

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.8383

Сырцова Е.А.

Преподаватель

Ефремов М.А.

Дата выполнения работы

07.03.2020

г. Санкт-Петербург

2020 г.

1. Постановка задачи

1.1. Цель работы

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

1.2. Сведения о функциях и структурах данных управляющей программы

Функции управляющей программы

Имя функции	Описание функции
TETR_TO_HEX	Функция шаблона, приведенного в методических указаниях. Функция переводит половину байта в шестнадцатеричную систему.
BYTE_TO_HEX	Функция шаблона, приведенного в методических указаниях. Байт в регистре AL переводится в два символа шестнадцатеричного числа в регистре AX.
WRD_TO_HEX	Функция шаблона, приведенного в методических указаниях. Функция переводит в шестнадцатеричную систему счисления 16-ти разрядное число.
BYTE_TO_DEC	Функция шаблона, приведенного в методических указаниях. Функция переводит байт в десятичную систему счисления.
PRINT	Функция выводит сообщение на экран.
TYPE_PC	Функция определяет код типа PC, который хранится в байте по адресу 0F000:0FFFEh, в последнем байте ROM BIOS.
PRINT_TYPE_PC	Функция определяет тип PC, в соответствии с полученным кодом и вызывает функцию для вывода результата на экран.
VERSION_DOS	Функция определяет версию PC, используя функцию BYTE_TO_DEC, и вызывает функцию для вывода результата на экран.
OEM_NUM	Функция определяет серийный номер OEM, используя функцию

	BYTE_TO_DEC, и вызывает функцию для вывода результата на экран.
USER_NUM	Функция определяет серийный номер пользователя, используя функции WRD_TO_HEX и BYTE_TO_HEX, и вызывает функцию для вывода результата на экран.

Структура данных управляющей программы

Имя	Тип	Назначение
PC	db	Вывод строки 'Type PC: PC'
PC_XT	db	Вывод строки 'Type PC: PC/XT'
AT	db	Вывод строки 'Type PC: AT'
PS2_1	db	Вывод строки 'Type PC: PS2 model 30'
PS2_2	db	Вывод строки 'Type PC: PS2 model 50 or 60'
PS2_3	db	Вывод строки 'Type PC: PS2 model 80'
PCjr	db	Вывод строки 'Type PC: PCjr'
PC_CONVERTIBLE	db	Вывод строки 'Type PC: PC Convertible'
VERS	db	Вывод строки 'Version MS DOS: '
OEM	db	Вывод строки 'OEM serial number: '
USER	db	Вывод строки 'User serial number: '

1.3. Последовательность действий, выполняемых утилитой

Программа определяет и выводит на экран следующие значения в заданном порядке: тип PC, версия ОС, серийный номер OEM, серийный номер пользователя.

2. Ход работы

2.1. Был написан текст исходного .COM модуля, который определяет тип PC и версию системы, а так же серийный номер OEM и серийный номер пользователя. В результате выполнения был получен «хороший» .COM модуль.

```
C:\>LR1.COM
Type PC: AT
Version MS DOS: 5.0
OEM serial number: 255
User serial number: 000000
C:\>
```

Рисунок 1 – «хороший».COM модуль

2.2. Был построен «плохой».EXE модуль, полученный из исходного текста для .COM модуля.

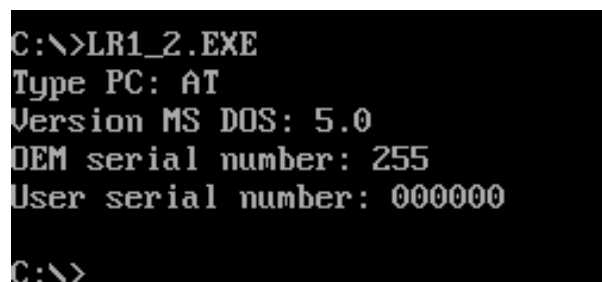


```
C:\>LR1.EXE

5 0
Type PC: PC
255
000000
Type PC: PC
Type PC: PC
C:\>
```

Рисунок 2 – «плохой».EXE модуль

2.3. Был написан текст исходного .EXE модуля, который выполняет те же функции. В результате был получен «хороший».EXE модуль.



```
C:\>LR1_2.EXE
Type PC: AT
Version MS DOS: 5.0
OEM serial number: 255
User serial number: 000000
C:\>
```

Рисунок 3 – «хороший».EXE модуль

3. Ответы на контрольные вопросы

3.1. Отличия исходных текстов COM и EXE программ

3.1.1. Сколько сегментов должна содержать COM-программа?

