**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4**

*дисциплина: Архитектура компьютера*

Студент: Грицко Сергей

Группа: НКАбд-02-25

**МОСКВА**

2025 г.

Оглавление

[Цель работы 3](#_bookmark0)

[Теоретическое введение 4](#_bookmark1)

* 1. [Основные принципы работы компьютера 4](#_bookmark2)
  2. [Ассемблер и язык ассемблера 5](#_bookmark3)
  3. [Процесс создания и обработки программы на языке ассемблера 7](#_bookmark4)

[Описание результатов выполнения лабораторной работы 8](#_bookmark5)

[Описание результатов выполнения заданий для самостоятельной работы 10](#_bookmark6)

[Выводы 11](#_bookmark7)

[Список литературы 12](#_bookmark8)

# Цель работы

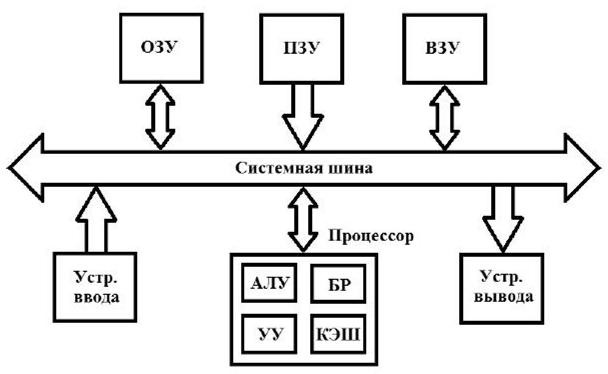
Освоение процедуры компиляции и сборки программ, написанных на ассемблере

NASM.

# Теоретическое введение

## Основные принципы работы компьютера

Основными функциональными элементами любой электронно-вычислительной машины (ЭВМ) являются центральный процессор, память и периферийные устройства. Взаимодействие этих устройств осуществляется через общую шину, к которой они подключены.[1]



*Рис. 1. Структурная схема ЭВМ*

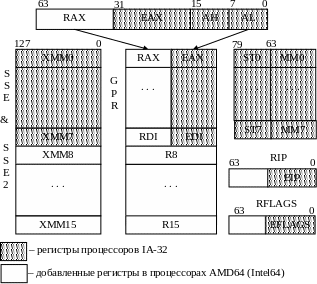
Основной задачей процессора является обработка информации и координация всех узлов компьютера. В состав центрального процессора (ЦП) входят:

* + - **Арифметико-логическое устройство (АЛУ)** — выполняет логические и арифметические действия.
    - **Устройство управления (УУ)** — обеспечивает управление и контроль всех устройств компьютера.
    - **Регистры** — сверхбыстрая оперативная память небольшого объёма для временного хранения промежуточных результатов.

Для программирования на ассемблере необходимо знать регистры процессора, так как большинство команд используют их в качестве операндов. Доступ к регистрам осуществляется по именам.[2]

Основные регистры общего назначения архитектуры x86:

* + - * **64-битные:** RAX, RCX, RDX, RBX, RSI, RDI
      * **32-битные:** EAX, ECX, EDX, EBX, ESI, EDI
      * **16-битные:** AX, CX, DX, BX, SI, DI
      * **8-битные:** AH, AL, CH, CL, DH, DL, BH, BL (старшая и младшая половины 16- битных регистров).



*Рис. 2. 64-битный регистр процессора ‘RAX’*

Другим важным узлом ЭВМ является оперативное запоминающее устройство (ОЗУ)

— быстродействующее энергозависимое хранилище для программ и данных, с которыми процессор работает в текущий момент.

В основе работы ЭВМ лежит принцип программного управления: компьютер решает задачу как последовательность действий (программу), состоящую из машинных команд.

Командный цикл процессора включает в себя формирование адреса команды, её считывание, дешифрацию и выполнение.

## Ассемблер и язык ассемблера

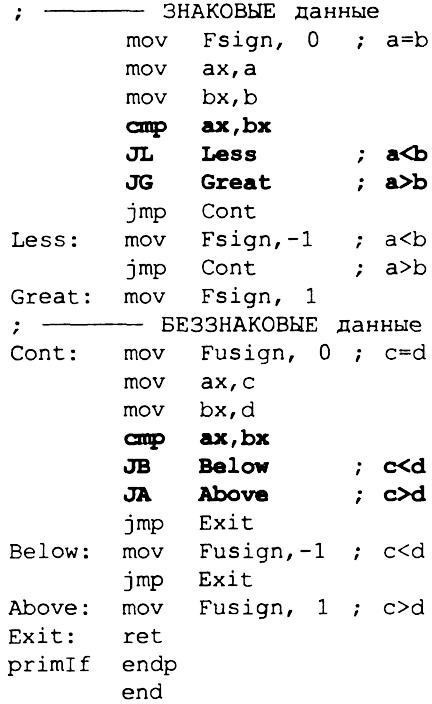
**Язык ассемблера (assembly language, asm)** — машинно-ориентированный язык низкого уровня, приближенный к архитектуре ЭВМ. Программы на ассемблере обращаются напрямую к ядру операционной системы, что дает более полный доступ к ресурсам компьютера по сравнению с языками высокого уровня.

