**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**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6383 |  | Любчук Д.В. |
| Преподаватель |  | Губкин А.Ф. |

Санкт-Петербург

2017

**Постановка задачи**

**Цель работы**

*Исследование различий в структурах исходных текстов модулей типов* ***.СОМ*** *и* ***.ЕХЕ,*** *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.*

*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 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 (Original Equipment Manufacturer) и серийным номером пользователя. Полученные строки выводятся на экран.*

*Далее необходимо отладить полученный исходный модуль, получить «хороший» .COM модуль и построить «плохой» .EXE, полученный из исходного текста для .COM модуля. Затем нужно написать текст исходного .EXE модуля, который выполняет те же функции, что и модуль .COM, далее его построить, отладить и сравнить исходные тексты для .COM и .EXE модулей.*

**Ход работы**

**Шаг 1.** Был написан 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 PC и выводить строку с названием модели.

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.

Был отлажен полученный исходный модуль.

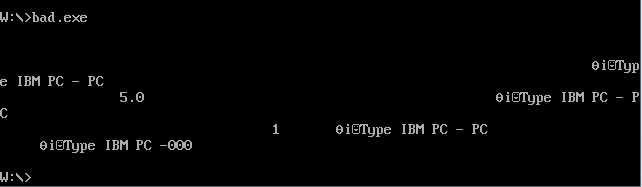
Результатом выполнения этого шага будет «хороший» .COM модуль, а также необходимо построить «плохой» .EXE, полученный из исходного текста для .COM модуля.

**Шаг 2.** Был написан текст исходного .EXE модуля, который выполняет те же функции, что и модуль в Шаге. Таким образом, будет получен «хороший» .EXE.

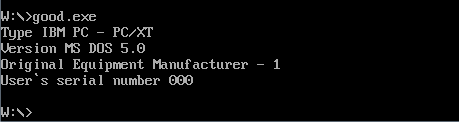
**Шаг 3.**  Были даны ответы на контрольные вопросы.

В результате выполнения лабораторной работы была написана программы, результаты работы которых показаны на рисунках ниже

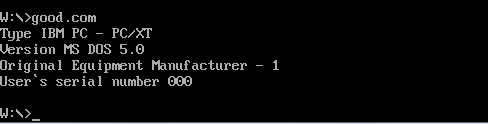
*Запуск плохого .EXE модуля:*



*Запуск хорошего .EXE модуля:*



*Запуск хорошего .COM модуля:*



**Ответы на контрольные вопросы:**

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

1. Сколько сегментов может содержать COM программа?

*Может содержать только один сегмент.*

1. EXE - программа?

*Может содержать больше одного сегмента.*

1. Какие директивы должны обязательно находиться в тексте COM – программы?

*Директива ORG 100h. Нужна для того, чтобы при загрузке модуля в оперативную память в начале COM-программы определяется 256-байтовый (100h) префикс программного сегмента, так что адресация имеет смещение в 256 байт от нулевого адреса.*

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

*Нет. В COM-программе один сегмент. Во время его загрузки нельзя ничего загрузить в регистр адреса сегмента этого модуля так как в момент ассемблирования и редактирования связей сегментное значение для сегмента неизвестно (если использовать команды : mov AX, seg data, mov DS, AX то модуль не соберется).*

*Ответ на примечание: Все будет работать, так как .EXE имеет информацию о сегментах (в таблице настройки адресов), в отличие от .COM файла.*

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

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

*Файл .COM состоит из одного сегмента. Код располагается с адреса 0h.*

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

*Файл «плохого» .EXE содержит сегмент PSP, после него идёт основной сегмент памяти. Заголовок располагается с адреса 0h. Код располагается с адреса 300h:*



*Ответ на примечание: PSP создается только непосредственно при запуске программы.*

*В .EXE файлах есть таблица размещений - relocation table, в которой размещаются адреса всех сегментов в памяти при загрузке программы, и когда мы используем команды типа mov ax, DATA , то есть копируем адрес сегмента данных, который, благодаря этой таблице определен в памяти, то программа нормально компилируется. А в .СОМ файлах этой таблицы нет и когда ты пытаешься использовать такую команду, то программа не соберется, потому что не знает где расположены сегменты.(И эта таблица занимает 200h памяти после заголовка в исполняемых .EXE файлах, если открывать через far и смотреть содержимое. В плохом exe 300h, потому что мы делаем смещение на 100h командой org.)Заголовок - содержит объем памяти, которая резервируется после загрузки под неинициализированные данные и стек. Это позволяет правильно определять адрес, по которому будет загружена следующая программа.*

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

«плохого» ЕХЕ?

