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. 0381 _____ Странникова Н.С.

Преподаватель _____ Губкин А.Ф.

Дата выполнения: 11 февраля

Санкт-Петербург

2022

I. ПОСТАНОВКА ЗАДАЧИ

Цель работы.

Исследование различий в структурах исходных текстов модулей типов .COM и .EXE, структур файлов загрузочных модулей и способов их загрузки в основную память.

Основные сведения.

В работе используются следующие процедуры:

TETR_TO_HEX — переводит 10-ые цифры в символьный код.

BYTE_TO_HEX — переводит байт в 16 системе счисления в символьный код(данная функция используется, когда при определении типа PC ни один код не совпал с рассматриваемыми значениями).

WRD_TO_HEX – переводит слова, представленные в 16 сс, в символьный код.

BYTE_TO_DEC – переводит байт, представленный в 16 сс, в символьный код, представленный в 10-ой сс.

Данные процедуры были взяты из раздела «общие сведения». Так же для корректной работы программы были написаны следующие процедуры:

OUTPUT – выводит значения в консоль.

GET_PC_TYPE – получает информацию о типе IBM PC, который хранится в байте по адресу 0F000:0FFFEh. После этого значения AL сравниваются с значениями из таблицы (см. ниже) для получения нужного типа.

GET_OS_VERSION – определяет версию MS DOS. Для этого используется функция 30H прерывания 21H.

Таблица соответствия кода и типа IBM PC:

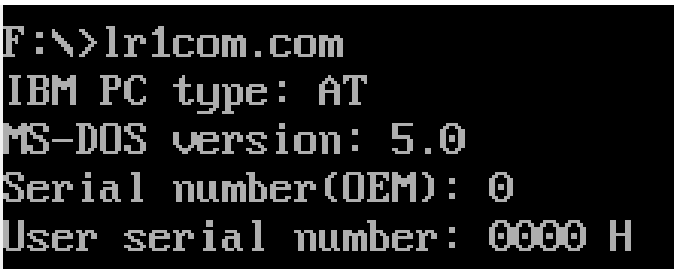
Тип IBM PC	Код
PC	FF
PC/XT	FE, FB
AT	FC
PS2 модель 30	FA
PS2 модель 50 или 60	FC
PS2 модель 80	F8
PCjr	FD
PC Convertible	F9

Так же были объявлены строки для вывода информации:

- TYPE_PC db 'IBM PC type: PC',0DH,0AH,'$'
- TYPE_PC_XT db 'IBM PC type: PC/XT',0DH,0AH,'$'
- TYPE_AT db 'IBM PC type: AT',0DH,0AH,'$'
- TYPE_PS2_30 db 'IBM PC type: PS2 model 30',0DH,0AH,'$'
- TYPE_PS2_50_OR_60 db 'IBM PC type: PS2 model 50 or 60',0DH,0AH,'$'
- TYPE_PS2_80 db 'IBM PC type: PS2 model 80',0DH,0AH,'$'
- TYPE_PCJR db 'IBM PC type: PCjr',0DH,0AH,'$'
- TYPE_PC_CONVERTIBLE db 'IBM PC type: PC Convertible',0DH,0AH,'$'
- VERSION db 'MS-DOS version: . ',0DH,0AH,'$'
- SERIAL_NUMBER db 'Serial number(OEM): ',0DH,0AH,'$'
- USER_NUMBER db 'User serial number: H $'

Выполнение работы.

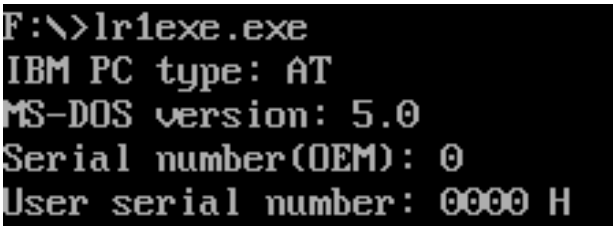
1) Был написан и отлажен исходный .COM модуль, который определяет тип PC и версию системы. После запуска программы в DOSBOX вывелось следующее:



Это «хороший» .COM модуль.

Из исходного текста для .COM модуля был построен «плохой» .EXE модуль:

2) Далее был написан текст исходного .EXE модуля, который выполняет те же функции, что и модуль из пункта 1). Для этого был добавлен сегмент стека, отделён сегмент данных и сегмент кода. Таким образом, получился «хороший» .EXE модуль:



3-6) Ответы на вопросы см. в разделе «Вопросы».

