**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО** **ЭВМ**

отчет

**по лабораторной работе №5**

**по дисциплине «Операционные Системы»**

Тема: Исследование структур загрузочных модулей

|  |  |  |
| --- | --- | --- |
| Студент гр. 9382 |  | Иерусалимов Н. |
| 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.

|  |  |
| --- | --- |
| 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 |

Таблица 1 – функции в программе

**Выполнение работы.**

Были написаны следующие процедуры:

PRINT – Выводит строку на экран.(Какие строки были объявлены смотрите ниже)

FIND\_INFO\_PC – Находит информацию о компьютере, после чего происходит сравнение с таблицей(рис.4), если были совпадения с ней, заносит в регистр dx нужную строку, после чего вызывается процедура для вывода на экран PRINT.

FIND\_OS\_VERSION – С помощью прерывания 30h в регистр ah заносится информация о версии DOSBOX. После этого мы переводим 16-ричную запись в 10 и выводим на экран значение версии, серийный номер и серийный номер пользователя.

Были объявлены строки для вывода информации:

1. T\_PC db 'Type: PC',0DH,0AH,'$'
2. T\_XT db 'Type: PC/XT',0DH,0AH,'$'
3. T\_AR db 'Type: AT',0DH,0AH,'$'
4. TPS2\_30 db 'Type: PS2 модель 30',0DH,0AH,'$'
5. TPS2\_80 db 'Type: PS2 модель 80',0DH,0AH,'$'
6. T\_JR db 'Type: PСjr',0DH,0AH,'$'
7. T\_CONV db 'Type: PC Convertible',0DH,0AH,'$'
8. VERSIONS db 'Version MS-DOS: . ',0DH,0AH,'$'
9. N\_SERIAL db 'Serial number OEM: ',0DH,0AH,'$'
10. N\_USER db 'User serial number: H $'

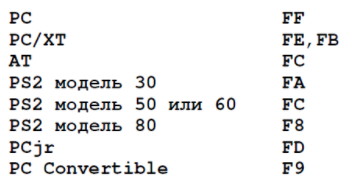


Рис.4

В результате работы программы получаем такой вывод(рис 1-3).

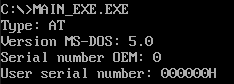


Рисунок 1 – “хороший EXE модуль”

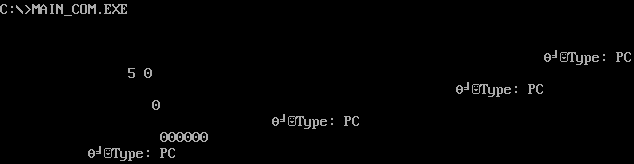


Рисунок 2 – “Плохой EXE модуль”

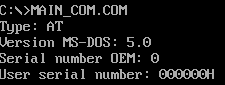


Рисунок 3 – “Хороший .COM модуль”

**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.

**Приложение А.**

**Ответы на контрольные вопросы.**

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

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

Программа содержит один сегмент. В этом сегменте находиться Код и данные, а стек генерируется автоматически.

1. **EXE-программа?**

Должна содержать не менее одного сегмента. Сегменты кода, данных и стека описываются отдельно друг от друга, но есть возможность не описывать сегмент стека, в таком случае будет использоваться стек DOS.

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

Должна быть обязательна директива ORG 100h, так как при загрузке модуля все сегментные регистры содержат адрес префикса программного сегмента (PSP), который является 256-байтовым(100H) 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. (ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING)

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

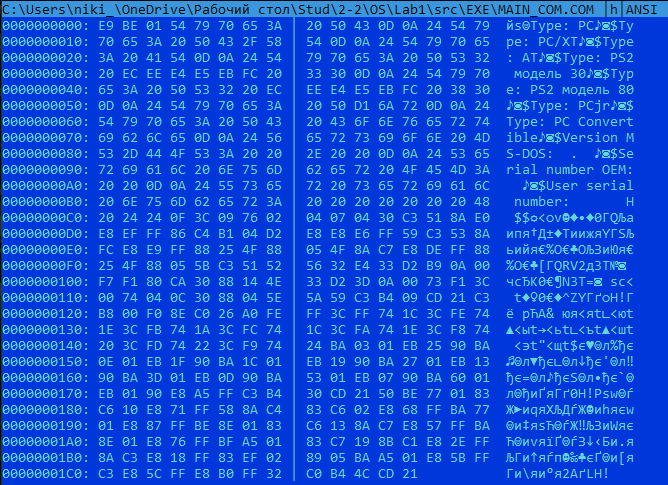
Не все форматы поддерживаются. Нельзя использовать команды вида mov <регистр>, seg <имя сегмента>, так как в .com-программе отсутствует таблица настроек (содержит описание адресов, которые зависят от размещения загрузочного модуля в ОП).

