

Отчёт по лабораторной работе №4

Дисциплина: Архитектура компьютеров

Мантуров Татархан Бесланович

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
5	Задания для самостоятельной работы	11
6	Выводы	12
	Список литературы	13

Список иллюстраций

Список таблиц

1 Цель работы

Освоение 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 Задание для самостоятельной работы

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

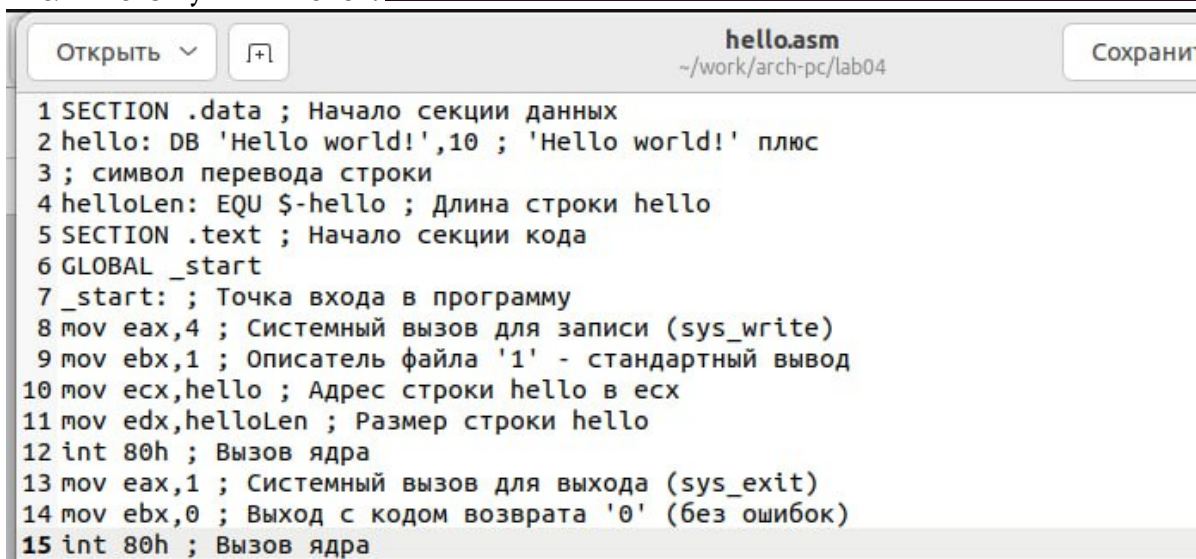
4 Выполнение лабораторной работы

Создаю каталог для работы с программами на языке ассемблера NASM и перехожу в него:

```
manturov@ubuntu:~$ mkdir -p ~/work/arch-pc/lab04 manturov@ubuntu:~$ cd ~/work/arch-pc/lab04
```

Создал текстовый файл с именем hello.asm: manturov@ubuntu:~/work/arch-pc/lab04\$ touch hello.asm

откройте этот файл с помощью любого текстового редактора, например, gedit и ввел в него нужный текст: manturov@ubuntu:~/work/arch-pc/lab04\$ gedit hello.asm



```
hello.asm
~/work/arch-pc/lab04
Сохранить

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

для компиляции приведённого выше текста программы «Hello World» написал:

```
manturov@ubuntu:~/work/arch-pc/lab04$ nasm -f elf hello.asm
```

скомпилировал исходный файл hello.asm в obj.o: manturov@ubuntu:~/work/arch-pc/lab04\$ nasm -o obj.o -f elf -g hello.asm

передаю на обработку компоновщику объектный файл: manturov@ubuntu:~/work/arch-pc/lab04\$ ld -m elf_386 obj.o -o hello.elf

Ключ -o с последующим значением задаёт в данном случае имя создаваемого исполняемого файла. Выполняю следующую команду: manturov@ubuntu:~/work/arch-pc/lab04\$ ld -m elf_386 obj.o -o hello.elf

Запустил на выполнение созданный исполняемый файл, находящийся в текущем каталоге, набрав в командной строке:

```
manturov@ubuntu:~/work/arch-pc/lab04$ ./he  
Hello world!
```

5 Задания для самостоятельной работы

1. В каталоге `~/work/arch-pc/lab04` с помощью команды `cp` создайте копию файла `hello.asm` с именем `lab4.asm`

```
manturov@ubuntu: ~/work/arch-pc/lab04$ cp ~/work/arch-pc/lab04/hello.asm ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04
```

2. С помощью любого текстового редактора внесите изменения в текст программы в файле `lab4.asm` так, чтобы вместо `Hello world!` на экран выводилась строка с вашими фамилией и именем.

```
manturov@ubuntu:~/work/arch-pc/lab04$ gedit lab4.asm
```

3. Оттранслируйте полученный текст программы `lab4.asm` в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.

```
manturov@ubuntu:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
manturov@ubuntu:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
manturov@ubuntu:~/work/arch-pc/lab04$ ./lab4
Manturov Tatarkhan
```

4. Скопируйте файлы `hello.asm` и `lab4.asm` в Ваш локальный репозиторий в каталог `~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/`. Загрузите файлы на Github.

```
manturov@ubuntu:~/work/arch-pc/lab04$ cp ~/work/arch-pc/lab04/hello.asm ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04/
manturov@ubuntu:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ git add .
manturov@ubuntu:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ git commit -am "feat(main):add files lab4"
[master 79546f3] feat(main):add files lab4
5 files changed, 30 insertions(+)
create mode 100644 labs/lab02/report.zip
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/lab4.asm
create mode 100644 labs/lab04/report/report.docx
create mode 100644 labs/lab04/report/report.pdf
manturov@ubuntu:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ git push
Перечисление объектов: 16, готово.
Получение объектов: 100% (16/16) - готово
```

6 Выводы

Я освоил процедуры компиляции и сборки программ, написанных на ассемблере NASM.

Список литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learning-bash-Shell-Programming-Nutshell/dp/0596009658>.
6. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
10. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс,
- 11.
12. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
13. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.
14. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВ- Петербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
15. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-

- е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix.
16. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
17. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер,
18. — 1120 с. — (Классика Computer Science).