

**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. 0381

\_\_\_\_\_

Кирильцев Д.А.

Преподаватель

\_\_\_\_\_

Ефремов М. А.

Санкт-Петербург

2022

### **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 РС 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.

Таблица 1 — Процедуры в программе.

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
model_print	Вывод строки на экран
PC_ver	Определение модели PC
OS_ver	Определение версии OS
OEM_num	Определение OEM
USER_num	Определение серийного номера пользователя

### Выполнение работы.

Данные объявленные в программе:

```
PC_m db 'PC',0Dh,0Ah,'$'  
XT_m db 'PC/XT',0Dh,0Ah,'$'  
AT_m db 'AT',0Dh,0Ah,'$'  
PS2_model30_m db 'PS2 model 30',0Dh,0Ah,'$'  
PS2_model80_m db 'PS2 model 80',0Dh,0Ah,'$'  
PCjr_m db 'PCjr',0Dh,0Ah,'$'  
PC_convertible_m db 'PC Convertible',0Dh,0Ah,'$'  
PC_custom_m db ' ',0Dh,0Ah,'$'  
DOS_ver db ' . ',0Dh,0Ah,'$'  
OEM db ' ',0Dh,0Ah,'$'  
USER db ' ',0Dh,0Ah,'$'
```

Программа последовательно выводит тип ПК, версию ОС, OEM и номер пользователя.

Далее предоставлены скриншоты полученных модулей.



Рис.1-  
"хороший"  
COM модуль



Рис. 2 - "плохой" EXE  
модуль

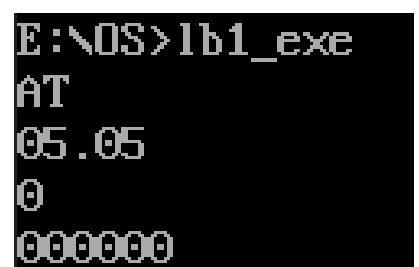


Рис.3-"хороший"  
EXE модуль

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

1. Сколько сегментов должна содержать COM-программа?

- Один, код и данные в COM-модуле располагаются в одном сегменте, а стек генерируется автоматически.

2. EXE-программа?

- EXE-модуль должен содержать сегмент кода и сегмент данных. Остальные сегменты являются опциональными. Если не объявить стек, то будет использоваться DOS-овский.

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

- 1. `ORG 100h` - Так как адресация начинается с шест. смещения 100 от начала PSP, то в программе после оператора `SEGMENT` кодируется директива `ORG 100H`.
- 2. `ASSUME` - для того, чтобы сегмент данных и сегмент кода указывали на один общий сегмент.

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

- Нет. Нельзя использовать команды вида `mov register, segment`, т. к. В момент ассемблирования и редактирования связей сегментное значение для сегмента неизвестно. Оно определяется только при загрузке программы. Поскольку файл типа `.COM` не может предоставить загрузчику перечня всех сегментных ссылок (информация для перемещения), то в данном случае программа будет выполняться неправильно.

