

Лабораторная работа номер 5.

Основы работы с Midnight Commander (mc). Структура программы на языке ассемблера NASM. Системные вызовы в ОС GNU Linux

Сорокин Кирилл

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	7
4.1	Самостоятельная работа	14
5	Выводы	19
	Список литературы	20

Список иллюстраций

4.1	Открытие Midnight Commander	7
4.2	Отображение папок в Midnight Commander	7
4.3	Создание папки	8
4.4	Вид созданной папки	8
4.5	Создание файла lab5-1.asm	8
4.6	Вид открытого файла	9
4.7	Файл с текстом программы	9
4.8	Сохранение файла	9
4.9	Просмотр содержимого файла	10
4.10	Содержимое папки после создания всех файлов	10
4.11	Работа первой программы	10
4.12	Вид двойного терминала	11
4.13	Копирование файла in_out.asm	11
4.14	Вид папки с in_out.asm	12
4.15	Создание файла lab5-2.asm	12
4.16	Редактирование lab5-2.asm	13
4.17	Вид папки с новыми файлами	13
4.18	Работа файла lab5-2.asm	13
4.19	Редактирование файла lab5-2.asm	14
4.20	Работа второй версии lab5-2	14
4.21	Начало самостоятельной работы	14
4.22	Изменение первого файла	15
4.23	Файлы lab5-1s	15
4.24	Работа lab5-1s	16
4.25	Копирование второго файла	16
4.26	Изменение второго файла	17
4.27	Файлы lab5-2s	17
4.28	Работа lab5-2s	18

1 Цель работы

Начиться работать с Midnight Commander, а также освоить инструкции `mov` и `int` в языке ассемблера.

2 Задание

Используя МС, написать программы, основываясь на приведённых материалах, а затем выполнить самостоятельную работу.

3 Теоретическое введение

Midnight Commander (или просто mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управлению файловой системой, т.е. mc является файловым менеджером. Midnight Commander позволяет сделать работу с файлами более удобной и наглядной. Инструкция языка ассемблера mov предназначена для дублирования данных источника в приёмнике. Инструкция языка ассемблера int предназначена для вызова прерывания с указанным номером.

4 Выполнение лабораторной работы

Откроем Midnight Commander командой mc (рис. 4.1).

```
kvsorokin@dk6n51 ~ $ mc
```

Рис. 4.1: Открытие Midnight Commander

Перейдём в необходимую нам дерикторию(рис. 4.2).

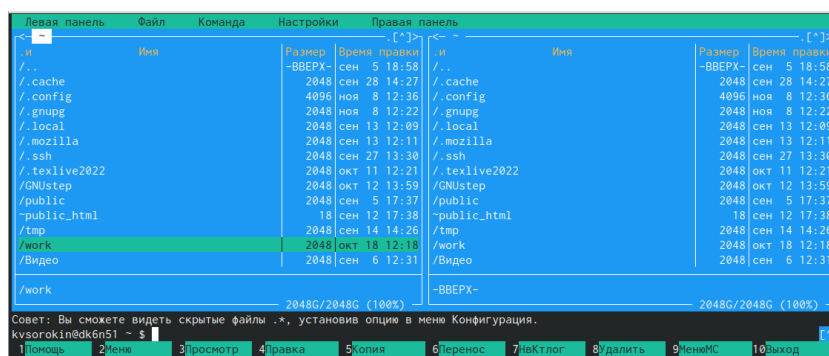


Рис. 4.2: Отображение папок в Midnight Commander

С помощью клавиши F7 создадим папку(рис. 4.3).

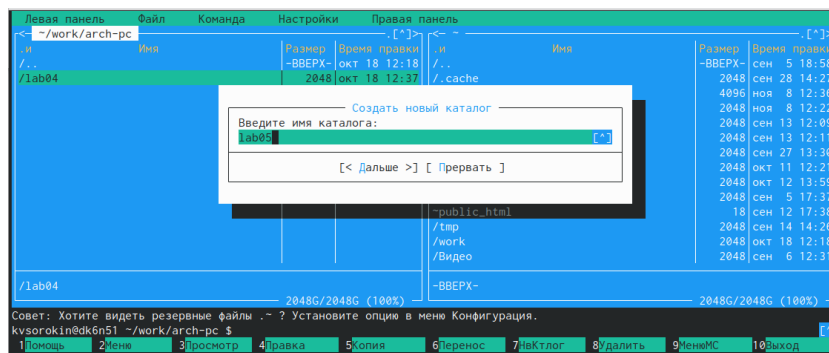


Рис. 4.3: Создание папки

Удостоверимся, что папка создана (рис. 4.4).

