

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. 0381

Котов Д.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

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

Постановка задачи.

Написать тексты исходных .COM и .EXE модулей, которые определяют тип PC и версию системы.

Ассемблерная программа должна читать содержимое предпоследнего байта ROM BIOS, по таблице (см. табл. 1), сравнивая коды, определять тип PC и выводить строку с названием модели. Если код не совпадает ни с одним значением, то двоичный код переводится в символьную строку, содержащую запись шестнадцатеричного числа и выводится на экран в виде соответствующего сообщения.

Затем программа должна определить версию 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.

Исходные данные.

За основу был взят предоставленный шаблон, содержащий процедуры:

TETR_TO_HEX, BYTE_TO_HEX_WRD_TO_HEX, BYTE_TO_DEC.

Таблица 1 – Соответствие типа IBM PC шестнадцатеричному коду.

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

Выполнение работы.

В файле `lb1_com.asm` был написан текст исходного `.COM` модуля.

Для вывода строк написана процедура `print`.

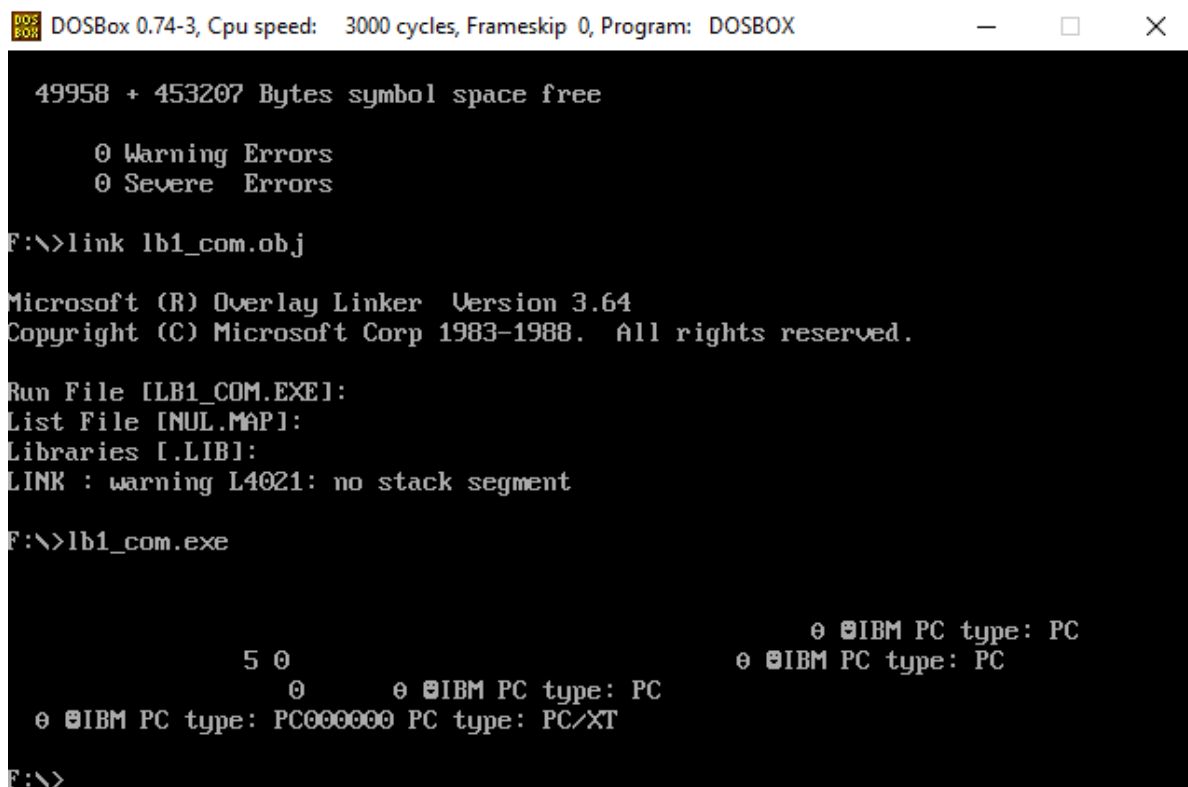
Подготовлены строки для вывода требуемых сообщений. Сообщения для типов IBM PC AT и PS2 модель 50 или 60 объединены в одно, т.к. их коды совпадают.

