Лабораторная работа №4

Студент: Кара Егор **Группа**: НБИбд-01-25

---

1. Цель работы Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.
2. Программа Hello world!
    1. Рассмотрим пример простой программы на языке ассемблера NASM. Традиционно первая программа выводит приветственное сообщение Hello world! на экран. Создаем каталог для работы с программами на языке ассемблера NASM:
    ```
    eakara@eakara-VirtualBox:~$ mkdir -p ~/work/arch-pc/lab04
    eakara@eakara-VirtualBox:~$
    ```
    Рис.1
    2. Перейдем в созданный каталог:
    ```
    eakara@eakara-VirtualBox:~$ cd ~/work/arch-pc/lab04
    eakara@eakara-VirtualBox:~/work/arch-pc/lab04$
    ```
    Рис.2
    3. Создаем текстовый файл с именем hello.asm:
    ```
    eakara@eakara-VirtualBox:~/work/arch-pc/
    eakara@eakara-VirtualBox:~/work/arch-pc/
    ```
    Рис.3
    4. Откроем этот файл с помощью любого текстового редактора, например, gedit
    ```
    eakara@eakara-VirtualBox:~/work/arch-pc/lab04$ gedit hello.asm
    ```
    Рис.4
    5. и введем в него следующий текст:

    Open — hello.asm — ~/work/arch-pc/lab04 — Save

    ```
    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
    17
    ```
    Рис.5
3. Транслятор NASM
    1. NASM превращает текст программы в объектный код. Например, для компиляции приведённого выше текста программы «Hello World» необходимо написать: nasm

-f elf hello.asm. С помощью команды ls проверим, что объектный файл был создан.

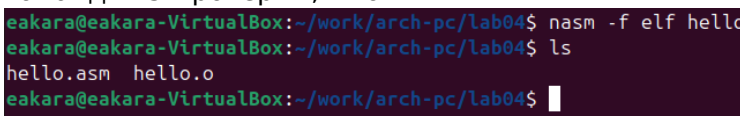

Рис.6

4. Расширенный синтаксис командной строки NASM

    1. Выполним следующую команду: nasm -o obj.o -f elf -g -l list.lst hello.asm Данная команда скомпилирует исходный файл hello.asm в obj.o (опция -o позволяет задать имя объектного файла, в данном случае obj.o), при этом формат выходного файла будет elf, и в него будут включены символы для отладки (опция -g), кроме того, будет создан файл листинга list.lst (опция -l). С помощью команды ls проверили, что файлы были созданы.

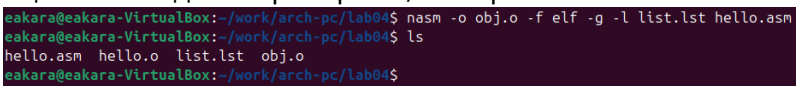

Рис.7

5. Компоновщик LD

    1. Как видно из схемы на рис. 4.3, чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику: ld -m elf_i386 hello.o -o hello

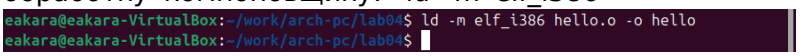

Рис.8 С помощью команды ls проверим, что исполняемый файл hello был создан.

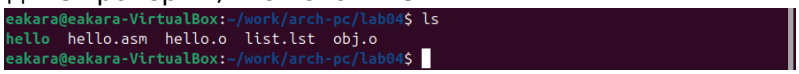

Рис.9 Компоновщик ld не предполагает по умолчанию расширений для файлов, но принято использовать следующие расширения: • o – для объектных файлов; • без расширения – для исполняемых файлов; • map – для файлов схемы программы; • lib – для библиотек. Ключ -o с последующим значением задаёт в данном случае имя создаваемого исполняемого файла. Выполним следующую команду: ld -m elf_i386 obj.o -o main

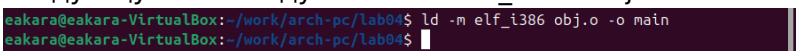

Рис.10 Какое имя будет иметь исполняемый файл? Какое имя имеет объектный файл из которого собран этот исполняемый файл?