Процессор понимает не команды ассемблера, а **машинные коды** (последовательности нулей и единиц). Преобразование команд с языка ассемблера в машинный код выполняет специальная программа-транслятор — **Ассемблер**.

Для каждой архитектуры процессора существует свой язык ассемблера. В нашем

курсе используется **NASM (Netwide Assembler)** — открытый ассемблер для архитектуры x86, использующий Intel-синтаксис.

Типичный формат команды в NASM:



*Рис. 3. Программа на assembler*

Программа также может содержать **директивы** — инструкции, управляющие работой транслятора (например, для определения данных).

## Процесс создания и обработки программы на языке ассемблера

Процесс создания ассемблерной программы состоит из четырех шагов:

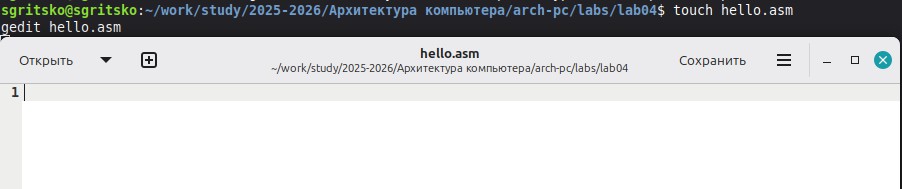


*Рис. 4. Процесс создания ассемблерной программы*

# Описание результатов выполнения лабораторной работы

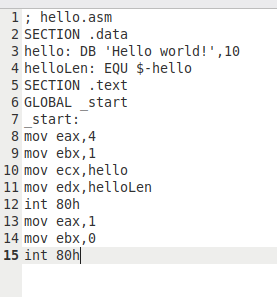
Для начала я перейду lab04 для выполнения работы

*Рис. 5 Переход в каталог лаб4*

Далее я создал файл hello.asm и открыл его в текстовом редакторе gedit, чтобы ввести предоставленный в методических указаниях код программы.

*Рис. 6 Переход в текстовый редактор*

Содержимое файла hello.asm:



Я выполнил трансляцию исходного кода в объектный файл hello.o в формате elf.

*Рис. 7 Трансляция кода*

После выполнения команды я проверил наличие созданного файла с помощью команды ls.

*Рис.8 Вывод команды ls, показывающий наличие файлов hello.asm и hello.o в каталоге.*

Команда ls показала, что объектный файл hello.o был успешно создан в текущем каталоге.

Затем я использовал команду с дополнительными параметрами для создания

объектного файла с другим именем (obj.o), добавления отладочной информации (-g) и генерации файла листинга (list.lst) и снова проверил содержимое каталога.

*Рис.9 Вывод команды ls, где видны файлы hello.asm, hello.o, obj.o и list.lst.*

Как и ожидалось, были созданы объектный файл obj.o и файл листинга list.lst.

Для получения исполняемого файла я использовал компоновщик ld. Сначала я скомпоновал первый объектный файл hello.o и проверил, был ли создан исполняемый файл hello.

*Рис.10 Вывод команды ls, где виден исполняемый файл hello с правами на выполнение.*

Исполняемый файл hello был успешно создан.

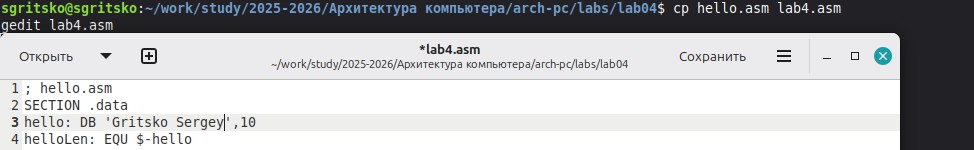
Затем я выполнил компоновку второго объектного файла obj.o в исполняемый файл с именем main и запустил первую созданную программу hello на выполнение.

*Рис.11 Вывод <<Hello world!>>.*

Программа успешно скомпилировалась и запустилась, выведя на экран приветственное сообщение. Это подтверждает, что все шаги от написания кода до компоновки были выполнены верно.

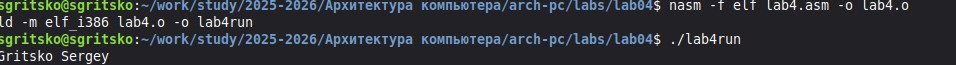
# Описание результатов выполнения заданий для самостоятельной работы

Я скопировал файл hello.asm в новый файл lab4.asm и открыл его для редактирования и файле lab4.asm я изменил строку "Hello world!" на свои фамилию и имя: "Грицко Сергей".

.