## Отличия форматов файлов .COM и .EXE программ

```
C:\Documents and Settings\Danil DU\Desktop\KAMI.COM
00000000: E9 80 01 50 43 0D 0A 24 50 43 2F 58 54 0D 0A 24  é?@PCJOSPC/XIJC$
00000001: 41 54 0D 0A 24 50 53 32 20 6D 6F 64 65 6C 20 33  AIJC$PS2 model 3
00000002: 30 0D 0A 24 50 53 32 20 6D 6F 64 65 6C 20 38 30 0JC$PS2 model 80
00000003: 0D 0A 24 50 D0 A1 6A 72 0D 0A 24 50 43 20 43 6F JJC$PD: jrJC$PC Co
00000004: 6E 76 65 72 74 69 62 6C 65 0D 0A 24 20 20 0D 0A nvertibleJC$ JC
00000005: 24 20 20 2E 20 20 0D 0A 24 20 20 20 0D 0A 24 20 $ JC$ JC$
00000006: 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
00000007: 20 20 20 20 24 24 0F 3C 09 76 02 04 07 04 30 C3 $S< OuB+♦♦0A
00000008: 51 8A E0 E8 EF FF 86 C4 B1 04 D2 E8 E8 E6 FF 59 QSaëiÿ+â±♦0èèæÿY
00000009: C3 53 8A FC E8 E9 FF 88 25 4F 88 05 4F 8A C7 E8 àSSüèéÿ^%0^âOSÇè
0000000A: DE FF 88 25 4F 88 05 5B C3 51 52 3C 00 74 1F 32 _ÿ^%0^âIaQR< t^2
0000000B: E4 33 D2 B9 0A 00 F7 F1 80 C2 30 88 14 4E 33 D2 à301 0 ^â?A0^IN30
0000000C: 3D 0A 00 73 F1 3C 00 0C 30 88 04 EB 05 90 0C 30 = 0 sã< 90^èè?90
0000000D: 88 04 5A 59 C3 B4 09 CD 21 C3 B8 00 F0 8E C0 26 ^♦ZYA^oi!A. dZA&
0000000E: A0 FE FF 3C FF 74 1F 3C FE 74 21 3C FB 74 1D 3C _ÿ<ÿt^<_t!<ÿt+<
0000000F: FC 74 1F 3C FA 74 21 3C F8 74 23 3C FD 74 32 3C ÿt^<ÿt!<ot#<ÿt2<
00000010: F9 74 34 EB 1F 90 BA 03 01 EB 2F 90 BA 08 01 EB ÿt4èÿ?^v&è/?^0è
00000011: 29 90 BA 10 01 EB 23 90 BA 15 01 EB 1D 90 BA 24 >?^0èè?^S0è+?^$
00000012: 01 EB 17 90 BE 4C 01 46 E8 55 FF BA 4C 01 EB 0A 0è±?_L0FèUÿèL0è0
00000013: 90 BA 33 01 EB 04 90 BA 3B 01 E8 98 FF C3 B4 30 ?^30è+?^;0è^ÿA^0
00000014: CD 21 50 BE 51 01 46 E8 5F FF 58 83 C6 04 E8 58 I!P_Q0Fè_ÿXfè♦èX
00000015: FF BA 51 01 E8 7E FF C3 BE 59 01 8A C7 E8 49 FF ÿèQ0è^ÿA_ÿSÇèIÿ
00000016: BA 59 01 E8 6F FF C3 BF 5F 01 83 C7 05 8B C1 E8 èÿ0è0ÿAè_0fÇèAè
00000017: 1F FF 8A C3 E8 09 FF 83 EF 02 89 05 BA 5F 01 E8 ▼ÿS0è0ÿfè0è0è
00000018: 53 FF C3 E8 54 FF E8 B5 FF E8 CC FF E8 D8 FF 32 SÿAèIÿèµÿèIÿè0ÿ2
00000019: C0 B4 4C CD 21 A'LI!
```

Рис. 4 - "Хороший" COM модуль

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

- COM-файл состоит из одного сегмента, сегмент стека генерируется автоматически при создании COM - модуля.
- COM-файл ограничен размером одного сегмента и не превышает 64 Кб.
- Программа, записанная в файле типа .COM может сразу выполняться (из-за постоянного смещения).
- Код начинается с адреса 0h, но при загрузке модуля устанавливается смещение в 100h. (в доказательство приведен рис. 4)

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

```

C:\Documents and Settings\Daniil_DU\Desktop\KAMI.EXE
00000000: 4D 5A 95 00 03 00 00 00 20 00 00 00 FF FF 00 00 MZ
00000010: 00 00 C0 D0 00 01 00 00 1E 00 00 00 01 00 00 00 AD
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000001A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000001B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000001C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000001D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000001E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000001F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000220: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000230: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000240: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000250: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000260: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000270: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000280: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000290: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000002A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000002B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000002C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000002D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000002E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000002F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000300: E9 80 01 50 43 0D 0A 24 50 43 2F 58 54 0D 0A 24
00000310: 41 54 0D 0A 24 50 53 32 20 6D 6F 64 65 6C 20 33

```

Рис. 5 - "плохой" EXE модуль

- У «плохого» EXE файла данные и код располагаются в одном сегменте, однако это не соответствует формату EXE.
- Код начинается с адреса 300h, а с адреса 0h идёт настраивающая таблица (заголовок EXE файла). (Это иллюстрирует Рис. 5)

