**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 загрузочного модулей**

|  |  |  |
| --- | --- | --- |
| Студент гр. 9382 |  | Рыжих Р.В. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2021

**Постановка задачи**

**Цель работы.**

Изучить основные принципы трансляции, отладки и выполнения программ на языке Ассемблера. Исследование различий в структурах исходных текстов модулей типов .COM и .EXE, структур файлов загрузочных модулей и способов их загрузки в основную память.

**Сведения о функциях и структурах данных.**

В данной программе используются следующие функции и структуры данных:

|  |  |
| --- | --- |
| Процедура | Описание |
| TETR\_TO\_HEX | Перевод десятичной цифры в код символа, который записывается в AL |
| BYTE\_TO\_HEX | Перевод значений байта в число 16-ой СС и его представление в виде двух символов |
| WRD\_TO\_HEX | Перевод слова в число 16-ой СС и представление его в виде четырех символов |
| BYTE\_TO\_DEC | Перевод значения байта в число 10-ой СС и представляет его в виду символов |
| PRINT\_STRING | Вывод строки на экран |
| PRINT\_PC\_TYPE | Печать на экран тип ПК |
| PRINT\_OS\_VERSION | Печать на экран версии ОС, серийного номера ОЕМ и серийного номера пользователя |

**Последовательность действий**

В ходе работы программа выполняет следующие действия:

1. Процедура PRINT\_PC\_TYPE, которая выводит на экран тип ПК пользователя. Информация о типе ПК находится в предпоследнем байте ROM BIOS по адресу 0F000:0FFFEh. Значение этого байта определяет тип: Ffh – PC, Feh/Fbh – PC/XT, FCh – AT, FAh – PS2 model 30, FCh – PS2 model 50 or 60, F8h – PS2 model 80, FDh – Pcjr, F9h – PC Convertible. Если значение байта не сходится со значениями типов ПК, то выводится сообщение об ошибке.
2. Процедура PRINT\_OS\_VERSION, которая выводит на экран версию ОС, серийный номер ОЕМ и серийный номер пользователя. В данной процедуре используется функция 30h прерывания 21h.
3. Завершение работы программы.

**Выполнение шагов лабораторной работы:**

**1 шаг:**

Был написан текст исходного .COM модуля Lab1\_COM.asm, который определяет тип ПК и версию его системы. Далее после компилирования был получен «плохой» .EXE модуль Lab1\_COM.exe. При помощи EXE2BIN.EXE и «плохого» модуля был получен «хороший» .COM модуль Lab1\_COM.com.



Рис. 1. - Пример работы "плохого" модуля .EXE Lab1\_COM.exe

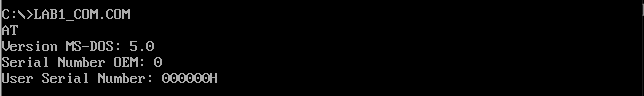


Рис. 2. - Пример работы "хорошего" .COM модуля Lab1\_COM.com

**2 шаг:**

Был написан исходный текст .EXE модуля lab1\_exe.asm, который выполняет те же функции, что и модуль в Шаге 1. Далее был получен «хороший» .EXE модуль lab1\_exe.exe.

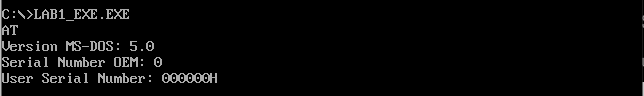


Рис. 3. - Пример работы хорошего .EXE модуля lab1\_exe.exe

**3 шаг:**

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

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

COM-программа должна содержать только один сегмент, потому что данные программы и сам хранятся в одном сегменте, а стек автоматически устанавливается на последнюю ячейку сегмента.

*2) EXE-программа?*

EXE-программа должна содержать один или более сегментов. Количество сегментов зависит от выбранной модели памяти.

*3) Какие директивы должны обязательно быть в тексте COM-программы?*

В COM-программе обязательно должна быть директива ORG 100h. Данная директива устанавливает CS:IP на конец PSP, так как после загрузки все сегментные регистры (как и CS) указывают на начало PSP, а IP = 0, а это значит, что программа не будет выполняться, начиная с этого адреса. Именно эта директива смещает все относительные адреса на 100h байт.

