

**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. 7381

\_\_\_\_\_

Кушкочева А.О.

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2019

### **Цель работы.**

Исследование различий в структурах исходных текстов модулей .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 0F000:0FFFE, в предпоследнем байте ROM BIOS. Соответствие кода и типа в таблице:

<b>PC</b>	<b>FF</b>
<b>PC/XT</b>	<b>FE,FB</b>
<b>AT</b>	<b>FC</b>
<b>PS2 модель 30</b>	<b>FA</b>
<b>PS2 модель 50 или 60</b>	<b>FC</b>
<b>PS2 модель 80</b>	<b>F8</b>
<b>PCjr</b>	<b>FD</b>
<b>PC Convertible</b>	<b>F9</b>

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:CH – 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 – Используется для перевода половины байта в шестнадцатеричную систему счисления.

BYTE\_TO\_HEX – Используется для перевода байта регистра AL в шестнадцатеричную систему счисления, помещая результат в AX.

WRD\_TO\_HEX – Используется для перевода двух байт регистра AX в шестнадцатеричную систему счисления, помещая результат в регистр DI.

BYTE\_TO\_DEC – Используется для перевода байта регистра AL в десятичную систему счисления, помещая результат в SI.

TYPE\_PC – Определяет тип IBM PC.

MS\_DOS\_VER – Определяет версию MS DOS, серийный номер OEM и серийный номер пользователя.

## Структуры данных.

Таблица 1 – Структуры данных

Название поля данных	Тип	Назначение
PC_TYPE	db	Тип IBM PC
PC	db	PC
PCXT	db	PC/XT
AT	db	AT
PS230	db	PS2 модель 30
PS250	db	PS2 модель 50 или 60
PS280	db	PS2 модель 80
PCJR	db	PCJR
PC_CONVERT	db	PC Convertible
MS_DOS_VERSION	db	Номер версии MS DOS
OEM	db	Серийный номер OEM
USER_NUM	db	Серийный номер пользователя

## Ход работы.

**Шаг 1.** Запуск «хорошего» .COM модуля.

```
C:\>lab1bad.com
PC type: AT
MS-DOS:  5.00
OEM:  FF
USER NUMBER:  000000h
```

Рисунок 1 – «Хороший» .COM модуль

Запуск «плохого» .EXE модуля.

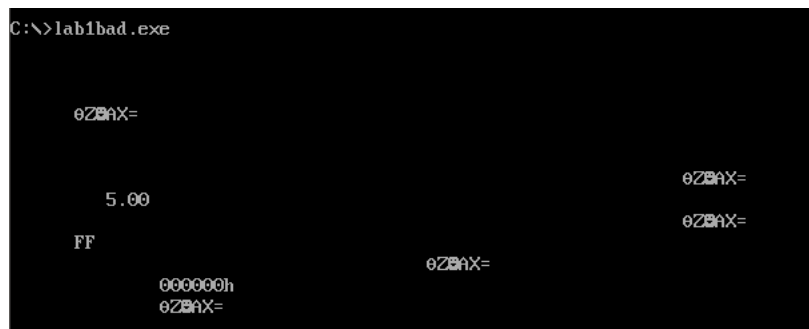


Рисунок 2 – «Плохой» .EXE модуль

## Шаг 2. Запуск «хорошего» .EXE модуля.

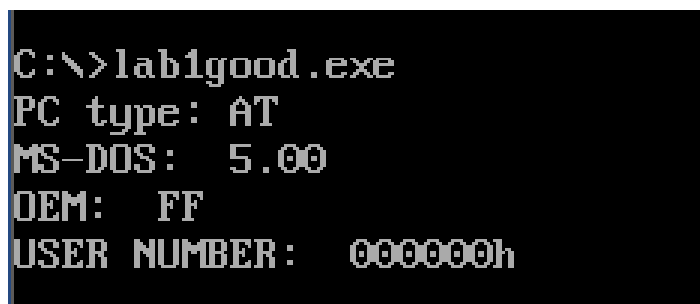


Рисунок 3 – «Хороший» .EXE модуль

**Шаг 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?**

Директива ORG 100h (смещение 100h), так как при загрузке COM-файла в память DOS занимает первые 256 байт (100h) блоком данных PSP и располагает код программы только после этого блока. Директива ASSUME, ставящая в соответствие начало программы сегментам кода и данных.

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

Нет, не все, так как в отличие от EXE-программы, в которой существует таблица настроек (таблица разметки), называемая Relocation Table, COM-

программа ею не располагает. Адреса сегментов определяются загрузчиком в момент запуска программы на основе информации о местоположении полей адресов в файле из Relocation Table. Следовательно, в связи с отсутствием этой таблицы в COM-программах, команды вида mov [регистр], seg [сегмент] недопустимы.

