

Отчет по лабораторной работе №4

Дисциплина: Архитектура компьютера

Мустафина Аделя Юрисовна

Содержание

Цель работы	1
Задание	1
Теоретическое введение	2
Выполнение лабораторной работы	4
4.3.1. Программа Hello world!	4
4.3.2. Транслятор NASM.....	7
4.3.3. Расширенный синтаксис командной строки NASM.....	7
4.4. Компоновщик LD	8
4.4.1. Запуск исполняемого файла	8
4.5. Задание для самостоятельной работы	9
Выводы.....	10
Список литературы	10

Цель работы

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

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.

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.

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, преобразование (арифметические или логические операции) данных хранящихся в регистрах.

В качестве примера приведем названия основных регистров общего назначения (именно эти регистры чаще всего используются при написании программ):

- RAX, RCX, RDX, RBX, RSI, RDI — 64-битные
- EAX, ECX, EDX, EBX, ESI, EDI — 32-битные
- AX, CX, DX, BX, SI, DI — 16-битные
- AH, AL, CH, CL, DH, DL, BH, BL — 8-битные

Другим важным узлом ЭВМ является оперативное запоминающее устройство (ОЗУ). ОЗУ — это быстродействующее энергозависимое запоминающее устройство, которое напрямую взаимодействует с узлами процессора, предназначенное для хранения программ

и данных, с которыми процессор непосредственно работает в текущий момент. ОЗУ состоит из одинаковых пронумерованных ячеек памяти. Номер ячейки памяти — это адрес хранящихся в ней данных.

Периферийные устройства в составе ЭВМ:

- устройства внешней памяти, которые предназначены для долговременного хранения больших объёмов данных.
- устройства ввода-вывода, которые обеспечивают взаимодействие ЦП с внешней средой.

В основе вычислительного процесса ЭВМ лежит принцип программного управления. Это означает, что компьютер решает поставленную задачу как последовательность действий, записанных в виде программы.

Коды команд представляют собой многоразрядные двоичные комбинации из 0 и 1. В коде машинной команды можно выделить две части: операционную и адресную. В операционной части хранится код команды, которую необходимо выполнить. В адресной части хранятся данные или адреса данных, которые участвуют в выполнении данной операции.

При выполнении каждой команды процессор выполняет определённую последовательность стандартных действий, которая называется командным циклом процессора. Он заключается в следующем:

1. формирование адреса в памяти очередной команды;
2. считывание кода команды из памяти и её дешифрация;
3. выполнение команды;
4. переход к следующей команде.

Язык ассемблера (assembly language, сокращённо asm) — машинно-ориентированный язык низкого уровня.

NASM — это открытый проект ассемблера, версии которого доступны под различные операционные системы и который позволяет получать объектные файлы для этих систем. В NASM используется Intel-синтаксис и поддерживаются инструкции x86-64.

Выполнение лабораторной работы

4.3.1. Программа Hello world!

Перемещаюсь по каталогам для создания каталога для работы с программами на языке ассемблера. Но далее я скопировала все в другой каталог для своего удобства.

```
aymustafina@vbox:~$ mkdir -p ~/work/arch-pc/lab04
```

Создаю текстовый файл с именем hello.asm с помощью команды touch. И открываю этот файл с помощью текстового редактора gedit. Дополнительно его загружаю, так как на моей fedora его нет.

```
aymustafina@vbox:~/work/arch-pc/lab04$ touch hello.asm
```

```
aymustafina@vbox:~/work/arch-pc/lab04$ gedit hello.asm
```

```
bash: gedit: команда не найдена...
```

```
Установить пакет «gedit», предоставляющий команду «gedit»? [N/y] y
```

```
* Ожидание в очереди...
```

```
Следующие пакеты должны быть установлены:
```

```
amtk-5.6.1-6.fc40.x86_64      Actions, Menus and Toolbars Kit for GTK+ applica  
tions
```

```
gedit-2:46.2-1.fc40.x86_64    Text editor for the GNOME desktop
```

```
libgedit-gtksourceview-299.0.5-1.fc40.x86_64  Gedit Technology - Source code e  
diting widget
```

```
tepl-6.8.0-2.fc40.x86_64      Text editor product line library
```

```
Продолжить с этими изменениями? [N/y] y
```