В COM-программе обязательно должна быть директива ASSUME. Данная директива указывает ассемблеру с каким сегментом или группой сегментов связаны регистры.

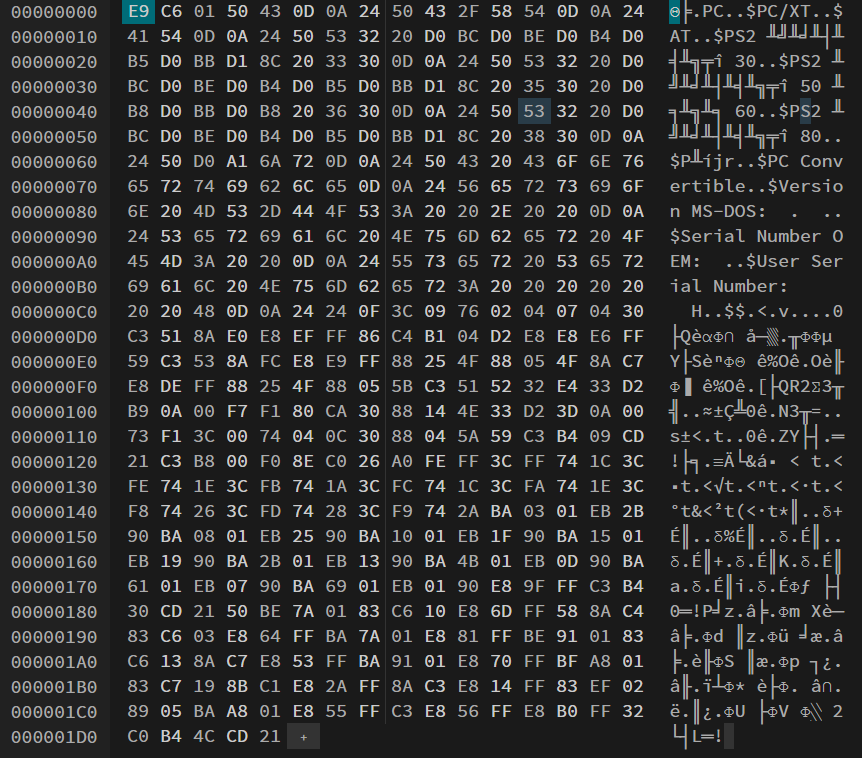
В COM-программе обязательно должна быть директива END. Данная директива завершает работу программы на ассемблере.

*4) Все ли форматы команд можно использовать в COM-программе?*

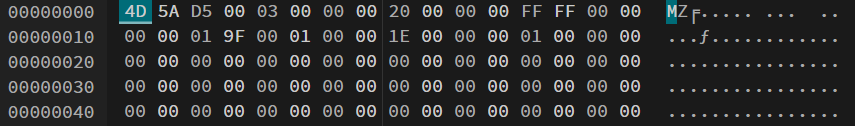
Нельзя использовать команды вида: seg NAME, где NAME – название сегмента, так как в COM-программе отсутствует таблица настройки. Таблица настроек содержит в себе описание необходимого адреса, которое зависит от размещения модуля в памяти.

**Шаг 4:**

Шестнадцатеричное представление модуля .COM:

  
Рис. 4. - Шестнадцатеричное представление модуля .COM:

Шестнадцатеричное представление плохого модуля .EXE:



Дальше идут нули до 300h строки

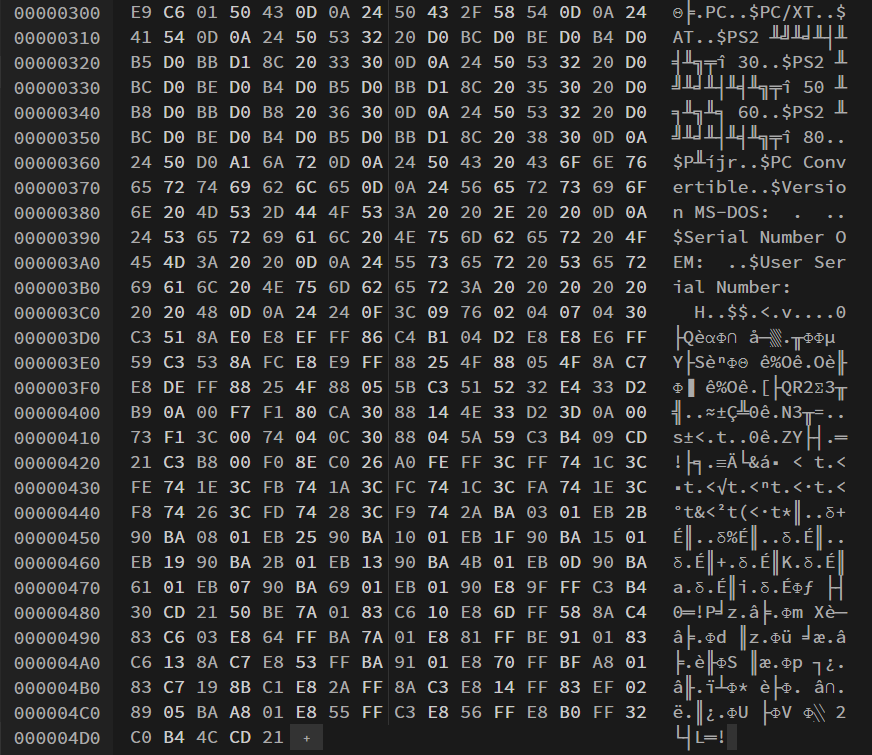
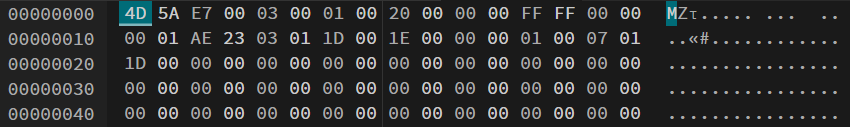


Рис. 5. - Шестнадцатеричное представление плохого модуля .EXE

Шестнадцатеричное представление хорошего модуля .EXE:



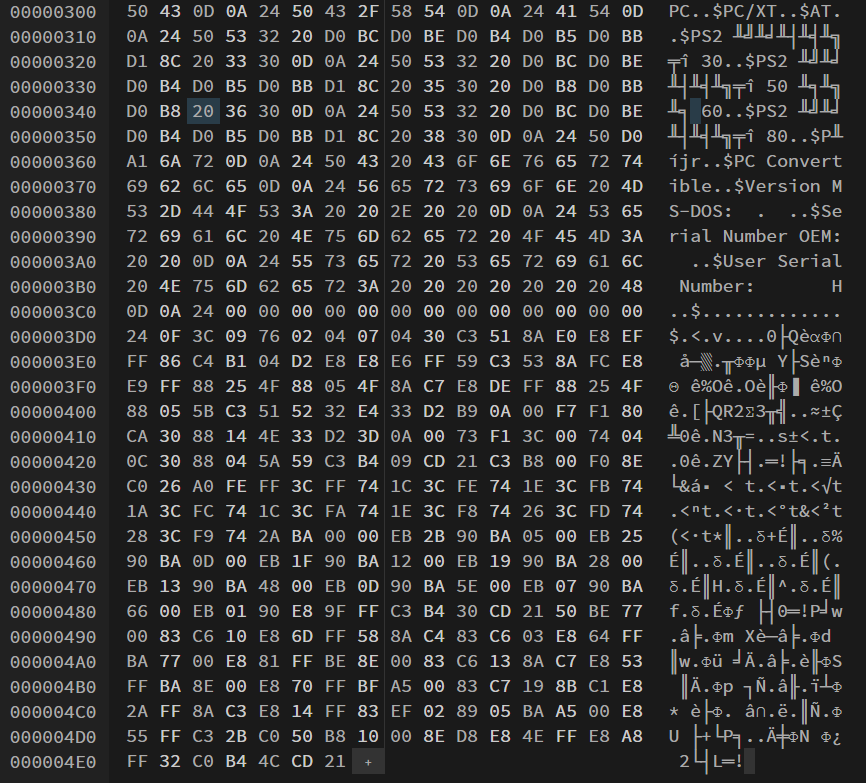


Рис. 6. - Шестнадцатеричное представление хорошего модуля .EXE

**«Отличие форматов файлов COM и EXE модулей»**

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

В данном файле код, данные и стек находятся в одном сегменте. Код и данные начинаются с адреса 0h (См. Рис. 4).