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

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

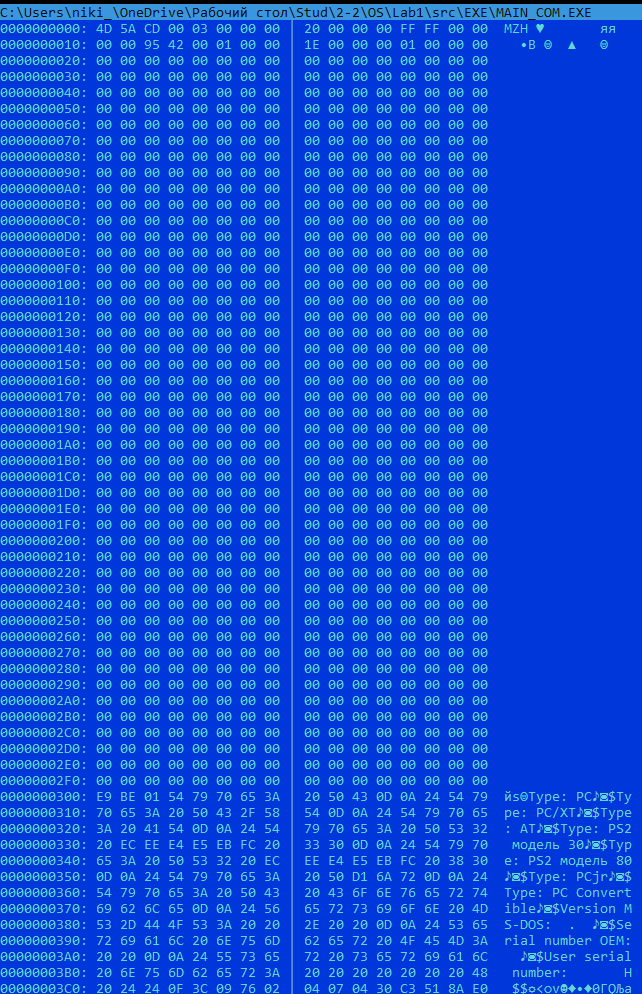
COM-файл состоит из одного сегмента, состоящего из сегмент кода и сегмент данных, сегмент стека генерируется автоматически при создании COM-программы.  СОМ-файл ограничен размером одного сегмента и не превышает 64 Кб

Код начинается с адреса 0h, но при загрузке модуля устанавливается смещение в 100h.



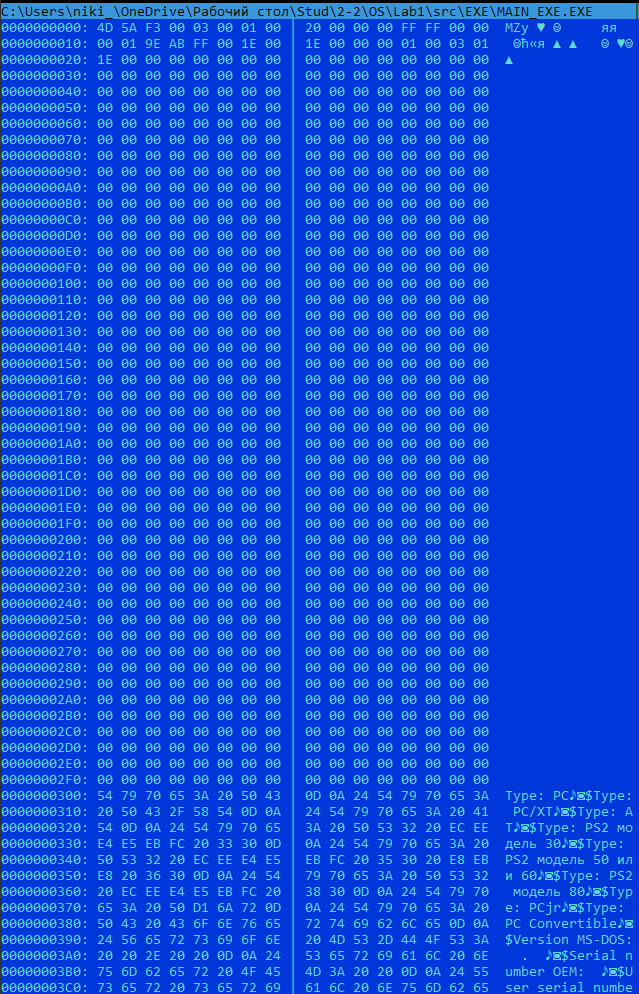
1. **Какова структура файла «плохого» EXE? С какого адреса располагается код? Что располагается с адреса 0?**

В «плохого» EXE данные и код располагаются в одном сегменте, что для EXE файла некорректно, так как код и данные должны быть разделены на отдельные сегменты. Код располагается с адреса 300h, а с адреса 0h идёт таблица настроек.



