

# **Лабораторная работа №4 Создание и процесс обработки программ на языке ассемблера NASM**

**Дисциплина: Архитектура ЭВМ**

Шевырев Иван

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
2.1	Программа “Hello world!” . . . . .	6
2.1.1	Создадим файл <code>hello.asm</code> . . . . .	6
2.1.2	Воспользуемся транслятором <code>NASM</code> . . . . .	7
2.1.3	Воспользуемся компоновщиком <code>LD</code> . . . . .	8
2.1.4	Скомпилируем в файл с именем <code>main</code> с помощью ключа <code>-o</code> . . . . .	8
2.1.5	Запустим исполняемый файл . . . . .	8
<b>3</b>	<b>Задания для самостоятельной работы</b>	<b>10</b>
3.0.1	Создадим копию файла . . . . .	10
3.0.2	Изменим содержимое файла . . . . .	10
3.0.3	Воспользуемся транслятором <code>Nasm</code> . . . . .	10
3.0.4	Воспользуемся компоновщиком . . . . .	11
3.0.5	Запустим <code>./lab05</code> . . . . .	11
3.1	Загрузим файлы на <code>GitHub</code> . . . . .	11
3.1.1	Скопируем файлы в локальный репозиторий . . . . .	11
3.1.2	Создадим <code>git commit</code> . . . . .	11
3.1.3	Выгрузим файлы на гитхаб . . . . .	12
<b>4</b>	<b>Выводы</b>	<b>13</b>

# Список иллюстраций

2.1	Создание директори . . . . .	6
2.2	Создание lab05.asm и команда ls . . . . .	6
2.3	Открытие файла через текстовый редактор gedit . . . . .	7
2.4	Код введенный в файл . . . . .	7
2.5	Трансляция кода . . . . .	8
2.6	Создание объектного файла с другим именем . . . . .	8
2.7	компоновщик ld . . . . .	8
2.8	Исполняемый файл с именем main . . . . .	8
2.9	Запуск ./hello . . . . .	8
3.1	Копирование файлов . . . . .	10
3.2	Редактирование через текстовый редактор gedit . . . . .	10
3.3	Трансляция lab05 . . . . .	10
3.4	Использование команды ld . . . . .	11
3.5	Исполнение lab05 . . . . .	11
3.6	Копирование всех asm файлов в локальный репозиторий . . . . .	11
3.7	git add и git commit . . . . .	11
3.8	git push . . . . .	12

## Список таблиц

# 1 Цель работы

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

## 2 Выполнение лабораторной работы

### 2.1 Программа “Hello world!”

#### 2.1.1 Создадим файл `hello.asm`

##### 2.1.1.1 Создадим каталог `lab05`

```
[idshevihryov@pivo ~]$ mkdir work/study/2022-2023/study_2022-2023_arh-pc/lab05
[idshevihryov@pivo ~]$ cd work/study/2022-2023/study_2022-2023_arh-pc/lab05
[idshevihryov@pivo lab05]$ ls
[idshevihryov@pivo lab05]$
```

Рис. 2.1: Создание директории

##### 2.1.1.2 Создадим текстовый файл `lab05.asm`

```
[idshevihryov@pivo lab05]$ touch hello.asm
[idshevihryov@pivo lab05]$ ls
hello.asm
[idshevihryov@pivo lab05]$
```

Рис. 2.2: Создание `lab05.asm` и команда `ls`

### 2.1.1.3 Откроем файл через Gedit

```
[idshevihryov@pivo lab05]$ gedit hello.asm

(gedit:14388): dbind-WARNING **: 11:23:44.642: Couldn't connect to database: No such file or directory

(gedit:14388): dconf-WARNING **: 11:23:44.676: Unable to open /etc/gconf/gconf.xml: No such file or directory

(gedit:14388): dconf-CRITICAL **: 11:23:44.676: unable to create schema: No such file or directory

(gedit:14388): dconf-CRITICAL **: 11:23:44.676: unable to create schema: No such file or directory

(gedit:14388): dconf-CRITICAL **: 11:23:44.676: unable to create schema: No such file or directory

(gedit:14388): dconf-CRITICAL **: 11:23:44.676: unable to create schema: No such file or directory

(gedit:14388): dconf-CRITICAL **: 11:23:44.676: unable to create schema: No such file or directory

(gedit:14388): dconf-CRITICAL **: 11:23:44.676: unable to create schema: No such file or directory
```

Рис. 2.3: Открытие файла через текстовый редактор gedit

### 2.1.1.4 Введем код в файл

```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3     hello: DB 'Hello world!',10 ; 'Hello world!' плюс
4                                     ; символ перевода строки
5
6     helloLen: EQU $-hello ; Длина строки hello
7
8 SECTION .text ; Начало секции кода
9     GLOBAL _start
10
11 _start: ; Точка входа в программу
12     mov eax,4 ; Системный вызов для записи (sys_write)
13     mov ebx,1 ; Описатель файла '1' - стандартный вывод
14     mov ecx,hello ; Адрес строки hello в ecx
15     mov edx,helloLen ; Размер строки hello
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
21
```

