**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

отчет

**по практической работе № 1**

**по дисциплине «Операционные системы»**

Тема: **Исследование структур загрузочных модулей**

|  |  |  |
| --- | --- | --- |
| Студент гр. 9383 |  | Чумак М.А. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2021

**Цель работы.**

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

**Постановка задачи.**

Требуется написать текст исходного .COM модуля, который определяет тип PC 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 переводиться в символьную строку, содержащую запись шестнадцатеричного числа и выводиться 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 (Original Equipment Manufacturer) 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. Затем нужно написать текст «хорошего» .EXE модуля, который выполняет те же функции, что и модуль .COM, далее его построить, отладить и сравнить исходные тексты для .COM и .EXE модулей.

Таблица 1 – функции в программе

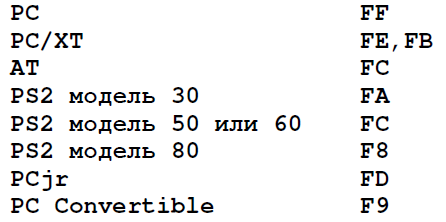
|  |  |
| --- | --- |
| Процедура | Описание |
| TETR\_TO\_HEX | Перевод десятичной цифры в код символа |
| BYTE\_TO\_HEX | Перевод байта в 16-ной с/с в символьный код |
| WRD\_TO\_HEX | Перевод слова в 16-ной с/с в символьный код |
| BYTE\_TO\_DEC | Перевод байта в 16-ной с/с в символьный код в 10-ной с/с |
| PRINT\_STRING | Вывод строки на экран |
| PC\_TYPE | Определение типа PC |
| OS\_VERSION | Определение характеристик OS |

**Выполнение работы.**

Были объявлены следующие строки для вывода информации:

1. PC\_TYPE\_PC db 'PC type: PC', 0dh, 0ah, '$'
2. PC\_TYPE\_PC\_XT db 'PC type: PC/XT', 0dh, 0ah, '$'
3. PC\_TYPE\_AT db 'PC type: AT', 0dh, 0ah, '$'
4. PC\_TYPE\_PS2\_30 db 'PC type: PS2 30', 0dh, 0ah, '$'
5. PC\_TYPE\_PS2\_50\_60 db 'PC type: PS2 50 or 60', 0dh, 0ah, '$'
6. PC\_TYPE\_PS2\_80 db 'PC type: PS2 80', 0dh, 0ah, '$'
7. PC\_TYPE\_PС\_JR db 'PC type: PСjr', 0dh, 0ah, '$'
8. PC\_TYPE\_PC\_CONVERTIBLE db 'PC type: PC Convertible', 0dh, 0ah, '$'
9. DOS\_VERSION db 'MS DOS version: . ', 0dh, 0ah, '$'
10. OEM\_NUMBER db 'OEM serial number: ', 0dh, 0ah, '$'
11. USER\_NUMBER db 'User serial number: H $'

Была объявлена функция, описанная выше, для определения типа ПК (PC\_TYPE) в соответствии с таблицей:

****

Была написана функция для определения характеристик операционной системы OS\_VERSION:

1. номер основной версии системы и её модификации;
2. номер OEM;
3. серийный номер пользователя.

В результате выполнения различных модулей были получены следующие результаты:

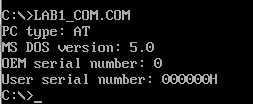


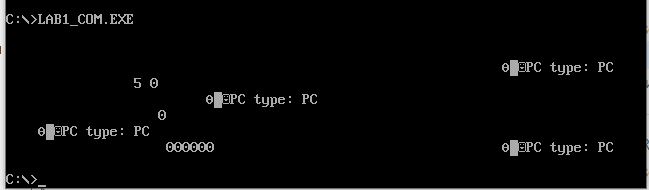
Рисунок 1 — “хороший” .COM модуль

Рисунок 2 — “плохой” .EXE модуль

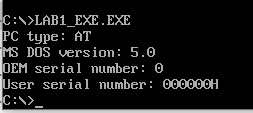


Рисунок 3 — “хороший” .EXE модуль

**Ответы на вопросы.**

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

1. *Сколько сегментов должна сдержать COM-программа?*

Один сегмент. Данные вместе с кодом находятся в одном сегменте, стек же генерируется автоматически.

