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

Студентка гр. 8383 _____ Ишанина Л.Н.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Исследование различий в структурах исходных текстов модулей типов .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 0F000:0FFFEh, 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 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.

Постановка задачи.

Требуется реализовать 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. 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.

Процедуры используемые в программе.

TETR_TO_HEX – процедура для перевода половины байта в шестнадцатеричную систему счисления.

BYTE_TO_HEX – процедура для перевода байта регистра AL в шестнадцатеричную систему счисления, помещая результат в AX.

WRD_TO_HEX – процедура для перевода двух байт регистра AX в шестнадцатеричную систему счисления, помещая результат в регистр DI.

BYTE_TO_DEC – процедура для перевода байта регистра AL в десятичную систему счисления, помещая результат в SI.

Ход работы.

1. Запуск «хорошего» .COM модуля.

Рисунок 1 – «Хороший» .COM модуль

Запуск «плохого» .EXE модуля.

Рисунок 2 – «Плохой» .EXE модуль

2. Запуск «хорошего» .EXE модуля.

Рисунок 3 – «Хороший» .EXE модуль

3. Ответы на контрольные вопросы. Отличия исходных текстов COM и EXE программ.

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

Один сегмент.

2) **EXE программа?**

EXE программа может содержать больше одного сегмента.

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

В тексте COM программы обязательно должны быть следующие директивы:

- ORG 100h

Данная директива необходима, так как она задает смещение для всех адресов программы на 256 байт для PSP.

- ASSUME

Эта директива ставит в соответствие начало программы сегментам кода и данных.

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

Нет, не все форматы команд можно использовать в COM-программе, так как COM-программа не располагает таблицей настроек(Relocation Table). Адреса сегментов определяются загрузчиком в момент запуска программы на основе информации о местоположении полей адресов в файле из этой таблицы. А значит, из-за отсутствия Relocation Table в COM-программах, такие команды как mov [регистр], seg [сегмент] невозможны.

4. COM модуль в шестнадцатеричном виде.

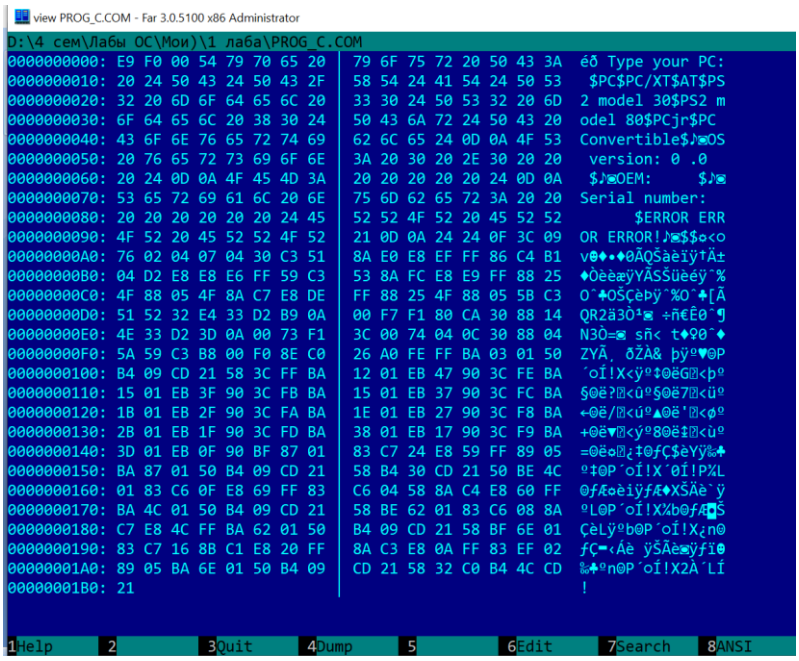


Рисунок 4 – .COM модуль в шестнадцатеричном виде

«Плохой» .EXE модуль в шестнадцатеричном виде.

