

Лабораторная работа №4

Создание и процесс обработки программ на языке ассемблера NASM

Софич Андрей Геннадьевич

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
4.1	Создание программы Hello world!	8
4.2	Работа с транслятором NASM.	9
4.3	Работа с компоновщиком LD.	10
4.4	Запуск исполняемого файла.	10
4.5	Выполнение заданий для самостоятельной работы.	11
5	Выводы	14
	Список литературы	15

Список иллюстраций

4.1	Перемещение по директориям и создание файла	8
4.2	Файл hello.asm	8
4.3	Программа	9
4.4	Создание бинарного файла	9
4.5	Создание файла листинга	10
4.6	Получение исполняемого файла	10
4.7	Значение main	10
4.8	Проверка программы	11
4.9	Копирование файла hello.asm	11
4.10	Изменение программы	11
4.11	Создание объектного файла	12
4.12	Получение исполняемого файла	12
4.13	Проверка программы	12
4.14	Добавление файлов в лабораторную работу	13
4.15	Отправление работы на github	13

Список таблиц

1 Цель работы

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

2 Задание

Создание программы Hello world!

Работа с транслятором NASM

Работа с расширенным синтаксисом командной строки NASM

Работа с компоновщиком LD

Запуск исполняемого файла

Выполнение заданий для самостоятельной работы.

3 Теоретическое введение

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.

4 Выполнение лабораторной работы

4.1 Создание программы Hello world!

С помощью команды `cd` перехожу в каталог и создаю пустой текстовый файл `hello.asm`(рис. 4.1).

```
[sofichandrey@fedora labs]$ cd lab04  
[sofichandrey@fedora lab04]$ touch hello.asm
```

Рис. 4.1: Перемещение по директориям и создание файла

Открываю файл в текстовом редакторе `gedit`(рис. 4.2).

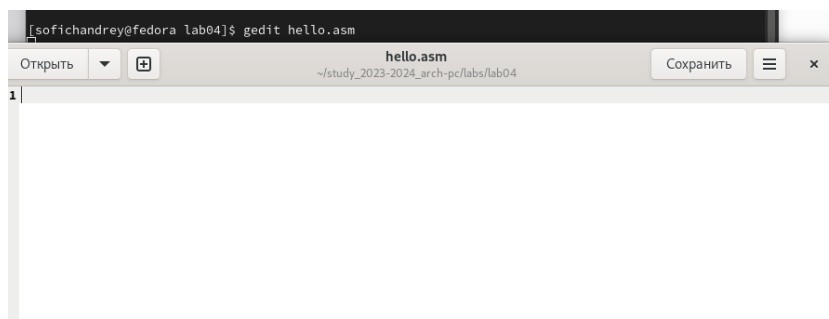
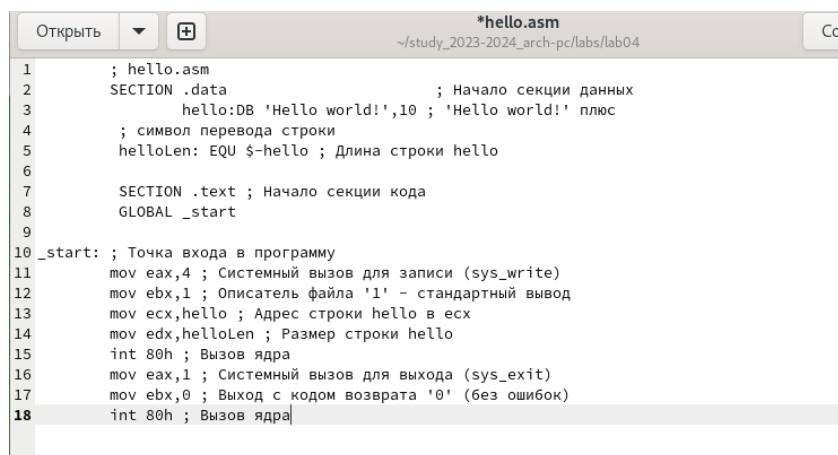


Рис. 4.2: Файл `hello.asm`

Заполняю файл, заполняю программу для вывода Hello world (рис. ??).