Рис. 2.4: Код введенный в файл

## 2.1.2 Воспользуемся транслятором NASM

Скомпилируем вышенаписанную программу с помощью команды `nasm -f elf hello.asm`

```
[idshevihryov@pivo lab05]$ nasm -f elf hello.asm
[idshevihryov@pivo lab05]$ ls
hello.asm hello.o
[idshevihryov@pivo lab05]$
```

Рис. 2.5: Трансляция кода

Создадим файл с другим именем используя `-o` и создадим листинг, с помощью `-l`

```
[idshevihryov@pivo lab05]$ nasm -o obj.o -f elf -g -l list.lst hello.asm
[idshevihryov@pivo lab05]$ ls
hello.asm hello.o list.lst obj.o
[idshevihryov@pivo lab05]$
```

Рис. 2.6: Создание объектного файла с другим именем

### 2.1.3 Воспользуемся компоновщиком LD

Выполним команду `ld -m elf_i386 hello.o -o hello`

```
[idshevihryov@pivo lab05]$ ld -m elf_i386 hello.o -o hello
[idshevihryov@pivo lab05]$ ls
hello hello.asm hello.o list.lst obj.o
[idshevihryov@pivo lab05]$
```

Рис. 2.7: компоновщик ld

С помощью команды `ls`, увидим что файл `hello` создался

### 2.1.4 Скомпонуем в файл с именем main с помощью ключа -o

```
[idshevihryov@pivo lab05]$ ld -m elf_i386 obj.o -o main
[idshevihryov@pivo lab05]$ ls
hello hello.asm hello.o list.lst main obj.o
[idshevihryov@pivo lab05]$
```

Рис. 2.8: Исполняемый файл с именем main

### 2.1.5 Запустим исполняемый файл

```
[idshevihryov@pivo lab05]$ ./hello
Hello world!
[idshevihryov@pivo lab05]$
```

Рис. 2.9: Запуск ./hello



На экран вывелось “Hello world!”

## 3 Задания для самостоятельной работы

### 3.0.1 Создадим копию файла

Скопируем содержимое из файла `hello.asm` в `lab5.asm`

```
idshevihryov@pivo study_2022-2023_arh-pc]$ cp lab05/hello.asm labs/lab05/lab5.asm
idshevihryov@pivo study_2022-2023_arh-pc]$ ls labs/lab05/
lab5.asm  presentation  report
idshevihryov@pivo study_2022-2023_arh-pc]$
```

Рис. 3.1: Копирование файлов

### 3.0.2 Изменим содержимое файла

С помощью редактора Gedit заменим строку с “Hello world” на свое имя и фамилию

```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3     hello: DB 'Шевырев Иван',10 ; 'Шевырев Иван' плюс
4                                     ; символ перевода строки
5
6     helloLen: EQU $-hello        ; Длина строки hello
7
```

Рис. 3.2: Редактирование через текстовый редактор gedit

### 3.0.3 Возпользуемся транслятором Nasm

```
idshevihryov@pivo lab05]$ nasm -o lab5.o -f elf -g -l list.lst lab5.asm
idshevihryov@pivo lab05]$ ls
lab5.asm  lab5.o  list.lst  presentation  report
idshevihryov@pivo lab05]$
```

Рис. 3.3: Трансляция lab05

### 3.0.4 Воспользуемся компоновщиком

С помощью команды `ld` создадим исполняемый файл `lab5`

```
[idshevihryov@pivo lab05]$ ld -m elf_i386 lab5.o -o lab5
[idshevihryov@pivo lab05]$ ls
lab5 lab5.asm lab5.o list.lst presentation report
[idshevihryov@pivo lab05]$
```

Рис. 3.4: Использование команды `ld`

### 3.0.5 Запустим `./lab05`

```
[idshevihryov@pivo lab05]$ ./lab5
Левырев Иван
[idshevihryov@pivo lab05]$
```

Рис. 3.5: Исполнение `lab05`

Видим как на экран выводятся наша имя и фамилия

## 3.1 Загрузим файлы на GitHub

### 3.1.1 Скопируем файлы в локальный репозиторий

```
[idshevihryov@pivo lab05]$ cp *.asm ../labs/lab05
[idshevihryov@pivo lab05]$ ls ../labs/lab05
hello.asm lab5.asm presentation report
[idshevihryov@pivo lab05]$
```

Рис. 3.6: Копирование всех `asm` файлов в локальный репозиторий

### 3.1.2 Создадим `git commit`

```
[idshevihryov@pivo study_2022-2023_arh-pc]$ git add .
[idshevihryov@pivo study_2022-2023_arh-pc]$ git commit -m "лабораторная 4 задания для самостоятельной работы"
[master 256b855] лабораторная 4 задания для самостоятельной работы
```

Рис. 3.7: `git add` и `git commit`

### 3.1.3 Выгрузим файлы на гитхаб

используем команду git push

```
[idshevihryov@pivo study_2022-2023_arh-pc]$ git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 1.03 KiB | 1.03 MiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:IvanShevyrev/study_2022-2023_arh-pc.git
   f3bc089..256b855  master -> master
[idshevihryov@pivo study_2022-2023_arh-pc]$
```

Рис. 3.8: git push

## 4 Выводы

За эту лабораторной работы мы научились переводить программы на языке ассемблера NASM в исполняемый файл с помощью трансляции и компоновки через `ld`.

Мы написали программу на NASM, которая выводит в терминал нашу фамилию и имя.