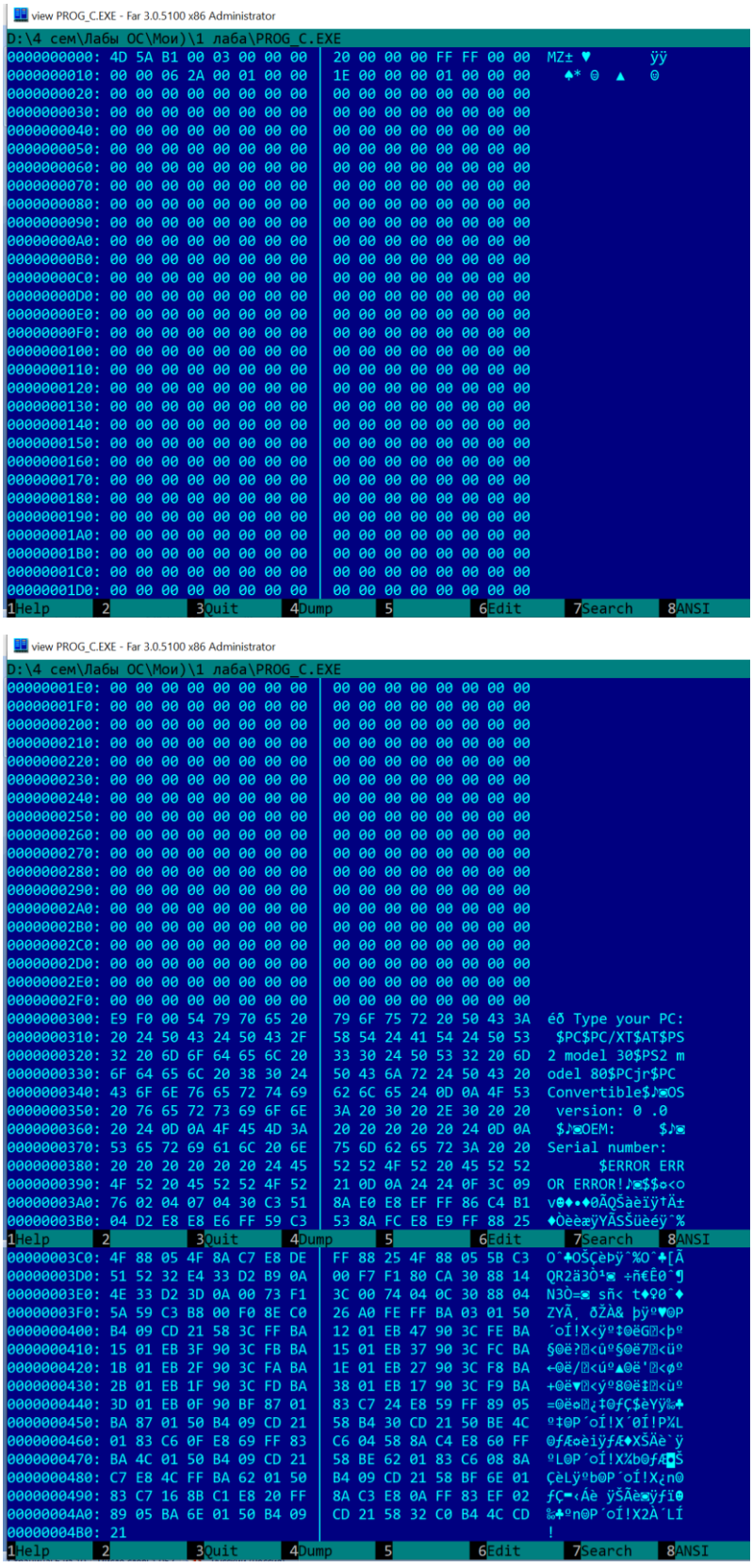
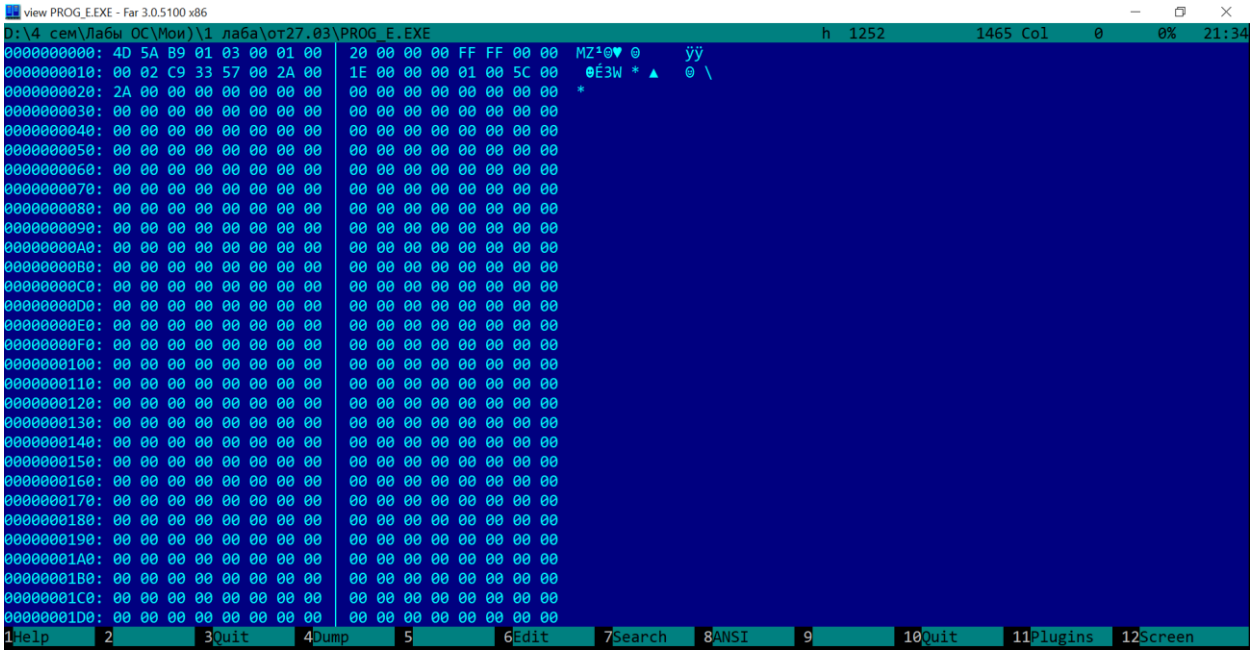