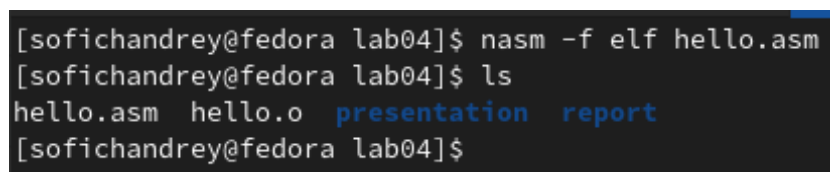


```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3     hello:DB 'Hello world!',10 ; 'Hello world!' плюс
4     ; символ перевода строки
5     helloLen: EQU $-hello ; Длина строки hello
6
7     SECTION .text ; Начало секции кода
8     GLOBAL _start
9
10 _start: ; Точка входа в программу
11    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

Рис. 4.3: Программа

4.2 Работа с транслятором NASM.

Превращаю текст программы для вывода “Hello world!” в объектный код с помощью транслятора NASM, используя команду `nasm -f elf hello.asm`, ключ `-f` указывает транслятору `nasm`, что требуется создать бинарный файл в формате ELF и с помощью утилиты `ls` проверяю создан ли файл `hello.o` (рис. 4.4).



```
[sofichandrey@fedora lab04]$ nasm -f elf hello.asm
[sofichandrey@fedora lab04]$ ls
hello.asm hello.o presentation report
[sofichandrey@fedora lab04]$
```

Рис. 4.4: Создание бинарного файла

##Работа с расширенным синтаксисом командной строки NASM. Ввожу команду, которая скомпилирует файл `hello.asm` в файл `obj.o`, при этом в файл будут включены символы для отладки (ключ `-g`), также с помощью ключа `-l` будет создан файл листинга `list.lst` и правильность выполнения команды. (рис. 4.5).

```
[sofichandrey@fedora lab04]$ nasm -o obj.o -f elf -g -l list.lst hello.asm
[sofichandrey@fedora lab04]$ ls
hello.asm  hello.o  list.lst  obj.o  presentation  report
[sofichandrey@fedora lab04]$
```

Рис. 4.5: Создание файла листинга

4.3 Работа с компоновщиком LD.

Передаю объектный файл hello.o на обработку компоновщику LD, чтобы получить исполняемый файл hello. Ключ -o задает имя создаваемого исполняемого файла. Далее проверяю с помощью утилиты ls правильность выполнения команды. (рис. 4.6).

```
[sofichandrey@fedora lab04]$ ls
hello  hello.asm  hello.o  list.lst  obj.o  presentation  report
```

Рис. 4.6: Получение исполняемого файла

Выполняю следующую команду. Исполняемый файл будет иметь имя main, т.к. после ключа -o было задано значение main. Объектный файл, из которого собран этот исполняемый файл, имеет имя obj.o. (рис. 4.7).

```
[sofichandrey@fedora lab04]$ ld -m elf_i386 obj.o -o main
[sofichandrey@fedora lab04]$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o  presentation  report
```

Рис. 4.7: Значение main

4.4 Запуск исполняемого файла.

Запускаю созданный файл, чтобы проверить программу. (рис. 4.8).

```
[sofichandrey@fedora lab04]$ ./hello
Hello world!
[sofichandrey@fedora lab04]$
```

Рис. 4.8: Проверка программы

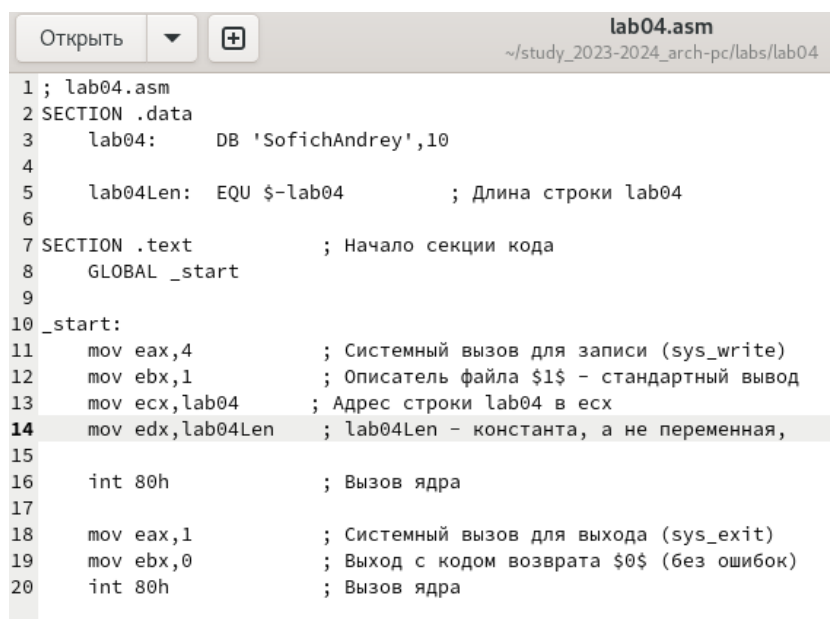
4.5 Выполнение заданий для самостоятельной работы.

Создаю копию файла, называю его lab04.asm. (рис. 4.9)

