МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Операционные системы»

Тема: Исследование структур загрузочных модулей

Студент гр. 0381 _____ Магнитов С.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.

Задание.

Напишите 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. Отладьте полученный исходный модуль. Результатом выполнения этого шага будет «хороший» .COM модуль, а также необходимо построить «плохой» .EXE, полученный из исходного текста для .COM модуля.

Напишите текст исходного .EXE модуля, который выполняет те же функции, что и модуль в Шаге 1 и постройте и отладьте его. Таким образом, будет получен «хороший».

Теоретические сведения.

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 lab1com.asm написан код исходного .COM модуля. В начале файла прописаны строки для вывода сообщений о типах PC, серийном номере OEM и серийном номере пользователя. Для вывода этой информации создана процедура PRINT.

Написана процедура TYPE_PC, которая определяет тип PC. В AL сохраняется значение байта, в котором записан код системы. Далее происходит сравнение этого значения с кодами из таблицы 1, после этого происходит переход к нужной метке, в которой в DX заносится смещение соответствующего сообщения, в конце в метке m_print вызывается процедура PRINT для печати сообщения.

Написана процедура VERSION, которая определяет версию системы, серийный номер OEM и номер пользователя. Нужные данные получаются с помощью функции 30h прерывания 21h. После этого выводится три сообщения с помощью процедуры PRINT: SYSTEM_VERSION, OEM и USER.

Командой masm lab1com.asm был получен объектный файл lab1com.obj, из которого затем командой link lab1com.obj собирается «плохой» .EXE-модуль. Результат запуска lab1com.exe представлен на рисунке 1.

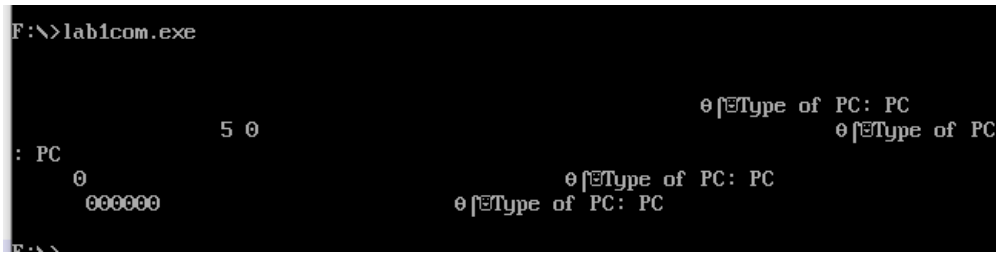


Рисунок 1 - Вывод модуля lab1com.exe

С помощью команды exe2bin lab1com.exe lab1com.com был получен .COM-модуль. Результат запуска lab1com.com представлен на рисунке 2.

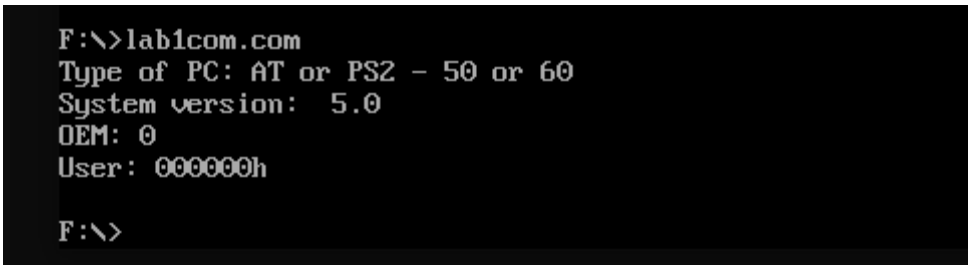


Рисунок 2 - Вывод модуля lab1com.com

В файле lab1exe.asm был написан код «хорошего» .EXE модуля. Для был взят код из файла lab1com.asm, а затем внесены корректировки. Во-первых, добавлены определения сегмента данных и сегмента стека, во-вторых, строки сообщений вынесены в сегмент данных, в-третьих, код, в котором вызывались процедуры TYPE_PC и VERSION вынесен в добавленную отдельную процедуру MAIN, в ней также присутствует загрузка адреса сегмента данных.

