**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

**ОТЧЕТ**

**по лабораторной работе №1**

**по дисциплине «Операционные системы»**

**Тема: Исследование структур загрузочных модулей**

Студентка гр. 6383 Терещенко В.Н.  
Преподаватель Губкин А.Ф.

Санкт-Петербург

2018

**Цель работы.**

Исследование различий в структурах исходных текстов модулей типов **.COM** и **.EXE**,структур файлов загрузочных модулей и способов их загрузки в основную память.

**Необходимые сведения для составления программы.**

Тип IBM PC хранится в байте по адресу 0F000:0FFFEh, в предпоследнем байте ROM BIOS. Соответствие кода и типа в таблице:

**PC FF**

**PC/XT FE, FB**

**AT FC**

**PS2 модель 30 FA**

**PS2 модель 50 или 60 FC**

**PS2 модель 80 F8**

**PCjr FD**

**PC Convertible F9**

Для определения версии MS DOS следует воспользоваться функцией 30Н прерывания 21Н. Входным параметром является номер функции в АН:

**MOV AH,30h**

**INT 2Ih**

Выходными параметрами являются:

AL - номер основной версии. Если 0, то < 2.0

АН - номер модификации

ВН - серийный номер OEM (Original Equipment Manufacturer)

BL:CX - 24-битовый серийный номер пользователя.

**Порядок выполнения и ход работы.**

Необходимо написать текста исходного **.COM** модуля, который определяет тип РС и версию системы.

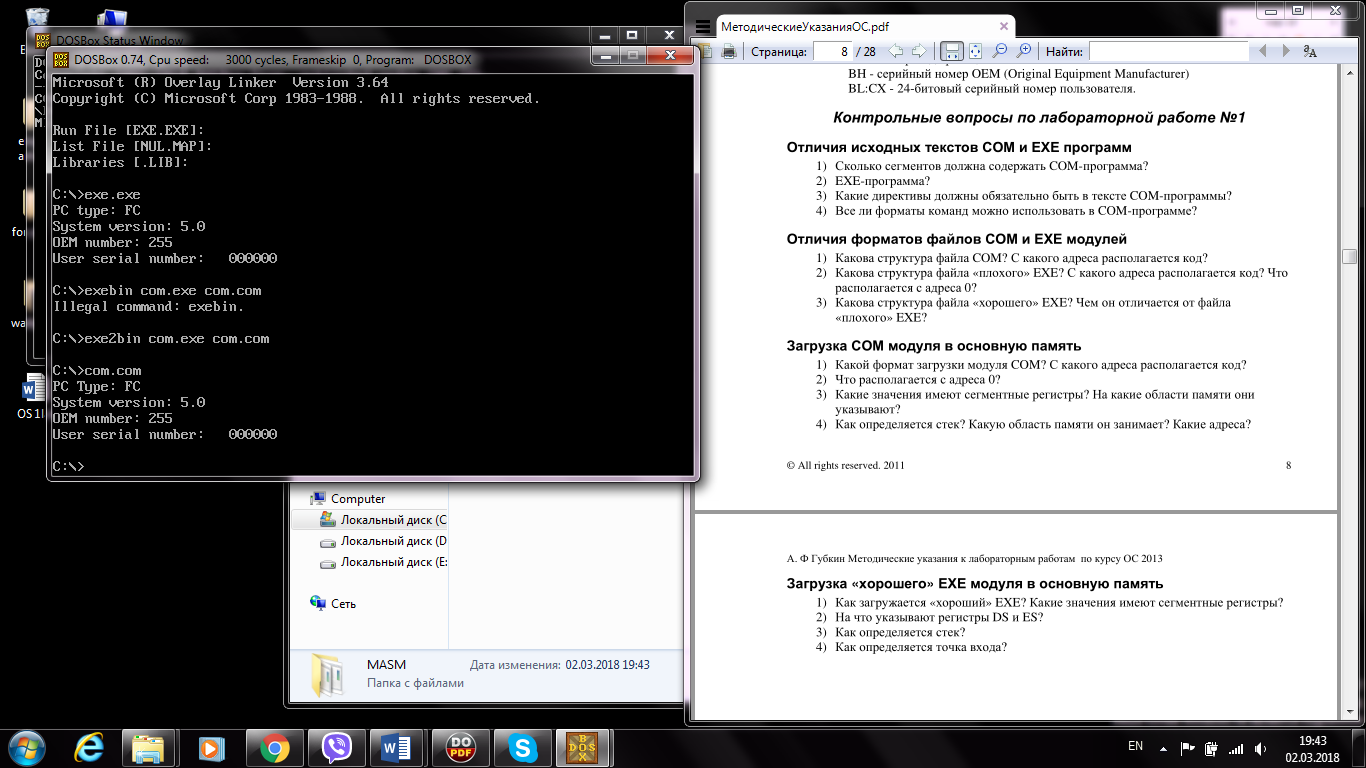
Ассемблерная программа должна читать содержимое предпоследнего байта ROM BIOS, по таблице, сравнивая коды, определять тип РС и выводить строку с названием модели. Если код не совпадает ни с одним значением, то двоичный код переводится в символьную строку, содержащую запись шестнадцатеричного числа и выводится на экран в виде соответствующего сообщения.