```
[sofichandrey@fedora lab04]$ cp hello.asm lab04.asm
```

Рис. 4.9: Копирование файла hello.asm

С помощью текстового редактора gedit открываю файл lab04.asm и меняю программу так, чтобы она выводила мои имя и фамилию.(рис. 4.10)



```

1 ; lab04.asm
2 SECTION .data
3     lab04:      DB 'SofichAndrey',10
4
5     lab04Len:   EQU $-lab04          ; Длина строки lab04
6
7 SECTION .text          ; Начало секции кода
8     GLOBAL _start
9
10 _start:
11     mov eax,4          ; Системный вызов для записи (sys_write)
12     mov ebx,1          ; Описатель файла $1$ - стандартный вывод
13     mov ecx,lab04      ; Адрес строки lab04 в ecx
14     mov edx,lab04Len   ; lab04Len - константа, а не переменная,
15
16     int 80h           ; Вызов ядра
17
18     mov eax,1          ; Системный вызов для выхода (sys_exit)
19     mov ebx,0          ; Выход с кодом возврата $0$ (без ошибок)
20     int 80h           ; Вызов ядра

```

Рис. 4.10: Изменение программы

Компилирую текст программы в объектный файл.Проверяю с помощью утилиты ls, что файл lab04.o создан.(рис. 4.11)

```
[sofichandrey@fedora lab04]$ nasm -f elf lab04.asm
[sofichandrey@fedora lab04]$ ls
hello      hello.o    lab04.o    main      presentation
hello.asm  lab04.asm  list.lst  obj.o     report
```

Рис. 4.11: Создание объектного файла

Передаю объектный файл lab04.o на обработку компоновщику LD, чтобы получить исполняемый файл lab04.(рис. 4.12)

```
[sofichandrey@fedora lab04]$ ld -m elf_i386 lab04.o -o lab04
[sofichandrey@fedora lab04]$ ls
hello      hello.o    lab04.asm  list.lst  obj.o      report
hello.asm  lab04     lab04.o    main      presentation
```

Рис. 4.12: Получение исполняемого файла

Запускаю исполняемый файл lab04, на экран действительно выводятся мои имя и фамилия.(рис. 4.13)

```
[sofichandrey@fedora lab04]$ ./lab04
SofichAndrey
```

Рис. 4.13: Проверка программы

Скидываю файлы hello.asm и lab04.asm в каталог 4 лабораторный работы.(рис. 4.14)

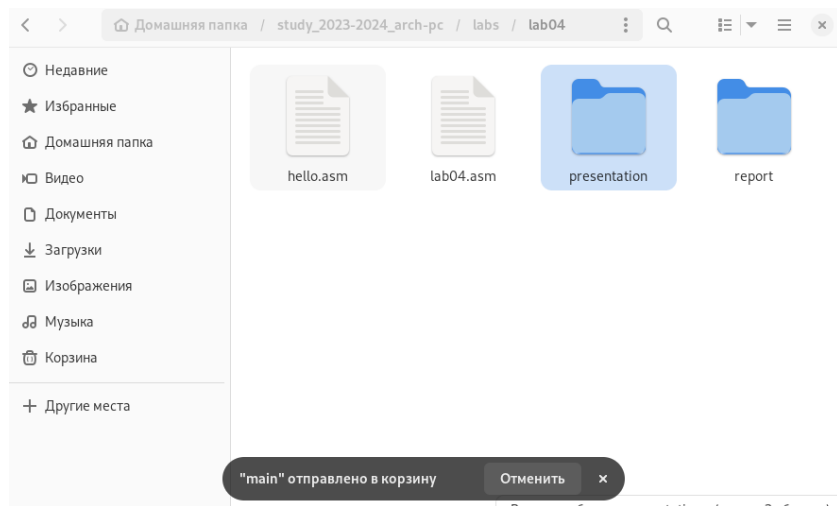


Рис. 4.14: Добавление файлов в лабораторную работу

Отправление всех изменений на github. (рис. 4.15)

```
[sofichandrey@fedora lab04]$ git add .
[sofichandrey@fedora lab04]$ git commit -am 'feat(main): add files lab-04'
warning: in the working copy of 'labs/lab02/report/Ло2_Софич_отчёт (1).md', CRLF
e replaced by LF the next time Git touches it
[master 6cb735c] feat(main): add files lab-04
 3 files changed, 267 insertions(+), 227 deletions(-)
 create mode 100644 labs/lab04/hello.asm
 create mode 100644 labs/lab04/lab04.asm
[sofichandrey@fedora lab04]$ git push
```

Рис. 4.15: Отправление работы на github

5 Выводы

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

Список литературы

::: {#Архитектура ЭВМ)