

Отчёт по лабораторной работе 4

язык ассемблера NASM

Ошкодер С.А.

Содержание

Цель работы
Задание
Теоретическое введение
Выполнение лабораторной работы
Выводы
Список литературы

Цель работы

Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM

Задание

1. Программа Hello world!
 - 1.1 Создать каталог для работы с программами на языке ассемблера NASM
 - 1.2 Перейти в созданный каталог
 - 1.3 Создать текстовый файл с именем hello.asm
 - 1.4 Открыть этот файл
 - 1.5 Ввести в него указанный текст
2. Транслятор NASM
 - 2.1 Выполнить компиляцию в объектный код
3. Расширенный синтаксис
 - 3.1 Выполнить компиляцию исходного файла
4. компоновщик LD
 - 4.1 Передать объектный файл на обработку компоновщику
5. Запустить исполняемый файл
6. Задания для самостоятельной работы
 - 6.1 Создать копию файла hello.asm с именем lab4.asm
 - 6.2 Изменить скопированный файл, чтобы выводилась строка с именем и фамилией
 - 6.3 Оттранслировать полученный текст программы lab4.asm в объектный

файл

6.4 Скопировать файлы hello.asm и lab4.asm в локальный репозиторий

Теоретическое введение

В процессе создания ассемблерной программы можно выделить четыре шага:

- Набор текста программы в текстовом редакторе и сохранение её в отдельном файле. Каждый файл имеет свой тип (или расширение), который определяет назначение файла. Файлы с исходным текстом программ на языке ассемблера имеют тип `asm`.
- Трансляция — преобразование с помощью транслятора, например `nasm`, текста программы в машинный код, называемый объектным. На данном этапе также может быть получен листинг программы, содержащий кроме текста программы различную дополнительную информацию, созданную транслятором. Тип объектного файла — `o`, файла листинга — `lst`.
- Компоновка или линковка — этап обработки объектного кода компоновщиком (`ld`), который принимает на вход объектные файлы и собирает по ним исполняемый файл. Исполняемый файл обычно не имеет расширения. Кроме того, можно получить файл карты загрузки программы в ОЗУ, имеющий расширение `map`.
- Запуск программы. Конечной целью является работоспособный исполняемый файл. Ошибки на предыдущих этапах могут привести к некорректной работе программы, поэтому может присутствовать этап отладки программы при помощи специальной программы — отладчика. При нахождении ошибки необходимо провести коррекцию программы, начиная с первого шага. Из-за специфики программирования, а также по традиции для создания программ на языке ассемблера обычно пользуются утилитами командной строки (хотя поддержка ассемблера есть в некоторых универсальных интегрированных средах)

Выполнение лабораторной работы

1. Программа Hello world (см рис1)
 - 1.1 создать каталог для работы с программами на языке ассемблера NASM (см рис1)
 - 1.2 перейти в созданный каталог (см рис1)
 - 1.3 создать текстовый файл с именем `hello.asm` (см рис1)
 - 1.4 открыть этот файл (см рис1)
 - 1.5 ввести в него указанный текст (см рис1)
2. Транслятор NASM (см рис1)
 - 2.1 выполнить компиляцию исходного файла (см рис1)
3. Расширенный синтаксис (см рис1)
 - 3.1 выполнить компиляцию исходного исходного файла (см рис1)
- 4 Компоновщик LD (см рис1)
 - 4.1 передать объектный файл на обработку компоновщику (см рис1)
- 5 Запустить исполняемый файл (см рис1-2)