Результат запуска lab1exe.exe представлен на рисунке 3.

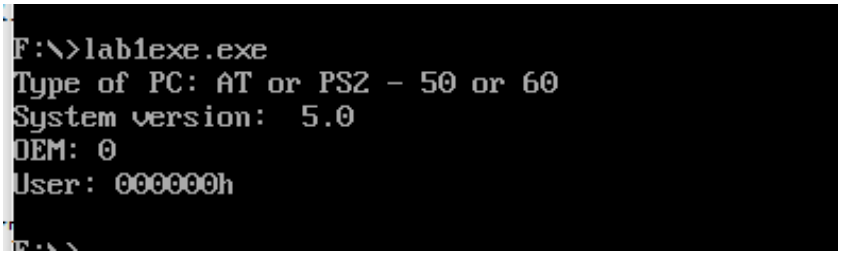


Рисунок 3 - Вывод модуля lab1exe.exe

Вывод.

Во время выполнения лабораторной работы были изучены различия в структурах исходных текстов модулей типов .COM и .EXE, структуры файлов загрузочных модулей и способы их загрузки в основную память.

Ответы на контрольные вопросы.

«Отличия исходных текстов COM и EXE программ»

1. Сколько сегментов должна содержать COM-программа?

COM-программа должна содержать один сегмент. Это сегмент кода, в котором определяются все данные. Для COM-программ стек генерируется автоматически.

2. EXE-программа?

EXE-программа как минимум должна содержать один сегмент. Это сегмент кода. Кроме этого она может содержать сегмент стека и сегмент данных. Данные обязательно нужно выносить в отдельный сегмент, а если не задать сегмент стека, то будет использован стек DOS.

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

В тексте COM-программы обязательно должны быть: директива ASSUME (указывает, что данные сегмент используется как сегмент кода и сегмент данных); директива ORG 100h (смещение кода от начала PSP, без этой директивы можно запустить модуль, но будет выведен некорректный результат).

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

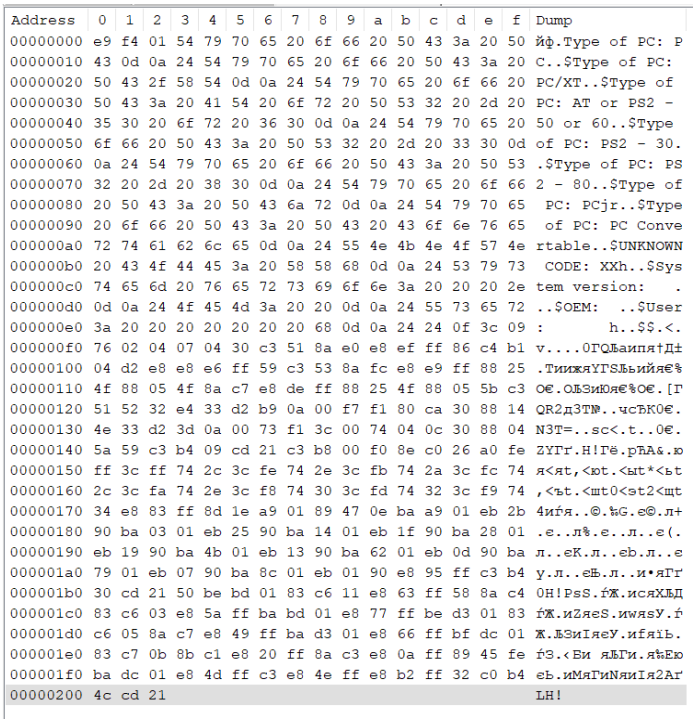
Нет, так как, например, команды, операндами которых являются сегменты, не могут быть выполнены из-за того, что в COM-модулях

отсутствует заголовок, в котором содержится таблица настройки. По этой таблице осуществляется поиск абсолютных адресов сегментов.