1. *EXE-программа?*

Не менее одного сегмента. Сегменты кода, данных и стека описываются отдельно друг от друга. Есть возможность не описывать сегмент стека, в это случае будет использоваться стек DOS.

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

Должна быть обязательна директива ORG 100h, потому что при загрузке модуля все сегментные регистры содержат адрес PSP, который является 256-байтовым блоком, поэтому адресация имеет смещение в 256 байт от нулевого адреса. Также необходима процедура ASSUME, для того чтобы сегмент данных и сегмент кода указывали на один общий сегмент.

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

Нет, не все. Нельзя использовать команды вида mov <регистр>, seg <имя сегмента>, потому что в программе .com отсутствует таблица настроек, содержащая описание адресов, которые зависят от размещения загрузочного модуля в ОП.

**Отличия форматов файлов .COM и.EXE программ:**

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

COM-файл состоит из одного сегмента, состоящего из сегмента кода и сегмента данных. Сегмент стека генерируется автоматически при создании COM-программы.  СОМ-файл ограничен размером одного сегмента и не превышает 64 Кб. Код начинается с адреса 0h. При загрузке модуля устанавливается смещение в 100h.

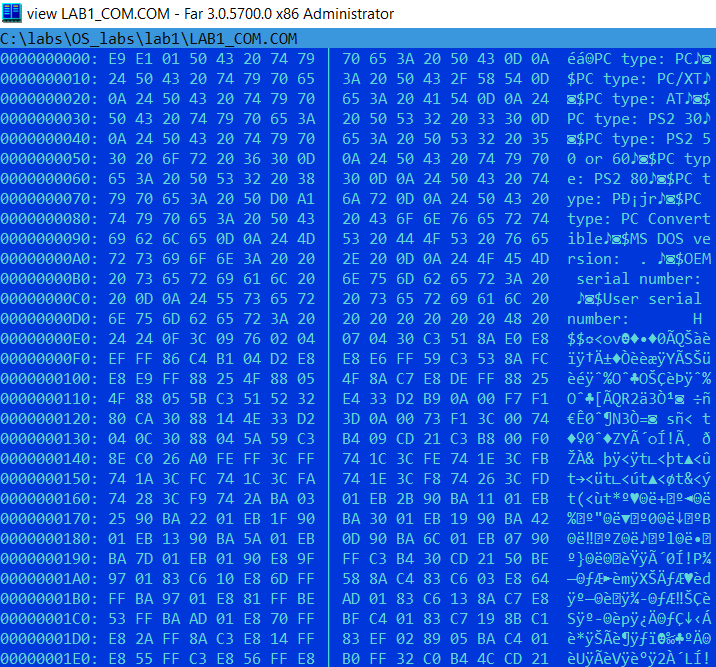


Рисунок 4 — Структура файла “хорошего” .COM модуля

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

В “плохом” файле EXE в нашем случае данные и код располагаются в одном сегменте, что для EXE файла некорректно: код и данные должны быть разделены на отдельные сегменты. Код располагается с адреса 300h, а с адреса 0h идёт таблица настроек.