Написана процедура `pc_type_defenition`, которая определяет тип PC. В этой процедуре в `AL` сохраняется значение байта, в котором записан код системы и проводится сравнение значения с кодами из табл. 1. Если обнаруживается совпадение, то происходит переход к метке, в которой в `DX` заносится смещение соответствующего сообщения, после чего вызывается процедура `print` для его печати. Иначе выводится сообщение о том, что соответствие не было найдено с кодом системы.

Написана процедура `version_defenition`, которая определяет версию системы, серийный номер OEM и номер пользователя. Необходимые данные получаются с помощью функции `30h` прерывания `21h`.

С помощью команды `masm lb1_com.asm` был получен объектный файл `lb1_com.obj`. Командой `link lb1_com.obj` был собран «плохой» `.EXE` модуль.

Запустив lb1_com.exe было получено следующее сообщение:



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

49958 + 453207 Bytes symbol space free

  0 Warning Errors
  0 Severe Errors

F:\>link lb1_com.obj

Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

Run File [LB1_COM.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment

F:\>lb1_com.exe

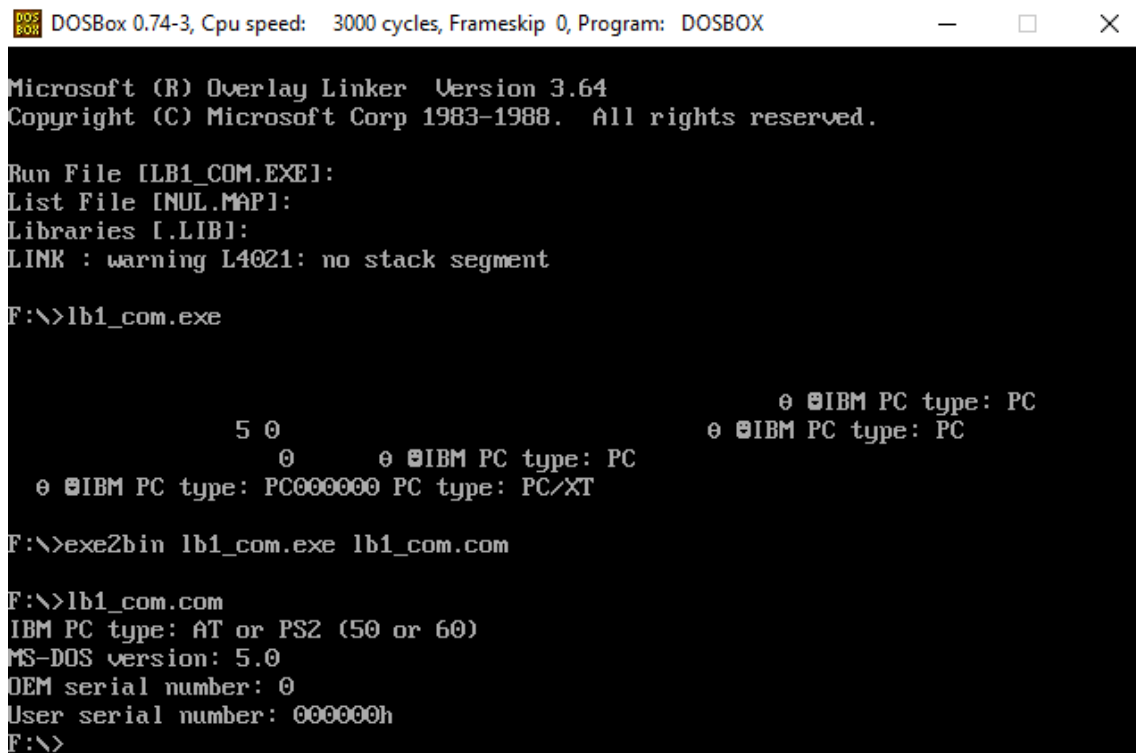
                    5 0
                    0 0 IBM PC type: PC
0 IBM PC type: PC0000000 PC type: PC/XT

