Отчёт по лабораторной работе №6

Дисциплина: Архитектура компьютера

Мишина Анастасия Алексеевна

Содержание

# 1 Цель работы

Приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера mov и int.

# 2 Выполнение лабораторной работы

Для начала открываем Midnight Commander с помощью команды mc, переходим в каталог ~/work/arch-pc (рис. 1). Создаем каталог для шестой работы, используя команду F7 (рис. 2).

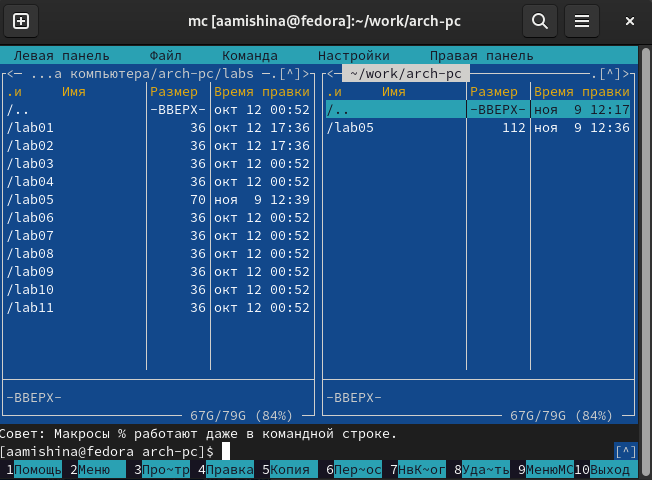


Рис. 1: Переход в каталог ~/work/arch-pc

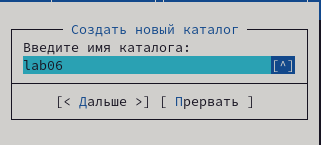


Рис. 2: Создание каталога lab06

В новом каталоге создаем файл lab6-1.asm (рис. 3) и открываем его с помощью клавиши F4. Теперь введем текст 6.1 из мануала в файл (рис. 4).

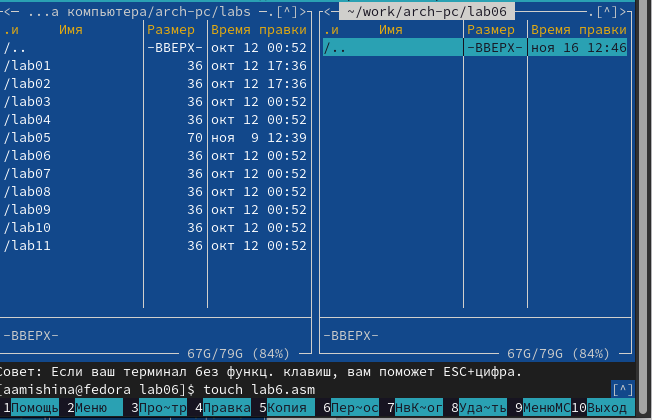


Рис. 3: Создание файла lab6-1.asm

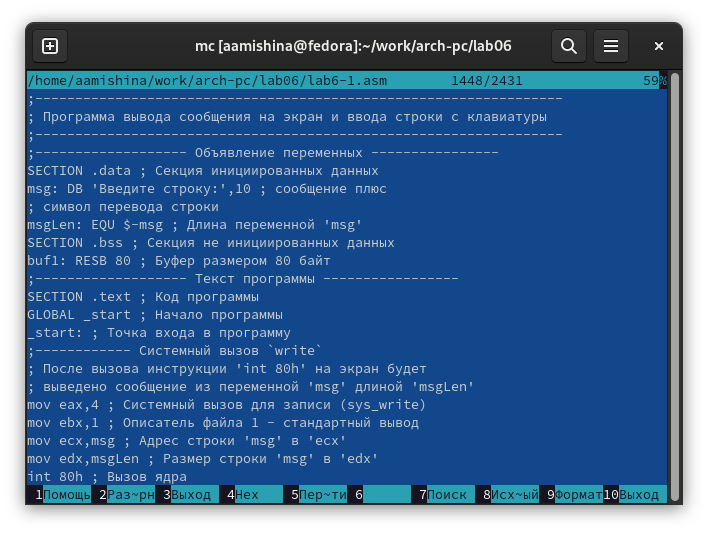


Рис. 4: Программа в lab6-1.asm

Программа lab6-1.asm:

;------------------------------------------------------------------  
; Программа вывода сообщения на экран и ввода строки с клавиатуры  
;------------------------------------------------------------------  
;------------------- Объявление переменных ----------------  
SECTION .data ; Секция инициированных данных  
msg: DB 'Введите строку:',10 ; сообщение плюс  
; символ перевода строки  
msgLen: EQU $-msg ; Длина переменной 'msg'  
SECTION .bss ; Секция не инициированных данных  
buf1: RESB 80 ; Буфер размером 80 байт  
;------------------- Текст программы -----------------  
SECTION .text ; Код программы  
GLOBAL \_start ; Начало программы  
\_start: ; Точка входа в программу  
;------------ Cистемный 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'  
int 80h ; Вызов ядра  
;------------ системный вызов `read` ----------------------  
; После вызова инструкции 'int 80h' программа будет ожидать ввода  
; строки, которая будет записана в переменную 'buf1' размером 80 байт  
mov eax, 3 ; Системный вызов для чтения (sys\_read)  
mov ebx, 0 ; Дескриптор файла 0 - стандартный ввод  
mov ecx, buf1 ; Адрес буфера под вводимую строку  
mov edx, 80 ; Длина вводимой строки  
int 80h ; Вызов ядра  
;------------ Системный вызов `exit` ----------------------  
; После вызова инструкции 'int 80h' программа завершит работу  
mov eax,1 ; Системный вызов для выхода (sys\_exit)  
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)  
int 80h ; Вызов ядра

Пользуясь клавишами F2, F10, F3 сохраняем программу, выходим из редактора и открываем файл для просмотра, убеждаемся, что все на месте.

Далее преобразовываем текст программы в объектный файл, выполняем компоновку и запускаем исполняемый файл. Вводим ФИО, программа отрабатывает (рис. 5).

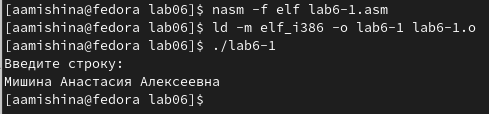


Рис. 5: Работа с файлом lab6-1.asm

Скачиваем файл in\_out.asm из ТУИСа, копируем его из каталога загрузок в каталог lab06 (рис. 6).

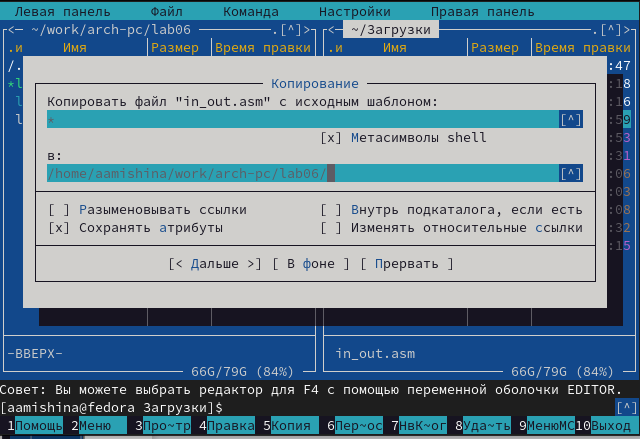


Рис. 6: Копирование файла in\_out.asm в папку lab06

Создаем копию файла lab6-1.asm с помощью клавиши F5, называем его lab6-2.asm (рис. 7). Вставляем код 6.2 в новый файл (рис. 8).