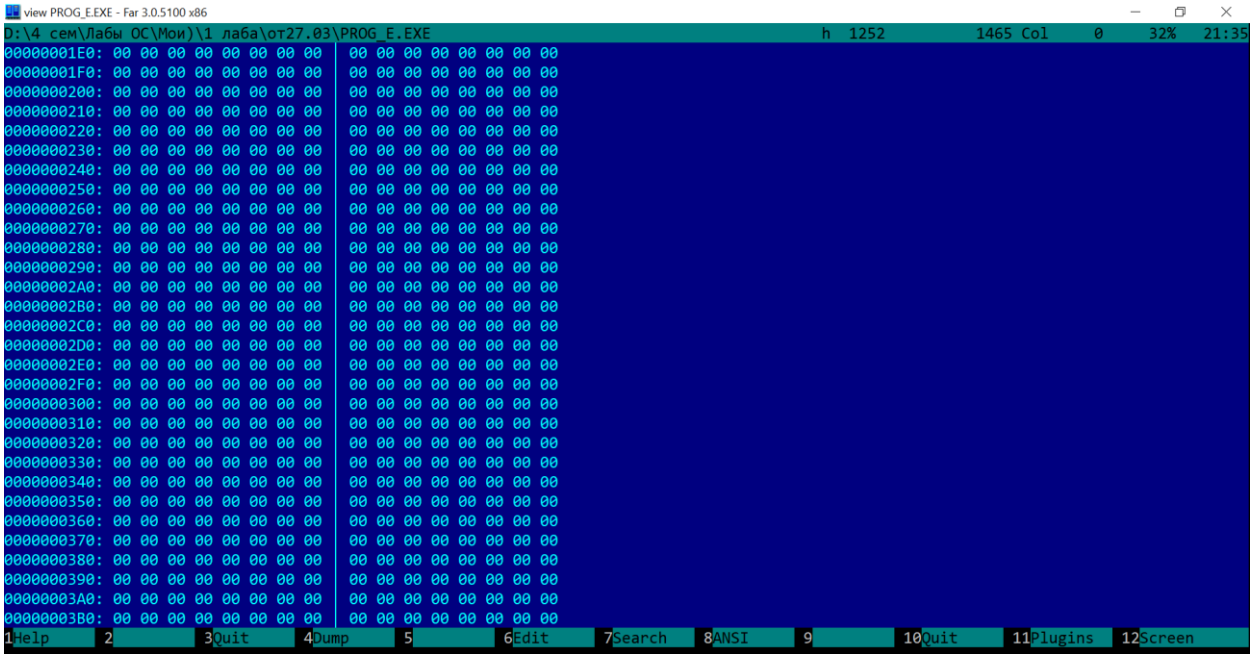