F:\>
```

Рисунок 1 – Вывод lb1_com.exe

Командой exe2bin lb1_com.exe lb1_com.com получен .COM модуль.

Запустив его выводятся корректное сообщение:



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

Run File [LB1_COM.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : warning L4021: no stack segment

F:\>lb1_com.exe

                    5 0
                    0 0 IBM PC type: PC
0 IBM PC type: PC0000000 PC type: PC/XT

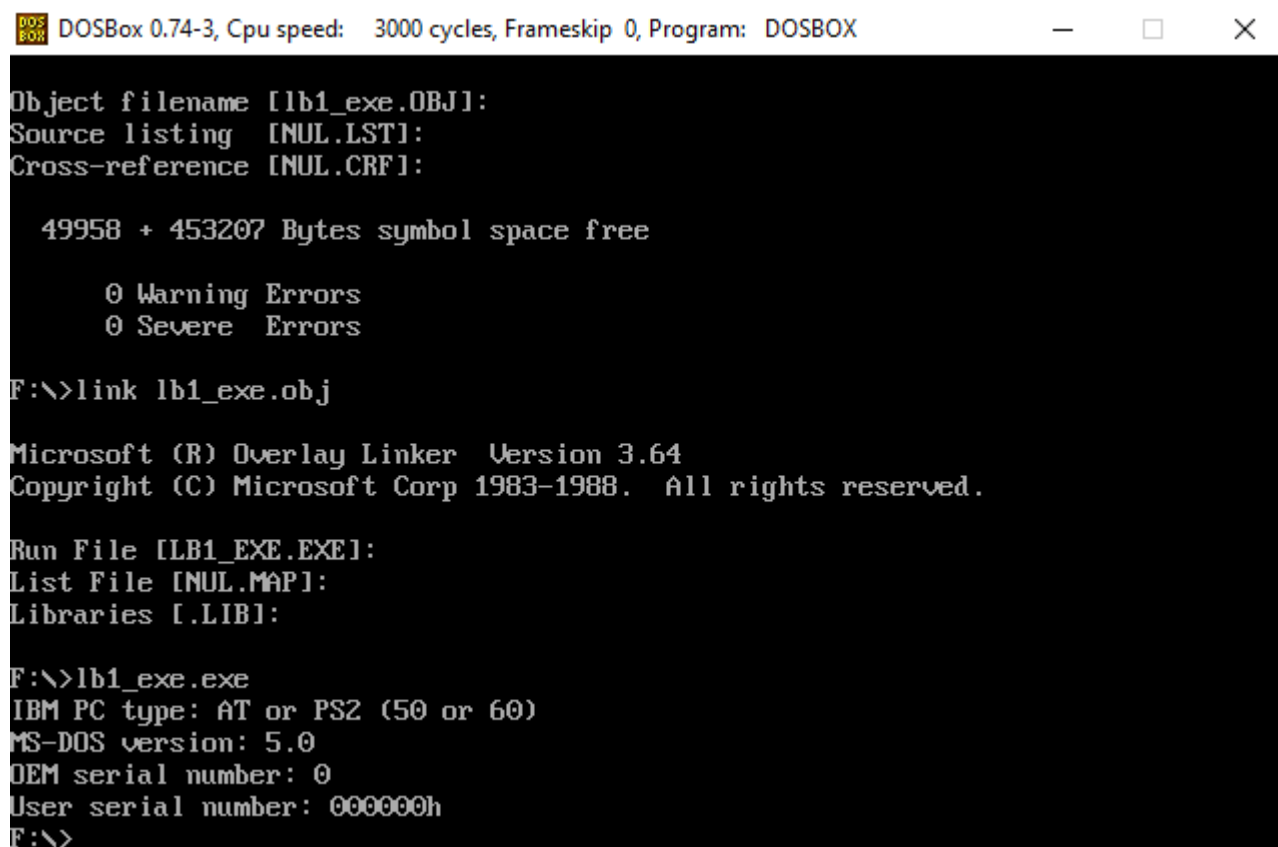
F:\>exe2bin lb1_com.exe lb1_com.com

F:\>lb1_com.com
IBM PC type: AT or PS2 (50 or 60)
MS-DOS version: 5.0
OEM serial number: 0
User serial number: 0000000h
F:\>
```

Рисунок 2 – Вывод lb1_com.com

В файле lb1_exe.asm был написан код «хорошего» .EXE модуля. Для этого был взят код из файла lb1_com.asm, после чего в него были внесены следующие изменения: добавлены определения сегмента стека и данных, строки перенесены в сегмент данных, была реализована процедура MAIN, в которой происходит загрузка адреса сегмента данных и вызов процедур pc_type_defenition и version_defenition.