«Отличия форматов файлов COM и EXE модулей»

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

В файле COM имеется только один сегмент. Это сегмент кода, в нем располагаются данные и код. Код располагается с адреса 0, здесь это команда jmp BEGIN, после неё размещены данные-строки, а затем продолжение кода. При запуске модуля ко всем адресам будет добавлено смещение 100h, потому что в COM модулях для выделения 256 байт под PSP используется директива ORG 100h.



Address	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	Dump
00000000	e9	f4	01	54	79	70	65	20	6f	66	20	50	43	3a	20	50	йф.Type of PC: P
00000010	43	0d	0a	24	54	79	70	65	20	6f	66	20	50	43	3a	20	C..$Type of PC:
00000020	50	43	2f	58	54	0d	0a	24	54	79	70	65	20	6f	66	20	PC/XT..$Type of
00000030	50	43	3a	20	41	54	20	6f	72	20	50	53	32	20	2d	20	PC: AT or PS2 -
00000040	35	30	20	6f	72	20	36	30	0d	0a	24	54	79	70	65	20	50 or 60..$Type
00000050	6f	66	20	50	43	3a	20	50	53	32	20	2d	20	33	30	0d	of PC: PS2 - 30.
00000060	0a	24	54	79	70	65	20	6f	66	20	50	43	3a	20	50	53	.$Type of PC: PS
00000070	32	20	2d	20	38	30	0d	0a	24	54	79	70	65	20	6f	66	2 - 80..$Type of
00000080	20	50	43	3a	20	50	43	6a	72	0d	0a	24	54	79	70	65	PC: PCjr..$Type
00000090	20	6f	66	20	50	43	3a	20	50	43	20	43	6f	6e	76	65	of PC: PC Conve
000000a0	72	74	61	62	6c	65	0d	0a	24	55	4e	4b	4e	4f	57	4e	rtable..$UNKNOWN
000000b0	20	43	4f	44	45	3a	20	58	58	68	0d	0a	24	53	79	73	CODE: XXh..$Sys
000000c0	74	65	6d	20	76	65	72	73	69	6f	6e	3a	20	20	20	2e	tem version: .
000000d0	0d	0a	24	4f	45	4d	3a	20	20	0d	0a	24	55	73	65	72	..$OEM: ..$User
000000e0	3a	20	20	20	20	20	20	20	68	0d	0a	24	24	0f	3c	09	: h..$$.<.
000000f0	76	02	04	07	04	30	c3	51	8a	e0	e8	ef	ff	86	c4	b1	v....0ГQЉаипя†Д±
00000100	04	d2	e8	e8	e6	ff	59	c3	53	8a	fc	e8	e9	ff	88	25	.ТиижяYГSЉьийяЄ%
00000110	4f	88	05	4f	8a	c7	e8	de	ff	88	25	4f	88	05	5b	c3	O€.OЉЗиЮяЄ%O€.[Г
00000120	51	52	32	e4	33	d2	b9	0a	00	f7	f1	80	ca	30	88	14	QR2д3Т№..чсЪК0€.
00000130	4e	33	d2	3d	0a	00	73	f1	3c	00	74	04	0c	30	88	04	N3T=..sc<.t..0€.
00000140	5a	59	c3	b4	09	cd	21	c3	b8	00	f0	8e	c0	26	a0	fe	ZYГг.H!Гё.рЋА&.ю
00000150	ff	3c	ff	74	2c	3c	fe	74	2e	3c	fb	74	2a	3c	fc	74	я<яt,<юt.<ыt*<ьt
00000160	2c	3c	fa	74	2e	3c	f8	74	30	3c	fd	74	32	3c	f9	74	,<ъt.<шt0<эt2<щt
00000170	34	e8	83	ff	8d	1e	a9	01	89	47	0e	ba	a9	01	eb	2b	4иѓя..©.‰G.е©.л+
00000180	90	ba	03	01	eb	25	90	ba	14	01	eb	1f	90	ba	28	01	.е..л%.е..л..е(.
00000190	eb	19	90	ba	4b	01	eb	13	90	ba	62	01	eb	0d	90	ba	л..еК.л..еb.л..е
000001a0	79	01	eb	07	90	ba	8c	01	eb	01	90	e8	95	ff	c3	b4	y.л..еЊ.л..и•яГґ
000001b0	30	cd	21	50	be	bd	01	83	c6	11	e8	63	ff	58	8a	c4	0H!PsS.ѓЖ.исяХЉД
000001c0	83	c6	03	e8	5a	ff	ba	bd	01	e8	77	ff	be	d3	01	83	ѓЖ.иZяєS.ишяsУ.ѓ
000001d0	c6	05	8a	c7	e8	49	ff	ba	d3	01	e8	66	ff	bf	dc	01	Ж.ЉЗиIяєУ.ифяїЬ.
000001e0	83	c7	0b	8b	c1	e8	20	ff	8a	c3	e8	0a	ff	89	45	fe	ѓЗ.‹Би яЉГи.я‰Ею
000001f0	ba	dc	01	e8	4d	ff	c3	e8	4e	ff	e8	b2	ff	32	c0	b4	єЬ.иМяГиNяиIя2Аґ
00000200	4c	cd	21														LH!

