

Создание и процесс обработки программ на языке ассемблера NASM

Отчёт

Сидорова Арина Валерьевна

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
2.1	Hello, world!	5
2.2	Транслятор NASM	6
2.3	Расширенный синтаксис командной строки NASM	6
2.4	Компоновщик LD	6
2.5	Запуск исполняемого файла	7
2.6	Задание для самостоятельной работы	7
3	Выводы	10

Список иллюстраций

2.1	Создаем каталоги и текстовый файл, открываем его	5
2.2	Вписываем команды в файл	6
2.3	Создаем файл и проверяем, создался ли он	6
2.4	Преобразование hello.asm в obj.o	6
2.5	Объектный файл передаем на обработку компоновщику с помощью ld	7
2.6	Используем ld	7
2.7	Запуск файла	7
2.8	Создание копии файла, внесение изменений в текст программы .	8
2.9	Редактируем файл	8
2.10	Оттранслируем полученный текст	8
2.11	Копируем созданные файлы в локальный репозиторий	9
2.12	Отправляем файлы на github	9

1 Цель работы

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

2 Выполнение лабораторной работы

2.1 Hello, world!

Создаем каталог для работы с программами на языке ассемблера NASM и создаем текстовый файл, открываем файл с помощью текстового редактора(рис. fig. 2.1).

```
avsidorova@avsidorova:~$ mkdir -p ~/work/arch-pc/lab04
avsidorova@avsidorova:~$ ls
snap  work  Документы  Изображения  Общедоступные  Шаблоны
tmp   Видео  Загрузки  Музыка       'Рабочий стол'
avsidorova@avsidorova:~$ cd work
avsidorova@avsidorova:~/work$ ls
arch-pc  study
avsidorova@avsidorova:~/work$ cd study
avsidorova@avsidorova:~/work/study$ ls
2023-2024
avsidorova@avsidorova:~/work/study$ cd
avsidorova@avsidorova:~$ cs ~/work/arch-pc/lab04
Команда «cs» не найдена, но может быть установлена с помощью:
sudo apt install csound
avsidorova@avsidorova:~$ cd ~/work/arch-pc/lab04
avsidorova@avsidorova:~/work/arch-pc/lab04$ touch hello.asm
avsidorova@avsidorova:~/work/arch-pc/lab04$ ls
hello.asm
avsidorova@avsidorova:~/work/arch-pc/lab04$ gedit hello.asm
```

Рис. 2.1: Создаем каталоги и текстовый файл, открываем его

Вписываем команды в файл(рис. fig. 2.2).

```

1 ; hello.asm
2 SECTION .data                ; Начало секции данных
3     hello: DB 'Hello,world!',10 ; 'hello, world!'
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
15    mov eax,1                 ; Системный вызов для выхода (sys_exit)
16    mov ebx,0                 ; Выход с кодом возврата '0' (без ошибок)
17    int 80h                   ; Вызов ядра

```

Рис. 2.2: Вписываем команды в файл

2.2 Транслятор NASM

Преобразуем текст программы в объектный код (рис. fig. 2.3)

```

avsidorova@avsidorova:~/work/arch-pc/lab04$ nasm -f elf hello.asm
avsidorova@avsidorova:~/work/arch-pc/lab04$ ls
hello.asm  hello.o

```

Рис. 2.3: Создаем файл и проверяем, создался ли он

2.3 Расширенный синтаксис командной строки NASM

Выполняем команды, скомпилируем исходный файл, проверим создался ли он (рис. fig. 2.4)

```

avsidorova@avsidorova:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.asm
avsidorova@avsidorova:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o

```

Рис. 2.4: Преобразование hello.asm в obj.o

2.4 Компоновщик LD

Объектный файл передаем на обработку компоновщику:(рис. fig. 2.5)

```
avsidorova@avsidorova:~/work/arch-pc/lab04$ ld -m elf_i386
hello.o -o hello
avsidorova@avsidorova:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o list.lst obj.o
avsidorova@avsidorova:~/work/arch-pc/lab04$
```

Рис. 2.5: Объектный файл передаем на обработку компоновщику с помощью ld

Выполняем следующую команду:(рис. fig. 2.6)

```
avsidorova@avsidorova:~/work/arch-pc/lab04$ ld -m elf_i386
obj.o -o main
avsidorova@avsidorova:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o list.lst main obj.o
avsidorova@avsidorova:~/work/arch-pc/lab04$
```