Ответ: COM-программа должна содержать один сегмент.

3.1.2. EXE-программа?

Ответ: EXE-программа может содержать более одного сегмента. В программе описываются три сегмента: команд, данных и стека.

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

Ответ: В тексте COM-программы обязательно должна быть директива `ORG 100H`, которая резервирует 256 байт для PSP. Так же обязательно должна быть директива `ASSUME`, которая устанавливает соответствие сегментного регистра CS сегменту команд, а сегментного регистра DS – сегменту данных.

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

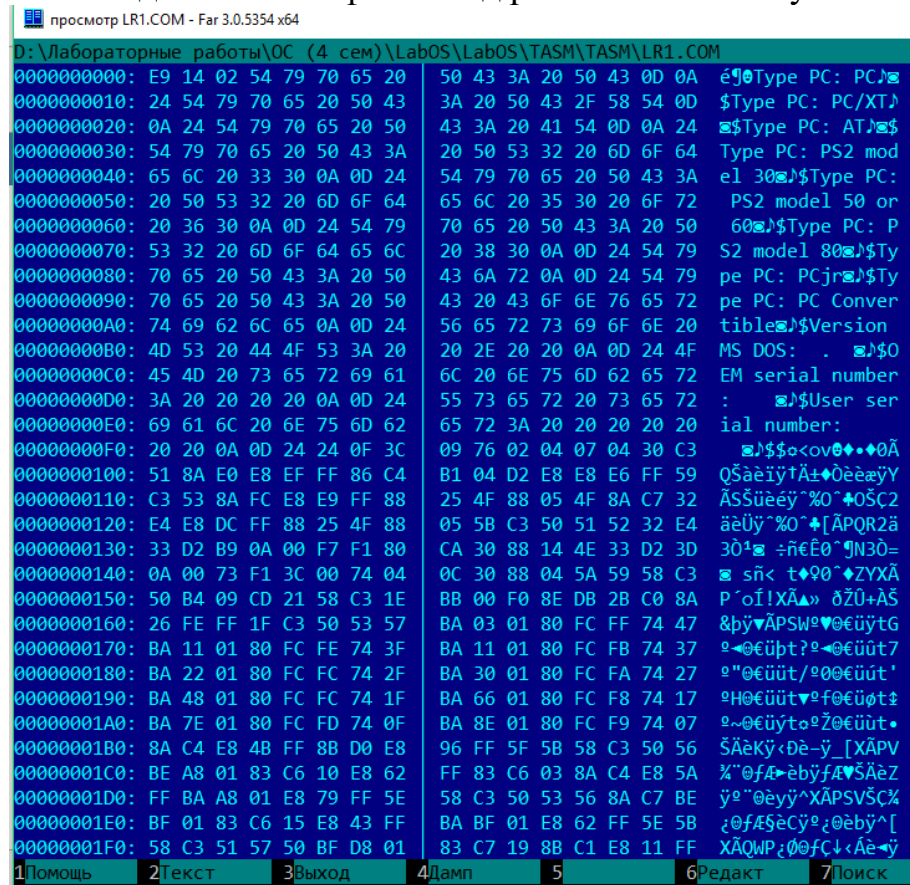
Ответ: В COM-программе нельзя использовать команды вида `mov rvalue` в виде адресов сегментов и команды, содержащие дальнюю адресацию. Это связано с тем, что в COM-программе отсутствует таблица настроек (Relocation Table), с помощью которой в момент

запуска программы загрузчик определяет и подставляет адреса сегментов.