После сборки и запуска lb1_exe.exe выводится корректное сообщение:



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Object filename [lb1_exe.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:

49958 + 453207 Bytes symbol space free

0 Warning Errors
0 Severe Errors

F:\>link lb1_exe.obj

Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

Run File [LB1_EXE.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:

F:\>lb1_exe.exe
IBM PC type: AT or PS2 (50 or 60)
MS-DOS version: 5.0
OEM serial number: 0
User serial number: 000000h
F:\>_
```

Рисунок 3 – Вывод lb1_exe.exe

Ответы на вопросы см. в разделе «Вопросы».

Выводы.

Были исследованы различия в структуре исходных текстов для модулей .COM и .EXE, структура загрузочных файлов этих типов и способ загрузки их в основную память.

ВОПРОСЫ

Отличия исходных текстов COM и EXE программ

- 1) Сколько сегментов должна содержать COM-программа?

Ровно один сегмент, содержащий данные и код. Стек генерируется автоматически.

- 2) EXE-программа?

Не менее одного сегмента, в котором содержится код. Могут присутствовать сегменты данных и стека. При наличии данные обязательно выносятся в отдельный сегмент. Если сегмент стека не задан, то используется стек DOS.

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

ORG 100h – смещение кода на 256 байт от нулевого адреса (пропуск области PSP)

ASSUME нужно использовать, чтобы сегментные регистры указывали на один сегмент.

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

Нет, команды с указанием сегментов использовать нельзя, так как в .COM модуле отсутствует таблица настройки адресов, по которой осуществляется поиск абсолютных адресов сегментов.

Отличия форматов файлов COM и EXE модулей

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

У COM файла есть только один сегмент, в котором располагаются код и данные. Модуль ограничен размером 64 Кб. Код размещается с нулевого адреса: сначала идёт команда jmp BEGIN, далее располагаются данные, дальше – код. Однако при загрузке модуля устанавливается смещение на 256 байт.

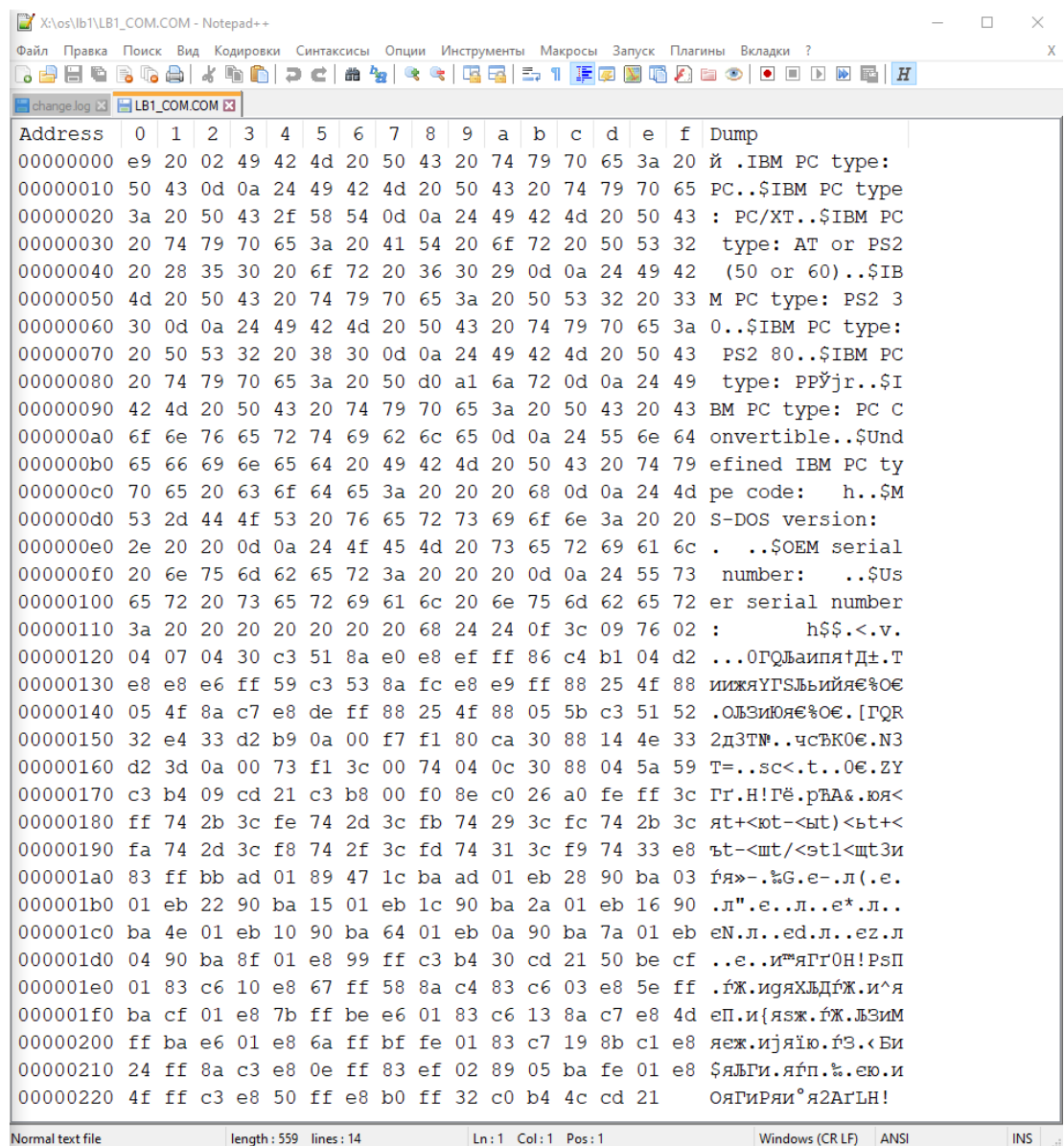


Рисунок 4 – Содержимое lb1_com.com

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

«Плохой» EXE файл содержит заголовок с технической информацией, необходимой для загрузки, таблицу настроек адресов и сегмент, в котором находятся данные и код. Код располагается с адреса 300h, так как до 200h размещены заголовок и таблица настроек, далее идёт смещение 100h, вызванное директивой ORG 100h.