Рис. 2.6: Используем ld

2.5 Запуск исполняемого файла

Запустим на выполнение созданный исполняемый файл, находящийся в текущем каталоге(рис. fig. 2.7)

```
avsidorova@avsidorova:~/work/arch-pc/lab04$ ./hello
Hello,world!
avsidorova@avsidorova:~/work/arch-pc/lab04$
```

Рис. 2.7: Запуск файла

2.6 Задание для самостоятельной работы

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

С помощью текстового редактора вносим изменения в текст программы в файле lab4.asm так, чтобы вместо Hello world! на экран выводилась строка с моим ФИО (рис. fig. 2.8) (рис. fig. 2.9).

```

avsidorova@avsidorova:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
avsidorova@avsidorova:~/work/arch-pc/lab04$ gedit lab4.asm
Gtk-Message: 01:16:54.083: Not loading module "atk-bridge": The f

```

Рис. 2.8: Создание копии файла, внесение изменений в текст программы

```

1 ; hello.asm
2 SECTION .data ; Начало секции данных
3     hello: DB 'Сидорова Арина Валерьевна!',10 ; 'Сидорова Арина Валерьевна!'
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
15    mov eax,1 ; Системный вызов для выхода (sys_exit)
16    mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
17    int 80h ; Вызов ядра

```

Рис. 2.9: Редактируем файл

Оттранслируем полученный текст программы lab4.asm в объектный файл. Выполним компоновку объектного файла и запустим получившийся исполняемый файл.(рис. fig. 2.10).

```

avsidorova@avsidorova:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
avsidorova@avsidorova:~/work/arch-pc/lab04$ nasm -o obj.o -f elf
-g -l list.lst lab4.asm
avsidorova@avsidorova:~/work/arch-pc/lab04$ ls
hello    hello.o  lab4.o   main
hello.asm lab4.asm list.lst obj.o
avsidorova@avsidorova:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o
-o hello
avsidorova@avsidorova:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o
-o main
avsidorova@avsidorova:~/work/arch-pc/lab04$ ./hello
Сидорова Арина Валерьевна!
avsidorova@avsidorova:~/work/arch-pc/lab04$

```

Рис. 2.10: Оттранслируем полученный текст

Скопируем файлы hello.asm и lab4.asm в локальный репозиторий в каталог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/. Загрузим файлы на Github.(рис. 2.11)(рис. fig. 2.12)


```

avsidorova@avsidorova:~/work/arch-pc/lab04$ cp hello.asm ~/work/st
udy/2023-2024/"Архитектура компьютера"/arch-pc/labs/lab04/
avsidorova@avsidorova:~/work/arch-pc/lab04$ cp lab4.asm ~/work/st
udy/2023-2024/"Архитектура компьютера"/arch-pc/labs/lab04/
avsidorova@avsidorova:~/work/arch-pc/lab04$ cd ~/work/study/2023-
2024/"Архитектура компьютера"/arch-pc/labs/lab04/
avsidorova@avsidorova:~/work/study/2023-2024/Архитектура компьюте
pa/arch-pc/labs/lab04$ ls
hello.asm lab4.asm presentation report
avsidorova@avsidorova:~/work/study/2023-2024/Архитектура компьюте
pa/arch-pc/labs/lab04$

```

Рис. 2.11: Копируем созданные файлы в локальный репозиторий

```

pa/arch-pc/labs/lab04$ git add .
avsidorova@avsidorova:~/work/study/2023-2024/Архитектура компьюте
pa/arch-pc/labs/lab04$ git commit -am 'feat(main) : add files lab
-4'
[master 6809f85] feat(main) : add files lab-4
2 files changed, 34 insertions(+)
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/lab4.asm
avsidorova@avsidorova:~/work/study/2023-2024/Архитектура компьюте
pa/arch-pc/labs/lab04$ git push
Перечисление объектов: 9, готово.
Подсчет объектов: 100% (9/9), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 1020 байтов | 23.00 КиБ/с, готово.
Всего 6 (изменений 3), повторно использовано 0 (изменений 0), пов
торно использовано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 2 local obje
cts.
To github.com:avsidorova/study_2023-2024_arh--pc.git
27bf62b..6809f85 master -> master
avsidorova@avsidorova:~/work/study/2023-2024/Архитектура компьюте
pa/arch-pc/labs/lab04$

```

Рис. 2.12: Отправляем файлы на github

3 Выводы

Мы познакомились с языком ассемблера NASM и создали две работающих программы.