Рисунок 4 - Содержимое lab1com.com

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

«Плохой» EXE содержит заголовок с технической информацией, а также единственный сегмент, в котором расположены данные и код. Заголовок

располагается с адреса 0, в нем содержатся сведения о размере модуля, адресе стека, относительных смещениях, адресе точки входа и др. А код начинается с адреса 300h.

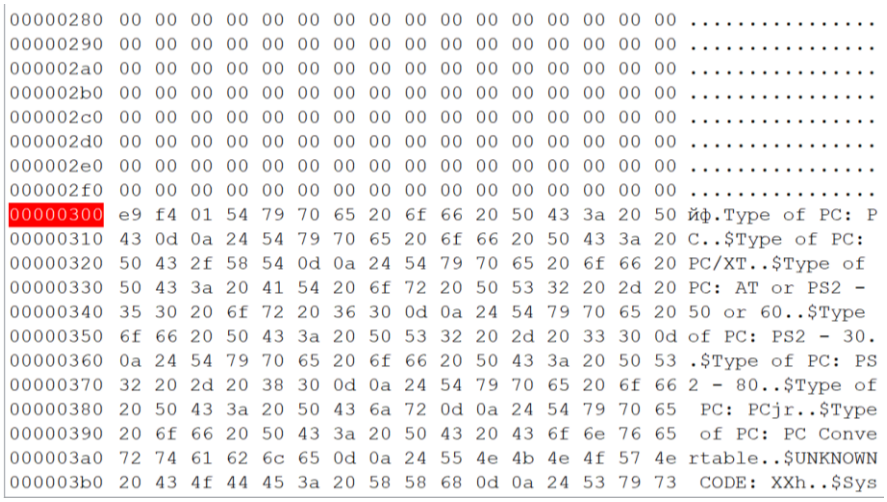


Рисунок 5 - Часть содержимого файла lab1com.exe. Красным выделен адрес начала кода

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

«Хороший» EXE в начале содержит заголовок с технической информацией, далее идет сегмент стека, у которого адреса 200h-600h, потому что на стек было выделено 512 слов по 2 байта, затем - сегмент данных с адресами 600h-6F0h и сегмент кода с адреса 6F0h и до конца файла. От «плохого» EXE данный модуль отличается тем, что у него происходит деление на сегменты, то есть данные отдельно от кода.