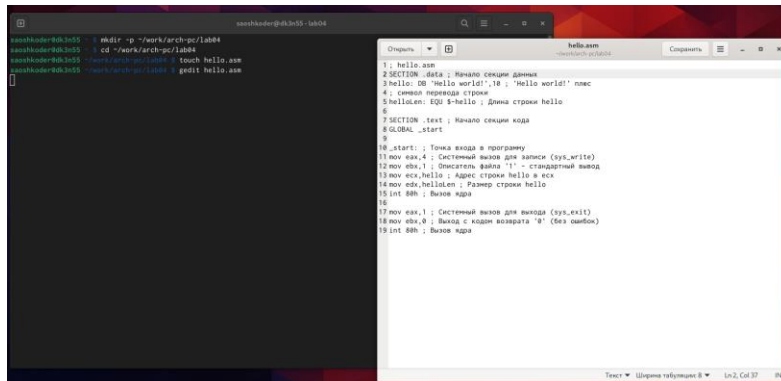
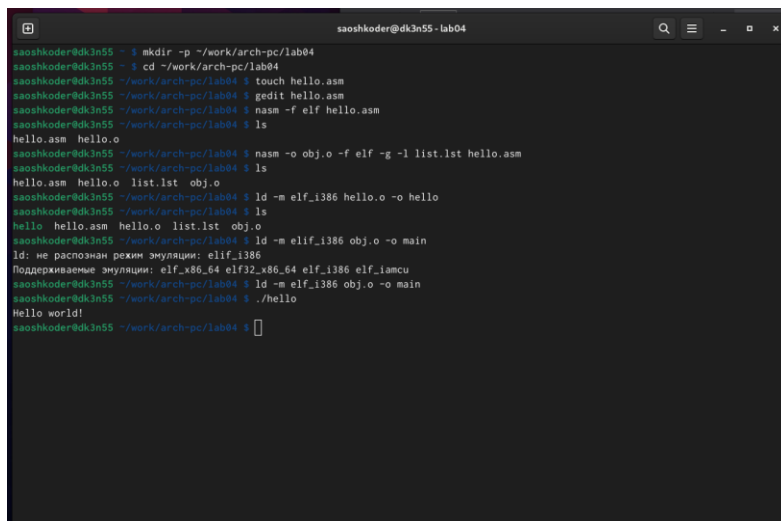
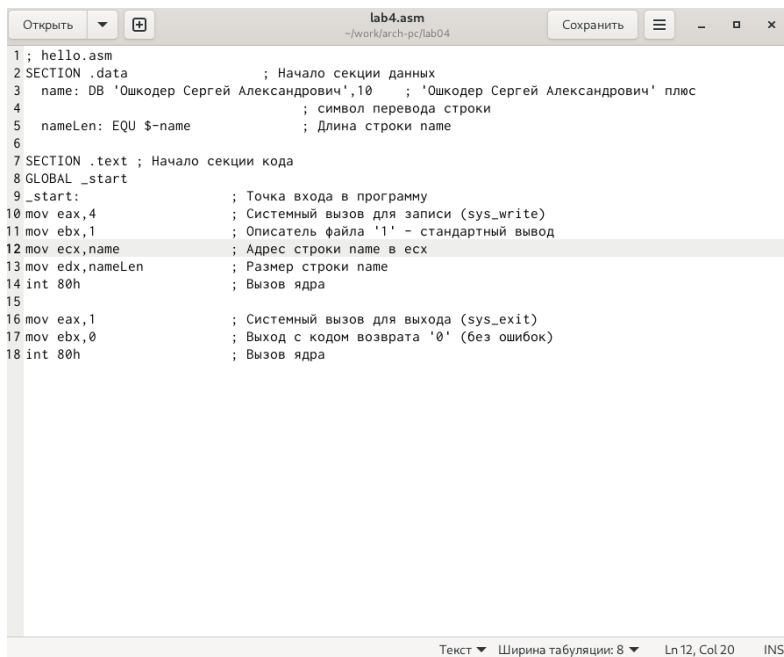


