

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра математического обеспечения и применения ЭВМ**

**ОТЧЕТ**  
**по лабораторной работе № 1**  
**по дисциплине «Операционные системы»**  
**Тема: Исследование структур загрузочных модулей**

Студент гр. 9383

Рыбников Р.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2021

### **Цель работы.**

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

### **Задача.**

Написать 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 и серийным номером пользователя. Полученные строки выводятся на экран.

Отладить полученный исходный модуль.

### **Функции программы.**

- TETR\_TO\_HEX: перевод десятичной цифры в код символа.
- BYTE\_TO\_HEX: перевод байта 16-й CC в символьный код.
- WRD\_TO\_HEX: перевод слова в 16-й CC в символьный код.
- BYTE\_TO\_DEC: перевод байта 16-й CC в десятичный код в 10-й CC.
- WRITESTRING: вывод строки на экран.
- PC\_TYPE: определение типа PC.
- OS\_VER: определение характеристик OS

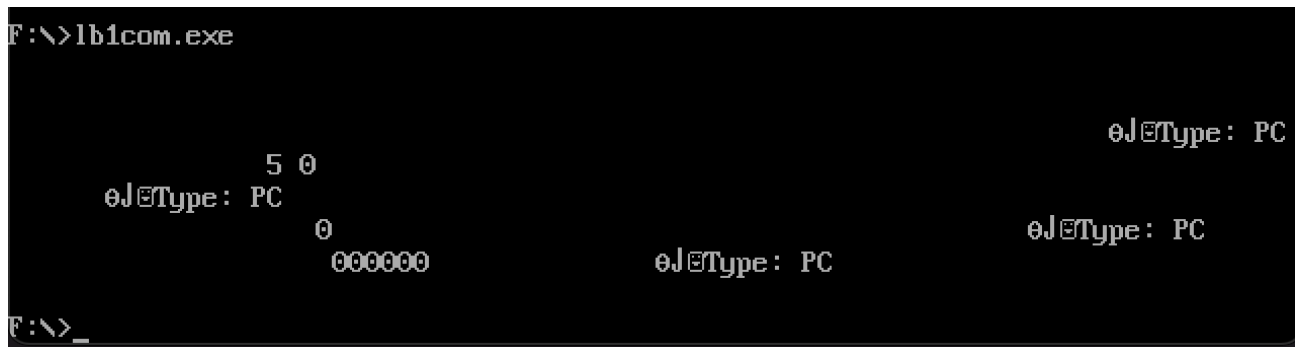
### Выполнение работы.

Была реализована программа для .COM модуля, из него получается некорректный .EXE модуль, а уже из некорректного .EXE модуля получается корректный .COM модуль.



```
F:\>lb1com.com
Type: AT
Version MS-DOS: 5.0
Serial number OEM: 0
User serial number: 000000H
F:\>
```

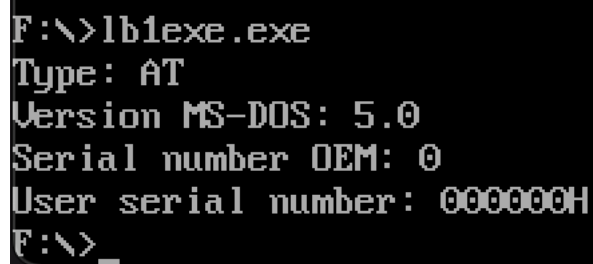
Рисунок 1 – корректный .COM модуль



```
F:\>lb1com.exe
                                     0J@Type: PC
5 0
0J@Type: PC
0
000000
0J@Type: PC
0J@Type: PC
F:\>_
```

Рисунок 2 – некорректный .EXE модуль

Так же была реализована программа для корректного .EXE модуля.



```
F:\>lb1exe.exe
Type: AT
Version MS-DOS: 5.0
Serial number OEM: 0
User serial number: 000000H
F:\>_
```

Рисунок 3 – корректный .EXE модуль

Были даны ответы на вопросы:

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

1. Сколько сегментов должна содержать COM-программа?

Ровно один сегмент должна содержать COM-программа. Потому что код и данные располагаются в одном сегменте.

2. EXE-программа?

Не менее одного сегмента.

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

В тексте COM-программы должна присутствовать директива `ORG 100h`, из-за того, что первые 100h байт занимает PSP, ASSUME, для того чтобы сегмент данных и сегмент кода указывали на общий сегмент.