3.2. Отличия форматов файлов COM и EXE модулей

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

Ответ: .COM файл состоит из одного сегмента, который содержит данные и команды. В .COM файле код располагается с нулевого адреса.



```
00000000: E9 14 02 54 79 70 65 20 50 43 3A 20 50 43 0D 0A  e9 14 02 54 79 70 65 20 50 43 3A 20 50 43 0D 0A  Type PC: PC
0000000010: 24 54 79 70 65 20 50 43 43 3A 20 50 43 2F 58 54 0D  24 54 79 70 65 20 50 43 43 3A 20 50 43 2F 58 54 0D  Type PC: PC/XT
0000000020: 0A 24 54 79 70 65 20 50 43 20 50 53 32 20 6D 6F 64  0A 24 54 79 70 65 20 50 43 20 50 53 32 20 6D 6F 64  Type PC: AT
0000000030: 54 79 70 65 20 50 43 3A 65 6C 20 33 30 0A 0D 24 54  54 79 70 65 20 50 43 3A 65 6C 20 33 30 0A 0D 24 54  Type PC: PS2
0000000040: 65 6C 20 33 30 0A 0D 24 54 79 70 65 20 50 43 3A  65 6C 20 33 30 0A 0D 24 54 79 70 65 20 50 43 3A  e1 30  Type PC:
0000000050: 20 50 53 32 20 6D 6F 64 70 65 20 50 43 3A 20 50  20 50 53 32 20 6D 6F 64 70 65 20 50 43 3A 20 50  PS2 model 50 or
0000000060: 20 36 30 0A 0D 24 54 79 20 38 30 0A 0D 24 54 79  20 36 30 0A 0D 24 54 79 20 38 30 0A 0D 24 54 79  60  Type PC: P
0000000070: 53 32 20 6D 6F 64 65 6C 70 65 20 50 43 3A 20 50  53 32 20 6D 6F 64 65 6C 70 65 20 50 43 3A 20 50  S2 model 80
0000000080: 70 65 20 50 43 3A 20 50 43 6A 72 0A 0D 24 54 79  70 65 20 50 43 3A 20 50 43 6A 72 0A 0D 24 54 79  pe PC: PCjr
0000000090: 70 65 20 50 43 3A 20 50 43 20 43 6F 6E 76 65 72  70 65 20 50 43 3A 20 50 43 20 43 6F 6E 76 65 72  pe PC: PC Conver
00000000A0: 74 69 62 6C 65 0A 0D 24 20 2E 20 20 0A 0D 24 4F  74 69 62 6C 65 0A 0D 24 20 2E 20 20 0A 0D 24 4F  tible
00000000B0: 4D 53 20 44 4F 53 3A 20 6C 20 6E 75 6D 62 65 72  4D 53 20 44 4F 53 3A 20 6C 20 6E 75 6D 62 65 72  MS DOS: .
00000000C0: 45 4D 20 73 65 72 69 61 6C 20 6E 75 6D 62 65 72  45 4D 20 73 65 72 69 61 6C 20 6E 75 6D 62 65 72  EM serial number
00000000D0: 3A 20 20 20 0A 0D 24 69 61 6C 20 6E 75 6D 62 65  3A 20 20 20 0A 0D 24 69 61 6C 20 6E 75 6D 62 65  :
00000000E0: 69 61 6C 20 6E 75 6D 62 09 76 02 04 07 04 30 C3  69 61 6C 20 6E 75 6D 62 09 76 02 04 07 04 30 C3  ial number:
00000000F0: 20 20 0A 0D 24 24 0F 3C B1 04 D2 E8 E8 E6 FF 59  20 20 0A 0D 24 24 0F 3C B1 04 D2 E8 E8 E6 FF 59  Q$aeiÿtÄ±ðèèæÿY
0000000100: 51 8A E0 E8 EF FF 86 C4 25 4F 88 05 4F 8A C7 32  51 8A E0 E8 EF FF 86 C4 25 4F 88 05 4F 8A C7 32  AS$üèÿ`%0`40$C2
0000000110: C3 53 8A FC E8 E9 FF 88 05 5B C3 50 51 52 32 E4  C3 53 8A FC E8 E9 FF 88 05 5B C3 50 51 52 32 E4  äèÿ`%0`[ÄPQR2ä
0000000120: E4 E8 DC FF 88 25 4F 88 CA 30 88 14 4E 33 D2 3D  E4 E8 DC FF 88 25 4F 88 CA 30 88 14 4E 33 D2 3D  3D¹ ÷ñèÿ`JN3D=
0000000130: 33 D2 B9 0A 00 F7 F1 80 0C 30 88 04 5A 59 58 C3  33 D2 B9 0A 00 F7 F1 80 0C 30 88 04 5A 59 58 C3  sñ< t±90`ZYXÄ
0000000140: 0A 00 73 F1 3C 00 74 04 BB 00 F0 8E DB 2B C0 8A  0A 00 73 F1 3C 00 74 04 BB 00 F0 8E DB 2B C0 8A  P`oí!XÄ» ðZÜ+Ä$
0000000150: 50 B4 09 CD 21 58 C3 1E BA 03 01 80 FC FF 74 47  50 B4 09 CD 21 58 C3 1E BA 03 01 80 FC FF 74 47  &þÿÄPSW♥œüÿtG
0000000160: 26 FE FF 1F C3 50 53 57 BA 11 01 80 FC FE 74 3F  26 FE FF 1F C3 50 53 57 BA 11 01 80 FC FE 74 3F  e-œüÿt?°œüÿt7
0000000170: BA 11 01 80 FC FE 74 3F BA 30 01 80 FC FA 74 27  BA 11 01 80 FC FE 74 3F BA 30 01 80 FC FA 74 27  e"œüÿt/°œüÿt'
0000000180: BA 22 01 80 FC FC 74 2F BA 66 01 80 FC F8 74 17  BA 22 01 80 FC FC 74 2F BA 66 01 80 FC F8 74 17  e"œüÿt°fœüÿt'
0000000190: BA 48 01 80 FC FC 74 1F BA 8E 01 80 FC F9 74 07  BA 48 01 80 FC FC 74 1F BA 8E 01 80 FC F9 74 07  e~œüÿt°œüÿt°
00000001A0: BA 7E 01 80 FC FD 74 0F 96 FF 5F 5B 58 C3 50 56  BA 7E 01 80 FC FD 74 0F 96 FF 5F 5B 58 C3 50 56  ŠÄèKÿ<ðè-ÿ [XÄPV
00000001B0: 8A C4 E8 4B FF 8B D0 E8 FF 83 C6 03 8A C4 E8 5A  8A C4 E8 4B FF 8B D0 E8 FF 83 C6 03 8A C4 E8 5A  %"øfÄ»èbÿfÄŠÄèZ
00000001C0: BE A8 01 83 C6 10 E8 62 58 C3 50 53 56 8A C7 BE  BE A8 01 83 C6 10 E8 62 58 C3 50 53 56 8A C7 BE  ÿ°"œÿÿ^XÄPSV$CZ
00000001D0: FF BA A8 01 E8 79 FF 5E BA BF 01 E8 62 FF 5E 5B  FF BA A8 01 E8 79 FF 5E BA BF 01 E8 62 FF 5E 5B  ÿ°øfÄ$èCÿ°;øèbÿ^
00000001E0: BF 01 83 C6 15 E8 43 FF 83 C7 19 8B C1 E8 11 FF  BF 01 83 C6 15 E8 43 FF 83 C7 19 8B C1 E8 11 FF  XÄQWP;øøfC↓Äè-ÿ
00000001F0: 58 C3 51 57 50 BF D8 01
```