рис1



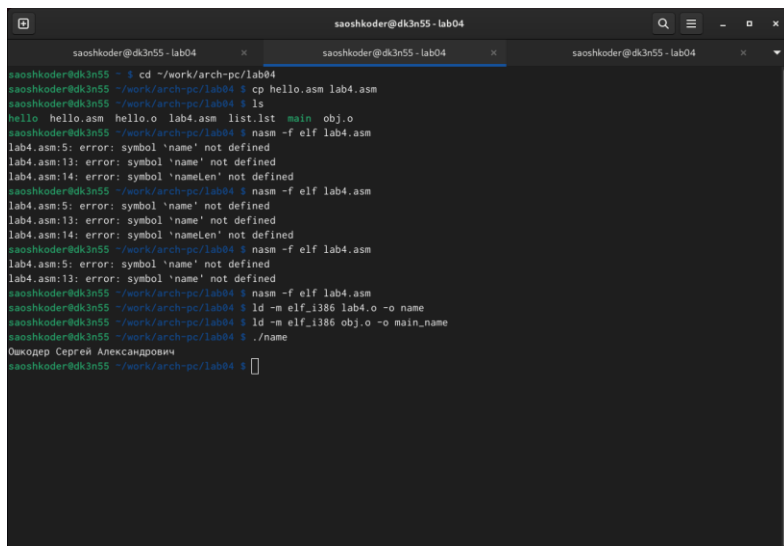
Выполнение лабораторной работы (рис1-2)

6. Задания для самостоятельной работы (см рис 3-4)
 - 6.1 создать копию файла hello.asm с именем lab4.asm (см рис 3-4)
 - 6.2 изменить скопированный файл, чтобы выводилась строка с именем и фамилией (см рис 3-4)
 - 6.3 оттранслировать полученный текст программы lab4.asm в объектный файл (см рис 3-4)
 - 6.4 скопировать файлы hello.asm и lab4.asm в локальный репозиторий (см рис 3-4)



```
1 ; hello.asm
2 SECTION .data                ; Начало секции данных
3   name: DB 'Ошкодер Сергей Александрович',10    ; 'Ошкодер Сергей Александрович' плюс
4                                           ; символ перевода строки
5   nameLen: EQU $-name          ; Длина строки name
6
7 SECTION .text ; Начало секции кода
8 GLOBAL _start
9 _start:                        ; Точка входа в программу
10 mov eax,4                     ; Системный вызов для записи (sys_write)
11 mov ebx,1                     ; Описатель файла '1' - стандартный вывод
12 mov ecx,name                  ; Адрес строки name в ecx
13 mov edx,nameLen               ; Размер строки name
14 int 80h                      ; Вызов ядра
15
16 mov eax,1                     ; Системный вызов для выхода (sys_exit)
17 mov ebx,0                     ; Выход с кодом возврата '0' (без ошибок)
18 int 80h                      ; Вызов ядра
```

рис3



```
saoshkoder@dk3n55 - lab04
saoshkoder@dk3n55: ~$ cd ~/work/arch-pc/lab04
saoshkoder@dk3n55: ~/work/arch-pc/lab04$ cp hello.asm lab4.asm
saoshkoder@dk3n55: ~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o
saoshkoder@dk3n55: ~/work/arch-pc/lab04$ nasm -f elf lab4.asm
lab4.asm:5: error: symbol 'name' not defined
lab4.asm:13: error: symbol 'name' not defined
lab4.asm:14: error: symbol 'nameLen' not defined
saoshkoder@dk3n55: ~/work/arch-pc/lab04$ nasm -f elf lab4.asm
lab4.asm:5: error: symbol 'name' not defined
lab4.asm:13: error: symbol 'name' not defined
lab4.asm:14: error: symbol 'nameLen' not defined
saoshkoder@dk3n55: ~/work/arch-pc/lab04$ nasm -f elf lab4.asm
lab4.asm:5: error: symbol 'name' not defined
lab4.asm:13: error: symbol 'name' not defined
lab4.asm:14: error: symbol 'nameLen' not defined
saoshkoder@dk3n55: ~/work/arch-pc/lab04$ nasm -f elf lab4.asm
saoshkoder@dk3n55: ~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o name
saoshkoder@dk3n55: ~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main_name
saoshkoder@dk3n55: ~/work/arch-pc/lab04$ ./name
Ошкодер Сергей Александрович
saoshkoder@dk3n55: ~/work/arch-pc/lab04$
```

рис4

Выводы

Я ознакомился с созданием и процессом обработки программ на языке ассемблера NASM

Список литературы

. GDB: The GNU Project Debugger. — URL: <https://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/>. 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/Learning-bash-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/>. 13. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВ- Петербург, 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). ::: {#refs} :::