Рисунок 5 - «Плохой» .EXE модуль в шестнадцатеричном виде «Хороший» .EXE модуль в шестнадцатеричном виде.


view PROG_E.EXE - Far 3.0.5100 x86
D:\4 сем\Лабы ОС\Мои)\1 лаба\от27.03\PROG_E.EXE h 1252 1465 Col 0 0% 21:34
0000000000: 4D 5A B9 01 03 00 01 00 20 00 00 00 FF FF 00 00 MZ¹☺♥ ☺ ÿÿ
0000000010: 00 02 C9 33 57 00 2A 00 1E 00 00 00 01 00 5C 00 ☻É3W * ▲ ☺ \
0000000020: 2A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 *
0000000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000000A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000000B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000000C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000000D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000000E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000000F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000001A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000001B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000001C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000001D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1Help 2 3Quit 4Dump 5 6Edit 7Search 8ANSI 9 10Quit 11Plugins 12Screen


view PROG_E.EXE - Far 3.0.5100 x86
D:\4 сем\Лабы ОС\Мои)\1 лаба\от27.03\PROG_E.EXE h 1252 1465 Col 0 32% 21:35
00000001E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000001F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000220: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000230: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000240: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000250: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000260: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000270: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000280: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000290: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000002A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000002B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000002C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000002D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000002E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000002F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000300: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000310: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000320: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000330: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000340: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000350: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000360: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000370: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000380: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000390: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000003A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000003B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1Help 2 3Quit 4Dump 5 6Edit 7Search 8ANSI 9 10Quit 11Plugins 12Screen