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

Нет. Нельзя использовать команды вида `mov регистр, seg имя_сегмента`, т.к. отсутствует таблица настроек.

На рисунке ниже .COM модуль представлен в 16-м виде, а ниже него представление корректного и некорректного .EXE модуля в 16-м виде.

192-168-0-104:job\_asm ramka178rus\$ hexyl lb1com.com

00000000	e9 f5 01 54 79 70 65 3a	20 50 43 0d 0a 24 54 79	xxType: PC__\$Type
00000010	70 65 3a 20 50 43 2f 58	54 0d 0a 24 54 79 70 65	pe: PC/X T__\$Type
00000020	3a 20 41 54 0d 0a 24 54	79 70 65 3a 20 50 53 32	: AT__\$Type: PS2
00000030	20 d0 bc d0 be d0 b4 d0	b5 d0 bb d1 8c 20 33 30	xxxxxx xxx 30
00000040	0d 0a 24 54 79 70 65 3a	20 50 53 32 20 d0 bc d0	__\$Type: PS2 xxx
00000050	be d0 b4 d0 b5 d0 bb d1	8c 20 35 30 20 d0 b8 d0	xxxxxxx x 50 xxx
00000060	bb d0 b8 20 36 30 0d 0a	24 54 79 70 65 3a 20 50	xxx 60__\$Type: P
00000070	53 32 20 d0 bc d0 be d0	b4 d0 b5 d0 bb d1 8c 20	S2 xxxxx xxxxxxx
00000080	38 30 0d 0a 24 54 79 70	65 3a 20 50 d0 a1 6a 72	80__\$Type: Pxxjr
00000090	0d 0a 24 54 79 70 65 3a	20 50 43 20 43 6f 6e 76	__\$Type: PC Conv
000000a0	65 72 74 69 62 6c 65 0d	0a 24 56 65 72 73 69 6f	ertible__\$Versio
000000b0	6e 20 4d 53 2d 44 4f 53	3a 20 20 2e 20 20 0d 0a	n MS-DOS: . __
000000c0	24 53 65 72 69 61 6c 20	6e 75 6d 62 65 72 20 4f	\$Serial number 0
000000d0	45 4d 3a 20 20 0d 0a 24	55 73 65 72 20 73 65 72	EM: __\$ User ser
000000e0	69 61 6c 20 6e 75 6d 62	65 72 3a 20 20 20 20 20	ial numb er:
000000f0	20 20 48 20 24 24 0f 3c	09 76 02 04 07 04 30 c3	H \$\$<_v.....0x
00000100	51 8a e0 e8 ef ff 86 c4	b1 04 d2 e8 e8 e6 ff 59	Qxxxxxxxx xxxxxxY
00000110	c3 53 8a fc e8 e9 ff 88	25 4f 88 05 4f 8a c7 e8	xSxxxxxxxx %0x0xxx
00000120	de ff 88 25 4f 88 05 5b	c3 51 52 32 e4 33 d2 b9	xxx%0x[ xQR2x3xx
00000130	0a 00 f7 f1 80 ca 30 88	14 4e 33 d2 3d 0a 00 73	_0xxxx0x N3x=_0s
00000140	f1 3c 00 74 04 0c 30 88	04 5a 59 c3 b4 09 cd 21	x<0t0x ZYxx_x!
00000150	c3 b8 00 f0 8e c0 26 a0	fe ff 3c ff 74 1c 3c fe	xx0xxx&xx xx<xt0<x
00000160	74 1e 3c fb 74 1a 3c fc	74 1c 3c fa 74 1e 3c f8	t0<xt0<x t0<xt0<x
00000170	74 26 3c fd 74 28 3c f9	74 2a ba 03 01 eb 2b 90	t&<xt(<x t*xxx+x
00000180	ba 0e 01 eb 25 90 ba 1c	01 eb 1f 90 ba 27 01 eb	xxx%xxx xxx'0x
00000190	19 90 ba 43 01 eb 13 90	ba 69 01 eb 0d 90 ba 85	xxxCxxx xi0x_xxx
000001a0	01 eb 07 90 ba 93 01 eb	01 90 e8 9f ff c3 b4 30	xxxxxxxx xxxxxx0
000001b0	cd 21 50 be aa 01 83 c6	10 e8 6d ff 58 8a c4 83	x!Pxxx xxmxXxxx
000001c0	c6 03 e8 64 ff ba aa 01	e8 81 ff be c1 01 83 c6	xxdxxx xxxxxx
000001d0	13 8a c7 e8 53 ff ba c1	01 e8 70 ff bf d8 01 83	xxxxSxxx xpxxxx
000001e0	c7 19 8b c1 e8 2a ff 8a	c3 e8 14 ff 83 ef 02 89	xxxx*xxx xxxxxx
000001f0	05 ba d8 01 e8 55 ff c3	e8 56 ff e8 b0 ff 32 c0	xxxUxx xVxxx2x
00000200	b4 4c cd 21		xLx!

