

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. 8382

Щеглов А.С.

Преподаватель

Ефремов М.А.

Санкт-Петербург

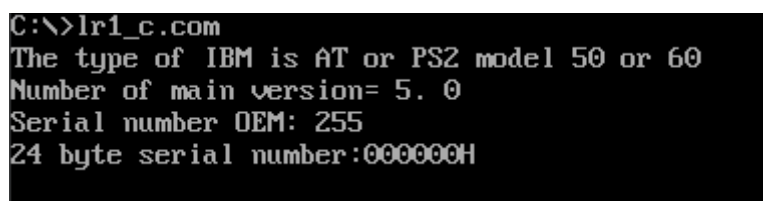
2020

Цель работы.

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

Ход работы.

Для определения типа PC и версии системы были написаны тексты .COM и .EXE модулей (см. Приложение А и В). Тип IBM PC хранится в байте по адресу 0F000:0FFFEh (предпоследний байт ROM BIOS). Для определения версии MS DOS была использована функция 30H прерывания 21H. После её вызова версия системы определяется значением регистров AL, AH. В регистре BH – серийный номер OEM, в BL:CH – 24-битовый серийный номер пользователя. Результат выполнения .COM модуля представлен на рис. 1. Результат выполнения «плохого» .EXE модуля, полученного из исходного текста для .COM модуля, представлен на рис. 2. Результат выполнения «хорошего» .EXE модуля представлен на рис. 3.



```
C:\>lr1_c.com
The type of IBM is AT or PS2 model 50 or 60
Number of main version= 5. 0
Serial number OEM: 255
24 byte serial number:0000000H
```

Рисунок 1. Результат работы lr1_c.com

```
C:\>lr1_c.exe
in version= .
5 0
0iNumber of main version= .
255
000000
0iNumber of main v
ersion= .
000000
0iNumber of main version= .
```

Рисунок 2. Результат работы lr1_c.exe

```
C:\>lr1_e.exe
The type of IBM is AT or PS2 model 50 or 60
Number of main version= 5. 0
Serial number OEM: 255
24 byte serial number:0000000H
```

Рисунок 3. Результат работы lr1_e.exe

Вид исходных файлов в шестнадцатеричном виде представлен на рисунках 4,5,6,7,8,9.