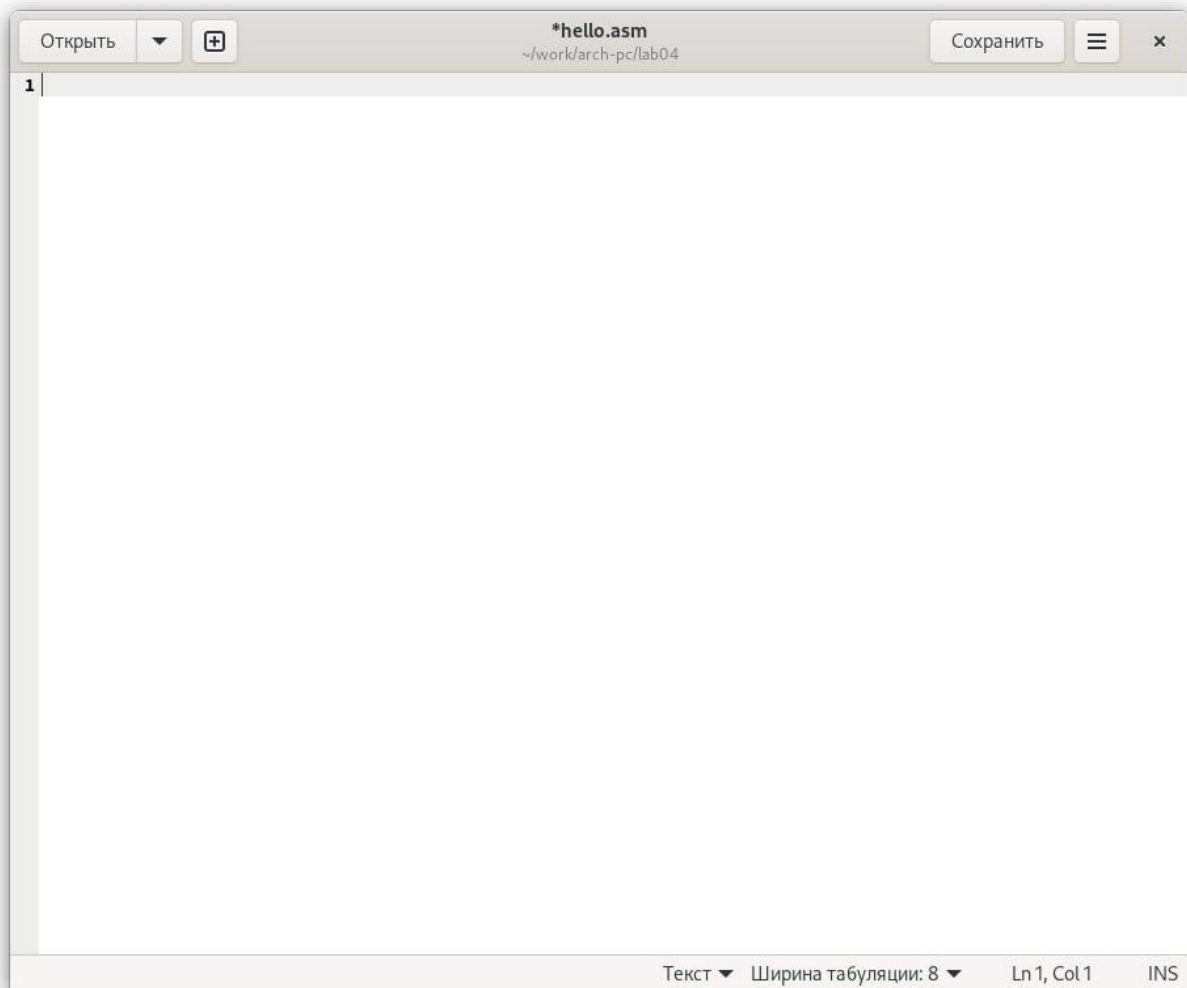
```
* Ожидание в очереди...
```

```
* Ожидание аутентификации...
```

```
* Ожидание в очереди...
```

```
* Загрузка пакетов...
```