Рисунок 4 -- 16-й вид .COM модуля



192-168-0-104:job\_asm ramka178rus\$ hexyl lb1com.exe

00000000	4d 5a 04 01 03 00 00 00	20 00 00 00 ff ff 00 00	MZ.....000	000xx00
00000010	00 00 66 cc 00 01 00 00	1e 00 00 00 01 00 00 00	00fx0.00	.000.000
00000020	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	00000000	00000000
*				
00000300	e9 f5 01 54 79 70 65 3a	20 50 43 0d 0a 24 54 79	xx.Type: PC__\$Ty	
00000310	70 65 3a 20 50 43 2f 58	54 0d 0a 24 54 79 70 65	pe: PC/X T__\$Type	
00000320	3a 20 41 54 0d 0a 24 54	79 70 65 3a 20 50 53 32	: AT__\$T type: PS2	
00000330	20 d0 bc d0 be d0 b4 d0	b5 d0 bb d1 8c 20 33 30	xxxxxxx xxxxx 30	
00000340	0d 0a 24 54 79 70 65 3a	20 50 53 32 20 d0 bc d0	__\$Type: PS2 xxx	
00000350	be d0 b4 d0 b5 d0 bb d1	8c 20 35 30 20 d0 b8 d0	xxxxxxx x 50 xxx	
00000360	bb d0 b8 20 36 30 0d 0a	24 54 79 70 65 3a 20 50	xxx 60__\$Type: P	
00000370	53 32 20 d0 bc d0 be d0	b4 d0 b5 d0 bb d1 8c 20	S2 xxxxx xxxxxxx	
00000380	38 30 0d 0a 24 54 79 70	65 3a 20 50 d0 a1 6a 72	80__\$Type: Pxxjr	
00000390	0d 0a 24 54 79 70 65 3a	20 50 43 20 43 6f 6e 76	__\$Type: PC Conv	
000003a0	65 72 74 69 62 6c 65 0d	0a 24 56 65 72 73 69 6f	ertible__\$Versio	
000003b0	6e 20 4d 53 2d 44 4f 53	3a 20 20 2e 20 20 0d 0a	n MS-DOS: . __	
000003c0	24 53 65 72 69 61 6c 20	6e 75 6d 62 65 72 20 4f	\$Serial number 0	
000003d0	45 4d 3a 20 20 0d 0a 24	55 73 65 72 20 73 65 72	EM:__\$ User ser	
000003e0	69 61 6c 20 6e 75 6d 62	65 72 3a 20 20 20 20 20	ial numb er:	
000003f0	20 20 48 20 24 24 0f 3c	09 76 02 04 07 04 30 c3	H \$\$<_v.....0x	
00000400	51 8a e0 e8 ef ff 86 c4	b1 04 d2 e8 e8 e6 ff 59	Qxxxxxxxx xxxxxxY	
00000410	c3 53 8a fc e8 e9 ff 88	25 4f 88 05 4f 8a c7 e8	xSxxxxxxxx %Ox.0xxx	
00000420	de ff 88 25 4f 88 05 5b	c3 51 52 32 e4 33 d2 b9	xxx%Ox.[ xQR2x3xx	
00000430	0a 00 f7 f1 80 ca 30 88	14 4e 33 d2 3d 0a 00 73	_0xxxxx0x .N3x=_0s	
00000440	f1 3c 00 74 04 0c 30 88	04 5a 59 c3 b4 09 cd 21	x<0t.0x .ZYxx_x!	
00000450	c3 b8 00 f0 8e c0 26 a0	fe ff 3c ff 74 1c 3c fe	xx0xxx&x xx<xt.<x	
00000460	74 1e 3c fb 74 1a 3c fc	74 1c 3c fa 74 1e 3c f8	t.<xt.<x t.<xt.<x	
00000470	74 26 3c fd 74 28 3c f9	74 2a ba 03 01 eb 2b 90	t&<xt(<x t*xx.x+x	
00000480	ba 0e 01 eb 25 90 ba 1c	01 eb 1f 90 ba 27 01 eb	x.xx%xx. .xxx'!x	
00000490	19 90 ba 43 01 eb 13 90	ba 69 01 eb 0d 90 ba 85	.xxC.x.x xix_x_xxx	
000004a0	01 eb 07 90 ba 93 01 eb	01 90 e8 9f ff c3 b4 30	.x.xxxx.x .xxxxxx0	
000004b0	cd 21 50 be aa 01 83 c6	10 e8 6d ff 58 8a c4 83	x!Pxx.x xmxXxxx	
000004c0	c6 03 e8 64 ff ba aa 01	e8 81 ff be c1 01 83 c6	x.xdxxx. xxxxx.x	
000004d0	13 8a c7 e8 53 ff ba c1	01 e8 70 ff bf d8 01 83	.xxxSxxx .xpxxx.x	
000004e0	c7 19 8b c1 e8 2a ff 8a	c3 e8 14 ff 83 ef 02 89	x.xxxx*xx xx.xxxx.x	
000004f0	05 ba d8 01 e8 55 ff c3	e8 56 ff e8 b0 ff 32 c0	.xx.xUxx xVxxxx2x	
00000500	b4 4c cd 21		xLx!	

