

РУДН. Архитектура компьютеров

Отчёт по лабораторной работе №4

Косинов Никита Андреевич, НПИМбв-02-20

Содержание

1	Цель работы	5
2	Ход работы	6
3	Написание простейшей программы	7
4	Создание исполняемого файла	9
5	Самостоятельная работа	11
6	Выводы	14

Список иллюстраций

3.1	Рабочая папка	7
3.2	Создание файла программы	7
3.3	Программа “Hello, world!”	8
4.1	Объектный файл	9
4.2	Расширенный синтаксис трансляции	9
4.3	Файл hello создан	9
4.4	Файл main создан	10
4.5	Запуск программ	10
5.1	Учётная запись с ключом	11
5.2	Создание папки	12
5.3	Репозиторий yamadharm	12
5.4	Создание репозитория	13

Список таблиц

1 Цель работы

Для взаимодействия компьютера и человека есть множество языков программирования (**группа C, Python, Java, Ruby** и пр.). В основном используются высокоуровневые языки, программы которых выглядят суть предложения на английском. Однако, у них есть общий недостаток: ограничения по доступу к возможностям ЭВМ и время работы из-за обработки зачастую лишних операций.

Ассемблерная программа же работает напрямую с ядром машины и, как следствие, наиболее приближена к машинному коду, за счёт чего делает ровно то, что от неё попросил программист. Поэтому она быстрее, но в то же время, и намного более громоздка. Но для выполнения простых задач или программирования простейших электронных устройств ассемблер необходим.

Цель данной работы - приобретение теоретических и практических навыков по написанию и дальнейшей компиляции простейшей программы, написанной на ассемблере **NASM**.

2 Ход работы

Лабораторная работа выполнена в терминале **ОС Linux** с использованием ассемблера и транслятора **NASM**. Действия по лабораторной работе представлены в следующем порядке:

1. Написание простейшей программы;
2. Создание исполняемого файла;
3. Самостоятельная работа.

3 Написание простейшей программы

По традиции, первой программой предлагается написать “*Hello world!*”.

1. Переходим в каталог *lab04* нашего локального репозитория.

```
nakosinov@dk4n68 ~ $ cd work/study/2023-2024/Архитектура\ компьютера/arch-pc/labs/lab04
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
presentation report
```

Рис. 3.1: Рабочая папка

2. Создаём файл формата *.asm с помощью команды создания файлов **touch**.

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ touch hello.asm
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm presentation report
```

Рис. 3.2: Создание файла программы

3. Открываем файл и записываем туда исполняемый код на языке ассемблер.

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ touch hello.asm
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm  presentation  report
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ gedit hello.asm
[
Открыть  hello.asm
~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04  Сохранить
1 SECTION .data
2     hello:      DB "Hello, world!",10
3
4     helloLen:   EQU $ - hello
5
6 SECTION .text
7     GLOBAL _start
8
9 _start:
10
11     mov eax, 4
12     mov ebx, 1
13     mov ecx, hello
14     mov edx, helloLen
15     int 80h
16
17     mov eax, 1
18     mov ebx, 0
19     int 80h
```

Рис. 3.3: Программа “Hello, world!”

4 Создание исполняемого файла

Чтобы компьютер исполнил нами задуманное, необходимо скомпилировать написанную программу в исполняемый файл. В данном случае будем использовать транслятор **NASM**.

1. Создаём файл формата *.o из написанной программы, используя ключ **-f**, чтобы полученный объектный файл был в необходимом формате **elf**.

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -f elf hello.asm
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm  hello.o  presentation  report
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 4.1: Объектный файл

2. Создаём второй объектный файл более полной командой трансляции, попутно получая файл листинга.

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm  hello.o  list.lst  obj.o  presentation  report
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 4.2: Расширенный синтаксис трансляции

3. Компонуем созданные объектные файлы в исполняемые посредством команды **ld**.