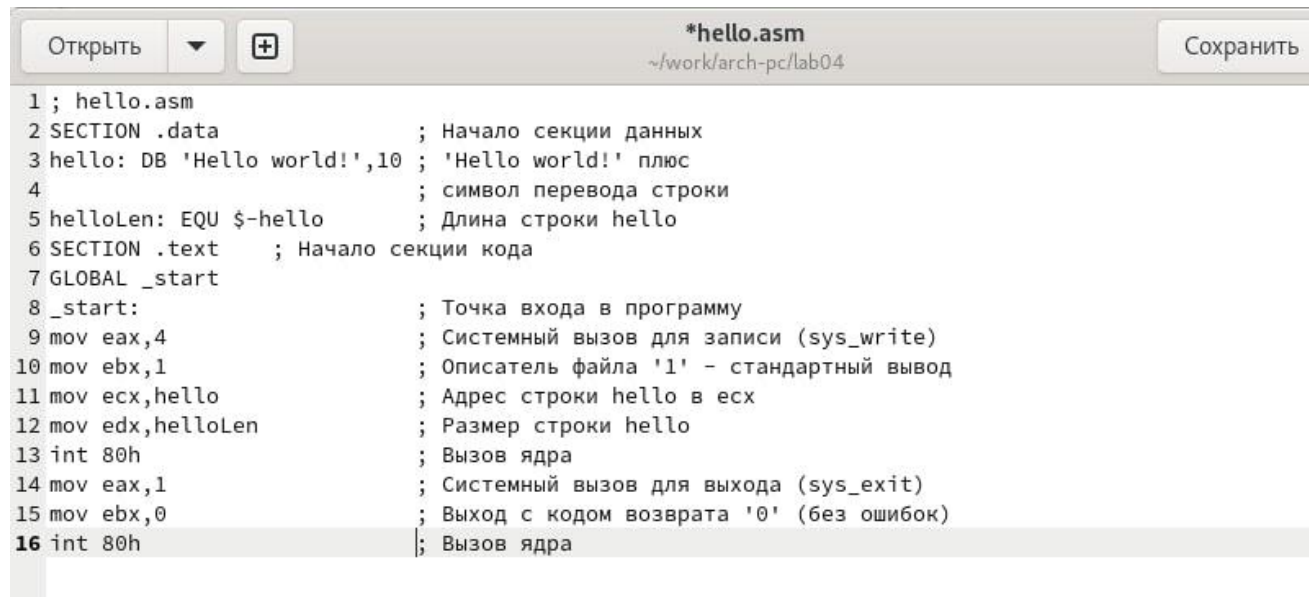
```
* Запрос данных...
```



Открытый исходный файл

Ввожу в данный файл код для вывода “Hello world!”.

4.3.2. Транслятор NASM



The screenshot shows a text editor window titled '*hello.asm' with the path '~/.work/arch-pc/lab04'. The editor contains the following assembly code:

```
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 SECTION .text          ; Начало секции кода
7 GLOBAL _start
8 _start:                ; Точка входа в программу
9 mov eax,4               ; Системный вызов для записи (sys_write)
10 mov ebx,1               ; Описатель файла '1' - стандартный вывод
11 mov ecx,hello           ; Адрес строки hello в ecx
12 mov edx,helloLen        ; Размер строки hello
13 int 80h                 ; Вызов ядра
14 mov eax,1               ; Системный вызов для выхода (sys_exit)
15 mov ebx,0               ; Выход с кодом возврата '0' (без ошибок)
16 int 80h                 ; Вызов ядра
```