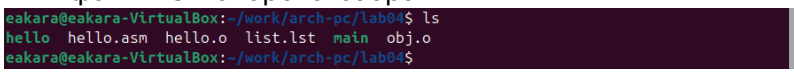

Рис.11

6. Запуск исполняемого файла

1. Запустить на выполнение созданный исполняемый файл, находящийся в текущем каталоге, можно, набрав в командной строке: ./hello

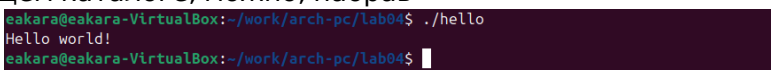

Рис.12

7. Самостоятельная работа

    1. В каталоге ~/work/arch-pc/lab04 с помощью команды cp создайте копию файла hello.asm с именем lab4.asm
    2. С помощью любого текстового редактора внесите изменения в текст программы в файле lab4.asm так, чтобы вместо Hello world! на экран выводилась строка с вашими фамилией и именем.
    3. Оттранслируйте полученный текст программы lab4.asm в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.
    4. Скопируйте файлы hello.asm и lab4.asm в Ваш локальный репозиторий в ката- лог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/. Загрузите файлы на Github.

---

8. Выполнение работы

Перешел в нужный каталог и сделал копию: (Рис.1)

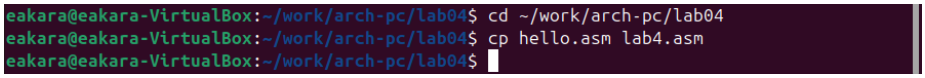

Рис.1

Открыл файл lab4.asm в редакторе nano: (Рис.2)

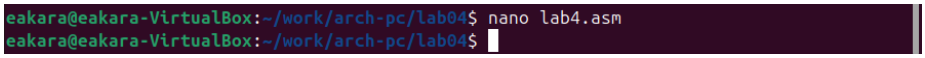

Рис.2

Нашел строку, где выводится Hello world!, и заменил ее на свою фамилию и имя: (Рис.3)

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

Рис.3

Использую NASM и LD (Рис.4)

```
eakara@eakara-VirtualBox:~/work/arch-pc/lab04$ nasm -f elf32 lab4.asm -o lab4.o
eakara@eakara-VirtualBox:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
eakara@eakara-VirtualBox:~/work/arch-pc/lab04$ ./lab4
Кара Егор
eakara@eakara-VirtualBox:~/work/arch-pc/lab04$
```

Рис.4

Скопировал оба файла в нужный каталог (Рис.5)

```
eakara@eakara-VirtualBox:~/work/arch-pc/lab04$ cp hello.asm lab4.asm ~/work/study/2025-2026/"Computer architecture"/arch-pc/labs/lab04/
eakara@eakara-VirtualBox:~/work/arch-pc/lab04$
```

Рис.5

Перешел в каталог репозитория и выполнил стандартные команды Git (Рис.6)

```
eakara@eakara-VirtualBox:~/work/arch-pc/lab04$ cd ~/work/study/2025-2026/"Computer architecture"/arch-pc/labs/lab04/
eakara@eakara-VirtualBox:~/work/study/2025-2026/Computer architecture/arch-pc/labs/lab04$ git add hello.asm lab4.asm
eakara@eakara-VirtualBox:~/work/study/2025-2026/Computer architecture/arch-pc/labs/lab04$ git commit -m "Добавлены файлы hello.asm и lab4.asm для лабораторной работы №4"
[master afc0157] Добавлены файлы hello.asm и lab4.asm для лабораторной работы №4
 2 files changed, 34 insertions(+)
 create mode 100644 labs/lab04/hello.asm
 create mode 100644 labs/lab04/lab4.asm
```

Рис.6

Выполнил команду git push origin master (Рис.7)

```
eakara@eakara-VirtualBox:~/work/study/2025-2026/Computer architecture/arch-pc/labs/lab04$ git push origin master
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 1014 bytes | 1014.00 KiB/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:EgorCara/study_2025-2026_arh-pc.git
   6bf18c..afc0157  master -> master
eakara@eakara-VirtualBox:~/work/study/2025-2026/Computer architecture/arch-pc/labs/lab04$
```

Рис.7

После этого файлы были загружены на GitHub

9. Вывод: Я освоил процедуры компиляции и сборки программ, написанных на ассемблере NASM.