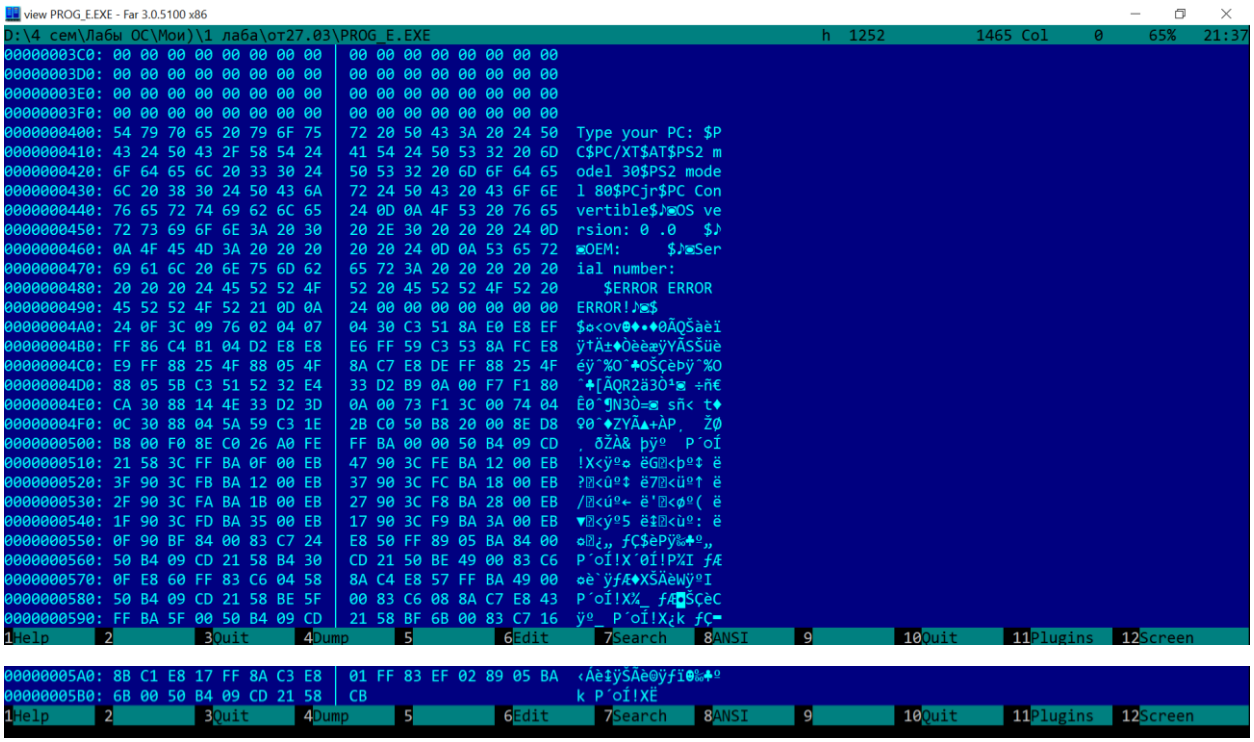


Рисунок 6 - «Хороший» .EXE модуль в шестнадцатеричном виде

5. Ответы на контрольные вопросы. Отличия форматов файлов COM и EXE программ.

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

Структура COM файла состоит из одного сегмента и содержит данные и машинные команды. Код начинается с адреса 0h, но при загрузке модуля устанавливается смещение в 100h.

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

Структура “плохого” EXE файла содержит данные и код в одном сегменте. Код располагается с адреса 300h. С адреса 0 располагается Relocation Table.

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

Структура “хорошего” EXE файла содержит информацию для загрузчика, сегмент стека, сегмент данных и сегмент кода, а именно 3

сегмента вместо одного как в “плохом” EXE. Так же код располагается с адреса 400h в отличие от 300h в “плохом” EXE файле.

6. Загрузка COM модуля в основную память.

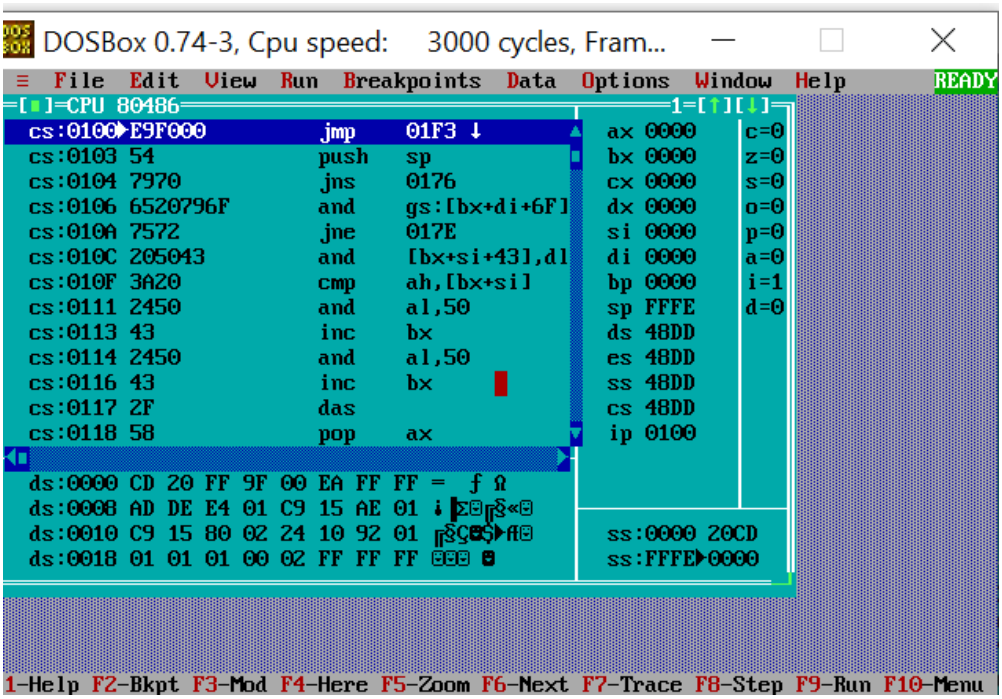


Рисунок 7 – Загрузка COM модуля в основную память

7. Ответы на контрольные вопросы. Загрузка COM модуля в основную память.