Теперь для того, чтобы скомпилировать данный текст программы использую транслятор NASM. Я снова загружаю недостающую команду.

```
aymustafina@vbox:~/work/arch-pc/lab04$ nasm -f elf64 hello.asm
bash: nasm: команда не найдена...
Установить пакет «nasm», предоставляющий команду «nasm»? [N/y] y
```

```
* Ожидание в очереди...
Следующие пакеты должны быть установлены:
nasm-2.16.01-7.fc40.x86_64      A portable x86 assembler which uses Intel-like s
yntax
Продолжить с этими изменениями? [N/y] y
```

После загрузки снова ввожу эту команды и проверяю скомпилировался ли необходимый мне файл. Объектный файл имеет имя “hello.o”.

```
aymustafina@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
aymustafina@vbox:~/work/arch-pc/lab04$
```

4.3.3. Расширенный синтаксис командной строки NASM

Теперь с помощью другой команды компилирую исходный файл hello.asm в obj.o, с ним же будет создан файл листинга list.lst. Снова проверяю правильность выполнения с помощью команды ls.

```
aymustafina@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
aymustafina@vbox:~/work/arch-pc/lab04$
```

```
aymustafina@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f elf64 -g -l list.lst hello.asm
```

Компиляция файлов(2)

На этом моменте я перехожу в другой каталог, скопировав все созданные файлы для своего удобства.

```
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ls
bib hello.asm hello.o image list.lst Makefile obj.o pandoc report.md
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$
```

4.4. компоновщик LD

Объектный файл передам на обработку компоновщику и проверю правильность выполнения команды.

```
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ld -m elf_x86_64 hello.o -o hello
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ls
bib hello hello.asm hello.o image list.lst Makefile obj.o pandoc report.md
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$
```

Далее выполняю еще одну команду. Исполняемый файл имеет имя “main”.

```
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ld -m elf_x86_64 obj.o -o main
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ls
bib hello hello.asm hello.o image list.lst main Makefile obj.o pandoc report.md
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$
```

4.4.1. Запуск исполняемого файла

Запускаю созданный исполняемый файл, набрав в командной строке `./hello`.

```
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ./hello
Hello world!
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$
```

4.5. Задание для самостоятельной работы

Создаю копию файла `hello.asm` с именем `lab4.asm` с помощью команды `cp`.

```
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ cp hello.asm lab4.asm
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ls
bib hello.asm image list.lst Makefile pandoc report.md lab4.asm
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$
```

С помощью текстового редактора вношу изменения в файл, чтобы он выводил на экран мое имя и фамилию.


```
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ls
bib hello hello.asm hello.o image lab4.asm list.lst main Makefile obj.o pandoc report.md
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ gedit lab4.asm
```

Открыть
+

*lab4.asm

Сохранить
≡
x

~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report

```

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

```

Выполняю те же команды как для файла с hello.

```
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ gedit lab4.asm
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ nasm -o obj.o -f elf64 -g -l list.lst lab4.asm
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ gedit lab4.asm
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ls
bib hello hello.asm hello.o image lab4.asm list.lst main Makefile obj.o pandoc report.md
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ nasm -f elf64 lab4.asm
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ls
bib hello hello.asm hello.o image lab4.asm lab4.o list.lst main Makefile obj.o pandoc report.md
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$
```

Преобразование файлов

```
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ld -m elf_x86_64 lab4.o -o lab4
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ls
bib hello hello.asm hello.o image lab4 lab4.asm lab4.o list.lst main Makefile obj.o pandoc report.md
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$
```

Преобразование файлов

Вывожу свое имя и фамилию.

```
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ ./lab4
Mustafina Adelya
aymustafina@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$
```

Выводы

При выполнении лабораторной работы я изучила компиляцию файлов написанных на ассемблере NASM.

Здесь кратко описываются итоги проделанной работы.

Список литературы

1. https://esystem.rudn.ru/pluginfile.php/2089084/mod_resource/content/0/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%964.%20%D0%A1%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%B8%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%20%D0%BE%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%20%D0%BD%D0%B0%20%D1%8F%D0%B7%D1%8B%D0%BA%D0%B5%20%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0%20NASM.pdf