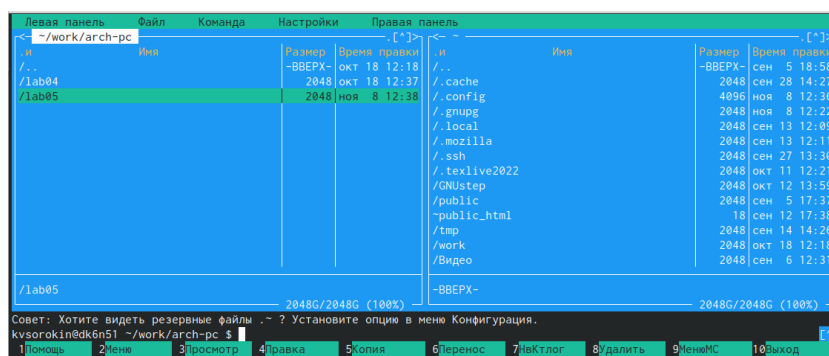


Рис. 4.4: Вид созданной папки

Командой touch создадим файл lab5-1.asm (рис. 4.5).



Рис. 4.5: Создание файла lab5-1.asm

Откроем файл для редактирование клавишей F4 (рис. 4.6).



Рис. 4.6: Вид открытого файла

Введём текст программы в файл (рис. 4.7).

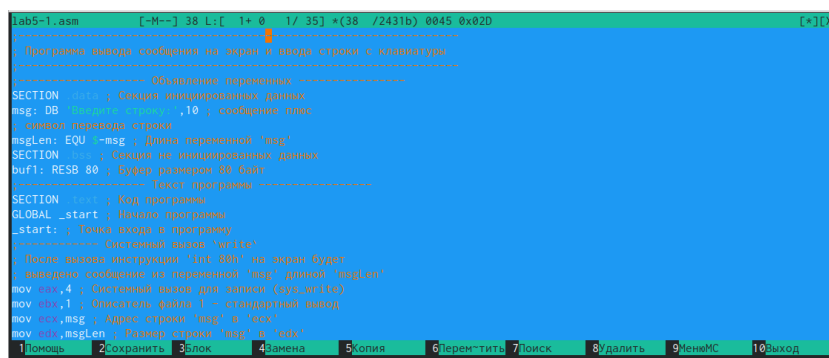


Рис. 4.7: Файл с текстом программы

Так как у нас редактор mscedit нажмем клавишу F2 для сохранения и клавишей F10 для выхода из редактирования файла (рис. 4.8).

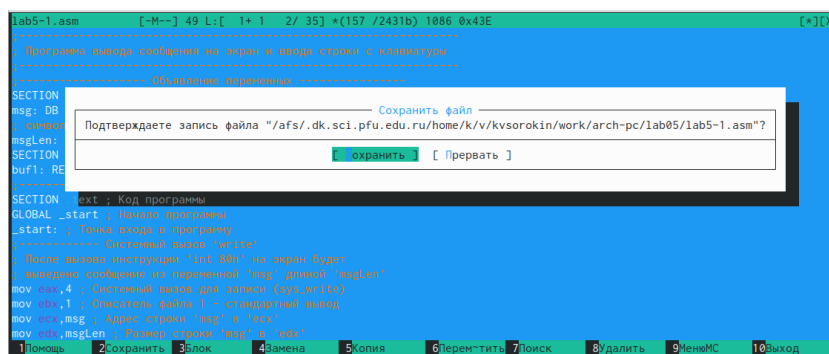


Рис. 4.8: Сохранение файла

Нажмём клавишу F3 для просмотра содержимого файла lab5-1.asm (рис. 4.9).

```

/afs/dk.sci.pfu.edu.ru/home/k/v/kvsorokin/work/arch-pc/lab05/lab5-1.asm 1418/2431 58
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов 'write' -----
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,msg ; Адрес строки 'msg' в 'ecx'
mov edx,msgLen ; Размер строки 'msg' в 'edx'
1Помощь 2Назад 3Выход 4Hex 5Перейти 6 7Поиск 8Исходный 9Формат 10Выход

```

Рис. 4.9: Просмотр содержимого файла

Оттранслируем текст lab5-1.asm в объектный файл, затем выполним его компоновку и запустим файл командой ./lab5-1 (рис. 4.10).