Вывод.

Были исследованы различия в структурах исходных текстов модулей типов .COM и .EXE, структур файлов загрузочных модулей и способов их загрузки в основную память.

II. ВОПРОСЫ

1. Отличия исходных текстов .COM и .EXE программ:

1) Сколько сегментов должна содержать COM-программа?

– COM-программа должна содержать один сегмент, так как стек генерируется автоматически, а код и данные не разделяются на разные сегменты, располагаясь в одном.

2) Сколько сегментов должна содержать EXE-программа?

– EXE-программа должна содержать 3 сегмента: сегмент кода, сегмент данных, сегмент стека(можно не объявлять, тогда будет использоваться стек DOS). Получается, что EXE-программа должна содержать не менее одного сегмента.

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

– Директива ORG 100h(обеспечение смещения в 245 байт от нулевого адреса, чтобы не попасть в область PSP) и ASSUME(позволяет сегменту данных и сегменту кода указывать на один общий сегмент).

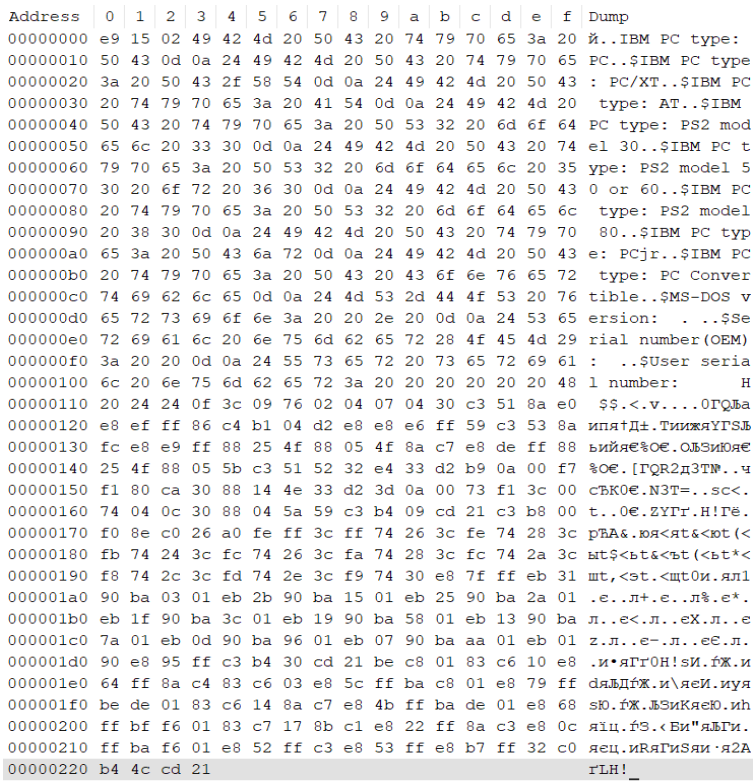
4) Все ли форматы команд можно использовать в COM-программе?

– Нет. Не могут быть выполнены команды с указанием сегментов, так как отсутствует таблица настроек(relocation table). То есть при загрузке программы к каждому сегментному адресу прибавляется значение начального сегмента программы, так как некоторые команды требуют указания не только смещения, но и сегмента адреса. Для получения расположения подобных элементов система использует таблицу настроек, которая находится в файле по некоторому смещению от его начала(само смещение хранится в заголовке в байтах 18-19h). Поэтому в COM-программе не могут быть выполнены команды с указанием сегментов из-за отсутствия таблицы настроек. Сама таблица состоит из нескольких блоков, каждый из которых записывает адрес в память. Таблица настроек используется для изменения адреса памяти, когда программа загружается в память.

2. Отличия форматов файлов .COM и .EXE модулей:

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

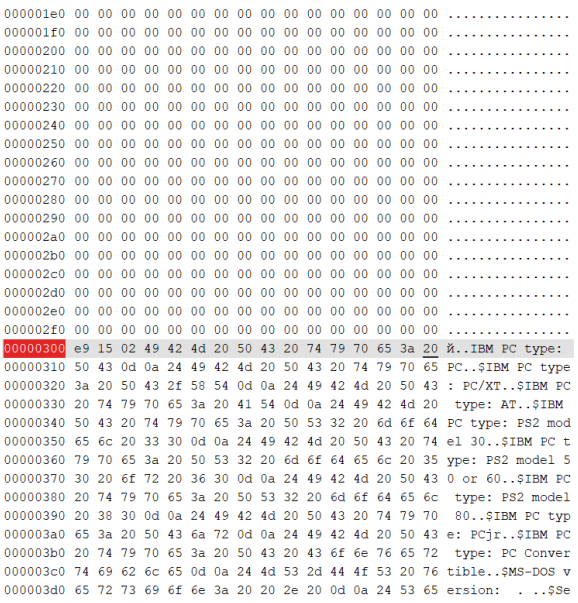
– Файл COM состоит из одного сегмента, который включает в себя код и данные. Сегмент стека генерируется автоматически. Так же COM файл не превышает размера в 64 Кб. Код располагается с адреса 100h, так как при загрузке модуля устанавливается смещение в 100h, поэтому адрес передвигается с 0h на 100h.



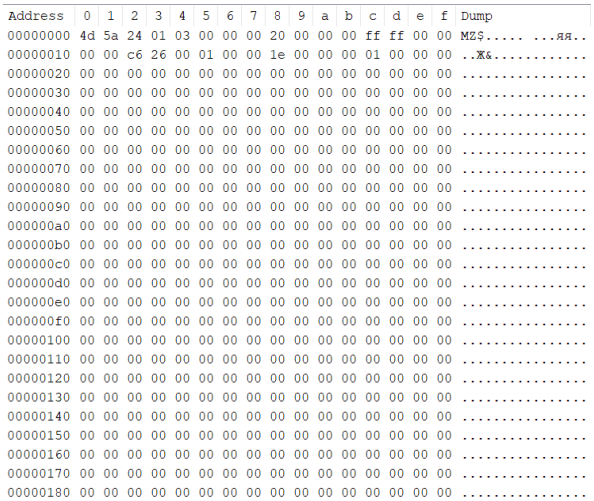
```
Address  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f  Dump
00000000 e9 15 02 49 42 4d 20 50 43 20 74 79 70 65 3a 20 й..IBM PC type:
00000010 50 43 0d 0a 24 49 42 4d 20 50 43 20 74 79 70 65 PC..$IBM PC type
00000020 3a 20 50 43 2f 58 54 0d 0a 24 49 42 4d 20 50 43 : PC/XT..$IBM PC
00000030 20 74 79 70 65 3a 20 41 54 0d 0a 24 49 42 4d 20  type: AT..$IBM
00000040 50 43 20 74 79 70 65 3a 20 50 53 32 20 6d 6f 64 PC type: PS2 mod
00000050 65 6c 20 33 30 0d 0a 24 49 42 4d 20 50 43 20 74 el 30..$IBM PC t
00000060 79 70 65 3a 20 50 53 32 20 6d 6f 64 65 6c 20 35 ype: PS2 model 5
00000070 30 20 6f 72 20 36 30 0d 0a 24 49 42 4d 20 50 43 0 or 60..$IBM PC
00000080 20 74 79 70 65 3a 20 50 53 32 20 6d 6f 64 65 6c  type: PS2 model
00000090 20 38 30 0d 0a 24 49 42 4d 20 50 43 20 74 79 70  80..$IBM PC typ
000000a0 65 3a 20 50 43 6a 72 0d 0a 24 49 42 4d 20 50 43 e: PCjr..$IBM PC
000000b0 20 74 79 70 65 3a 20 50 43 20 43 6f 6e 76 65 72  type: PC Conver
000000c0 74 69 62 6c 65 0d 0a 24 4d 53 2d 44 4f 53 20 76 tible..$MS-DOS v
000000d0 65 72 73 69 6f 6e 3a 20 20 2e 20 0d 0a 24 53 65 ersion:  . ..$Se
000000e0 72 69 61 6c 20 6e 75 6d 62 65 72 28 4f 45 4d 29 rial number(OEM)
000000f0 3a 20 20 0d 0a 24 55 73 65 72 20 73 65 72 69 61 :  ..$User seria
00000100 6c 20 6e 75 6d 62 65 72 3a 20 20 20 20 20 20 48 l number:      H
00000110 20 24 24 0f 3c 09 76 02 04 07 04 30 c3 51 8a e0  $$.<.v....0ГQЉа
00000120 e8 ef ff 86 c4 b1 04 d2 e8 e8 e6 ff 59 c3 53 8a ипят‡Д±.ТиижяYГSЉ
00000130 fc e8 e9 ff 88 25 4f 88 05 4f 8a c7 e8 de ff 88 ьийяЂ%О€.ОЉЗиЮяЂ
00000140 25 4f 88 05 5b c3 51 52 32 e4 33 d2 b9 0a 00 f7 %О€.[ГQR2д3ТЉ..ч
00000150 f1 80 ca 30 88 14 4e 33 d2 3d 0a 00 73 f1 3c 00 сЪК0€.N3Т=..sc<.
00000160 74 04 0c 30 88 04 5a 59 c3 b4 09 cd 21 c3 b8 00 t..0€.ZYГґ.Н!Гё.
00000170 f0 8e c0 26 a0 fe ff 3c ff 74 26 3c fe 74 28 3c рЋА&.юя<яt&<юt(<
00000180 fb 74 24 3c fc 74 26 3c fa 74 28 3c fc 74 2a 3c ыt$<ьt&<ъt(<ьt*<
00000190 f8 74 2c 3c fd 74 2e 3c f9 74 30 e8 7f ff eb 31 шt,<эt.<щt0и.ял1
000001a0 90 ba 03 01 eb 2b 90 ba 15 01 eb 25 90 ba 2a 01 .е..л+.е..л%.е*.
000001b0 eb 1f 90 ba 3c 01 eb 19 90 ba 58 01 eb 13 90 ba л..е<.л..еX.л..е
000001c0 7a 01 eb 0d 90 ba 96 01 eb 07 90 ba aa 01 eb 01 z.л..е-.л..еЄ.л.
000001d0 90 e8 95 ff c3 b4 30 cd 21 be c8 01 83 c6 10 e8 .и•яГґ0Н!sИ.ѓЖ.и
000001e0 64 ff 8a c4 83 c6 03 e8 5c ff ba c8 01 e8 79 ff dяЊДѓЖ.и\яєИ.иуя
000001f0 be de 01 83 c6 14 8a c7 e8 4b ff ba de 01 e8 68 sЮ.ѓЖ.ЉЗиКяєЮ.ин
00000200 ff bf f6 01 83 c7 17 8b c1 e8 22 ff 8a c3 e8 0c яїц.ѓЗ.‹Би"яЊГи.
00000210 ff ba f6 01 e8 52 ff c3 e8 53 ff e8 b7 ff 32 c0 яєц.иRяГиSяи·я2А
00000220 b4 4c cd 21                                     ґLН!
```