Рисунок 5 -- 16-й вид некорректного .EXE модуля



```

job_asm — -bash — 87x43
192-168-0-104:job_asm ramka178rus$ hexyl lb1exe.exe
00000000 4d 5a 17 01 03 00 01 00 20 00 00 00 ff ff 00 00 MZ.....0 000xx00
00000010 00 01 71 10 03 01 20 00 1e 00 00 00 01 00 07 01 0.q.... 0.000.0..
00000020 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0000000 000000000
00000030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0000000 000000000
*
00000300 54 79 70 65 3a 20 50 43 0d 0a 24 54 79 70 65 3a Type: PC __$Type:
00000310 20 50 43 2f 58 54 0d 0a 24 54 79 70 65 3a 20 41 PC/XT __$Type: A
00000320 54 0d 0a 24 54 79 70 65 3a 20 50 53 32 20 d0 bc T__$Type: PS2 xx
00000330 d0 be d0 b4 d0 b5 d0 bb d1 8c 20 33 30 d0 0a 24 xxxxxxxx xx 30__$
00000340 54 79 70 65 3a 20 50 53 32 20 d0 bc d0 be d0 b4 Type: PS 2 xxxxxx
00000350 d0 b5 d0 bb d1 8c 20 35 30 20 d0 b8 d0 bb d0 b8 xxxxxx 5 0 xxxxxx
00000360 20 36 30 0d 0a 24 54 79 70 65 3a 20 50 53 32 20 60__$Ty pe: PS2
00000370 d0 bc d0 be d0 b4 d0 b5 d0 bb d1 8c 20 38 30 d0 xxxxxxxx xxxx 80_
00000380 0a 24 54 79 70 65 3a 20 50 d0 a1 6a 72 0d 0a 24 __$Type: Pxxjr__$
00000390 54 79 70 65 3a 20 50 43 20 43 6f 6e 76 65 72 74 Type: PC Convert
000003a0 69 62 6c 65 0d 0a 24 56 65 72 73 69 6f 6e 20 4d ible__$V
000003b0 53 2d 44 4f 53 3a 20 20 2e 20 20 0d 0a 24 53 65 S-DOS: . __$Se
000003c0 72 69 61 6c 20 6e 75 6d 62 65 72 20 4f 45 4d 3a rial num ber OEM:
000003d0 20 20 0d 0a 24 55 73 65 72 20 73 65 72 69 61 6c __$Use r serial
000003e0 20 6e 75 6d 62 65 72 3a 20 20 20 20 20 20 20 48 number: H
000003f0 20 24 00 00 00 00 00 00 00 00 00 00 00 00 00 00 $000000 00000000
00000400 24 0f 3c 09 76 02 04 07 04 30 c3 51 8a e0 e8 ef $.<_v... .0xQxxxx
00000410 ff 86 c4 b1 04 d2 e8 e8 e6 ff 59 c3 53 8a fc e8 xxxx.xxx xxYxSxxx
00000420 e9 ff 88 25 4f 88 05 4f 8a c7 e8 de ff 88 25 4f xxx%0x.0 xxxxxx%0
00000430 88 05 5b c3 51 52 32 e4 33 d2 b9 0a 00 f7 f1 80 x.[xQR2x 3xx_0xxx
00000440 ca 30 88 14 4e 33 d2 3d 0a 00 73 f1 3c 00 74 04 x0x.N3x=_0sx<0t.
00000450 0c 30 88 04 5a 59 c3 b4 09 cd 21 c3 b8 00 f0 8e _0x.ZYxx _x!xx0xx
00000460 c0 26 a0 fe ff 3c ff 74 1c 3c fe 74 1e 3c fb 74 x&xxx<xt .<xt.<xt
00000470 1a 3c fc 74 1c 3c fa 74 1e 3c f8 74 26 3c fd 74 .<xt.<xt .<xt&<xt
00000480 28 3c f9 74 2a ba 00 00 eb 2b 90 ba 0b 00 eb 25 (<xt*x00 x+xx.0x%
00000490 90 ba 19 00 eb 1f 90 ba 24 00 eb 19 90 ba 40 00 xx.0x.xx $0x.xx00
000004a0 eb 13 90 ba 66 00 eb 0d 90 ba 82 00 eb 07 90 ba x.xx f0x_ xxx0x.xx
000004b0 90 00 eb 01 90 e8 9f ff c3 b4 30 cd 21 50 be a7 x0x.xxxx xx0x!Pxx
000004c0 00 83 c6 10 e8 6d ff 58 8a c4 83 c6 03 e8 64 ff 0xx.xmX xxxxx.xdx
000004d0 ba a7 00 e8 81 ff be be 00 83 c6 13 8a c7 e8 53 xx0xxxxx 0xx.xxxxS
000004e0 ff ba be 00 e8 70 ff bf d5 00 83 c7 19 8b c1 e8 xxx0xpxx x0xx.xxxx
000004f0 2a ff 8a c3 e8 14 ff 83 ef 02 89 05 ba d5 00 e8 *xxxx.xx x.x.xx0x
00000500 55 ff c3 2b c0 50 b8 10 00 8e d8 e8 4e ff e8 a8 Uxx+ xPx. 0xxxNxxx
00000510 ff 32 c0 b4 4c cd 21 x2xxLx!