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 hello.o -o hello
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  list.lst  obj.o  presentation  report
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 4.3: Файл hello создан

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 obj.o -o main
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello hello.asm hello.o list.lst main obj.o presentation report
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 4.4: Файл main создан

4. Запускаем оба исполняемых файлов и убеждаемся, что оба работают исправно!

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ./hello
Hello, world!
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ./main
Hello, world!
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 4.5: Запуск программ

5 Самостоятельная работа

Для закрепления и большей проработки навыков использования языка ассемблер, напомним программу, выводящую на экран фамилию, имя и номер группы автора.

1. Копируем написанную ранее программу **hello.asm**. Дадём копии новое имя и открываем для редактирования.



The image shows a terminal window at the top with the following commands and output:

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ cp hello.asm lab4.asm
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ gedit lab4.asm
```

Below the terminal is a text editor window titled "lab4.asm" showing the following assembly code:

```
1 SECTION .data
2     hello:    DB "Hello, world!",10
3
4     helloLen: EQU $ - hello
5
6 SECTION .text
7     GLOBAL _start
8
9 _start:
10    mov eax, 4
11    mov ebx, 1
12    mov ecx, hello
13    mov edx, helloLen
14    int 80h
15
16    mov eax, 1
17    mov ebx, 0
18    int 80h
```

Рис. 5.1: Учётная запись с ключом

2. Изменяем код таким образом, чтобы выводились фамилия и имя на первой строке и номер группы на второй.

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ gedit lab4.asm
lab4.asm
1 SECTION .data
2     Id:      DB "Kosinov Nikita",10,"02-20 Group",10
3
4     IdLen:   EQU $ - Id
5
6 SECTION .text
7     GLOBAL _start
8
9 _start:
10    mov eax, 4
11    mov ebx, 1
12    mov ecx, Id
13    mov edx, IdLen
14    int 80h
15
16    mov eax, 1
17    mov ebx, 0
18    int 80h
```

Рис. 5.2: Создание папки

3. Создаём объектный файл **labObj.o** и далее исполняемый **Last_First_Name**.
Запускаем последний и видим, что он работает исправно.

```
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ gedit lab4.asm
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -o labObj.o -f elf
-g lab4.asm
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 labObj.o -o Last_First_Name
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ./Last_First_Name
Kosinov Nikita
02-20 Group
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 5.3: Репозиторий yamadharma

4. Заливаем все сделанные в течение лабораторной работы изменения в глобальный репозиторий на **github**.

```

nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ./Last_First_Name
Kosinov Nikita
02-20 Group
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ git add .
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ git commit -am 'add files lab4'
[master a98104a] add files lab4
10 files changed, 55 insertions(+), 1 deletion(-)
delete mode 100644 labs/lab03/report/..lock.report.docx#
create mode 100755 labs/lab04/Last_First_Name
create mode 100755 labs/lab04/hello
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/hello.o
create mode 100644 labs/lab04/lab4.asm
create mode 100644 labs/lab04/labObj.o
create mode 100644 labs/lab04/list.lst
create mode 100755 labs/lab04/main
create mode 100644 labs/lab04/obj.o
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ git push
Перечисление объектов: 20, готово.
Подсчет объектов: 100% (20/20), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (15/15), готово.
Запись объектов: 100% (15/15), 3.46 КиБ | 3.46 МиБ/с, готово.
Всего 15 (изменений 8), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (8/8), completed with 3 local objects.
To github.com:kosinovna-1/study_2023-2024_arch-pc.git
   ed931ef..a98104a  master -> master
nakosinov@dk4n68 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $

```

Рис. 5.4: Создание репозитория

6 Выводы

Ассемблерная программа - язык программирования, обращающийся к самому низкому доступному человеку уровню работы ЭВМ. За счёт этого он даёт возможность использовать потенциал машины максимально возможно.

При написании программы на ассемблере важно помнить, что каждая команда должна располагаться на отдельной строке, также важно следить за регистром. Как и любую другую программу, перед запуском её необходимо скомпилировать.