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

– В таком файле код и данные располагаются в одном сегменте, однако это неправильно, так как .EXE файл требует, чтобы сегмент кода и сегмент данных разделялись. Код располагается с адреса 300h. С адреса 0h располагается заголовок(информация для загрузчика) данного модуля. MZ является частью заголовка(00h-01h). Адрес первого элемента таблицы настроек адресов относительно начала файла начинается с 18h.



```
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 15 02 49 42 4d 20 50 43 20 74 79 70 65 3a 20 й..IBM PC type:
00000310 50 43 0d 0a 24 49 42 4d 20 50 43 20 74 79 70 65 PC..$IBM PC type
00000320 3a 20 50 43 2f 58 54 0d 0a 24 49 42 4d 20 50 43 : PC/XT..$IBM PC
00000330 20 74 79 70 65 3a 20 41 54 0d 0a 24 49 42 4d 20  type: AT..$IBM
00000340 50 43 20 74 79 70 65 3a 20 50 53 32 20 6d 6f 64 PC type: PS2 mod
00000350 65 6c 20 33 30 0d 0a 24 49 42 4d 20 50 43 20 74 el 30..$IBM PC t
00000360 79 70 65 3a 20 50 53 32 20 6d 6f 64 65 6c 20 35 ype: PS2 model 5
00000370 30 20 6f 72 20 36 30 0d 0a 24 49 42 4d 20 50 43 0 or 60..$IBM PC
00000380 20 74 79 70 65 3a 20 50 53 32 20 6d 6f 64 65 6c  type: PS2 model
00000390 20 38 30 0d 0a 24 49 42 4d 20 50 43 20 74 79 70  80..$IBM PC typ
000003a0 65 3a 20 50 43 6a 72 0d 0a 24 49 42 4d 20 50 43 e: PCjr..$IBM PC
000003b0 20 74 79 70 65 3a 20 50 43 20 43 6f 6e 76 65 72  type: PC Conver
000003c0 74 69 62 6c 65 0d 0a 24 4d 53 2d 44 4f 53 20 76 tible..$MS-DOS v
000003d0 65 72 73 69 6f 6e 3a 20 20 2e 20 0d 0a 24 53 65 ersion:  . ..$Se
```



```
Address  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f  Dump
00000000 4d 5a 24 01 03 00 00 00 20 00 00 00 ff ff 00 00 MZ$..... ...яя..
00000010 00 00 c6 26 00 01 00 00 1e 00 00 00 01 00 00 00 ..Ж&............
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 ................
```