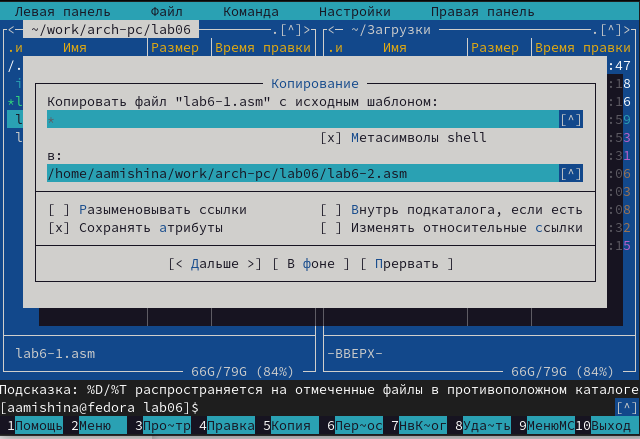


Рис. 7: Создание копии файла lab6-1.asm

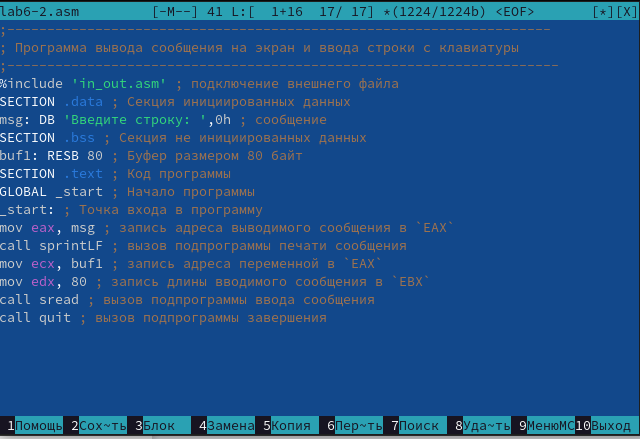


Рис. 8: Программа с кодом 6.2

Программа lab6-2.asm:

;--------------------------------------------------------------------  
; Программа вывода сообщения на экран и ввода строки с клавиатуры  
;---------------------------------------------------------------------  
%include 'in\_out.asm' ; подключение внешнего файла  
SECTION .data ; Секция инициированных данных  
msg: DB 'Введите строку: ',0h ; сообщение  
SECTION .bss ; Секция не инициированных 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 sprintLF ; вызов подпрограммы печати сообщения  
mov ecx, buf1 ; запись адреса переменной в `EAX`  
mov edx, 80 ; запись длины вводимого сообщения в `EBX`  
call sread ; вызов подпрограммы ввода сообщения  
call quit ; вызов подпрограммы завершения

Создаем исполняемый файл, убеждаемся в его работоспособности (рис. 9).

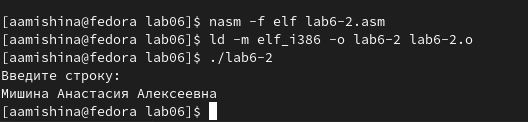


Рис. 9: Работа с файлом lab6-2.asm

Заменяем подпрограмму spinrLF на sprint (рис. 10). Пересоздаем исполняемый файл, заметим следующую разницу: теперь нет символа перевода строки при выводе надписи “Введите строку”, следовательно ввод текста будет происходить без перехода на новую строку (рис. 11).

Рис. 10: Замена подпрограммы spinrLF на sprint

Рис. 10: Замена подпрограммы spinrLF на sprint

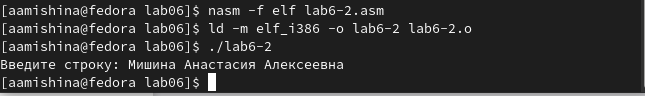


Рис. 11: Работа с измененным файлом lab6-2.asm

# 3 Выполнение заданий самостоятельной работы

Для начала создадим копию файла lab6-1.asm (рис. 12). Как мы знаем, эта программа уже умеет выводить приглашение типа “Введите строку”, также она считывает нашу строку и записывает ее в buf1. Ориентируясь на вывод сообщения, создаем кусочек кода, который выводит строку, записанную в buf1 (рис. 13). Убеждаемся в работоспособности программы (рис. 14).