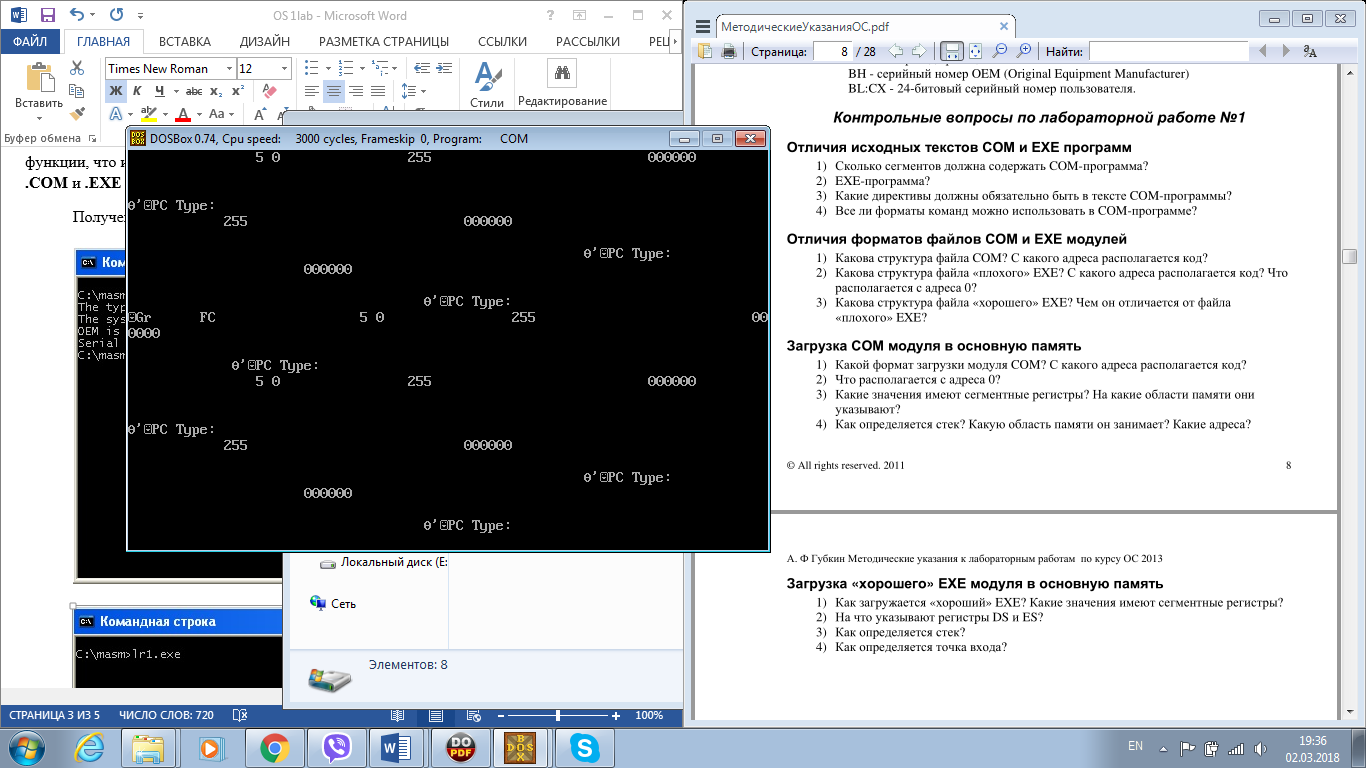
Затем определяется версия системы. Ассемблерная программа должна по значениям регистров AL и AH формировать текстовую строку в формате xx.yy, где xx - номер основной версии, а yy - номер модификации в десятичной системе счисления, формировать строки с серийным номером OEM и серийным номером пользователя. Полученные строки выводятся на экран.