### 3. Какова структура «хорошего» EXE? Чем он отличается от файла «плохого» EXE?

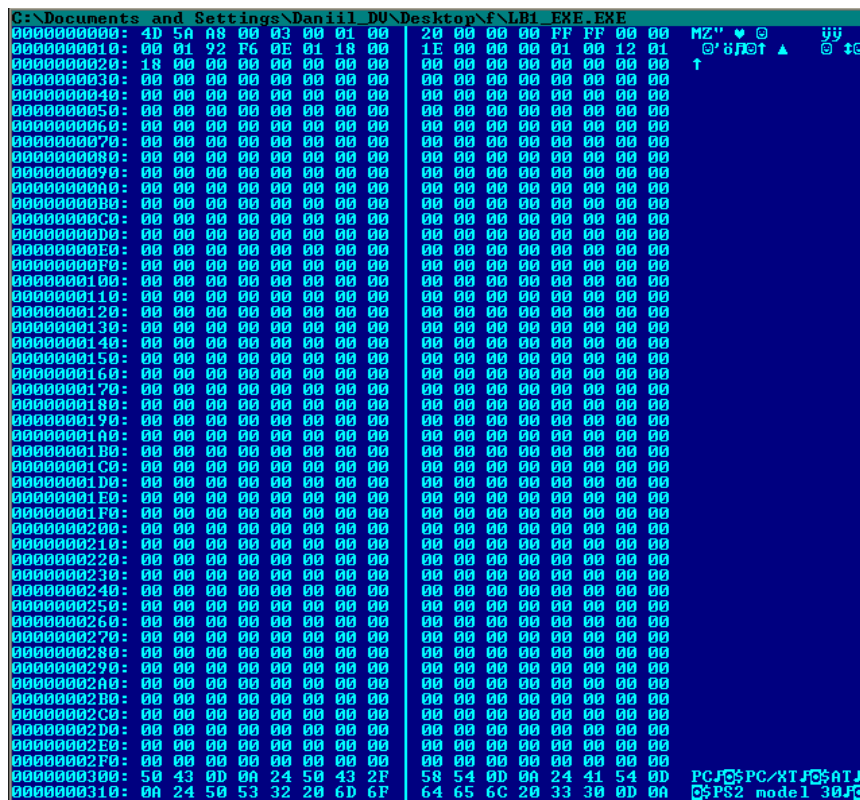


Рис. 6 - "хороший" EXE модуль

- В EXE-модуле код и данные являются отдельными сегментами, также присутствует таблица связей, заголовков, отвечающий за настройку адресов.
- В «хорошем» EXE-модуле происходит разделение сегментов (кода и данных), необходимое для правильного форматирования, а в «плохом» содержится лишь один сегмент, объединяющий код и данные. «Плохой» EXE начинает код с 300h, так как он получается из COM модуля, в котором изначально сегмент кода смещён на 100h, Но, так как, происходит создание EXE-модуля, добавляется еще и сдвиг PSP (200h). В «хорошем» EXE



присутствует только смещение для PSP модуля, поэтому код начинается с 200h.

- В данном случае смещение кода 300h так как выделяется память под стек (в размере 100h), память под стек находится между PSP и кодом. (Как показано на рис. 6)

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

- Стек находится между PSP и данными и занимает с 100h до 300h

### **Загрузка COM модуля в основную память**

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

- Определяется сегментный адрес участка Основной Памяти, у которого достаточно места для загрузки программы, образ COM-файла считывается с диска и помещается в память, начиная с адреса PSP 100h. После загрузки двоичного образа COM-модуля сегментные регистры CS, DS, ES и SS указывают на PSP(в данном случае сегментные регистры указывают на 48DD), SP указывает на конец сегмента PSP (FFFE), слово 00H помещено в стек, IP содержит 100H. (Это можно увидеть на Рис. 7)