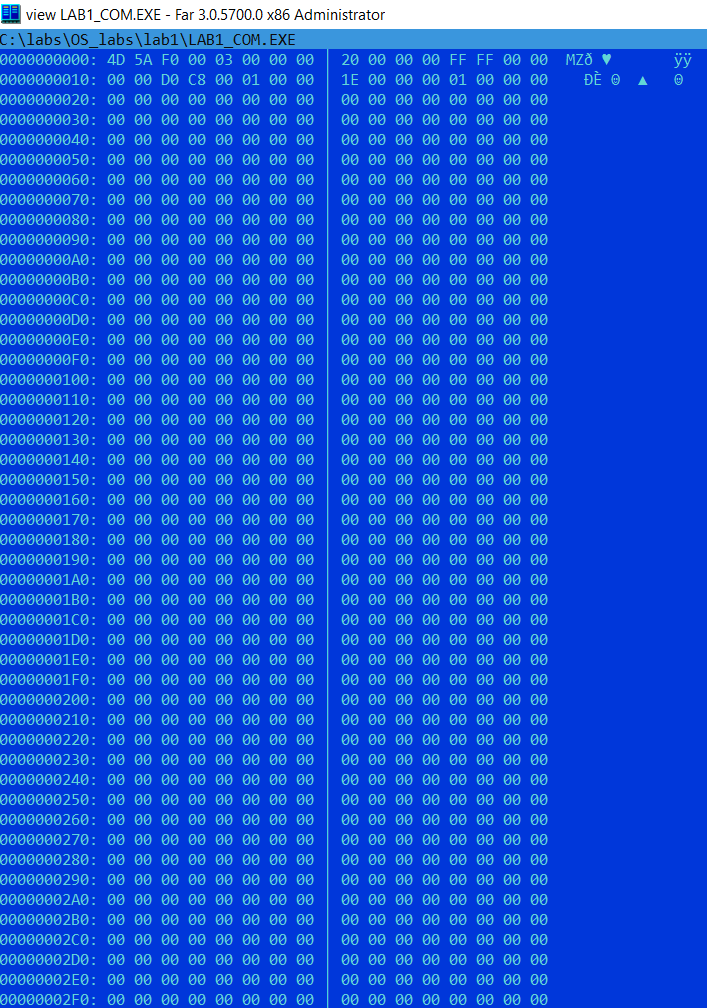


Рисунок 5 — Структура файла “плохого” .EXE модуля (1)



Рисунок 6 — Структура файла “плохого” .EXE модуля (2)

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

В «хорошем» EXE код, данные и стек разделены на сегменты. EXE файл может иметь любой размер. У «хорошего» EXE код, данные и стек находятся в разных сегментах, а в «плохом» - в одном сегменте.

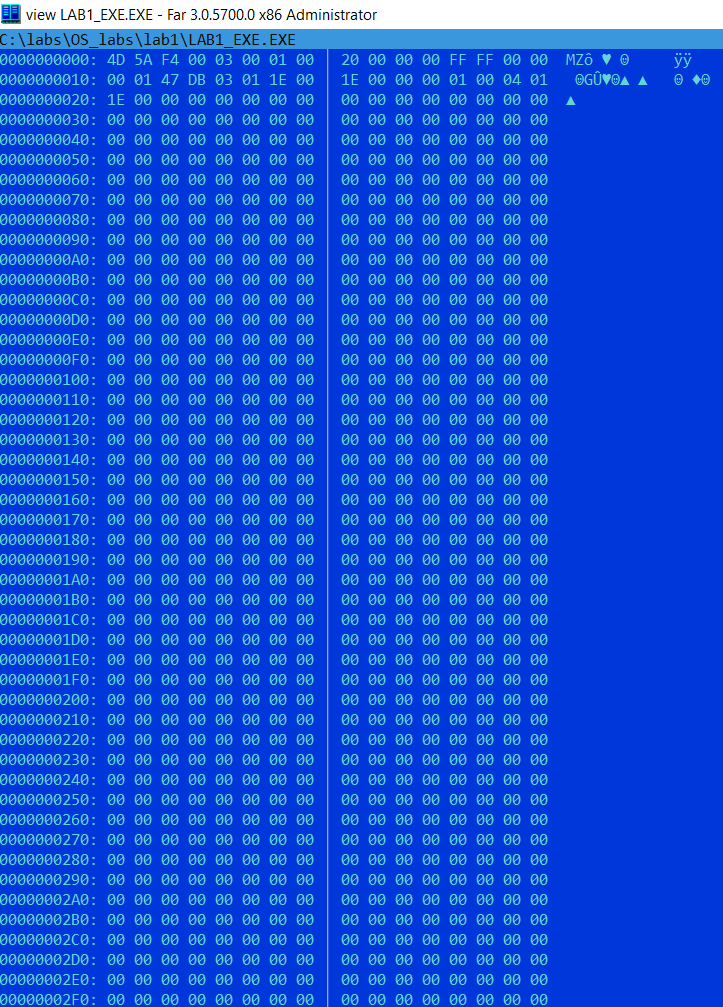


Рисунок 7 — Структура файла “хорошего” .EXE модуля (1)