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

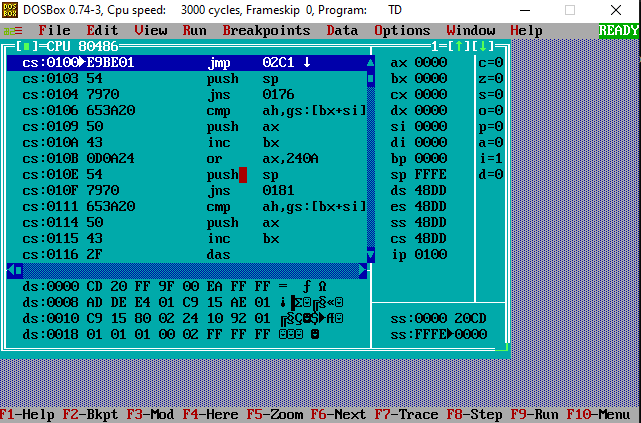
В EXE-программе код, данные и стек поделены на сегменты. Программа в формате ЕХЕ может иметь любой размер. EXE-файл имеет заголовок, который используется при его загрузке. Заголовок состоит из форматированной части, содержащей сигнатуру и данные, необходимые для загрузки EXE-файла, и таблицы для настройки адресов. В отличии от «плохого» EXE в «хорошем» EXE присутствуют три сегмента: сегмент кода, сегмент данных и сегмент стека, а «плохой» EXE содержит один сегмент, совмещающий код и данные. Также в «плохом» EXE адресация кода начинается с 300h, так как он получается из .COM файла, в котором изначально сегмент кода смещён на 100h, а при создании «плохого» EXE к этому смещению добавляется размер PSP модуля(200h). А в «хорошем» EXE присутствует только смещение для PSP модуля, поэтому код начинается с 200h. В данной случае смещение кода 400h так как выделяется память под стек (200h), память под стек находится между PSP и кодом.



**Загрузка COM модуля в основную память:**

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

Определяется сегментный адрес участка ОП, у которого достаточно места для загрузки программы, образ COM-файла считывается с диска и помещается в память, начиная с PSP:0100h. После загрузки двоичного образа COM-программы сегментные регистры CS, DS, ES и SS указывают на PSP(в данном случае сегментные регистры указывают на 48DD), SP указывает на конец сегмента PSP(обычно FFFE), слово 00H помещено в стек, IP содержит 100H в результате команды JMP PSP:100H.



1. **Что располагается с адреса 0?**

Программный сегмент PSP, размером 256 байт (100h), за резервируемый операционной системой.

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

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

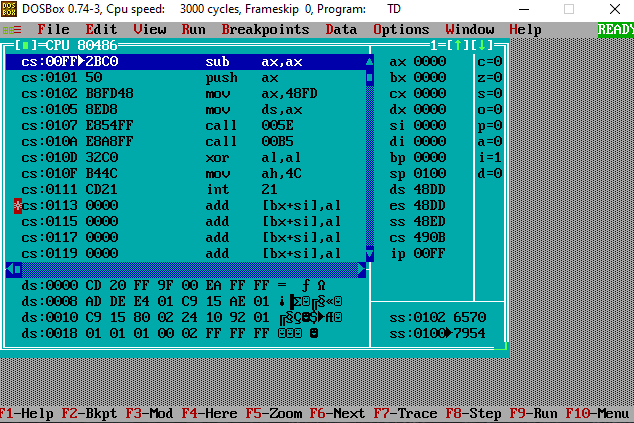
1. **Как определяется стек? Какую область памяти он занимает? Какие адреса?**

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 (FFFEh, – последний адрес, кратный двум).

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

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

EXE-файл загружается, начиная с адреса PSP:0100h. В процессе загрузки считывается информация заголовка (PSP) EXE в начале файла и выполняется перемещение адресов сегментов, то есть DS и ES устанавливаются на начало сегмента PSP(DS=ES=48DD), SS(SS=48ED) – на начало сегмента стека, CS(CS=490D) – на начало сегмента команд. В IP загружается смещение точки входа в программу, которая берётся из метки после директивы END. Причём дополнительный программный сегмент (PSP) присутствует в каждом EXE-файле.



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 исполнение регистр SS указывает на начало сегмента стека, а SP на конца стека(его смещение).

1. **Как определяется точка входа?**

Точка входа определяется при помощи директивы END.