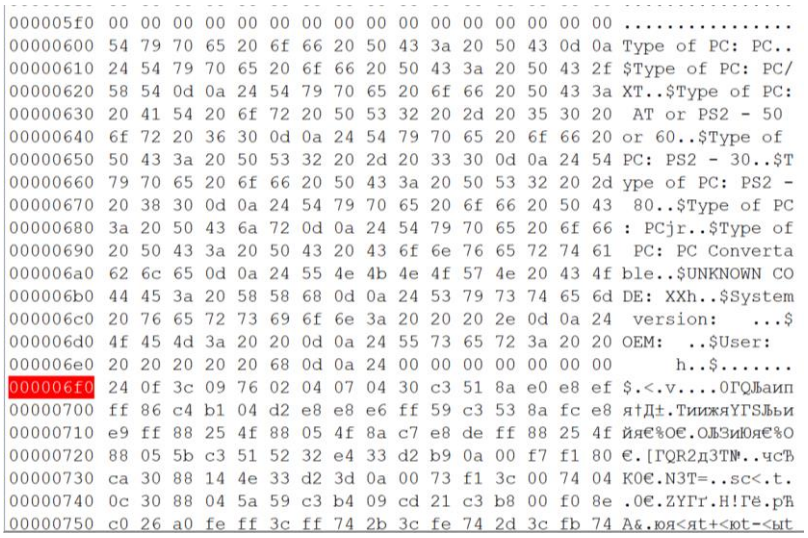


Рисунок 6 - Часть содержимого файла lab1exe.exe. Красным выделен адрес начала кода и сегмента кода

«Загрузка COM модуля в основную память»

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

Для загрузки программы определяется сегментный адрес свободного участка памяти, далее создается блок памяти для переменных среды и блок памяти для PSP и программы. В блок памяти переменных среды помещается пусть к файлу программы, заполняется PSP. Чтение программы выполняется с ее записью по адресу PSP:0100h. Код начинается с адреса 0100h.

2. Что располагается с адреса 0?

С адреса 0 располагается префикс программного сегмента (PSP).

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

Сегментные регистры CS, DS, SS, ES имеют значения 48DD и в начале программы указывают на начало PSP.

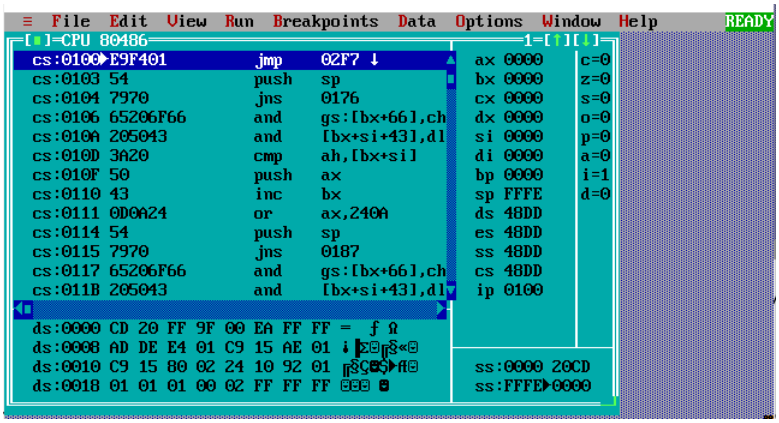


Рисунок 7 - Отладчик td.exe с открытым COM-файлом.

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

При запуске программы стек определяется автоматически и располагается в сегменте кода. Указатель стека установлен на конец сегмента – FFFE, это говорит о том, что под стек отводится часть сегмента после данных и кода.

«Загрузка «хорошего» EXE модуля в основную память»

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

Сначала определяется сегментный адрес свободного участка памяти для загрузки программы, затем создается блок памяти для переменных среды и блок памяти для PSP и программы. В блок памяти переменных среды помещается пусть к файлу программы, заполняется PSP. Считывается форматированная часть заголовка файла, на основе данных в ней определяется размер загрузочного модуля, смещение его начала.

2. На что указывают регистры DS и ES?

В начале выполнения программы регистры DS и ES указывают на начало PSP.

3. Как определяется стек?

Стек определяется с помощью директивы .STACK, которой обозначается начало сегмента стека (после неё задается размер стека). Также его можно определить с помощью стандартной директивы SEGMENT, использовав команду:

Название SEGMENT STACK - Название ENDS.

4. Как определяется точка входа?

Точка входа определяется в сегменте кода после директивы END.

END Название_точки_выхода