*Файл «хорошего» .EXE также содержит и заголовок. В «плохом» .EXE выделяется дополнительная память для заголовка. Код «хорошего» EXE-файла начинается с адреса 200h:*



*Ответ на примечание: В плохом EXE присутствует директива org 100h из-за которой выделяется дополнительная память. 200h дополнительной памяти отводится на заголовок, содержащий форматированную часть(сигнатура программы, необходимая для загрузки) и таблицу настройки адресов. Поэтому в плохом получаем. Поэтому в плохом получаем 300h.*

**Загрузка COM модуля в основную память:**

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

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

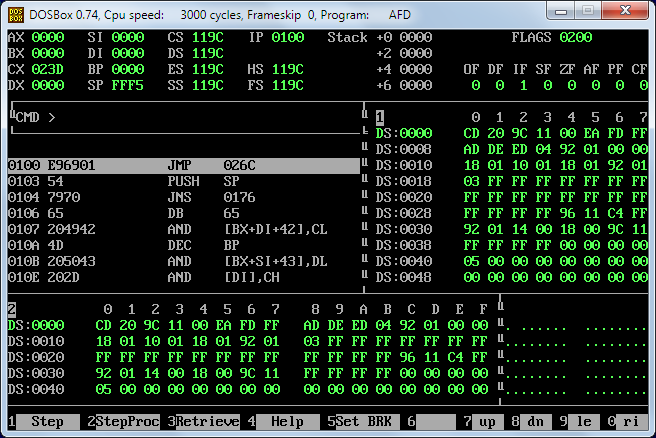
1. Что располагается с адреса 0?

*Блок данных PSP.*

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

*При загрузке .COM модулей все сегментные регистры указывают на адрес PSP.*

*В начале программы они имеют значения 119С:*



4) Как определяется стек? Какую область памяти он занимает? Какие адреса?

*Стек должен входить в какой-то сегмент или образовывать отдельный сегмент. Адрес верхнего элемента хранится в регистре SP – указателе стека. В SS хранится сегментный адрес этого сегмента. Пара регистров SS:SP описывают адрес доступной ячейки стека: в SS хранится сегментный адрес стека, а в SP — относительный адрес доступной ячейки.*

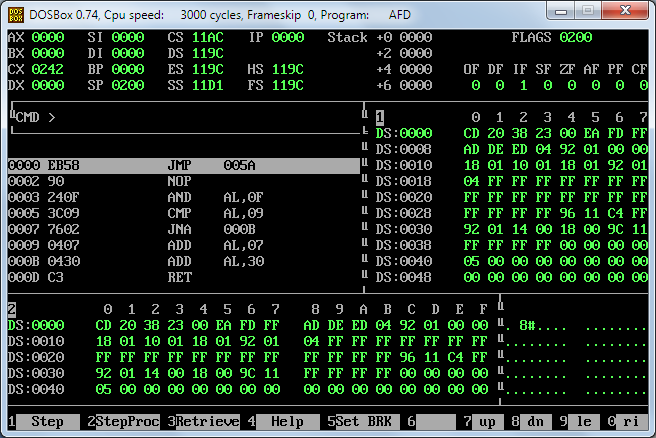
*Если в программе отсутствует явное объявление стека, система сама создает стек по умолчанию в конце сегмента команд по адресу FFFEh относительно начала сегмента команд.*

**Загрузка «хорошего» EXE. Модуля в основную память:**

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

*Регистр CS – начальный адрес сегмента кода. Регистры DS, ES – начало сегмента PSP. Регистр SS – начальный адрес сегмента стека. Регистр DS – начальный адрес сегмента данных. Сегментный адрес PSP заносится в регистр DS в момент исполнения программы.*

*На момент начала SS=11D1, DS=119C, CS = 11AC, ES=119C.*



1. На что указывают регистры DS и ES?

*Регистры DS и ES указывают на начало сегмента PSP (в отличие от .COM-модуля, где при запуске на PSP указывается ещё и регистр CS).*

1. Как определяется стек?

*Стек определяется при помощи команды .STACK (регистрам SS и SP придаются значения, указанные в заголовке, затем к SS прибавляется сегментный адрес начального сегмента).*

1. Как определяется точка входа?

*Смещение точки входа в программу загружается в указатель команд IP . Значение узнаем с помощью адреса, с которого начинается выполнение программы, определяется операндом директивы END.*

**Вывод**

В результате выполнения данной лабораторной работы были исследованы различия в структурах исходных текстов модулей типов .СОМ и .ЕХЕ, структур файлов загрузочных модулей , способов их загрузки в основную память.