000000000:	E9 A1 01 4E 75 6D 62 65	72 20 6F 66 20 6D 61 69	é;@Number of mai
000000010:	6E 20 76 65 72 73 69 6F	6E 3D 20 20 2E 20 20 20	n version= .
000000020:	0D 0A 24 53 65 72 69 61	6C 20 6E 75 6D 62 65 72	Serial number
000000030:	20 4F 45 4D 3A 20 20 20	20 20 20 20 20 20 20 20	OEM:
000000040:	20 20 20 20 20 20 20 0D	0A 24 32 34 20 62 79 74	\$24 byt
000000050:	65 20 73 65 72 69 61 6C	20 6E 75 6D 62 65 72 3A	e serial number:
000000060:	20 20 20 20 20 20 48 0D	0A 24 54 68 65 20 74 79	H;\$The ty
000000070:	70 65 20 6F 66 20 49 42	4D 20 69 73 20 50 43 0D	pe of IBM is PC
000000080:	0A 24 54 68 65 20 74 79	70 65 20 6F 66 20 49 42	\$The type of IB
000000090:	4D 20 69 73 20 50 43 2F	58 54 0D 0A 24 54 68 65	M is PC/XT;\$The
0000000A0:	20 74 79 70 65 20 6F 66	20 49 42 4D 20 69 73 20	type of IBM is
0000000B0:	50 53 32 20 6D 6F 64 65	6C 20 33 30 0D 0A 24 54	PS2 model 30;\$T
0000000C0:	68 65 20 74 79 70 65 20	6F 66 20 49 42 4D 20 69	he type of IBM i
0000000D0:	73 20 41 54 20 6F 72 20	50 53 32 20 6D 6F 64 65	s AT or PS2 mode
0000000E0:	6C 20 35 30 20 6F 72 20	36 30 0D 0A 24 54 68 65	l 50 or 60;\$The
0000000F0:	20 74 79 70 65 20 6F 66	20 49 42 4D 20 69 73 20	type of IBM is
000000100:	50 53 32 20 6D 6F 64 65	6C 20 38 30 0D 0A 24 54	PS2 model 80;\$T
000000110:	68 65 20 74 79 70 65 20	6F 66 20 49 42 4D 20 69	he type of IBM i
000000120:	73 20 50 43 6A 72 0D 0A	24 54 68 65 20 74 79 70	s PCjr;\$The typ
000000130:	65 20 6F 66 20 49 42 4D	20 69 73 20 50 43 20 43	e of IBM is PC C
000000140:	6F 6E 76 65 72 74 69 62	6C 65 0D 0A 24 24 0F 3C	onvertible;\$<
000000150:	09 76 02 04 07 04 30 C3	51 8A E0 E8 EF FF 86 C4	ov\$ÀQ\$àèiÿtÄ
000000160:	B1 04 D2 E8 E8 E6 FF 59	C3 53 8A FC E8 E9 FF 88	±ÖèæÿVÄS\$üèÿ^
000000170:	25 4F 88 05 4F 8A C7 E8	DE FF 88 25 4F 88 05 5B	%0^*O\$Çèÿ^%0^*[
000000180:	C3 51 52 32 E4 33 D2 B9	0A 00 F7 F1 80 CA 30 88	ÄQR2ä30^÷ñÊ0^
000000190:	14 4E 33 D2 3D 0A 00 73	F1 3C 00 74 04 0C 30 88	¶N30= sñ< t*90^
0000001A0:	04 5A 59 C3 B8 00 F0 8E	C0 26 A0 FE FF 3C FF 74	♦ZYÄ, ðŽÄ& þÿ<ÿt
0000001B0:	1C 3C FE 74 22 3C FB 74	1E 3C FC 74 2E 3C FA 74	L<þt"<ût▲<üt.<üt
0000001C0:	20 3C F8 74 30 3C FD 74	36 3C F9 74 3C BA 6A 01	<øt0<ÿt6<üt<øj0
0000001D0:	B4 09 CD 21 EB 3D 90 BA	82 01 B4 09 CD 21 EB 33	´oÍ!ë=,ø´oÍ!ë3
0000001E0:	90 BA 9D 01 B4 09 CD 21	EB 29 90 BA BF 01 B4 09	®®®´oÍ!ë)®:ø´o
0000001F0:	CD 21 EB 1F 90 BA ED 01	B4 09 CD 21 EB 15 90 BA	Í!ë®iø´oÍ!ë\$®
000000200:	0F 02 B4 09 CD 21 EB 0B	90 BA 29 02 B4 09 CD 21	±ø´oÍ!ë®)ø´oÍ!
000000210:	EB 01 90 B4 30 CD 21 BE	03 01 83 C6 18 E8 61 FF	ë®´0Í!%øfA†èaÿ
000000220:	BE 03 01 83 C6 1B 8A C4	E8 56 FF BA 03 01 B4 09	%øfA←ŠÄèVÿø´o
000000230:	CD 21 B4 30 CD 21 8A C7	BE 23 01 83 C6 15 E8 40	Í!´0Í!ŠÇ%#øfA\$è@
000000240:	FF 8B C1 BF 4A 01 83 C7	1B E8 1D FF 8A C3 E8 07	ÿ<Ä¿JøfÇ←èÿŠÄè•
000000250:	FF BF 4A 01 83 C7 16 89	05 BA 23 01 B4 09 CD 21	ÿ¿JøfÇ-%*#ø´oÍ!
000000260:	BA 4A 01 B4 09 CD 21 32	C0 B4 4C CD 21	øJ0´oÍ!2Ä´LÍ!

Рисунок 4. lr1_c.com в шестнадцатеричном виде

0000000000: 4D 5A 6D 01 03 00 00 00	20 00 00 00 FF FF 00 00	MZmG♥	ÿÿ
0000000010: 00 00 00 00 00 01 00 00	3E 00 00 00 01 00 FB 50	0 > 0 ùP	
0000000020: 6A 72 00 00 00 00 00 00	00 00 00 00 00 00 00 00	jr	
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		
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		

Рисунок 5. lr1_c.exe в шестнадцатеричном виде (1)