Левая панель	Файл	Команда	Настройки	Правая панель
	~/work/arch-pc/lab05			
	./	Имя	Размер	Имя
	./	-BBEPX-	ноя 8 12:39	-BBEPX-
	lab5-1.asm	2431	ноя 8 12:43	2048 сен 5 18:58
	lab5-1.o	752	ноя 8 12:44	2048 сен 28 14:27
				4096 ноя 8 12:36
				2048 ноя 8 12:22
				2048 сен 13 12:09
				2048 сен 13 12:11
				2048 сен 27 13:30
				2048 окт 11 12:21
				2048 окт 12 13:59
				2048 сен 5 17:37
				18 сен 12 17:38
				2048 сен 14 14:26
				2048 окт 18 12:18
				2048 сен 6 12:31
	lab5-1.asm			-BBEPX-
				2048G/2048G (100%)
				Совет: Установив переменную CDPATH, вы сэкономите усилия при наборе команды cd.
				kvsorokin@dk6n51 ~/work/arch-pc/lab05 \$./lab5-1
				Введите строку:
				Сорокин Кирилл Васильевич
				1Помощь 2Меню 3Просмотр 4Правка 5Копия 6Перенос 7Журнал 8Удалить 9МенюМС 10Выход

Рис. 4.10: Содержимое папки после создания всех файлов

Убедимся в правильности работы программы (рис. 4.11).

```

kvsorokin@dk6n51 ~/work/arch-pc/lab05 $ ./lab5-1
Введите строку:
Сорокин Кирилл Васильевич

```

Рис. 4.11: Работа первой программы

Для удобства откроем дополнительную панель терминала и найдём заранее скаченный файл in_out.asm (рис. 4.12).

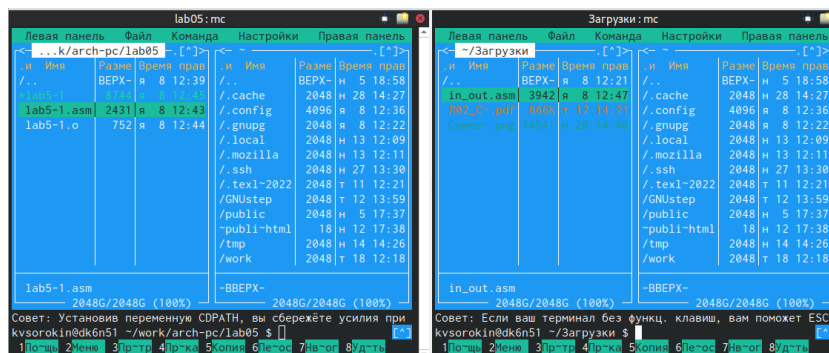


Рис. 4.12: Вид двойного терминала

Скопируем файл в наш рабочий каталог с помощью клавиши F5 (рис. 4.13).

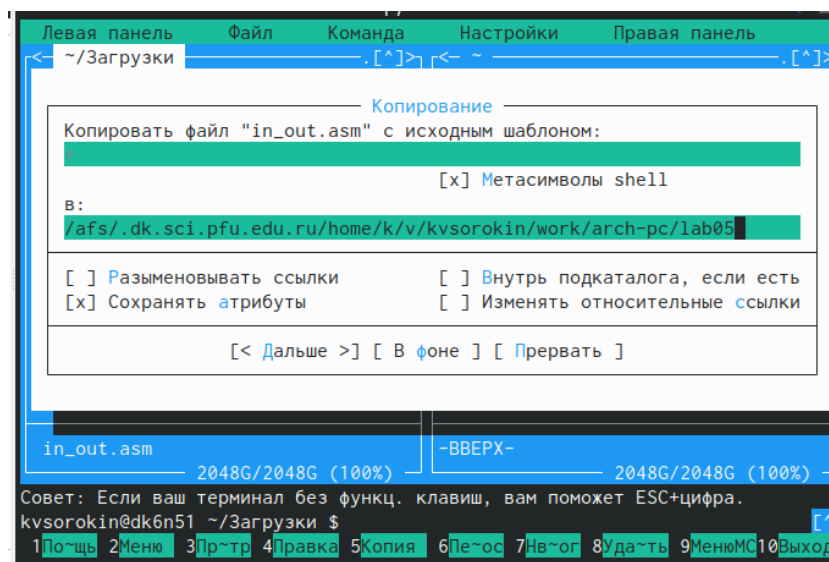


Рис. 4.13: Копирование файла in_out.asm

Убедимся, что он корректно перенёсся (рис. 4.14).