#### Шаг 4. .COM модуль в шестнадцатеричном виде.

C:\MASM\LAB1_1.COM																	
0000000000:	E9	80	01	92	A8	AF	20	49	4D	42	20	50	43	20	24	50	щАТип IMB PC \$P
0000000010:	43	0D	0A	24	50	43	2F	58	54	0D	0A	24	41	54	0D	0A	С \$PC/XT \$AT \$
0000000020:	24	50	53	32	20	AC	AE	A4	A5	AB	EC	20	33	30	0D	0A	\$PS2 модель 30 \$
0000000030:	24	50	53	32	20	AC	AE	A4	A5	AB	EC	20	35	30	20	A8	\$PS2 модель 50 и
0000000040:	AB	A8	20	36	30	0D	0A	24	50	53	32	20	AC	AE	A4	A5	ли 60 \$PS2 моде
0000000050:	AB	EC	20	38	30	0D	0A	24	50	43	6A	72	0D	0A	24	50	ль 80 \$PCjr \$P
0000000060:	43	20	43	6F	6E	76	65	72	74	69	62	6C	65	0D	0A	24	С Convertible \$
0000000070:	8D	AE	AC	A5	E0	20	A2	A5	E0	E1	A8	A8	20	4D	53	20	Номер версии MS
0000000080:	44	4F	53	3A	20	20	2E	20	20	20	20	0D	0A	24	91	A5	DOS: . \$Ce
0000000090:	E0	A8	A9	AD	EB	A9	20	AD	AE	AC	A5	E0	20	4F	45	4D	рийный номер OEM
00000000A0:	3A	20	20	20	20	0D	0A	24	91	A5	E0	A8	A9	AD	EB	A9	: \$Серийный
00000000B0:	20	AD	AE	AC	A5	E0	20	AF	AE	AB	EC	A7	AE	A2	A0	E2	номер пользоват
00000000C0:	A5	AB	EF	3A	20	20	20	20	20	20	0D	0A	24	24	0F	3C	еля: \$\$\$<
00000000D0:	09	76	02	04	07	04	30	C3	51	8A	E0	E8	EF	FF	86	C4	в\$QKршя Ж-
00000000E0:	B1	04	D2	E8	E8	E6	FF	59	C3	53	8A	FC	E8	E9	FF	88	тшщ Y SKNшщ И
00000000F0:	25	4F	88	05	4F	8A	C7	E8	DE	FF	88	25	4F	88	05	5B	%OIOK ш I %OIO[
0000000100:	C3	51	52	32	E4	33	D2	B9	0A	00	F7	F1	80	CA	30	88	QR2фЗт \$ üäA-ои
0000000110:	14	4E	33	D2	3D	0A	00	73	F1	3C	00	74	04	0C	30	88	IN3т=\$ sè< t\$Oи
0000000120:	04	5A	59	C3	1E	B8	00	F0	8E	D8	2B	DB	B7	FE	1F	C3	ZY \$ĚO+ \$ \$ \$
0000000130:	50	56	BE	70	01	83	C6	15	E8	C6	FF	BE	70	01	83	C6	PV p\$Г \$ш  \$p\$Г
0000000140:	17	8A	C4	E8	BB	FF	5E	58	C3	50	53	56	BE	8E	01	83	\$K-ш ^X PSV O\$Г
0000000150:	C6	16	8A	C7	E8	AA	FF	5E	5B	58	C3	53	51	57	50	BF	\$K шк ^X SQWP
0000000160:	A8	01	83	C7	22	8B	C1	E8	7F	FF	8A	C3	E8	69	FF	BF	и\$Г "л-ш \$ K wi
0000000170:	A8	01	83	C7	1D	89	05	58	5F	59	5B	C3	50	B4	09	CD	и\$Г \$Й\$X_Y \$P \$=
0000000180:	21	58	C3	E8	9E	FF	BA	03	01	E8	F0	FF	BA	0F	01	80	!X ш0   \$шĚ   \$A
0000000190:	FF	FF	74	47	BA	14	01	80	FF	FE	74	3F	BA	14	01	80	tG   \$A ■t?   \$A
00000001A0:	FF	FB	74	37	BA	1C	01	80	FF	FC	74	2F	BA	21	01	80	√t7   \$A №t/ ! \$A
00000001B0:	FF	FA	74	27	BA	31	01	80	FF	FC	74	1F	BA	48	01	80	·t' 1 \$A №t\$ H \$A
00000001C0:	FF	F8	74	17	BA	58	01	80	FF	FD	74	0F	BA	5F	01	80	°t\$ X \$A \$t\$ _ \$A
00000001D0:	FF	F9	74	07	8A	C7	E8	FF	FE	8B	D0	E8	9E	FF	B4	30	·t\$K ш ■лш0  θ
00000001E0:	CD	21	E8	4B	FF	E8	61	FF	E8	70	FF	BA	70	01	E8	8B	=!шK ша шp   p\$шл
00000001F0:	FF	BA	8E	01	E8	85	FF	BA	A8	01	E8	7F	FF	32	C0	B4	O\$шE   и\$ш\$ 2L
0000000200:	4C	CD	21														L=!

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.





COM файл 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?**

В «плохом» EXE файле данные и код содержатся в одном сегменте. С 0 адреса располагается "подпись" компоновщика, указывающая, что файл является файлом EXE. Код располагается с адреса 300h.