0000000300:	E9 A1 01 4E 75 6D 62 65	72 20 6F 66 20 6D 61 69	é;@Number of mai
0000000310:	6E 20 76 65 72 73 69 6F	6E 3D 20 20 2E 20 20 20	n version= .
0000000320:	0D 0A 24 53 65 72 69 61	6C 20 6E 75 6D 62 65 72	№\$Serial number
0000000330:	20 4F 45 4D 3A 20 20 20	20 20 20 20 20 20 20 20	OEM:
0000000340:	20 20 20 20 20 20 20 0D	0A 24 32 34 20 62 79 74	№\$24 byt
0000000350:	65 20 73 65 72 69 61 6C	20 6E 75 6D 62 65 72 3A	e serial number:
0000000360:	20 20 20 20 20 20 48 0D	0A 24 54 68 65 20 74 79	H)№\$The ty
0000000370:	70 65 20 6F 66 20 49 42	4D 20 69 73 20 50 43 0D	pe of IBM is PC)
0000000380:	0A 24 54 68 65 20 74 79	70 65 20 6F 66 20 49 42	№\$The type of IB
0000000390:	4D 20 69 73 20 50 43 2F	58 54 0D 0A 24 54 68 65	M is PC/XT)№\$The
00000003A0:	20 74 79 70 65 20 6F 66	20 49 42 4D 20 69 73 20	type of IBM is
00000003B0:	50 53 32 20 6D 6F 64 65	6C 20 33 30 0D 0A 24 54	PS2 model 30)№\$T
00000003C0:	68 65 20 74 79 70 65 20	6F 66 20 49 42 4D 20 69	he type of IBM i
00000003D0:	73 20 41 54 20 6F 72 20	50 53 32 20 6D 6F 64 65	s AT or PS2 mode
00000003E0:	6C 20 35 30 20 6F 72 20	36 30 0D 0A 24 54 68 65	l 50 or 60)№\$The
00000003F0:	20 74 79 70 65 20 6F 66	20 49 42 4D 20 69 73 20	type of IBM is
0000000400:	50 53 32 20 6D 6F 64 65	6C 20 38 30 0D 0A 24 54	PS2 model 80)№\$T
0000000410:	68 65 20 74 79 70 65 20	6F 66 20 49 42 4D 20 69	he type of IBM i
0000000420:	73 20 50 43 6A 72 0D 0A	24 54 68 65 20 74 79 70	s PCjr)№\$The typ
0000000430:	65 20 6F 66 20 49 42 4D	20 69 73 20 50 43 20 43	e of IBM is PC C
0000000440:	6F 6E 76 65 72 74 69 62	6C 65 0D 0A 24 24 0F 3C	onvertible)№\$<
0000000450:	09 76 02 04 07 04 30 C3	51 8A E0 E8 EF FF 86 C4	ov♦♦0AQŠaëiÿtÄ
0000000460:	B1 04 D2 E8 E8 E6 FF 59	C3 53 8A FC E8 E9 FF 88	±0ëëäÿYÄSŠüëÿˆ
0000000470:	25 4F 88 05 4F 8A C7 E8	DE FF 88 25 4F 88 05 5B	%0ˆ40ŠCëbÿˆ%0ˆ+[(
0000000480:	C3 51 52 32 E4 33 D2 B9	0A 00 F7 F1 80 CA 30 88	ÄQR2ä3Dˆ1 ÷ñëË0ˆ
0000000490:	14 4E 33 D2 3D 0A 00 73	F1 3C 00 74 04 0C 30 88	ŸN3Dˆ=Š sñ< t+Q0ˆ
00000004A0:	04 5A 59 C3 B8 00 F0 8E	C0 26 A0 FE FF 3C FF 74	♦ZYÄ, ðŽ&Š þÿ<ÿt
00000004B0:	1C 3C FE C7 22 3C FB 74	1E 3C FC 74 2E 3C FA 74	L<þt" <ütÄ<üt.<üt
00000004C0:	20 3C F8 74 30 3C FD 74	36 3C F9 74 3C BA 6A 01	<þt0<ÿt6<üt<ëj0
00000004D0:	B4 09 CD 21 EB 3D 90 BA	82 01 B4 09 CD 21 EB 33	ˆoÍ!ë=0ë,0ˆoÍ!ë3
00000004E0:	90 BA 9D 01 B4 09 CD 21	EB 29 90 BA BF 01 B4 09	0ë00ˆoÍ!ë)0ëë0ˆo
00000004F0:	CD 21 EB 1F 90 BA ED 01	B4 09 CD 21 EB 15 90 BA	Í!ëˆ0ëëi0ˆoÍ!ëŠ0ë
0000000500:	0F 02 B4 09 CD 21 EB 08	90 BA 29 02 B4 09 CD 21	00ˆoÍ!ëŠ0ë)0ˆoÍ!
0000000510:	EB 01 90 B4 30 CD 21 BE	03 01 83 C6 18 E8 61 FF	ë000ˆoÍ!%0ˆfA†ëäÿ
0000000520:	BE 03 01 83 C6 18 8A C4	E8 56 FF BA 03 01 B4 09	%0ˆofA<ŠÄëVÿëˆ0ˆo
0000000530:	CD 21 B4 30 CD 21 8A C7	BE 23 01 83 C6 15 E8 40	Í!ˆoÍ!ŠC%0ˆofAŠë@
0000000540:	FF 8B C1 BF 4A 01 83 C7	1B E8 1D FF 8A C3 E8 07	ÿ<Ä;J0fC<ë÷ÿŠÄë•
0000000550:	FF BF 4A 01 83 C7 16 89	05 BA 23 01 B4 09 CD 21	ÿ;J0fCˆ%+ë#0ˆoÍ!
0000000560:	BA 4A 01 B4 09 CD 21 32	C0 B4 4C CD 21	ëJ0ˆoÍ!2ÄˆLÍ!