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

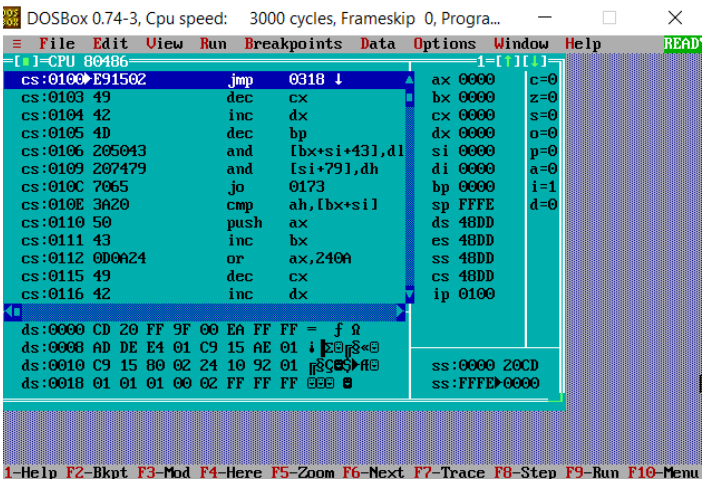
– В «хорошем» .EXE файле сегменты разделены на сегмент кода, сегмент стека, сегмент данных. В «плохом» .EXE файле код и данные располагаются в одном сегменте. Программа «хорошего» .EXE файла может иметь любой размер. Так же EXE-файл имеет заголовок, который используется при его загрузке. Заголовок состоит из форматированной части, содержащей сигнатуру и данные, необходимые для загрузки EXE-файла, и таблицы для настройки адресов. Так как «плохой» .EXE файл был построен из .COM файла, то в нём адресация кода начинается с 300h(изначально сегмент данных смещён на 100h + размер заголовка). В «хорошем» .EXE файле смещается код из-за выделения памяти под стек(16 байт → 200h) и под данные(220h).

```
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 49 42 4d 20 50 43 20 74 79 70 65 3a 20 50 43 0d IBM PC type: PC.
00000230 0a 24 49 42 4d 20 50 43 20 74 79 70 65 3a 20 50 .$IBM PC type: P
00000240 43 2f 58 54 0d 0a 24 49 42 4d 20 50 43 20 74 79 C/XT..$IBM PC ty
00000250 70 65 3a 20 41 54 0d 0a 24 49 42 4d 20 50 43 20 pe: AT..$IBM PC
00000260 74 79 70 65 3a 20 50 53 32 20 6d 6f 64 65 6c 20 type: PS2 model
00000270 33 30 0d 0a 24 49 42 4d 20 50 43 20 74 79 70 65 30..$IBM PC type
00000280 3a 20 50 53 32 20 6d 6f 64 65 6c 20 35 30 20 6f : PS2 model 50 o
00000290 72 20 36 30 0d 0a 24 49 42 4d 20 50 43 20 74 79 r 60..$IBM PC ty
000002a0 70 65 3a 20 50 53 32 20 6d 6f 64 65 6c 20 38 30 pe: PS2 model 80
000002b0 0d 0a 24 49 42 4d 20 50 43 20 74 79 70 65 3a 20 ..$IBM PC type:
000002c0 50 43 6a 72 0d 0a 24 49 42 4d 20 50 43 20 74 79 PCjr..$IBM PC ty
000002d0 70 65 3a 20 50 43 20 43 6f 6e 76 65 72 74 69 62 pe: PC Convertib
000002e0 6c 65 0d 0a 24 4d 53 2d 44 4f 53 20 76 65 72 73 le..$MS-DOS vers
000002f0 69 6f 6e 3a 20 20 2e 20 0d 0a 24 53 65 72 69 61 ion:  . ..$Seria
00000300 6c 20 6e 75 6d 62 65 72 28 4f 45 4d 29 3a 20 20 l number(OEM):
00000310 0d 0a 24 55 73 65 72 20 73 65 72 69 61 6c 20 6e ..$User serial n
00000320 75 6d 62 65 72 3a 20 20 20 20 20 20 48 20 24 00 umber:      H $.
```

