**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Операционные системы»**

**Тема: Исследование структур загрузочных модулей**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 6381 |  | Сергухин В.Ю. |
| Преподаватель |  | Губкин А. Ф. |

Санкт-Петербург

2018

**Постановка задачи**

**Цель работы.**

Исследование различий в структурах исходных текстов модулей .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 0F000:0FFFE, 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 30H прерывания 21H. Входным параметром является номер функции в AH:

MOV AH,30h

INT 21h

Выходными параметрами являются:

AL – номер основной версии. Если 0, то <2.0;

AH – номер модификации;

BH – серийный номер OEM (Original Equipment Manufacturer);

BL:CX – 24-битовый серийный номер пользователя

**Описание функций.**

|  |  |
| --- | --- |
| Название | Назначение |
| TETR\_TO\_HEX | перевод половины байта в 16-ую систему счисления |
| BYTE\_TO\_HEX | перевод байта регистра AL в 16-ую систему счисления в AX |
| WRD\_TO\_HEX | перевод двух байт регистра AX в 16-ую систему счисления в DI |
| BYTE\_TO\_DEC | перевод байта регистра AL в 10-ую систему счисления, помещая результат в SI |
| GET\_PC\_INFO | определяет тип IBM PC |
| GET\_SYSTEM\_INFO | определяет версию MS DOS |
| GET\_OEM\_INFO | определяет серийный номер OEM |
| GET\_USER\_INFO | определяет серийный номер пользователя |
| PrintMsg | выводит строку на экран |

**Описание данных.**

|  |  |
| --- | --- |
| Наименование | Назначение |
| TYPE\_PC | Тип IBM PC |
| SYS\_VER | Версия системы |
| OEM\_NUM | Номер OEM |
| USER\_NUM | Серийный номер пользователя |

**Ход работы**

1. Запуск .COM модуля:

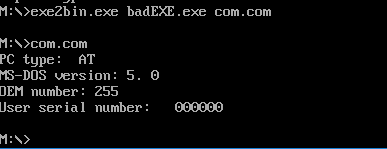


рис.1 Запуск» СОМ-модуля

1. Запуск «плохого» .EXE модуля:

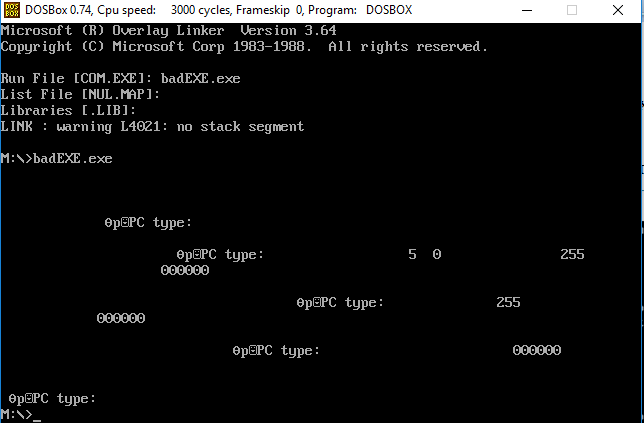


рис.2 Запуск «плохого» EXE-модуля

1. Запуск «хорошего» .EXE модуля



рис 3. Запуск «хорошего» EXE-модуля

**Описание результатов исследования проблем, поставленных в лабораторной работе**

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

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

**Ответ:** COM-программа состоит из одного сегмента, в котором располагаются данные, код и стек.

1. *EXE-программа?*

**Ответ:** EXE-программа может содержать более одного сегмента. В таких программах могут быть сегменты и для кода, и для данных, и для стека.

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

**Ответ:** В тексте COM-программы должна обязательно быть директива ORG 100h (смещение 100h), так как при загрузке СОМ файла в память DOS занимает первые 256 байт (100h) блоком данных PSP, после него располагает код программы. Кроме того, чтобы сообщить транслятору какой сегмент к какому сегментному регистру привязан, используется директива ASSUME. Это связано с тем, что все сегменты сами по себе равноправны, т.к. директивы SEGMENT и ENDS не содержат информацию о функциональном назначении сегментов. Для того чтобы использовать их как сегменты кода, данных или стека, и необходима директива ASSUME. В противном случае, программа не скомпилируется из-за невозможности обнаружения начала сегмента кода.