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

Сегментные регистры указывают на начало PSP. Код располагается с адреса 100h.

2) **Что располагается с 0 адреса?**

Адрес начала PSP.

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

Сегментные регистры имеют значения 48DDh. Они указывают на начало PSP.

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

Стек определяется автоматически так, что 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. Адреса расположены в диапазоне 0000h-FFFEh.

Загрузка «хорошего» EXE модуля в память.

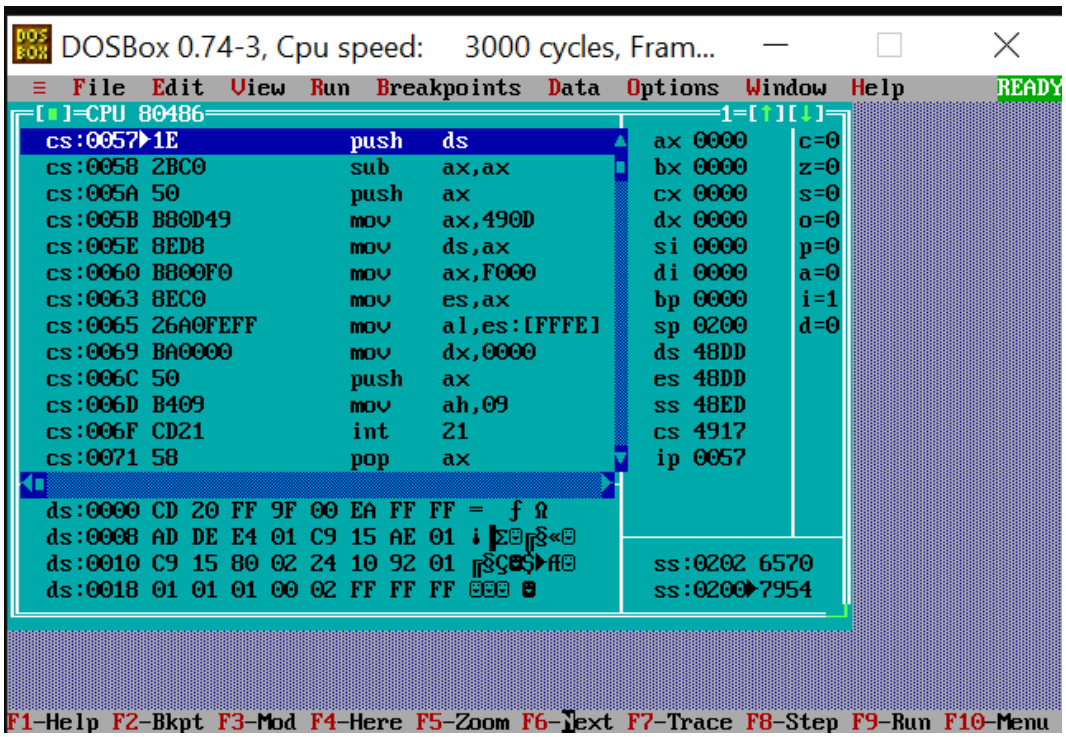


Рисунок 8 – Загрузка «хорошего» EXE модуля в память

8.Ответы на контрольные вопросы. Загрузка «хорошего» EXE модуля в память.

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

В области памяти строится PSP, стандартная часть заголовка считывается в память, определяется длина тела загрузочного модуля, определяется начальный сегмент, загрузочный модуль считывается в начальный сегмент, таблица настройки считывается в рабочую память, определяются значения сегментных регистров. DS и ES(48DD)

устанавливаются на начало PSP, SS(48ED) - на начало стека, CS(4917) - на начало сегмента кода.

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

Регистры DS и ES указывают на начало PSP.

3) **Как определяется стек?**

В исходном коде модуля стек определяется при помощи директивы STACK, а при исполнении в регистры SS и SP записываются адрес начала сегмента стека и его вершины соответственно.

4) **Как определяется точка входа?**

Точка входа определяется при помощи директивы END.

Вывод.

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.