```
X:\os\lb1\LB1_COM.EXE - Notepad++
Файл Правка Поиск Вид Кодировки Синтаксисы Опции Инструменты Макросы Запуск Плагины Вкладки ?
change.log LB1_COM.COM LB1_COM.EXE
Address 0 1 2 3 4 5 6 7 8 9 a b c d e f Dump
00000000 4d 5a 2f 01 03 00 00 00 20 00 00 00 ff ff 00 00 MZ/.....яя..
00000010 00 00 05 06 00 01 00 00 1e 00 00 00 01 00 00 00 .....
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 .....
00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
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 .....
Normal text file length: 1327 lines: 14 Ln: 1 Col: 1 Pos: 1 Windows (CR LF) ANSI INS
```

Рисунок 5 – Содержимое «плохого» EXE файла. Начало

```
X:\os\lb1\LB1_COM.EXE - Notepad++
Файл Правка Поиск Вид Кодировки Синтаксисы Опции Инструменты Макросы Запуск Плагины Вкладки ?
change.log LB1_COM.COM LB1_COM.EXE
Address 0 1 2 3 4 5 6 7 8 9 a b c d e f Dump
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 20 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 20 6f 72 20 50 53 32 type: AT or PS2
00000340 20 28 35 30 20 6f 72 20 36 30 29 0d 0a 24 49 42 (50 or 60)..$IB
00000350 4d 20 50 43 20 74 79 70 65 3a 20 50 53 32 20 33 M PC type: PS2 3
00000360 30 0d 0a 24 49 42 4d 20 50 43 20 74 79 70 65 3a 0..$IBM PC type:
00000370 20 50 53 32 20 38 30 0d 0a 24 49 42 4d 20 50 43 PS2 80..$IBM PC
00000380 20 74 79 70 65 3a 20 50 d0 a1 6a 72 0d 0a 24 49 type: PPyjr..$I
00000390 42 4d 20 50 43 20 74 79 70 65 3a 20 50 43 20 43 BM PC type: PC C
000003a0 6f 6e 76 65 72 74 69 62 6c 65 0d 0a 24 55 6e 64 convertible..$Und
000003b0 65 66 69 6e 65 64 20 49 42 4d 20 50 43 20 74 79 efined IBM PC ty
000003c0 70 65 20 63 6f 64 65 3a 20 20 20 68 0d 0a 24 4d pe code: h..$M
000003d0 53 2d 44 4f 53 20 76 65 72 73 69 6f 6e 3a 20 20 S-DOS version:
000003e0 2e 20 20 0d 0a 24 4f 45 4d 20 73 65 72 69 61 6c . ..$OEM serial
000003f0 20 6e 75 6d 62 65 72 3a 20 20 20 0d 0a 24 55 73 number: ..$Us
00000400 65 72 20 73 65 72 69 61 6c 20 6e 75 6d 62 65 72 er serial number
00000410 3a 20 20 20 20 20 20 20 68 24 24 0f 3c 09 76 02 : h$$$.<.v.
00000420 04 07 04 30 c3 51 8a e0 e8 ef ff 86 c4 b1 04 d2 ...0QЪаипл†Д†.Т
00000430 e8 e8 e6 ff 59 c3 53 8a fc e8 e9 ff 88 25 4f 88 иижяYГSЪийяе%Ое