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

**Ответ:** Нельзя использовать команды, связанные с адресом сегмента, например, команды вида: mov register,segment , ибо до загрузки адрес сегмента неизвестен (так как в COM-программе не содержится таблица настроек (Relocation Table), которая содержит описание адресов, зависящих от размещения загрузочного модуля в ОП, поскольку подобные адреса в нем запрещены), поэтому загрузчик не сможет его определить. По этим же причинам нельзя использовать команды содержащие дальнюю адресацию far: переход на метку возможен только в результате межсегментной передачи управления, а в COM-файле только один сегмент и отсутствует relocation table, следовательно, такие переходы невозможны.

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

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

**Ответ:** файл COM состоит из данных, используемых в программе, и команд. Код располагается с нулевого адреса;

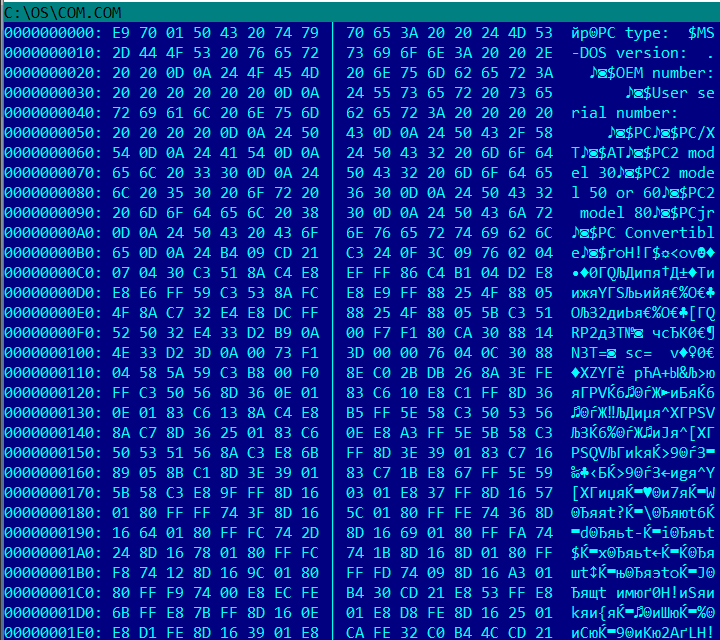


рис 4. Просмотр HEX СOM в Far

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

**Ответ:** В «плохом» EXE файле данные и код располагаются в одном сегменте. С адреса 0h располагается Relocation Table (таблица разметки), а код - с адреса 300h, т.к. заголовок занимает 200h байт(байты 8 и 9 указывают, сколько параграфов занимает заголовок) и команда ORG 100h «сдвигает» код на дополнительные 100h. В первых двух байтах можно увидеть символы MZ, означающие, что формат файла – 16-битный и его следует запускать в соответствии со структурой EXE-файлов.