Рисунок 4 - .COM модуль в шестнадцатеричном виде.

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

Ответ: «плохой» .EXE модуль не разделен на сегменты. Данные и код содержатся в одном сегменте. Код «плохого» .EXE модуля располагается с адреса 300h. С адреса 0h располагается заголовок.

```

просмотр LR1.EXE - Far 3.0.5354 x64
D:\Лабораторные работы\ОС (4 сем)\LabOS\LabOS\TASM\TASM\LR1.EXE
00000000: 4D 5A 30 01 03 00 00 00 20 00 00 00 FF FF 00 00 MZ0000  yy
00000001: 00 00 00 00 00 01 00 00 3E 00 00 00 01 00 FB 50 0 > 0 ūP
00000002: 6A 72 00 00 00 00 00 00 00 00 00 00 00 00 00 00 jr
00000003: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000004: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000005: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000006: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000007: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000008: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000009: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000A: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000B: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000C: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000D: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000E: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000000F: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000011: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000012: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000013: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000014: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000015: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000016: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000017: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000018: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000019: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000001A: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000001B: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000001C: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000001D: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000001E: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000001F: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1Помощь 2Текст 3Выход 4Дамп 5 6Редакт 7Поиск

```

```

просмотр LR1.EXE - Far 3.0.5354 x64
D:\Лабораторные работы\ОС (4 сем)\LabOS\LabOS\TASM\TASM\LR1.EXE
00000030: E9 14 02 54 79 70 65 20 50 43 3A 20 50 43 0D 0A éType PC: PC/XT
00000031: 24 54 79 70 65 20 50 43 3A 20 50 43 2F 58 54 0D $Type PC: PC/XT
00000032: 0A 24 54 79 70 65 20 50 43 3A 20 41 54 0D 0A 24 $Type PC: AT/$
00000033: 54 79 70 65 20 50 43 3A 20 50 53 32 20 6D 6F 64 Type PC: PS2 mod
00000034: 65 6C 20 33 30 0A 0D 24 54 79 70 65 20 50 43 3A el 30$Type PC:
00000035: 20 50 53 32 20 6D 6F 64 65 6C 20 35 30 20 6F 72 PS2 model 50 or
00000036: 20 36 30 0A 0D 24 54 79 70 65 20 50 43 3A 20 50 60$Type PC: P
00000037: 53 32 20 6D 6F 64 65 6C 20 38 30 0A 0D 24 54 79 S2 model 80$Ty
00000038: 70 65 20 50 43 3A 20 50 43 6A 72 0A 0D 24 54 79 pe PC: PCjr$Ty
00000039: 70 65 20 50 43 3A 20 50 43 20 43 6F 6E 76 65 72 pe PC: PC Conver
0000003A: 74 69 62 6C 65 0A 0D 24 56 65 72 73 69 6F 6E 20 tible$Version
0000003B: 4D 53 20 44 4F 53 3A 20 20 2E 20 20 0A 0D 24 4F MS DOS: . $0
0000003C: 45 4D 20 73 65 72 69 61 6C 20 6E 75 6D 62 65 72 EM serial number
0000003D: 3A 20 20 20 20 0A 0D 24 55 73 65 72 20 73 65 72 : $User ser
0000003E: 69 61 6C 20 6E 75 6D 62 65 72 3A 20 20 20 20 20 ial number:
0000003F: 20 20 0A 0D 24 24 0F 3C 09 76 02 04 07 04 30 C3 $ $o<ov0+0A
00000040: 51 8A E0 E8 EF FF 86 C4 B1 04 D2 E8 E8 E6 FF 59 Q$æiÿtÄ±0èæÿY
00000041: C3 53 8A FC E8 E9 FF 88 25 4F 88 8C 53 8A C7 32 Å$Ûèÿ`%0`+0$C2
00000042: E4 E8 DC FF 88 25 4F 88 05 5B C3 50 51 52 32 E4 æÿ`%0`+[ÄPQR2ä
00000043: 33 D2 B9 0A 00 F7 F1 80 CA 30 88 14 4E 33 D2 3D 3D+ ñèË0`JN3D=
00000044: 0A 00 73 F1 3C 00 74 04 0C 30 88 04 5A 59 58 C3 sñ< t00`ZYXÄ
00000045: 50 B4 09 CD 21 58 C3 1E BB 00 F0 8E DB 2B C0 8A P`oI!XÄ) øZÜ+Ä$
00000046: 26 FE FF 1F C3 50 53 57 BA 03 01 80 FC FF 74 47 &pýÄPSW0æÿtG
00000047: BA 11 01 80 FC FE 74 3F BA 11 01 80 FC FB 74 37 æüpt?æüüt7
00000048: BA 22 01 80 FC FC 74 2F BA 30 01 80 FC FA 74 27 æüüt/æüüt'
00000049: BA 48 01 80 FC FC 74 1F BA 66 01 80 FC F8 74 17 æüütæfæüüt±
0000004A: BA 7E 01 80 FC FD 74 0F BA 8E 01 80 FC F9 74 07 æüütæZæüüt•
0000004B: 8A C4 E8 4B FC 8B D0 E8 96 FF 5F 5B 58 C3 50 56 ŠÀèKÿ<Dè-ÿ_[XÄPV
0000004C: BE A8 01 83 C6 10 E8 62 FF 83 C6 03 8A C4 E8 5A %`ofæ-èbÿfÄ$ŠæZ
0000004D: FF BA A8 01 E8 79 FF 5E 58 C3 50 53 56 8A C7 BE ÿæÿÿ`XÄPSV$C%
0000004E: BF 01 83 C6 15 E8 43 FF BA BF 01 E8 62 FF 5E 5B æfA$èCÿæèbÿ^[
0000004F: 58 C3 51 57 50 BF D8 01 83 C7 19 8B C1 E8 11 FF XÄQWPæøfC±æ-ÿ