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

В «плохом» EXE файле код, данные и стек находятся в одном сегменте. Код и данные начинаются с адреса 300h. С адреса 0h находится управляющая информация загрузчика, которая содержит заголовок и таблицу настроек. (См. Рис. 5-6)

*3) Какова структура «хорошего» EXE? Чем он отличается от файла «плохого» EXE?*

У «хорошего» EXE код, данные и стек находятся в разных сегментах, а в «плохом» - в одном сегменте. С адреса 0 в «хорошем» EXE располагается заголовок, который содержит в себе таблицу для настройки адресов, данные и сигнатуру, в отличие от «плохого» EXE. У «хорошего» EXE выделяется память под стек между PSP и кодом.

У «плохого» EXE смещение 300h, т.к. изначально смещение 100h, а при создании EXE появляется смещение 200h с информацией для загрузчика.

У «хорошего» EXE смещение также 300h, т.к. выделяется сегмент под стэк на 100h и также добавляется смещение при создании EXE.

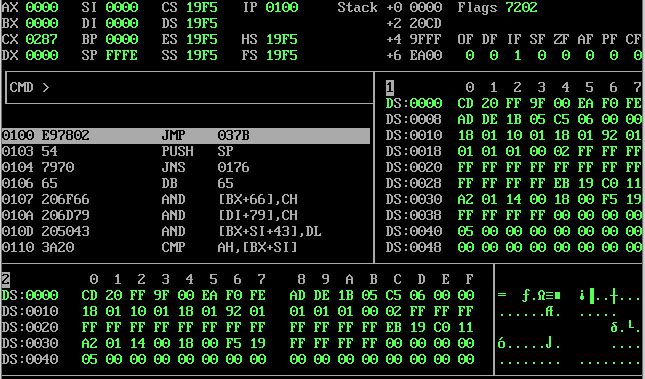
**Шаг 5:**

**«Загрузка COM модуля в основную память»**

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

В начале определяется сегментный адрес участка ОП, способного вместить загрузку программы, затем создается блок памяти для PSP и программы. После считывания COM-файл помещается в память с 100h. После сегментные регистры устанавливаются в начало PSP. SP устанавливается в конец PSP, 0000h помещается в стек, а в IP записывается 100h.

Код располагается с адреса 100h.

  
Рис. 9. - .COM в отладчике

*2) Что располагается с адреса 0?*

С адреса 0 располагается PSP размером в 100h байт.

*3) Какие значения имеют сегментные регистры? На какие области памяти они указывают?*

Регистры DS, ES, CS, SS указывают на начало блока PSP.

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

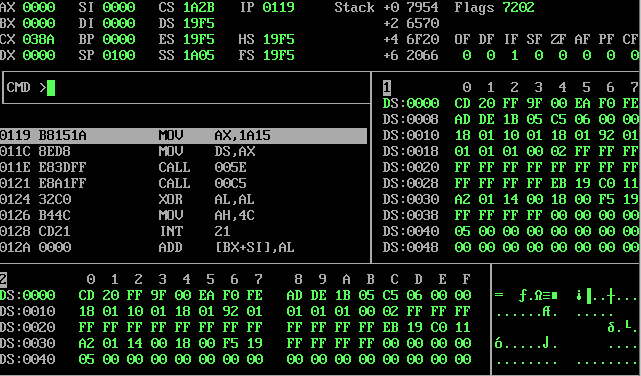
Стек генерируется автоматически. Регистр SS указывает на начало блока PSP, а SP на конец стека. Стек расположен между адресами SS:0000h – SS:FFFFh и заполняется с конца модуля в сторону уменьшения адресов.

**Шаг 6:**

**«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.

  
Рис. 10. - .EXE в отладчике

*2) На что указывают регистры DS и ES?*

ES и DS указывают на начало сегмента PSP.

*3) Как определяется стек?*

Стек определяется на основе директивы .stack с указанием размера стека. SS указывает на начало сегмента стека, а SP указывает на конец.

*4) Как определяется точка входа?*

Точка входа определяется параметром после директивы END.

**Заключение.**

В результате выполнения лабораторной работы были изучены структурные отличия .COM и .EXE модулей и получены навыки работы с отладчиком TD.EXE.