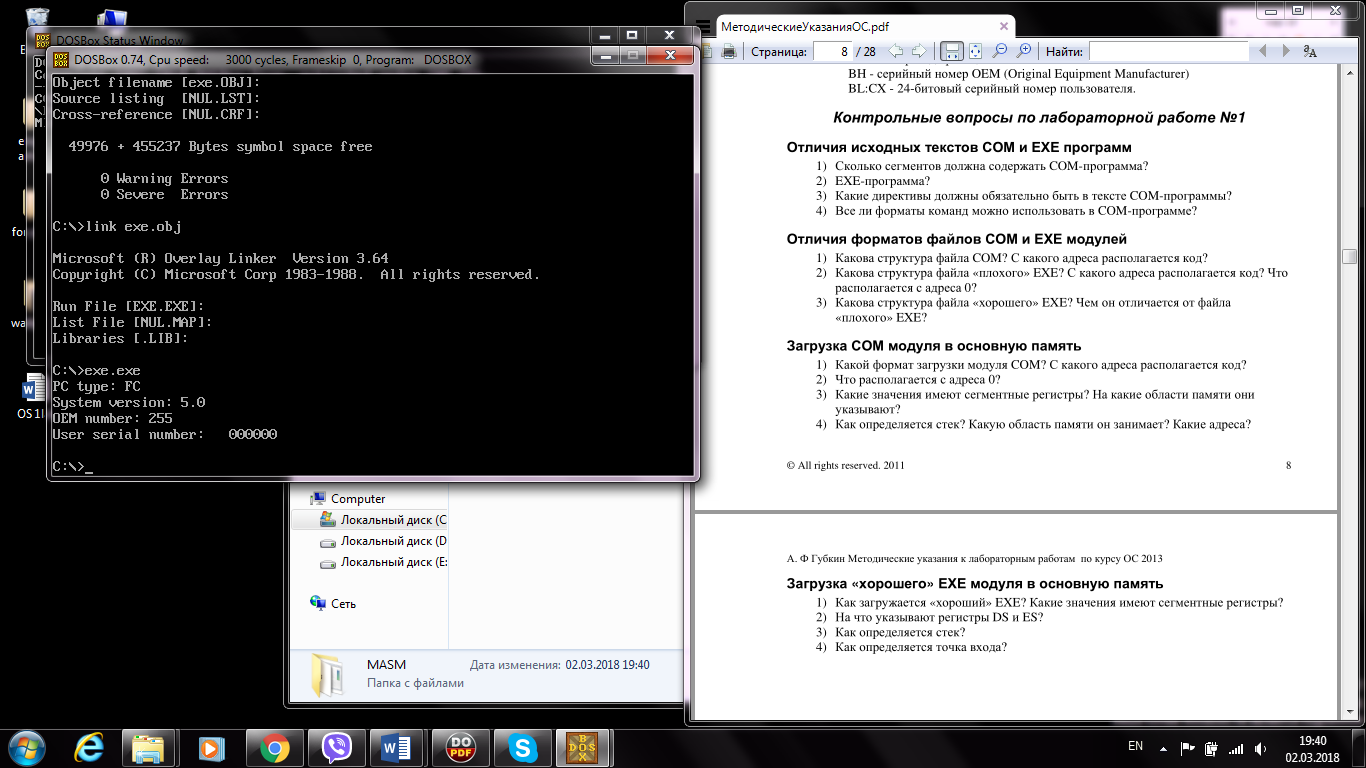
Далее необходимо отладить полученный исходный модуль и получить «хороший» **.COM** модуль, а также необходимо построить «плохой» **.EXE**, полученный из исходного текста для **.COM** модуля.

После нужно написать текст исходного **.EXE** модуля, который выполняет те же функции, что и модуль **.COM**, построить его, отладить и сравнить исходные тексты для **.COM** и **.EXE** модулей.

Полученные результаты приведены на рисунках 1, 2 и 3:

  
Рис. 1. «Хороший»**.COM** модуль

 Рис. 2. «Плохой»**.EXE**

   
Рис. 3. «Хороший»**.EXE**

**Ответы на контрольные вопросы.**

***Отличия исходных текстов COM и EXE программ***

1. Сколько сегментов должна содержать com-программа?