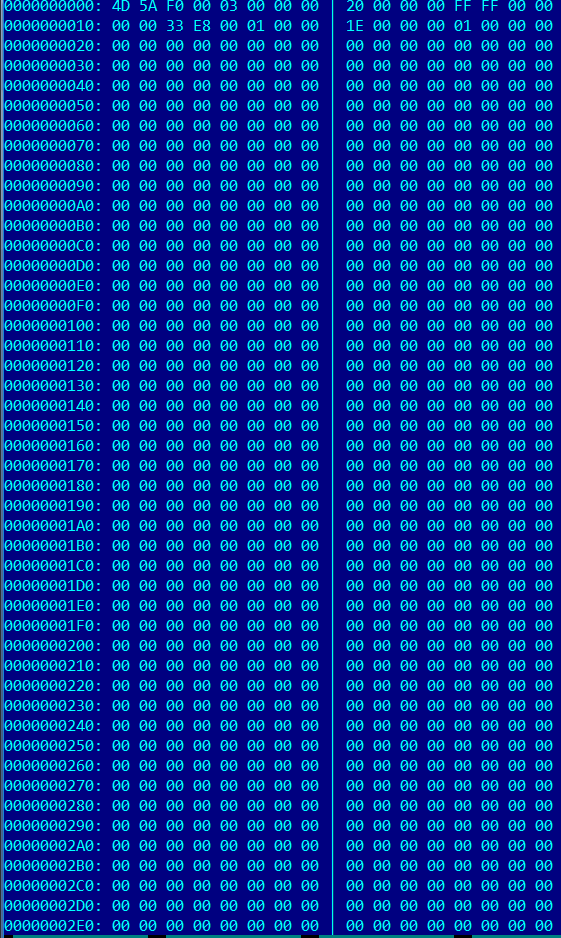


рис 5. Просмотр HEX «плохого» EXE в Far



рис 6. Просмотр HEX «плохого» EXE в Far (продолжение)

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

**Ответ:** В «хорошем» файле EXE содержится информация для загрузчика, сегмент стека, сегмент данных и сегмент кода, то есть 3 сегмента в отличии от одного в «плохом».EXE. Первые 200h байт содержат заголовк, за ним 400h байт памяти занимает сегмент стека, а за ним следует сегменты данных и кода с 600h байта. Это связано с размером стека в 512 слов в десятиричной с.с. (400h байт) выделенным самостоятельно в программе.

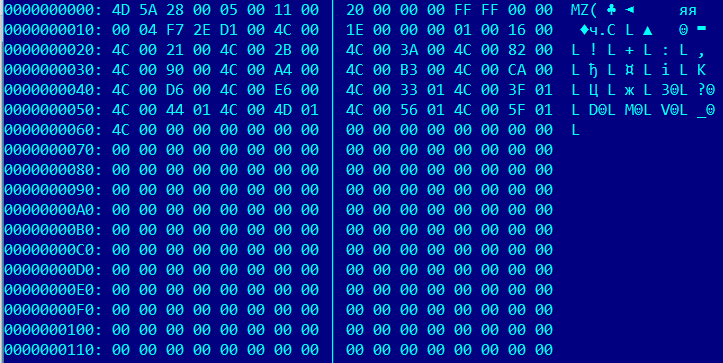


рис 7. Просмотр HEX «хорошего» EXE в Far



рис 8. Просмотр HEX «хорошего» EXE в Far ( продолжение)

***Загрузка COM модуля в основную память:***

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

**Ответ:** После загрузки COM-программы в память сегментные регистры указывают на начало PSP. Порядок загрузки: PSP, данные и код, стек. Код располагается с адреса 100h;

1. *Что располагается с 0 адреса?*

**Ответ:** С нулевого адреса располагается заголовок PSP.

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

**Ответ:** 48DD. Они указывают на начало PSP.

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

**Ответ:** Стек определяется автоматически, указатель стека устанавливается на конец сегмента. Если для программы размер сегмента в 64КБ является достаточным, то DOS устанавливает в регистре SP адрес конца сегмента – FFFEh. Если 64К байтовый сегмент не имеет достаточно места для стека, то DOS устанавливает стек в конце памяти. Адреса расположены в диапазоне 0000h-FFFEh (стек растет от больших адресов к меньшим).

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

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

**Ответ:** В области памяти строится PSP, считывается стандартная часть заголовка в память, определяется длина тела загрузочного модуля, начальный сегмент, загрузочный модуль считывается в начальный сегмент, таблица настройки считывается в рабочую память, определяются значения сегментных регистров. DS и ES устанавливаются на начало PSP, SS - на начало стека, CS - на начало сегмента кода.

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

**Ответ:** Изначально регистры DS и ES указывают на начало PSP.

1. *Как определяется стек?*

**Ответ:** Стек определяется благодаря директиве STACK. Регистры SS и SP принимают значения, указанные в заголовке, потом к SS прибавляется сегментный адрес начального сегмента.

1. *Как определяется точка входа?*

**Ответ:** Смещение точки входа в программу загружается в указатель команд IP. IP, а именно адрес, с которого начинается выполнение программы, определяется операндом директивы END, который называется точкой входа.

**Заключение:**

В ходе работы было проведено исследование различий в структурах исходных текстов модулей .COM и .EXE, , структур файлов загрузочных модулей и способов их загрузки в основную память. Была осуществлена загрузка EXE- модуля с кодом спроектированным для COM-файла в целях наглядной демонстрации различий в способе загрузки СOM и EXE файлов.