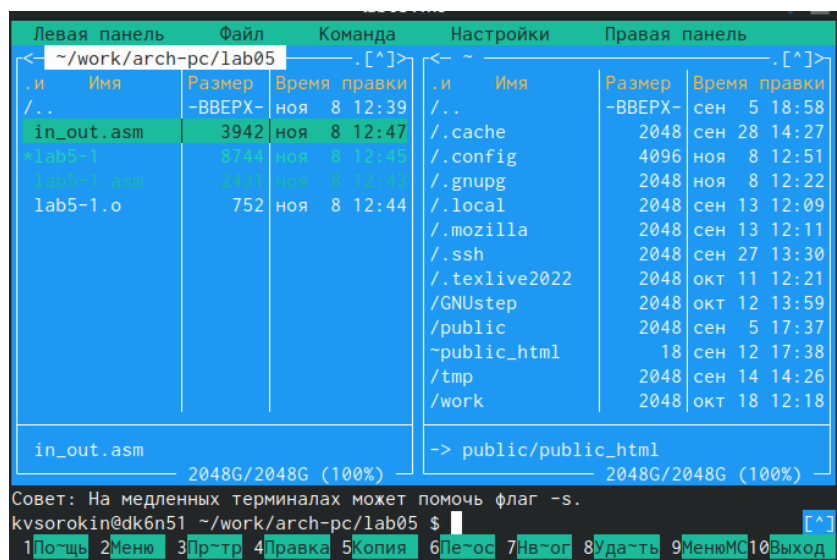


Рис. 4.14: Вид папки с in_out.asm

Создадим копию файла lab5-1.asm с именем lab5-2.asm (рис. 4.15).

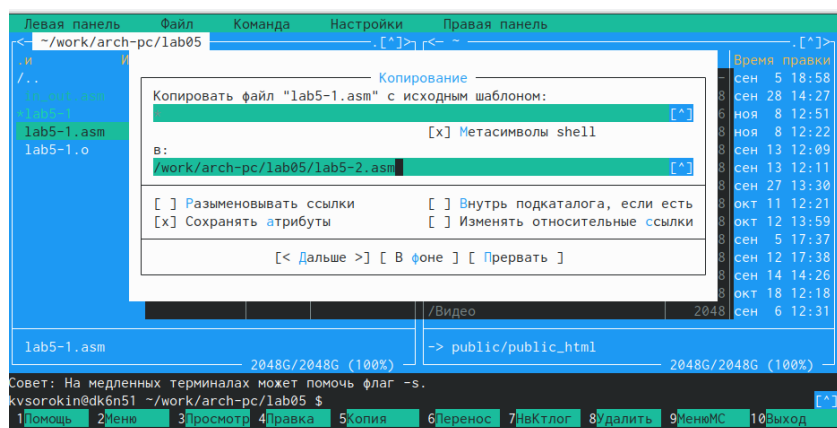


Рис. 4.15: Создание файла lab5-2.asm

Отредактируем его, так чтобы он использовал файл in_out.asm (рис. 4.16).

```

lab5-2.asm      [-M--] 41 L: [ 1+16 17/ 17] *(1224/1224b) <EOF>      [*][X]

; Программа вывода сообщения на экран и ввода строки с клавиатуры
;~~~~~
%include "lab5-1.asm" ; подключение внешнего файла
SECTION .data ; Секция инициализированных данных
msg: DB "Введите строку: ",0h ; сообщение
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
mov eax, msg ; запись адреса выводимого сообщения в 'EAX'
call sprintf ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'EAX'
mov edx, 80 ; запись длины вводимого сообщения в 'EBX'
call read ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения

```

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перенести 7Поиск 8Удалить 9МенюМС 10Выход

Рис. 4.16: Редактирование lab5-2.asm

Создадим его объектный файл и файл программы (рис. 4.17).