*Рис.12 Изменение текста для вывода.*

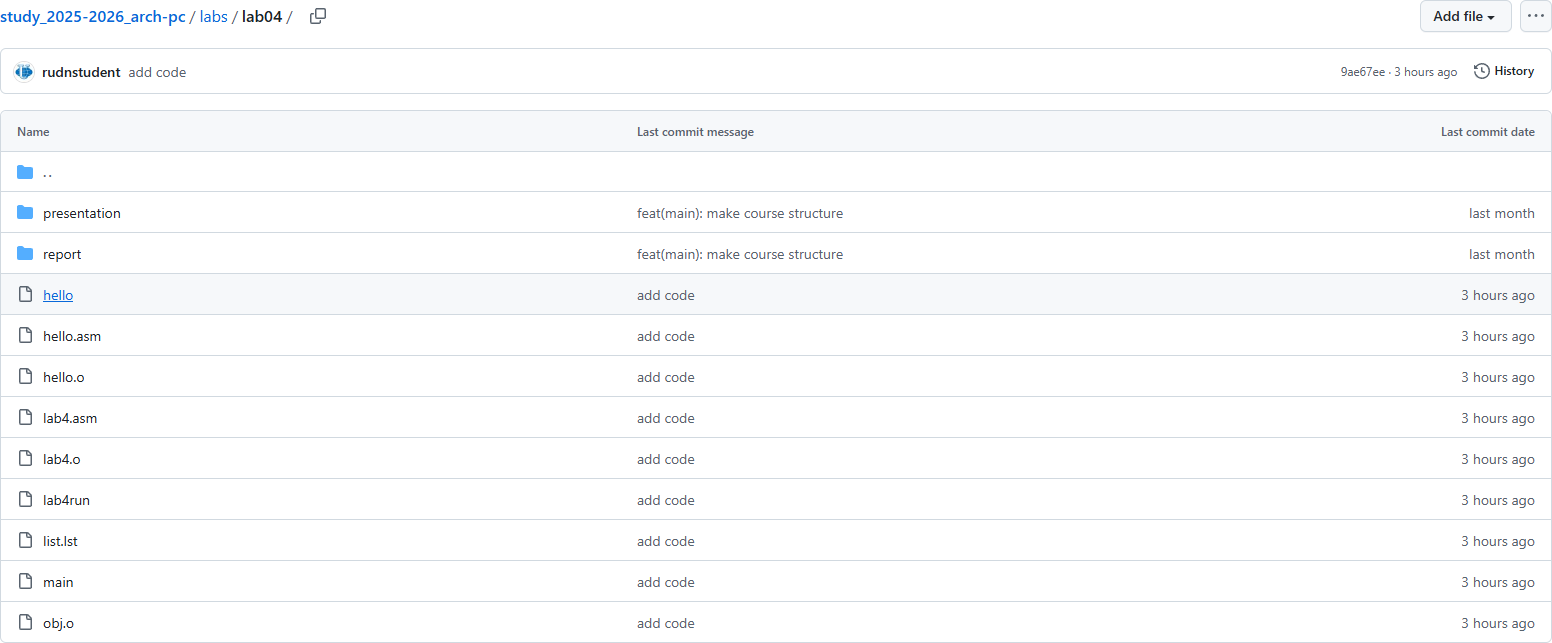
Я выполнил трансляцию и компоновку нового файла, создав исполняемый файл

lab4run и затем я запустил полученную программу.

*Рис.13 Вывод программы lab4run, где на экране отображается мое имя.*

*.*

Теперь я загружаю файлы на GitHub.



*Рис.14 Исходные коды программ были успешно загружены в мой удаленный репозиторий на GitHub.*

# Выводы

В ходе выполнения данной лабораторной работы я освоил полную процедуру создания, трансляции, компоновки и запуска программ на языке ассемблера NASM. Я научился использовать утилиты командной строки: транслятор nasm для преобразования исходного кода в объектные файлы и компоновщик ld для сборки исполняемых файлов. Также я получил практический навык внесения изменений в ассемблерный код и управления файлами с помощью git для их версионирования и загрузки на GitHub. Цель работы была полностью достигнута.

# Список литературы

1. GDB: The GNU Project Debugger. — URL: https://[www.gnu.org/software/gdb/.](http://www.gnu.org/software/gdb/)
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: https://midnight-commander. org/.
4. NASM Assembly Language Tutorials. — 2021. — URL: [https://asmtutor.com/.](https://asmtutor.com/)
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O’Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: [http://www.amazon.com/Learningbash-Shell-](http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658) [Programming-Nutshell/dp/0596009658](http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658).
6. Robbins A. Bash Pocket Reference. — O’Reilly Media, 2016. — 156 с. — ISBN 978- 1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
10. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон- Пресс, 2017.
11. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
12. Расширенный ассемблер: NASM. — 2021. — URL: https://[www.opennet.ru/docs/RUS/nasm/.](http://www.opennet.ru/docs/RUS/nasm/)
13. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
14. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: <http://www.stolyarov.info/books/asm_unix>.
15. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science). 16. Таненбаум Э., Бос Х. Современные операционные системы.

— 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science)