```

Рисунки 5,6-«плохой» .EXE модуль в шестнадцатеричном виде.

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

Ответ: «хороший» .EXE модуль, в отличие от «плохого» .EXE модуля содержит 3 отдельных сегмента – сегмент стека, сегмент данных,

сегмент кода. Код «хорошего» .EXE модуля располагается с адреса 400h.

```
просмотр LR1_2.EXE - Far 3.0.5354 x64
D:\Лабораторные работы\ОС (4 сем)\LabOS\LabOS\TASM\TASM\LR1_2.EXE
00000000: 4D 5A 45 00 04 00 01 00 20 00 00 00 FF FF 00 00 MZE 0 0 0 0 0 0 0 0
00000010: 00 02 00 00 00 00 30 00 3E 00 00 00 01 00 FB 50 0 0 > 0 0P
00000020: 6A 72 00 00 00 00 00 00 00 00 00 00 00 00 00 00 jr
00000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 26 01 0 0 0 0 0 0 0 0
00000040: 30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
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

просмотр LR1_2.EXE - Far 3.0.5354 x64
D:\Лабораторные работы\ОС (4 сем)\LabOS\LabOS\TASM\TASM\LR1_2.EXE
0000003D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000003E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000003F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000400: 54 79 70 65 20 50 43 3A 20 50 43 0D 0A 24 54 79 Type PC: PC\%$Ty
000000410: 70 65 20 50 43 3A 20 50 43 2F 58 54 0D 0A 24 54 pe PC: PC/XT\%$T
000000420: 79 70 65 20 50 43 3A 20 50 41 54 0D 0A 24 54 79 ype PC: AT\%$Typ
000000430: 65 20 50 43 3A 20 50 53 32 20 AC AE A4 A5 AB EC e PC: PS2 -%H%«i
000000440: 20 33 30 0A 0D 24 54 79 70 65 20 50 43 3A 20 50 30%$Type PC: P
000000450: 53 32 20 AC AE A4 A5 AB EC 20 35 30 20 6F 72 20 S2 -%H%«i 50 or
000000460: 36 30 0A 0D 24 54 79 70 65 20 50 43 3A 20 50 53 60%$Type PC: PS
000000470: 32 20 AC AE A4 A5 AB EC 20 38 30 0A 0D 24 54 79 2 -%H%«i 80%$Ty
000000480: 70 65 20 50 43 3A 20 50 43 6A 72 0A 0D 24 54 79 pe PC: PCjr\%$Ty
000000490: 70 65 20 50 43 3A 20 50 43 20 43 6F 6E 76 65 72 pe PC: PC Conver
0000004A0: 74 69 62 6C 65 0A 0D 24 56 65 72 73 69 6F 6E 20 tible%$Version
0000004B0: 4D 53 20 44 4F 53 3A 20 20 2E 20 20 0A 0D 24 4F MS DOS: . %$0
0000004C0: 45 4D 20 73 65 72 69 61 6C 20 6E 75 6D 62 65 72 EM serial number
0000004D0: 3A 20 20 20 20 0A 0D 24 55 73 65 72 20 73 65 72 : %$User ser
0000004E0: 69 61 6C 20 6E 75 6D 62 65 72 3A 20 20 20 20 20 ial number:
0000004F0: 20 20 0A 0D 24 00 00 00 00 00 00 00 00 00 00 00 %$
000000500: E9 22 01 24 0F 3C 09 76 02 04 07 04 30 C3 51 8A é"0$«ov00♦♦0A0$
000000510: E0 E8 EF FF 86 C4 B1 04 D2 E8 E8 E6 FF 59 C3 53 àeiÿtA±0èèÿVYAS
000000520: 8A FC E8 E9 FF 88 25 4F 88 05 4F 8A C7 32 E4 E8 $üèÿ"%"*+0$C2âè
000000530: DC FF 88 25 4F 88 05 5B C3 50 51 52 32 E4 33 D2 Üÿ"%"*+[\APQR2â30
000000540: B9 0A 00 F7 F1 80 CA 30 88 14 4E 33 D2 3D 0A 00 1  ÷ñ€E0"JN30=
000000550: 73 F1 3C 00 74 04 0C 30 88 04 5A 59 58 C3 50 B4 sñ< t♦90"ZYXÂP`
000000560: 09 CD 21 58 C3 1E BB 00 F0 8E DB 2B C0 8A 26 FE oI!XÂ» δZU+AS&b
000000570: FF 1F C3 50 53 57 BA 00 00 80 FC FF 74 47 BA 0E ÿ▼APSW€ €üÿtG€
000000580: 00 80 FC FE 74 3F BA 0E 00 80 FC FB 74 37 BA 1F €üÿt?€ €üÿt7€▼
000000590: 00 80 FC FC 74 2F BA 2D 00 80 FC FA 74 27 BA 46 €üÿt/€- €üÿt'€F
0000005A0: 00 80 FC FC 74 1F BA 65 00 80 FC F8 74 17 BA 7E €üÿt▼€e €üÿt±€~
0000005B0: 00 80 FC FD 74 0F BA 8E 00 80 FC F9 74 07 8A C4 €üÿto€Z €üÿt•$A
0000005C0: E8 4B FF 8B D0 E8 96 FF 5F 5B 58 C3 50 56 BE A8 èKÿ<Dè-ÿ [XÂPvX"
1Помощь 2Дамп 3Выход 4Текст 5 6Редакт 7Поиск
```