00000440 05 4f 8a c7 e8 de ff 88 25 4f 88 05 5b c3 51 52 .ОЪийяе%Ое.[ГQР
00000450 32 e4 33 d2 b9 0a 00 f7 f1 80 ca 30 88 14 4e 33 2дЗТW..чс%КOе.NЗ
00000460 d2 3d 0a 00 73 f1 3c 00 74 04 0c 30 88 04 5a 59 T=..sc<.t..Ое.ZY
00000470 c3 b4 09 cd 21 c3 b8 00 f0 8e c0 26 a0 fe ff 3c Гр.Н!Г%.pPA&..юя<
00000480 ff 74 2b 3c fe 74 2d 3c fb 74 29 3c fc 74 2b 3c ят+<ют-<ют><ют+<
00000490 fa 74 2d 3c f8 74 2f 3c fd 74 31 3c f9 74 33 e8 ьт-<шт/<эт1<шт3и
000004a0 83 ff bb ad 01 89 47 1c ba ad 01 eb 28 90 ba 03 фя»-.%G.e-л(.e.
000004b0 01 eb 22 90 ba 15 01 eb 1c 90 ba 2a 01 eb 16 90 .л".e..л..e*.л..
000004c0 ba 4e 01 eb 10 90 ba 64 01 eb 0a 90 ba 7a 01 eb eN..ед.л..ез.л
000004d0 04 90 ba 8f 01 e8 99 ff c3 b4 30 cd 21 50 be cf ..e..и"яГ0Н!PсП
000004e0 01 83 c6 10 e8 67 ff 58 8a c4 83 c6 03 e8 5e ff .фж.идяXдфж.и"я
000004f0 ba cf 01 e8 7b ff be e6 01 83 c6 13 8a c7 e8 4d eП.(яжж.фж.ЪSиM
00000500 ff ba e6 01 e8 6a ff bf fe 01 83 c7 19 8b c1 e8 ясжж.ижялю.фS.<Би
00000510 24 ff 8a c3 e8 0e ff 83 ef 02 89 05 ba fe 01 e8 $яЪГи.ялп.%ею.и
00000520 4f ff c3 e8 50 ff e8 b0 ff 32 c0 b4 4c cd 21 ОяГиPяи"я2ATLH!
```

Рисунок 5 – Содержимое «плохого» EXE файла. Конец

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

«Хороший» EXE файл содержит заголовок и таблицу настройки адресов, их общая длина 200h. После таблицы идут три сегмента: стека, данных и кода, а в «плохом» EXE файле есть только один сегмент.

Загрузка COM модуля в основную память

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

Определяется сегментный адрес участка памяти, у которого достаточно места для загрузки программы. Он заносится в сегментные регистры. Первые 256 байт этого сегмента занимает PSP. С адреса 100h загружается содержимое COM-файла. Указатель стека SP устанавливается на конец этого сегмента и в стек записывается 0000h. В регистр IP записывается значение 100h.

- 2) Что располагается с адреса 0?

Префикс программного кода (PSP).

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

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFDPRO

AX 0000	SI 0000	CS 19F5	IP 0100	Stack +0 0000	Flags 7202
BX 0000	DI 0000	DS 19F5		+2 20CD	
CX 022F	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF
DX 0000	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 0 0 0 0 0

CMD >

0100 E92002	JMP	0323
0103 49	DEC	CX
0104 42	INC	DX
0105 4D	DEC	BP
0106 205043	AND	[BX+SI+43],DL
0109 207479	AND	[SI+79],DH
010C 7065	JD	0173
010E 3A20	CMP	AH,[BX+SI]

DS:0000	CD 20 FF 9F 00 EA F0 FE	AD DE 1B 05 C5 06 00 00
DS:0010	1B 01 10 01 18 01 92 01	01 01 01 00 02 FF FF FF
DS:0020	FF FF FF FF FF FF FF FF	FF FF FF FF EB 19 C0 11
DS:0030	A2 01 14 00 18 00 F5 19	FF FF FF FF 00 00 00 00
DS:0040	05 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00

1 Step 2ProcStep 3Retrieve 4Help ON 5BRK Menu 6 7 ↑ 8 ↓ 9 ← 10 →

Рисунок 6 – Отладчик AFDPRO.EXE с открытым COM-файлом

Из рис. 6 видно, что все сегментные регистры имеют одно и тоже значение (в данном случае 19F5), в этот сегмент и загружена программа.

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

Стек генерируется автоматически при создании программы и располагается в сегменте кода. Регистр SP указывает на конец стека, а SS – на начало. Адреса стека расположены в диапазоне 0000h – FFFEh.

Загрузка «хорошего» EXE модуля в основную память

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

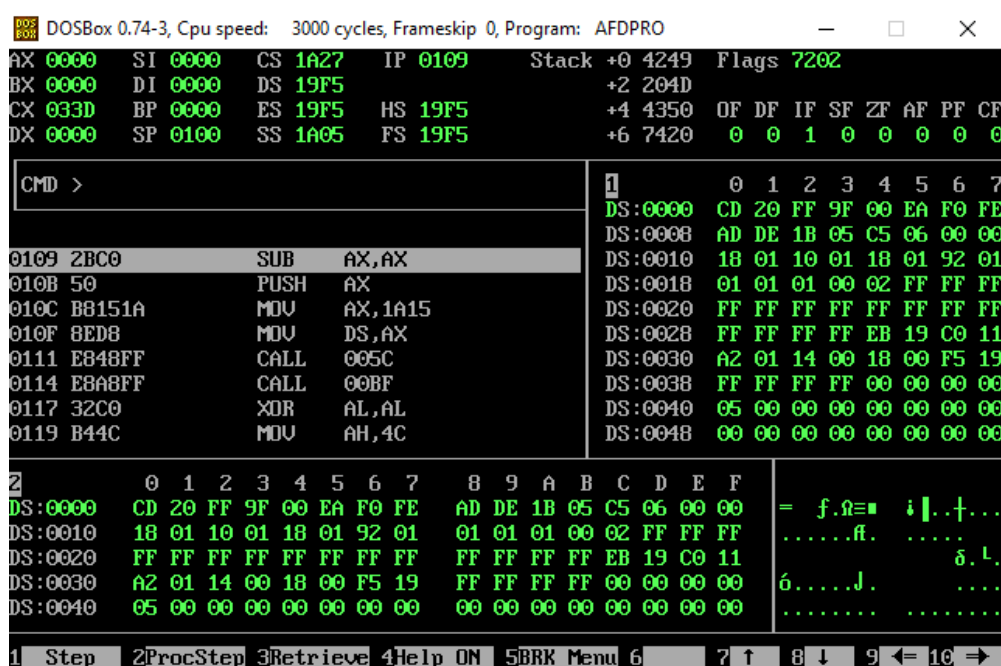


Рисунок 7 – Отладчик AFDPRO.EXE с открытым EXE-файлом

Определяется сегментный адрес свободного участка памяти, размера которого будет хватать для размещения программы. EXE-файл загружается, начиная с адреса PSP:0100h. В процессе загрузки считывается информация заголовка EXE в начале файла и выполняется перемещение адресов сегментов, DS и ES устанавливаются на начало сегмента PSP (DS = ES = 19F5), SS (SS = 1A05) – на начало сегмента стека, CS (CS = 1A27) – на начало сегмента кода. В IP загружается смещение точки входа в программу, которая берётся из метки после директивы END.

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`, которой обозначается начало сегмента стека или стандартной директивы `segment`, используя данную конструкцию:

```
<segment_name> segment stack
```

...

```
<segment_name> ends
```

Регистр SP указывает на конец стека, а SS – на начало.

4) Как определяется точка входа?

Точка входа определяется при помощи директивы END. После этой директивы указывается метка, куда переходит программа при запуске.