```

Рисунок 6 -- 16-й вид корректного .EXE модуля

## Отличия форматов файлов .COM и .EXE программ:

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

COM-файл состоит из одного сегмента, состоящего из сегмент кода и сегмент данных. Код начинается с адреса 0h.

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

.EXE файл состоит из одного сегмента. Сегмент с кодом и сегмент с данными начинаются с адреса 300h. А с адреса 0h находится информация загрузчика, которая содержит таблицу настроек и заголовок.

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

Корректный .EXE файл состоит из: сегмента стека, сегмента кода и сегмента данных. Данный файл не имеет ограничений в размере. С адреса 0h в корректном .EXE файле располагается таблица настроек. Отличие корректного от некорректного .EXE в выделение памяти под стек между PSP и кодом.

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

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

Сначала идёт определение сегментного адреса участка ОП, который способен вместить загрузку программы, после этого образ COM-файла считывается с диска и помещается в память, начиная с адреса 100h в PSP. После COM-файла, все сегментные регистры указывают на PSP. SP устанавливается в конец PSP, 0000h помещается в стек, а в IP записывается 100h.

2. Что располагается с адреса 0?

PSP в 100h байт.

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

Сегментные регистры CS, DS, ES и SS указывают на PSP.



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

Стек генерируется автоматически при создании COM-программы. SS – указывает на начало PSP, регистр SP указывает на конец стека. Адреса стека расположен между адресами SS:0000H -- SS:FFFFh.

#### **Загрузка «хорошего» EXE модуля в основную память:**

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

EXE-файл загружается, начиная с адреса PSP:0100h. В процессе загрузки считывается информация заголовка EXE и выполняется перемещение адресов сегментов. В IP загружается смещение точки входа в программу. CS указывает на начало сегмента команд

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