COM-программа содержит один сегмент.

1. Exe-программа?

EXE-программа содержит любое количество сегментов.

1. Какие директивы должны обязательно быть в тексте com-программы?

Обязательно должна быть директива оrg 100h.

Также необходимо присутствие директивы ASSUME, с помощью которой устанавливается соответствие между сегментными регистрами и программными сегментами.

1. Все ли форматы команд можно использовать в com-программе?

Все сегментные регистры в com-программе определяются в момент запуска, а не в момент компиляции, т.к. во время загрузки единственного сегмента com- программы нельзя ничего загрузить в регистр адреса сегмента этого модуля из-за того, что в момент ассемблирования и редактирования связей сегментное значение для сегмента неизвестно.

Нельзя использовать адреса сегмента этого модуля (например, нельзя использовать команду mov ax, data), т.к. для определения адреса загрузчику нужна информация о местоположении в файле полей адресов, требующих настройки на адреса основной памяти. Эта информация хранится в таблице настройки Relocation table, которая отсутствует в com-модуле, поэтому загрузчику негде взять информацию о сегменте.

***Отличия форматов файлов COM и EXE модулей***

1. Какова структура файла COM? С какого адреса располагается код?

Файл COM содержит только один сегмент. Код располагается с адреса 0h.

1. Какова структура файла «плохого» EXE? С какого адреса располагается код? Что располагается с адреса 0?

«Плохой» EXE состоит из сегмента PSP (создается в начале блока процесса после распределения блока памяти) и основного сегмента памяти. Код располагается с 300h. С адреса 0 располагается Relocation Table (таблица разметки).

1. Какова структура файла «хорошего» EXE? Чем он отличается от файла «плохого» EXE?

В «плохом» EXE все содержится в одном сегменте, в то время, как в «хорошем» EXE код, данные и стек находятся в разных сегментах. Код в «хорошем» EXE располагается с адреса 200h (а не с 300h), т.к. нет директивы org 100h, выделяющей место под PSP.

***Загрузка com модуля в основную память***

1. Какой формат загрузки модуля COM? С какого адреса располагается код?

После загрузки com-программы сегментные регистры указывают на начало PSP. Код располагается с адреса 100h.

1. Что располагается с адреса 0?

Сегмент PSP.

1. Какие значения имеют сегментные регистры? На какие области памяти они указывают?

Сегментные регистры содержат адреса памяти, с которых начинаются соответствующие сегменты (рис. 4):

SS – сегмент стека

DS – сегмент данных

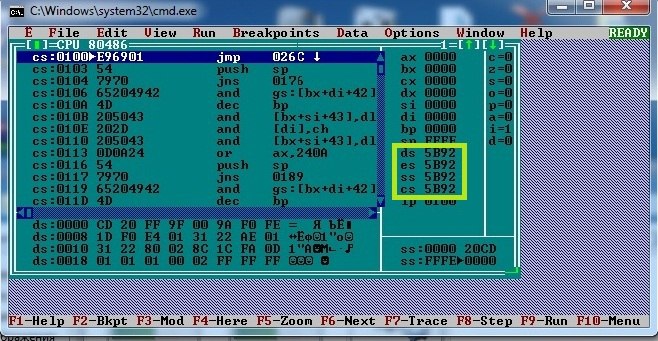
CS – сегмент кода

Рис. 4

1. Как определяется стек? Какую область памяти он занимает? Какие адреса?

Регистр SP указывает на конец стека, а SS – на начало. В отсутствие объявления стека, он определяется автоматически. Указатель стека устанавливается на последнее слово сегмента (SP = FFFEh) (см. рис. 4).

***Загрузка «хорошего» EXE модуля в основную память***

1. Как загружается «хороший» EXE? Какие значения имеют сегментные регистры?

Сегментный адрес PSP заносится в регистр DS в момент исполнения программы. Регистры указывают на начальные адреса соответствующих сегментов.

SS – сегмент стека

CS – сегмент кода

DS, ES – сегмент PSP

1. На что указывают регистры DS u ES?

На начало сегмента PSP.

1. Как определяется стек?

В коде программы прописывается команда .stack.

1. Как определяется точка входа?

Точка входа определяется директивой end. После нее указывается адрес, куда переходит программа при запуске.