Рисунок 6. lrl_с.ехе в шестнадцатеричном виде (2)

0000000000:	4D 5A 7C 00 04 00 01 00	20 00 00 00 FF FF 00 00	MZ ♦ @	яя
0000000010:	00 02 00 00 00 00 35 00	3E 00 00 00 01 00 FB 50	0 5 >	0 ыP
0000000020:	6A 72 00 00 00 00 00 00	00 00 00 00 00 00 00 00	jr	
0000000030:	00 00 00 00 00 00 00 00	00 00 00 00 00 00 5F 00		
0000000040:	35 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	5	
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		
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		

[illegible]

0000000400:	4E 75 6D 62 65 72 20 6F	66 20 6D 61 69 6E 20 76	Number of main v
0000000410:	65 72 73 69 6F 6E 3D 20	20 2E 20 20 20 0D 0A 24	ersion= .)\$
0000000420:	53 65 72 69 61 6C 20 6E	75 6D 62 65 72 20 4F 45	Serial number 0E
0000000430:	4D 3A 20 20 20 20 20 20	20 20 20 20 20 20 20 20	M:
0000000440:	20 20 20 20 0D 0A 24 32	34 20 62 79 74 65 20 73)\$24 byte s
0000000450:	65 72 69 61 6C 20 6E 75	6D 62 65 72 3A 20 20 20	erial number:
0000000460:	20 20 20 48 0D 0A 24 54	68 65 20 74 79 70 65 20	H)\$The type
0000000470:	6F 66 20 49 42 4D 20 69	73 20 50 43 0D 0A 24 54	of IBM is PC)\$T
0000000480:	68 65 20 74 79 70 65 20	6F 66 20 49 42 4D 20 69	he type of IBM i
0000000490:	73 20 50 43 2F 58 54 0D	0A 24 54 68 65 20 74 79	s PC/XT)\$The ty
00000004A0:	70 65 20 6F 66 20 49 42	4D 20 69 73 20 50 53 32	pe of IBM is PS2
00000004B0:	20 6D 6F 64 65 6C 20 33	30 0D 0A 24 54 68 65 20	model 30)\$The
00000004C0:	74 79 70 65 20 6F 66 20	49 42 4D 20 69 73 20 41	type of IBM is A
00000004D0:	54 20 6F 72 20 50 53 32	20 6D 6F 64 65 6C 20 35	T or PS2 model 5
00000004E0:	30 20 6F 72 20 36 30 0D	0A 24 54 68 65 20 74 79	0 or 60)\$The ty
00000004F0:	70 65 20 6F 66 20 49 42	4D 20 69 73 20 50 53 32	pe of IBM is PS2
0000000500:	20 6D 6F 64 65 6C 20 38	30 0D 0A 24 54 68 65 20	model 80)\$The
0000000510:	74 79 70 65 20 6F 66 20	49 42 4D 20 69 73 20 50	type of IBM is P
0000000520:	43 6A 72 0D 0A 24 54 68	65 20 74 79 70 65 20 6F	Cjr)\$The type o
0000000530:	66 20 49 42 4D 20 69 73	20 50 43 20 43 6F 6E 76	f IBM is PC Conv
0000000540:	65 72 74 69 62 6C 65 0D	0A 24 00 00 00 00 00 00	ertible)\$
0000000550:	EB 58 90 24 0F 3C 09 76	02 04 07 04 30 C3 51 8A	лХђ\$о<ов0♦♦0Г0Љ
0000000560:	E0 E8 EF FF 86 C4 B1 04	D2 E8 E8 E6 FF 59 C3 53	аипя†Д±♦ТиижяYFS
0000000570:	8A FC E8 E9 FF 88 25 4F	88 05 4F 8A C7 E8 DE FF	Љъийя€%O€♦OЉ3иЮя
0000000580:	88 25 4F 88 05 5B C3 51	52 32 E4 33 D2 B9 0A 00	€%O€♦[ГQR2д3ТЉ
0000000590:	F7 F1 80 CA 30 88 14 4E	33 D2 3D 0A 00 73 F1 3C	чсЪK0€ЉN3Т= sс<
00000005A0:	00 74 04 0C 30 88 04 5A	59 C3 1E 2B C0 50 B8 20	t♦Q0€♦ZYГ▲+APё
00000005B0:	00 8E D8 B8 00 F0 8E C0	26 A0 FE FF 3C FF 74 1C	џШё рџА8 юя<ятL
00000005C0:	3C FE 74 22 3C FB 74 1E	3C FC 74 2E 3C FA 74 20	<ют"<ыт▲<ьт.<ьт
00000005D0:	3C F8 74 30 3C FD 74 36	3C F9 74 3C BA 67 00 B4	<шт0<эт6<шт<ег r
00000005E0:	09 CD 21 EB 3D 90 BA 7F	00 B4 09 CD 21 EB 33 90	оН!л=ђео гоН!л3ђ
00000005F0:	BA 9A 00 B4 09 CD 21 EB	29 90 BA BC 00 B4 09 CD	ель гоН!л)ђеј гоН
0000000600:	21 EB 1F 90 BA EA 00 B4	09 CD 21 EB 15 90 BA 0C	!лџђек гоН!л\$ђеQ
0000000610:	01 B4 09 CD 21 EB 08 90	BA 26 01 B4 09 CD 21 EB	0гоН!л\$ђе&0гоН!л
0000000620:	01 90 B4 30 CD 21 BE 00	00 83 C6 18 E8 58 FF BE	0ђг0Н!s ѓЖ†иХяs
0000000630:	00 00 83 C6 1B 8A C4 E8	4D FF BA 00 00 B4 09 CD	ѓЖ<ЉдиМяе гоН
0000000640:	21 B4 30 CD 21 8A C7 BE	20 00 83 C6 15 E8 37 FF	!г0Н!Љ3s ѓЖ\$и7я
0000000650:	8B C1 BF 47 00 83 C7 1B	E8 14 FF 8A C3 E8 FE FE	<БЇG ѓ3<иЉЉГиюю
0000000660:	BF 47 00 83 C7 16 89 05	BA 20 00 B4 09 CD 21 BA	iG ѓ3=€е гоН!е
0000000670:	47 00 B4 09 CD 21 32 C0	B4 4C CD 21	G гоН!2ArLH!