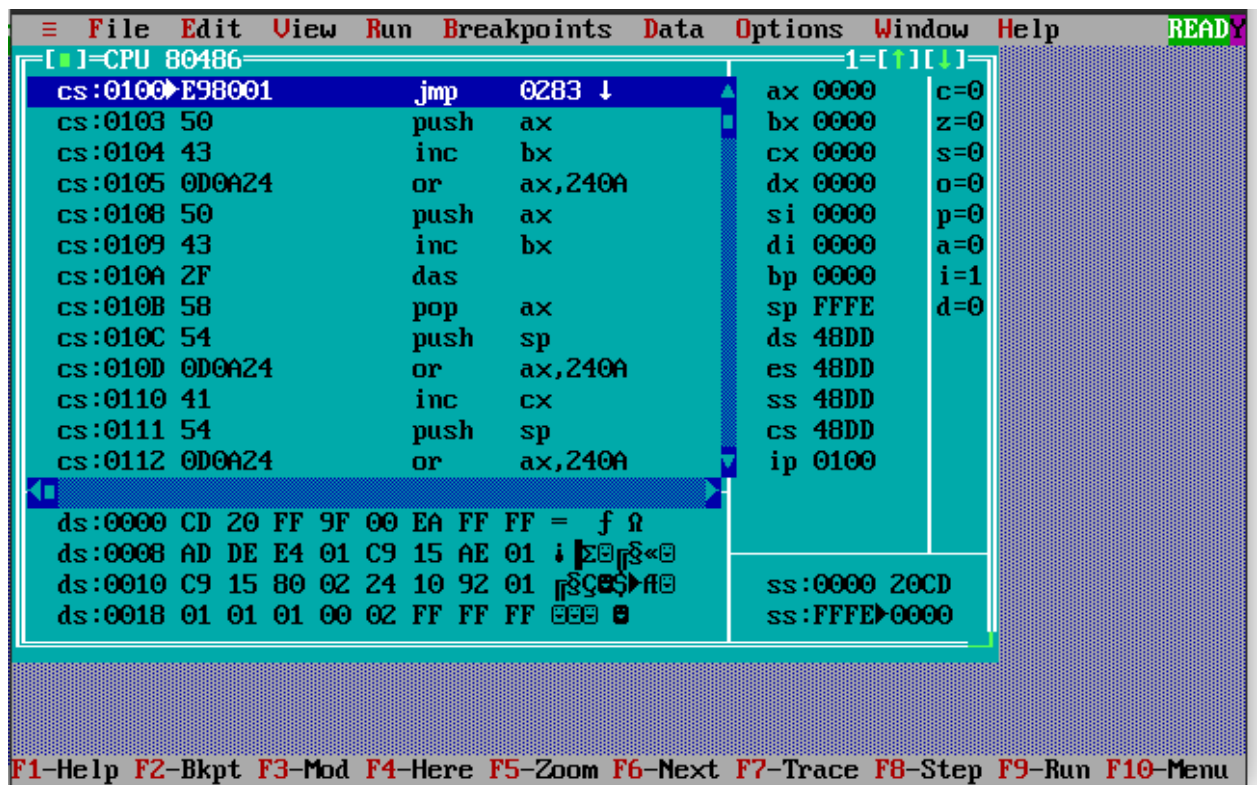


Рис. 7 - Отладчик для "Хорошего" COM-модуля

2. Что располагается с адреса 0?

- Программный сегмент PSP, размером 256 байт (100h), зарезервируемый операционной системой.

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

- Сегментные регистры CS, DS, ES и SS указывают на PSP и имеют значения 48DD. (Это можно увидеть на Рис. 7)

4. 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). (Это можно увидеть на Рис. 7)

## Загрузка «хорошего» EXE модуля в основную память

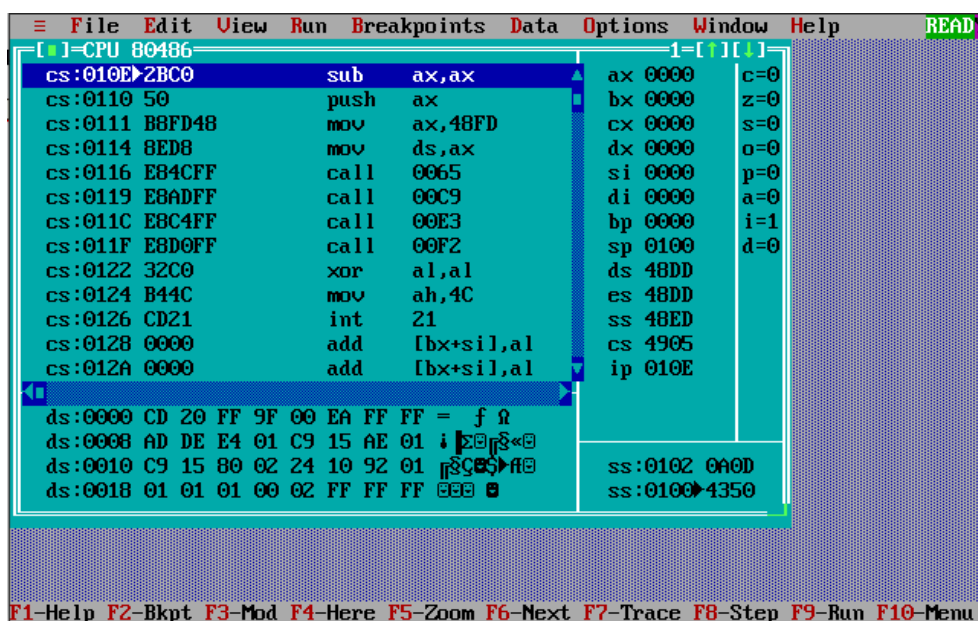


Рис. 8 - Отладчик "хорошего" EXE-модуля

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=4905) – на начало сегмента команд. В IP загружается смещение точки входа в программу, которая берётся из метки после директивы END. (Это иллюстрирует рисунок 8)

2. На что указывают регистры DS и ES?

- Регистры DS и ES указывают на начало PSP.

### 3. Как определяется стек?

- Стек определяется с помощью Stack Segment, после которой задаётся размер стека. При исполнении регистр SS указывает на начало сегмента стека, а SP на конца стека(его смещение).

### 4. Как определяется точка входа?

- Точка входа определяется при помощи директивы END.

### **Выводы.**

Были написаны COM и EXE модули, на основе которых производилось сравнение данных форматов. Также были выявлены недостатки и преимущества каждого из них.