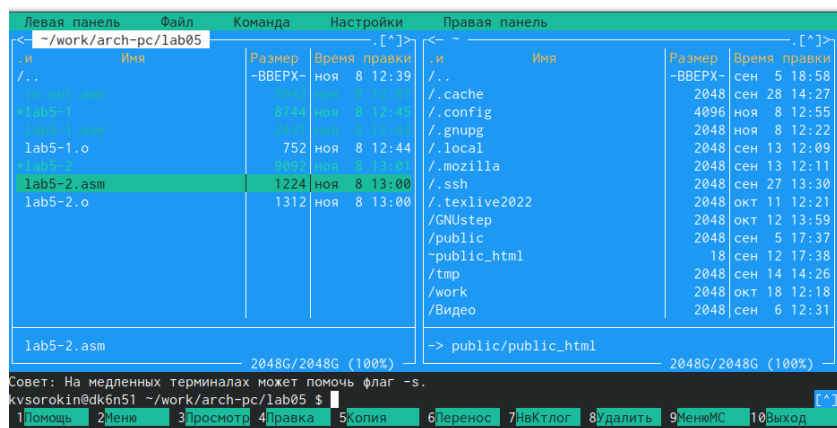


Рис. 4.17: Вид папки с новыми файлами

Проверим его работу (рис. 4.18).

```

kvsorokin@dk6n51 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку:
Сорокин Кирилл Васильевич

```

Рис. 4.18: Работа файла lab5-2.asm

Заменяем подпрограмму sprintf на sprintf (рис. 4.19).

```

mov eax, msg ; запись адреса выводимого сообщения в 'EAX'
call sprintf ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'EAX'

```

Рис. 4.19: Редактирование файла lab5-2.asm

Выполним новую программу и заметим, что теперь ввод текста идёт не с новой строки, а той же на какой был напечатан изначальный текст (рис. 4.20).

```

kvsorokin@dk6n51 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку: Сорокин Кирилл Васильевич

```

Рис. 4.20: Работа второй версии lab5-2

4.1 Самостоятельная работа

Создадим копию файла lab5-1.asm с именем lab5-1s.asm для самостоятельной работы (рис. 4.21).

lab5-1s.asm	2431	ноя 8 12:43
-------------	------	-------------

Рис. 4.21: Начало самостоятельной работы

Для того чтобы новая программа выполняла дополнительные действия добавим вывод, ранее введённых данных, и запросим новый ввод, чтобы была возможность отследить результат вывода (рис. 4.22).

```

mov  eax,4 ; Системн
mov  ebx,1 ; Описате
mov  ecx,buf1 ; адре
mov  edx,80 ; Размер
int  80h ; Вызов ядр

mov  eax, 3 ; Систем
mov  ebx, 0 ; Дескри
mov  ecx, buf2 ; Адр
mov  edx, 80 ; Длина
int  80h ; Вызов ядр

```

Рис. 4.22: Изменение первого файла

Создадим все файлы для работы программы (рис. 4.23).

```

lab5-1s.asm
lab5-1s.o
*lab5-1sm

```

Рис. 4.23: Файлы lab5-1s

Запустим программу и убедимся, что она корректно выполняет заданное задание (рис. 4.24).

```
kvsorokin@dk6n51 ~/work/arch-pc/lab05 $ ./lab5-1s
Введите строку:
Сорокин Кирилл Васильевич
Сорокин Кирилл Васильевич
```

Рис. 4.24: Работа lab5-1s

Создадим копию файла lab5-2.asm с именем lab5-2s.asm для самостоятельной работы (рис. 4.25).

```
Копирование
Копировать файл "lab5-2.asm" с исходным шаблоном:
*
[x] Метасимволы sh
В:
e/k/v/kvsorokin/work/arch-pc/lab05/lab5-2s.asm
```

Рис. 4.25: Копирование второго файла

От редактируем новую программу. Чтобы она могла выполнить все необходимые действия добавим вывод, ранее введенных данных, и запросим новый ввод, чтобы была возможность отследить результат вывода (рис. 4.26).


```
mov eax, buf1 ;  
call sprintf ;  
  
mov ecx, buf2 ;  
mov edx, 80 ;  
call sread ;
```

Рис. 4.26: Изменение второго файла

Создадим все файлы для работы программы (рис. 4.27).

```
*lab5-2s  
lab5-2s.asm  
lab5-2s.o
```

Рис. 4.27: Файлы lab5-2s

Запустим программу и убедимся, что она корректно выполняет заданное задание, но из-за того, что в изначальном файле lab5-2.asm мы оставили `sprint`, а не `sprintf`, то вид ввода немного отличается от lab5-1s (рис. 4.28).

```
kvsorokin@dk6n51 ~/work/arch-pc/lab05 $ ./lab5-2s  
Введите строку: Сорокин Кирилл Васильевич  
Сорокин Кирилл Васильевич
```

Рис. 4.28: Работа lab5-2s

5 Выводы

Мы научились использовать Midnight Commander, а также написали несколько программ на языке ассемблера с использованием новых инструкций.

Список литературы

1. 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://midnightcommander.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,

17. — 1120 с. — (Классика Computer Science)