Рисунок 9. lr1_e.exe в шестнадцатеричном виде (3)

Вид файлов модулей .COM и .EXE в отладчике представлен на рисунках 10 и 11

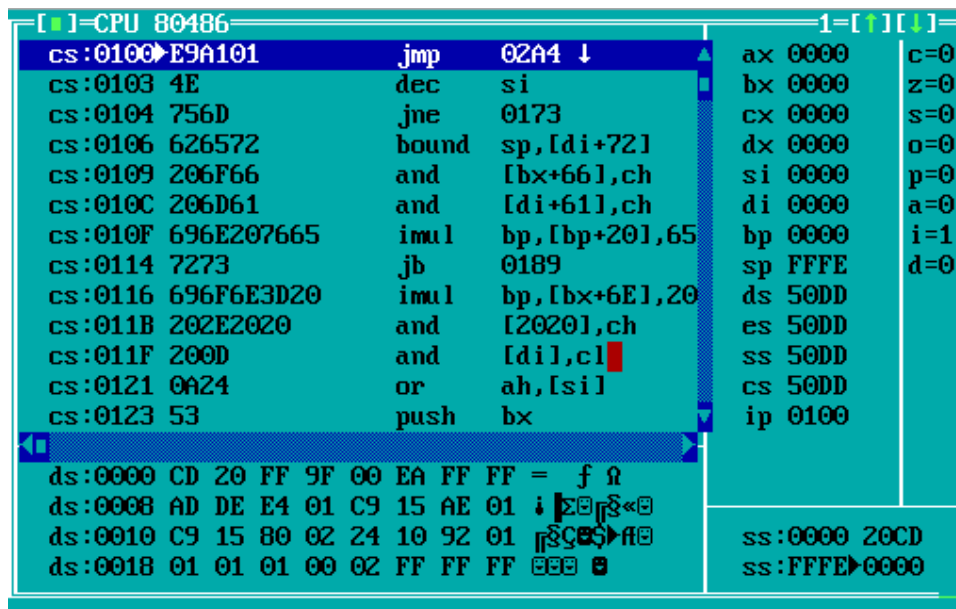


Рисунок 10. lr1_c.com в отладчике td.exe

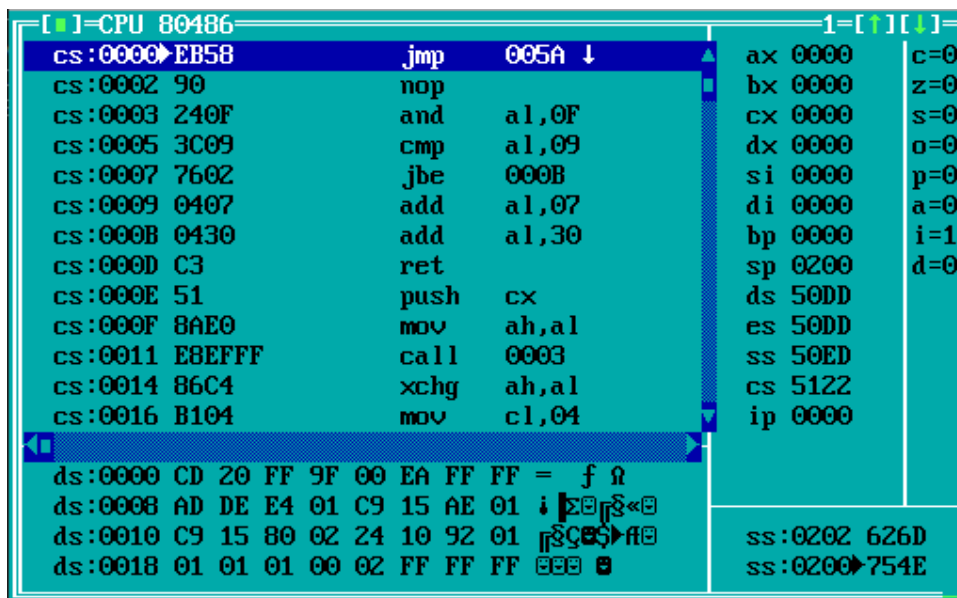


Рисунок 11. lr1_e.exe в отладчике td.exe

Контрольные вопросы.

Отличия исходных текстов COM и EXE программ

1. Сколько сегментов должна содержать COM-программа?

COM-программа содержит один сегмент.

2. EXE-программа?

EXE-программа может содержать различное количество сегментов в зависимости от модели памяти (например compact – один сегмент кода, несколько сегментов данных, сегмент стека)

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

ASSUME(устанавливает соответствие сегментных регистров и сегмента)
и ORG 100h(устанавливает смещение в 100h для PSP в начале программы)

4. Все ли форматы команд можно использовать в COM-программе?
Нельзя использовать команды, использующие адрес сегмента, так как он определяется после запуска программы. Также нельзя создать больше одного сегмента.

Отличия форматов файлов COM и EXE модулей

1. Какова структура файла COM? С какого адреса располагается код? COM содержит один сегмент, а код располагается с адреса 0h.

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

«Плохой» EXE также содержит один сегмент, код располагается с адреса 300h, с адреса 0 располагается заголовок(сигнатура файла, число элементов и адрес таблицы настройки адресов, длина заголовка, объём памяти, требующийся для выделения и прочие данные) и таблица настройки адресов(состоит из длинных указателей вида смещение: сегмент на слова в загрузочном модуле, которые содержат настраиваемые сегментные адреса).

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