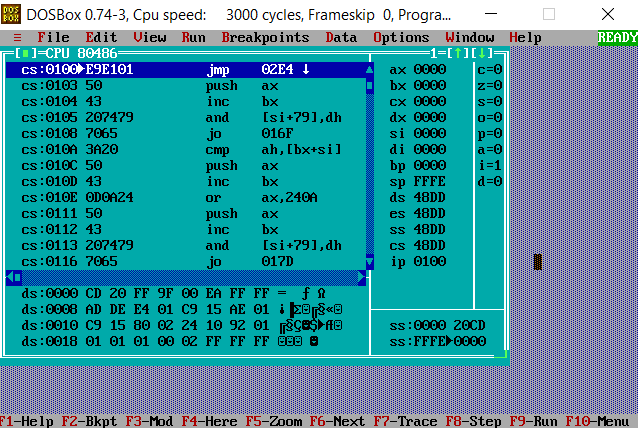


Рисунок 8 — Структура файла “хорошего” .EXE модуля (2)

**Загрузка COM модуля в основную память:**

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

Сначала операционная система ищет подходящее по размеру место в оперативной памяти для COM модуля, после чего ОС помещает в это место PSP и по смещению в 100h помещает модуль.



1. *Что располагается с адреса 0?*

Программный сегмент PSP, размером 256 байт (100h).

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

Сегментные регистры CS, DS, ES и SS указывают на PSP и имеют значения 48DD.

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

Стек генерируется автоматически при создании COM-программы. SS – на начало (0h), регистр SP указывает на конец стека (FFFEh). Адреса стека расположены в диапазоне 0h – FFFEh.

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

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

Данный EXE загружается со считыванием информации заголовка EXE, выполняется перемещение адресов сегментов, ES и DS устанавливаются в начало PSP, SS – на начало сегмента стека, а CS – на начало сегмента команд. В IP загружается смещение точки входа в программу.

1. *На что указывают регистры DS и ES?*

Регистры DS и ES указывают на начало сегмента PSP.

1. *Как определяется стек?*

Стек определяется с помощью директивы .stack, после которой задаётся размер стека. При исполнении регистр SS указывает на начало сегмента стека, а SP на конец стека.

1. *Как определяется точка входа?*

Точка входа определяется при помощи директивы END.

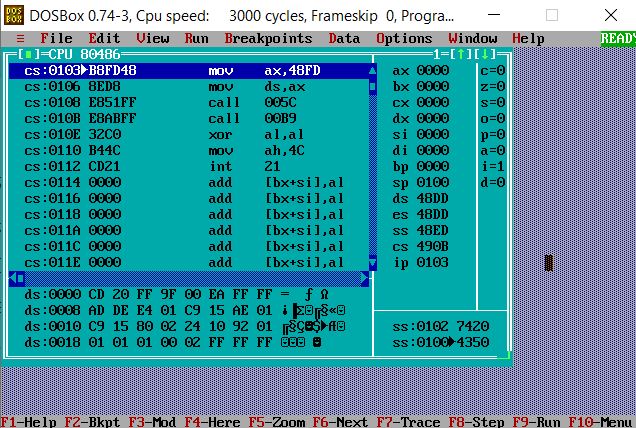


Рисунок 10 — EXE модуль в отладчике TD.EXE

**Выводы.**

В ходе лабораторной работы были исследованы различия в структурах исходных текстов модулей типов .COM и .EXE, структур файлов загрузочных модулей и способов их загрузки в основную память.