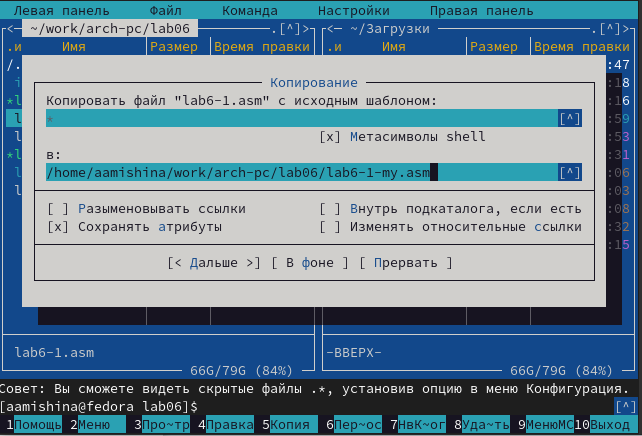


Рис. 12: Создание копии файла lab6-1.asm

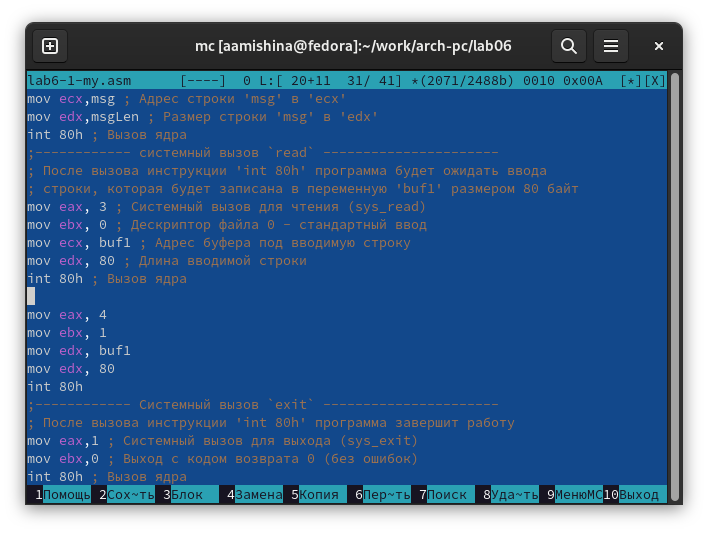


Рис. 13: Добавление вывода

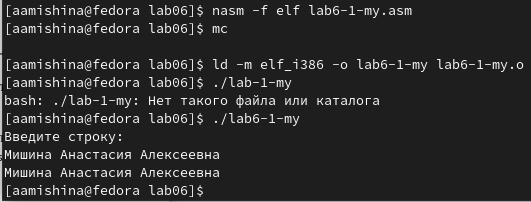


Рис. 14: Работа с файлом lab6-1-my.asm

Программа lab6-1-my.asm:

;------------------------------------------------------------------  
; Программа вывода сообщения на экран и ввода строки с клавиатуры  
;------------------------------------------------------------------  
;------------------- Объявление переменных ----------------  
SECTION .data ; Секция инициированных данных  
msg: DB 'Введите строку:',10 ; сообщение плюс  
; символ перевода строки  
msgLen: EQU $-msg ; Длина переменной 'msg'  
SECTION .bss ; Секция не инициированных данных  
buf1: RESB 80 ; Буфер размером 80 байт  
;------------------- Текст программы -----------------  
SECTION .text ; Код программы  
GLOBAL \_start ; Начало программы  
\_start: ; Точка входа в программу  
;------------ Cистемный 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'  
int 80h ; Вызов ядра  
;------------ системный вызов `read` ----------------------  
; После вызова инструкции 'int 80h' программа будет ожидать ввода  
; строки, которая будет записана в переменную 'buf1' размером 80 байт  
mov eax, 3 ; Системный вызов для чтения (sys\_read)  
mov ebx, 0 ; Дескриптор файла 0 - стандартный ввод  
mov ecx, buf1 ; Адрес буфера под вводимую строку  
mov edx, 80 ; Длина вводимой строки  
int 80h ; Вызов ядра  
  
mov eax, 4  
mov ebx, 1  
mov edx, buf1  
mov edx, 80  
int 80h  
;------------ Системный вызов `exit` ----------------------  
; После вызова инструкции 'int 80h' программа завершит работу  
mov eax,1 ; Системный вызов для выхода (sys\_exit)  
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)  
int 80h ; Вызов ядра