Структура файла аналогична, за исключением наличия трёх сегментов вместо одного. Теперь сегмент кода расположен по адресу 400h, так как перед ним появился сегмент стека размером 100h. В «плохом» файле код всегда начинается с адреса 300h, в «хорошем» файле вначале также идет заголовок и таблица настроек, но т. к. в нем определен сегмент стека, то адресация начинается с 200h + размер стека, 400h в данном случае.

Загрузка COM модуля в основную память

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

Код располагается с адреса 100h, перед ним – PSP.

2. Что располагается с адреса 0?

PSP — структура данных, которая используется в операционных системах семейства DOS и CP/M для сохранения состояния компьютерных программ.

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

Сегментные регистры имеют значение 50DD, указывающее на начало PSP.

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

Указатель стека (регистр SP) устанавливается на конец основного сегмента. В стек записывается 0000h (адрес возврата для команды ret).

Загрузка «хорошего» EXE модуля в основную память

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

DS, ES (50DD) – начало PSP; SS (50ED) – начало сегмента стека; CS (5122) – начало сегмента кода.

2. На что указывают регистры DS и ES?

Эти регистры указывают на начало PSP.

3. Как определяется стек? Директивой .STACK или с помощью ASSUME SS:<ссылка на сегмент, который будет определен как сегмент стека>

4. Как определяется точка входа?

Директивой END <ссылка на точку входа>.

Вывод.

В ходе выполнения данной лабораторной работы было исследовано различие в структурах исходных текстов модулей типов .COM и .EXE, структур файлов загрузочных модулей и способов их загрузки в основную память с помощью двух вариантов программы, которая определяет тип и версию системы.