MZ в начале EXE модуля это формат исполняемых файлов MS DOS, он не является частью таблицы настроек адресов, так как таблица состоит из элементов, число которых записано в байтах 06-07. Элемент таблицы настройки состоит из двух полей: 2-х байтного смещения и 2-х байтного сегмента, и указывает слова в загрузочном модуле, содержащее адрес, который должен быть настроен на место памяти, в которое загружается задача.

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

В «хорошем» файле EXE содержится информация для загрузчика, сегмент стека, сегмент данных и сегмент кода (3 сегмента вместо одного в «плохом» .EXE). Код располагается с адреса 200h в отличии от 300h в «плохом» .EXE файле.

**Шаг 5.** Загрузка COM модуля в основную память.

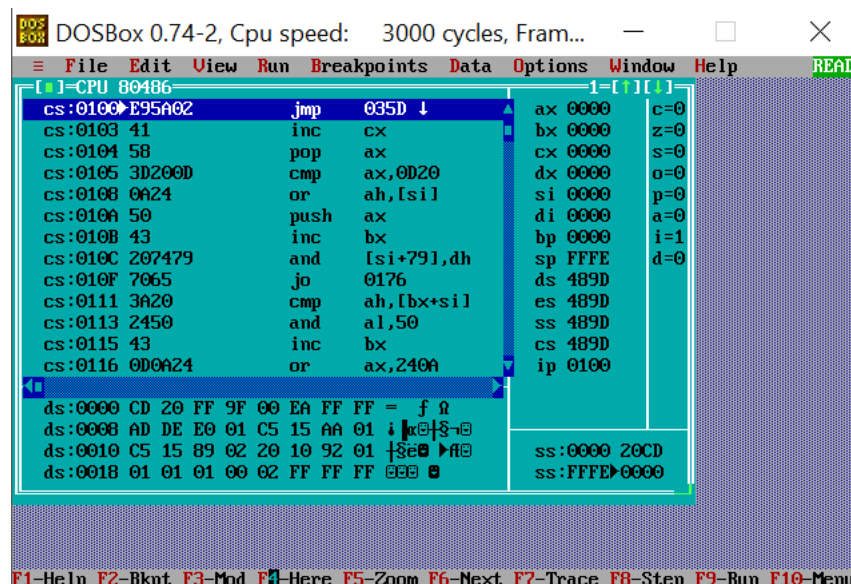


Рисунок 7 – Загрузка COM модуля в основную память

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

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

После загрузки COM-программы в память сегментные регистры указывают на начало PSP. Код располагается с адреса 100h (ip = 0100h).

- 2) **Что располагается с 0 адреса?**

Адрес начала PSP.

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

48DDh. Они указывают на начало PSP.

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

Стек определяется автоматически, указатель стека устанавливается на конец сегмента. Если для программы размер сегмента в 64КБ является достаточным, то DOS устанавливает в регистре SP адрес конца сегмента – FFFh. Адреса расположены в диапазоне 0000h-FFFh.

**Шаг 6.** Загрузка «хорошего» EXE модуля в память.

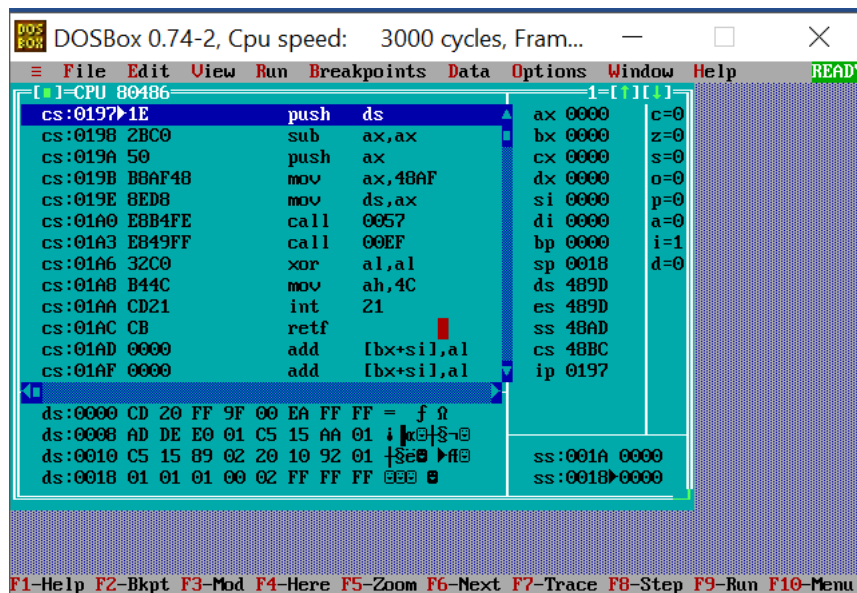


Рисунок 8 – Загрузка «хорошего» EXE модуля в память

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 устанавливаются на начало PSP, SS - на начало стека, CS - на начало сегмента кода.

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

DS и ES указывают на начало PSP. После выполнения команд `mov ax, @data` и `mov ds, ax` регистре DS содержит адрес начала сегмента данных.

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?**

При помощи команды `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.