Рисунки 7,8-«хороший» .EXE модуль в шестнадцатеричном виде.
3.3. Загрузка СОМ модуля в основную память

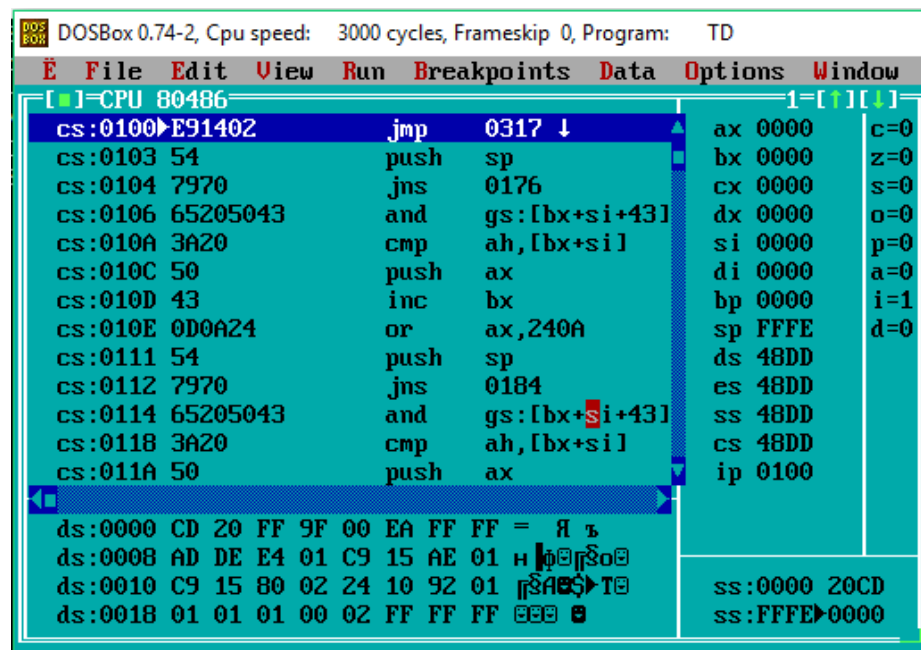


Рисунок 9 – отладчик TD.EXE для файла LR1.COM

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

Ответ: Формат загрузки модуля COM:

- 1)Выделение сегмента памяти для модуля.
- 2)Установка всех сегментных регистров на начало выделенного сегмента памяти.
- 3)Построение в первых 100h байтах памяти PSP.
- 4)Загрузка содержимого COM-файла и присваивание регистру IP значения 100h.
- 5)Регистр SP устанавливается в конец сегмента. Код начинается с адреса, содержащимся в CS, в нашем случае это 48DD.

3.3.2. Что располагается с адреса 0?

Ответ: С нулевого адреса располагается адрес начала PSP.

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

Ответ: Сегментные регистры DS, ES, SS, CS имеют значение 48DD. Они указывают на PSP.

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

Ответ: Стек генерируется автоматически. Стек занимает весь сегмент .COM программы. Сегментный регистр SS указывает на начало сегмента, а SP=FFFE на конец сегмента.

3.4. Загрузка «хорошего» EXE модуля в основную память

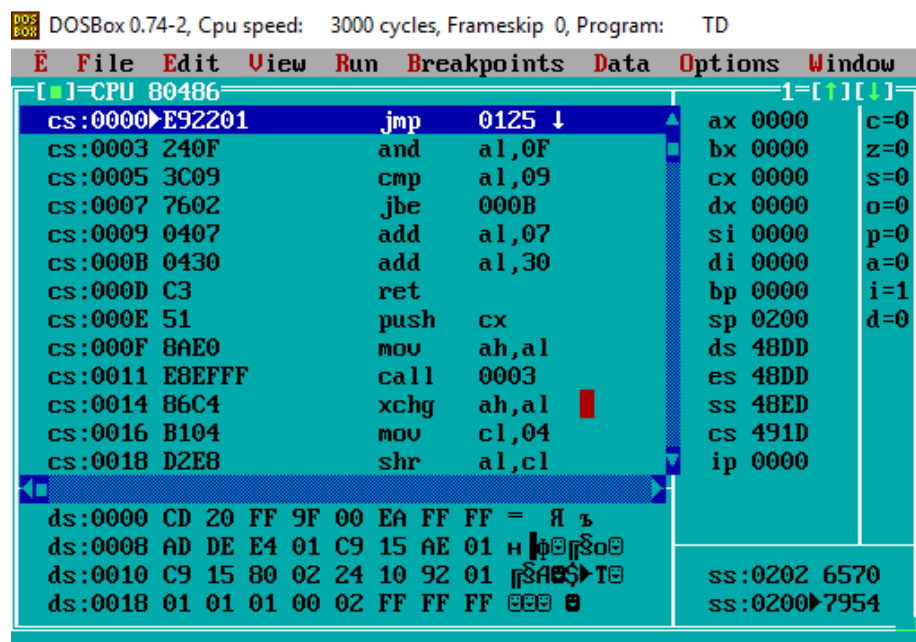


Рисунок 10 - TD.EXE для файла LR1_2.EXE

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

Ответ: В процессе загрузки .EXE модуля в память система пристраивает к началу программы дополнительный сегмент PSP. Система, загрузив программу в память, инициализирует сегментные регистры DS и ES, CS, SS. В данном случае DS=ES=48DD, CS=491D, SS=48ED.

3.4.2. На что указывают регистры DS и ES?

Ответ: Сегментные регистры DS и ES указывают на начало PSP.

3.4.3. Как определяется стек?

Ответ: Стек определяется с помощью директивы STACK. В момент исполнения выделяется указанный блок памяти. В регистр SS записывается адрес начала сегмента. Регистр SP указывает на вершину стека.

3.4.4. Как определяется точка входа?

Ответ: Последняя строка программы содержит директиву end. В качестве операнда этой директивы указывается точка входа в программу, т.е адрес первой выполняемой строки. В данном случае это метка START.

4. Заключение

В результате выполнения лабораторной работы были исследованы различия в структурах исходных текстов .COM и .EXE модулей, структур файлов загрузочных модулей и способов их загрузки в основную память.