ПРИЛОЖЕНИЕ А

Исходный код программы lr1_e.asm

```
AStack SEGMENT STACK
    DW 100h DUP(5353H)
AStack ENDS
DATA SEGMENT
VERS db 'Number of main version=  .  ',0DH,0AH,'$'
OEM   db'Serial number OEM:          ',0DH,0AH,'$'
USER  db'24 byte serial number:      H',0DH,0AH,'$'

TYPE1 db 'The type of IBM is PC',0DH,0AH,'$'
TYPE2 db 'The type of IBM is PC/XT',0DH,0AH,'$'
TYPE3 db 'The type of IBM is PS2 model 30',0DH,0AH,'$'
TYPE4 db 'The type of IBM is AT or PS2 model 50 or 60',0DH,0AH,'$'
TYPE5 db 'The type of IBM is PS2 model 80',0DH,0AH,'$'
TYPE6 db 'The type of IBM is PCjr',0DH,0AH,'$'
TYPE7 db 'The type of IBM is PC Convertible',0DH,0AH,'$'

DATA ENDS
TESTPC SEGMENT
    ASSUME CS:TESTPC, DS:DATA, ES:NOTHING, SS:AStack
START: JMP BEGIN

TETR_TO_HEX PROC near
and AL,0Fh
cmp AL,09
jbe NEXT1
add AL,07
NEXT1: add AL,30h
ret
TETR_TO_HEX ENDP

BYTE_TO_HEX PROC near
push CX
mov AH,AL
call TETR_TO_HEX
xchg AL,AH
mov CL,4
shr AL,CL
call TETR_TO_HEX
pop CX
ret
BYTE_TO_HEX ENDP

WRD_TO_HEX PROC near
push BX
mov BH,AH
call BYTE_TO_HEX
mov [DI],AH
dec DI
```

```

mov [DI],AL
dec DI
mov AL,BH
call BYTE_TO_HEX
mov [DI],AH
dec DI
mov [DI],AL
pop BX
ret
WRD_TO_HEX ENDP

```

```

BYTE_TO_DEC PROC near
push CX
push DX
xor AH,AH
xor DX,DX
mov CX,10
loop_bd: div CX
or DL,30h
mov [SI],DL
dec SI
xor DX,DX
cmp AX,10
jae loop_bd
cmp AL,00h
je end_1
or AL,30h
mov [SI],AL
end_1: pop DX
pop CX
ret

```

```

BYTE_TO_DEC ENDP
BEGIN:
    push DS
    sub AX,AX
    push AX
    mov AX,DATA
    mov DS, AX
    mov ax,0F000h
    mov es,ax
    mov al,es:[0FFFEh]

    cmp al,00FFh
    je Wt1
    cmp al,00FEh
    je Wt2
    cmp al,00FBh
    je Wt2
    cmp al,00FCh
    je Wt4

```

```

cmp al,00FAh
je Wt3
cmp al,00F8h
je Wt5
cmp al,00FDh
je Wt6
cmp al,00F9h
je Wt7
Wt1:
    mov DX,offset TYPE1
    mov AH,09h
    int 21h
    jmp next
Wt2:
    mov DX,offset TYPE2
    mov AH,09h
    int 21h
    jmp next
Wt3:
    mov DX,offset TYPE3
    mov AH,09h
    int 21h
    jmp next
Wt4:
    mov DX,offset TYPE4
    mov AH,09h
    int 21h
    jmp next
Wt5:
    mov DX,offset TYPE5
    mov AH,09h
    int 21h
    jmp next
Wt6:
    mov DX,offset TYPE6
    mov AH,09h
    int 21h
    jmp next
Wt7:
    mov DX,offset TYPE7
    mov AH,09h
    int 21h
    jmp next
next:

mov ah,30h
int 21h
mov si, offset VERS
add si, 24
call BYTE_TO_DEC
mov si, offset VERS

```



```

    add si, 27
    mov al,ah
    call BYTE_TO_DEC

    mov DX,offset VERS
    mov AH,09h
    int 21h

    mov ah, 30h
    int 21h
    mov al, bh

    mov si, offset OEM
    add si, 21
    call BYTE_TO_DEC

    mov ax, cx
    mov di, offset USER
    add di, 27
    call WRD_TO_HEX
    mov al, bl
    call BYTE_TO_HEX
    mov di, offset USER
    add di, 22

    mov [di], ax
    mov DX,offset OEM
    mov AH,09h
    int 21h

    mov DX,offset USER
    mov AH,09h
    int 21h

    xor AL,AL
    mov AH,4Ch
    int 21H
TESTPC ENDS
END START ;

