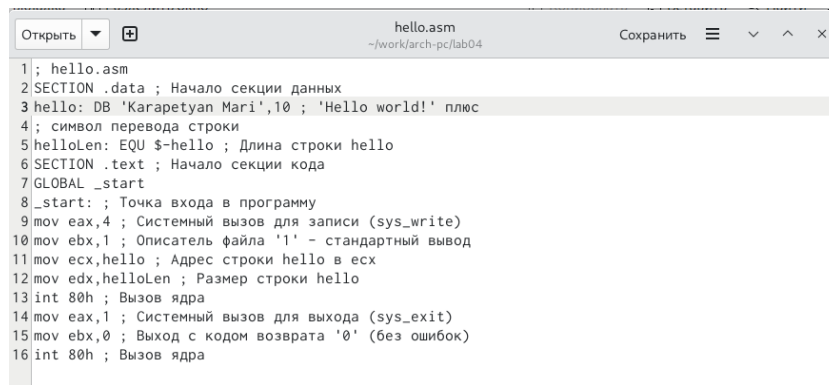
Рис. 4.12: Запуск на выполнение созданный исполняемый файл

Создадим копию файла hello.asm с именем lab4.asm (Рис.@fig:014)

```
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm
```

Рис. 4.13: Создание копию файла hello.asm с именем lab4.asm

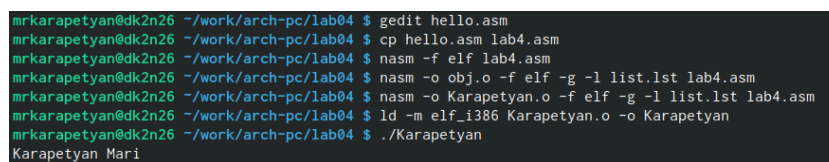
Внесем изменения в текст программы в файле lab4.asm (Рис.@fig:015)



```
1; hello.asm
2SECTION .data ; Начало секции данных
3hello: DB 'Karapetyan Mari',10 ; 'Hello world!' плюс
4; символ перевода строки
5helloLen: EQU $-hello ; Длина строки hello
6SECTION .text ; Начало секции кода
7GLOBAL _start
8_start: ; Точка входа в программу
9mov eax,4 ; Системный вызов для записи (sys_write)
10mov ebx,1 ; Описатель файла '1' - стандартный вывод
11mov ecx,hello ; Адрес строки hello в ecx
12mov edx,helloLen ; Размер строки hello
13int 80h ; Вызов ядра
14mov eax,1 ; Системный вызов для выхода (sys_exit)
15mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
16int 80h ; Вызов ядра
```

Рис. 4.14: Внесем изменения в текст программы

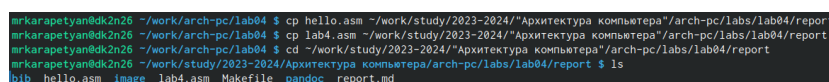
Оттранслируем полученный текст программы lab4.asm в объектный файл. Выполним компоновку объектного файла и запустим получившийся исполняемый файл (Рис.@fig:016)



```
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ gedit hello.asm
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst lab4.asm
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ nasm -o Karapetyan.o -f elf -g -l list.lst lab4.asm
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ ld -m elf_i386 Karapetyan.o -o Karapetyan
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ ./Karapetyan
Karapetyan Mari
```

Рис. 4.15: Оттранслирование, компоновка, запуск

Скопировала файлы hello.asm и lab4.asm в локальный репозиторий в каталог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/ с помощью утилиты cp и проверила наличие файлов с помощью утилиты ls (Рис.@fig:017)



```
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ cp hello.asm ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/report
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ cp lab4.asm ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/report
mrkarapetyan@dk2n26 ~/work/arch-pc/lab04 $ cd ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/report
mrkarapetyan@dk2n26 ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/report $ ls
b1b hello.asm image lab4.asm Makefile pandoc report.md
```

Рис. 4.16: Копирование файлов в локальный репозиторий

Загружаю файлы на Github # Выводы

В ходе выполнения работы, я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM

Список литературы