Теперь делаем копию файла lab6-2.asm (рис. 15), там также добавляем вывод нашей строки (рис. 16). Создаем исполняемый файл, убеждаемся, что все работает (рис. 17).

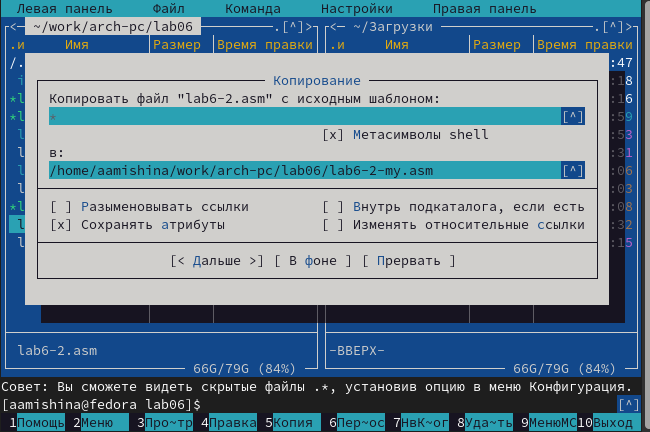


Рис. 15: Создание копии файла lab6-2.asm

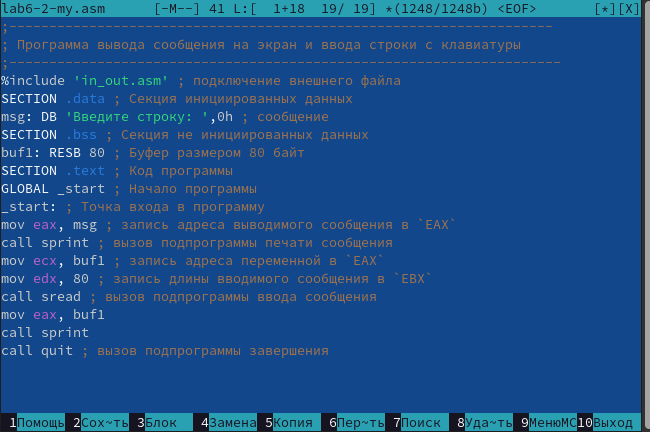


Рис. 16: Добавление вывода

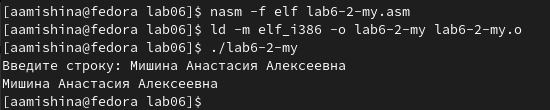


Рис. 17: Работа с файлом lab6-2-my.asm

Программа lab6-2-my.asm:

;--------------------------------------------------------------------  
; Программа вывода сообщения на экран и ввода строки с клавиатуры  
;---------------------------------------------------------------------  
%include 'in\_out.asm' ; подключение внешнего файла  
SECTION .data ; Секция инициированных данных  
msg: DB 'Введите строку: ',0h ; сообщение  
SECTION .bss ; Секция не инициированных 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 sprint ; вызов подпрограммы печати сообщения  
mov ecx, buf1 ; запись адреса переменной в `EAX`  
mov edx, 80 ; запись длины вводимого сообщения в `EBX`  
call sread ; вызов подпрограммы ввода сообщения  
mov eax, buf1  
call sprint  
call quit ; вызов подпрограммы завершения

# 4 Выводы

В ходе выполнения данной лабораторной работы я приобрела практические навыки работы в Midnight Commander. Также я освоила инструкции языка ассемблера mov и int. Вся моя работа была записана и показана в данной лабораторной.