3. Загрузка .COM модуля в основную память:

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

– Определяется сегментный адрес участка ОП, у которого достаточно места для загрузки программы, образ COM-файла считывается с диска и помещается в память, начиная с PSP:0100h. После загрузки двоичного образа COM-программы сегментные регистры CS, DS, ES и SS указывают на PSP(на рисунке видно, что в данном случае сегментные регистры указывают на 48DD), SP указывает на конец сегмента PSP(обычно FFFE), слово 00H помещено в стек, IP содержит 100H в результате команды JMP PSP:100H.



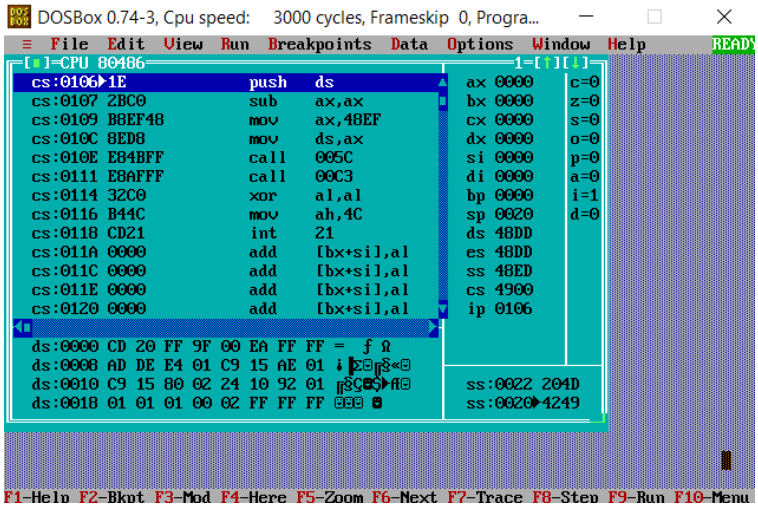
2) Что располагается с адреса 0?
– Сегмент PSP.

3) Какие значения имеют сегментные регистры? На какие области памяти они указывают?
– В начале выполнения программы все сегментные регистры имеют значение 48DD, так как они все указывают на PSP. Сегментные регистры CS, DS, ES, SS указывают на сегмент кода, сегмент данных, дополнительные данные, стек.

4) Как определяется стек? Какую область памяти он занимает? Какие адреса?
– Стек в .COM файле генерируется автоматически при создании программы. Регистр SS=0h указывает на начало стека, регистр SP=FFFEh указывает на конец стека. Таким образом, адреса стека располагаются в диапазоне от 0h до FFFEh.

4. Загрузка «хорошего» .EXE модуля в основную память

1) Как загружается «хороший» EXE? Какие значения имеют сегментные регистры?
– Сначала определяется сегментный адрес свободного участка памяти, размер которого достаточен для размещения программы. Создаётся блок памяти для PSP и программы, заполняются поля PSP соответствующими значениями. Далее считывается стандартная часть заголовка в память. После этого определяется длина тела загрузочного модуля(разность длины файла 04-07 и длины заголовка 08-09 + число байт в последнем блоке 02-03). В зависимости от признака, указывающего загружать задачу в конец памяти или в начало, определяются сегментный адрес для загрузки. Далее загрузочный модуль считывается в начальный сегмент(это тот сегмент, адрес которого был определен ранее). После этого таблица настройки считывается в рабочую память. Для каждого элемента этой таблицы к полю сегмента прибавляется сегментный адрес начального сегмента. Таким образом, элементы таблицы указывают на нужные слова(к ним так же прибавляется сегментный адрес начального сегмента) в памяти. Далее регистрам SS и SP придаются значения, указанные в заголовке, к SS прибавляется сегментный адрес начального сегмента. В ES и DS засылается сегментный адрес начала PSP. Управление передаётся загруженной задаче по адресу, указанному в заголовке (байты 14-17).



2) На что указывают регистры DS и ES?
– Регистры DS и ES указывают на начало сегмента PSP.

3) Как определяется стек?
– Стек определяется с помощью директивы .STACK, после которой задаётся размер стека.

4) Как определяется точка входа?
– Точка входа определяется директивой END.