```

ПРИЛОЖЕНИЕ Б

Исходный код программы lr1_c.asm

```
TESTPC SEGMENT
    ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING
    ORG 100H
START: JMP BEGIN
VERS db 'Number of main version=  .  ',0DH,0AH,'$'
OEM   db'Serial number OEM:          ',0DH,0AH,'$'
USER  db'24 byte serial number:      H',0DH,0AH,'$'

TYPE1 db 'The type of IBM is PC',0DH,0AH,'$'
TYPE2 db 'The type of IBM is PC/XT',0DH,0AH,'$'
TYPE3 db 'The type of IBM is PS2 model 30',0DH,0AH,'$'
TYPE4 db 'The type of IBM is AT or PS2 model 50 or 60',0DH,0AH,'$'
TYPE5 db 'The type of IBM is PS2 model 80',0DH,0AH,'$'
TYPE6 db 'The type of IBM is PCjr',0DH,0AH,'$'
TYPE7 db 'The type of IBM is PC Convertible',0DH,0AH,'$'

TETR_TO_HEX PROC near
and AL,0Fh
cmp AL,09
jbe NEXT1
add AL,07
NEXT1: add AL,30h
ret
TETR_TO_HEX ENDP

BYTE_TO_HEX PROC near
push CX
mov AH,AL
call TETR_TO_HEX
xchg AL,AH
mov CL,4
shr AL,CL
call TETR_TO_HEX
pop CX
ret
BYTE_TO_HEX ENDP

WRD_TO_HEX PROC near
push BX
mov BH,AH
call BYTE_TO_HEX
mov [DI],AH
```

```

dec DI
mov [DI],AL
dec DI
mov AL,BH
call BYTE_TO_HEX
mov [DI],AH
dec DI
mov [DI],AL
pop BX
ret
WRD_TO_HEX ENDP

```

```

BYTE_TO_DEC PROC near
push CX
push DX
xor AH,AH
xor DX,DX
mov CX,10
loop_bd: div CX
or DL,30h
mov [SI],DL
dec SI
xor DX,DX
cmp AX,10
jae loop_bd
cmp AL,00h
je end_1
or AL,30h
mov [SI],AL
end_1: pop DX
pop CX
ret

```

```

BYTE_TO_DEC ENDP

```

```

BEGIN:

```

```

    ;push DS
    ;sub AX,AX
    ;push AX
    ;mov AX,TESTPC
    ;mov DS, AX
    mov ax,0F000h
    mov es,ax
    mov al,es:[0FFFEh]

```

```

    cmp al,00FFh
    je Wt1
    cmp al,00FEh
    je Wt2
    cmp al,00FBh
    je Wt2
    cmp al,00FCh

```

```

je Wt4
cmp al,00FAh
je Wt3
cmp al,00F8h
je Wt5
cmp al,00FDh
je Wt6
cmp al,00F9h
je Wt7
Wt1:
    mov DX,offset TYPE1
    mov AH,09h
    int 21h
    jmp next
Wt2:
    mov DX,offset TYPE2
    mov AH,09h
    int 21h
    jmp next
Wt3:
    mov DX,offset TYPE3
    mov AH,09h
    int 21h
    jmp next
Wt4:
    mov DX,offset TYPE4
    mov AH,09h
    int 21h
    jmp next
Wt5:
    mov DX,offset TYPE5
    mov AH,09h
    int 21h
    jmp next
Wt6:
    mov DX,offset TYPE6
    mov AH,09h
    int 21h
    jmp next
Wt7:
    mov DX,offset TYPE7
    mov AH,09h
    int 21h
    jmp next
next:

mov ah,30h
int 21h
mov si, offset VERS
add si, 24
call BYTE_TO_DEC

```

```

    mov si, offset VERS
    add si, 27
    mov al,ah
    call BYTE_TO_DEC

    mov DX,offset VERS
    mov AH,09h
    int 21h

    mov ah, 30h
    int 21h
    mov al, bh

    mov si, offset OEM
    add si, 21
    call BYTE_TO_DEC

    mov ax, cx
    mov di, offset USER
    add di, 27
    call WRD_TO_HEX
    mov al, bl
    call BYTE_TO_HEX
    mov di, offset USER
    add di, 22

    mov [di], ax
    mov DX,offset OEM
    mov AH,09h
    int 21h

    mov DX,offset USER
    mov AH,09h
    int 21h

    xor AL,AL
    mov AH,4Ch
    int 21H
TESTPC ENDS
END START ;

```