

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. 0381

Балакин В.

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

1. Напишите текст исходного **.COM** модуля, который определяет тип РС и версию системы. Это довольно простая задача и для тех, кто уже имеет опыт программирования на ассемблере, это будет небольшой разминкой. Для тех, кто раньше не сталкивался с программированием на ассемблере, это неплохая задача для первого опыта. За основу возьмите шаблон, приведенный в разделе «Основные сведения». Необходимые сведения о том, как извлечь требуемую информацию, представлены в следующем разделе. Ассемблерная программа должна читать содержимое предпоследнего байта ROM BIOS, по таблице, сравнивая коды, определять тип РС 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.

2. 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» **.EXE**.

3. Сравнение исходных текстов для **.COM** и **.EXE** модулей. Ответьте на контрольные вопросы «Отличия исходных текстов COM и EXE программ».

4. Запустите FAR и откройте (F3/F4) файл загрузочного модуля .COM и файл «плохого» .EXE в шестнадцатеричном виде. Затем откройте (F3/F4) файл загрузочного модуля «хорошего» .EXE и сравните его с предыдущими файлами. Ответьте на контрольные вопросы «Отличия форматов файлов COM и EXE модулей».

5. Откройте отладчик TD.EXE и загрузите .COM. Ответьте на контрольные вопросы «Загрузка COM модуля в основную память». Представьте в отчете план загрузки модуля .COM в основную память.

6. Откройте отладчик TD.EXE и загрузите «хороший» .EXE. Ответьте на контрольные вопросы «Загрузка «хорошего» EXE модуля в основную память».

7. Оформление отчета в соответствии с требованиями. В отчете необходимо привести скриншоты. Для файлов их вид в шестнадцатеричном виде, для загрузочных модулей – в отладчике.

Таблица 1. Процедуры.

TETR_TO_HEX	-	Перевод десятичной цифры в AL в код 10 с.с.
BYTE_TO_HEX	-	Перевод значения AX в 16 с.с.
WRD_TO_HEX	-	Перевод значения AL в 16 с.с.
BYTE_TO_DEC	-	Перевод значения AL в 10 с.с.

Таблица 2. Макросы.

WRITE_MSG mgs	-	Вывод строки через int 21h – AH = 09h
MACRO_IF val, pctype	-	Сравнение AL и val, при совпадении выполняется вывод строки pctype

Выполнение работы.

Объявлены строки для вывода информации:

- PC db 'IBM PC type: PC',0DH,0AH,'\$' ;FF
- PCXT db 'IBM PC type: PC/XT',0DH,0AH,'\$' ;FE, FB
- PCJR db 'IBM PC type: PCjr',0DH,0AH,'\$' ;FD
- AT db 'IBM PC type: AT',0DH,0AH,'\$' ;FC
- PSTWOTHIRTY db 'IBM PC type: PS model 30',0DH,0AH,'\$' ;FA
- PCC db 'IBM PC type: PC Convertible',0DH,0AH,'\$' ;F9
- PSTWOEIGHTY db 'IBM PC type: PC model 80',0DH,0AH,'\$' ;F8
- VERSION db 'MS DOS version: 01. ',0DH,0AH,'\$'
- OEM db 'OEM: ',0DH,0AH,'\$'
- USER db 'User: H',0DH,0AH,'\$'

В результате были получены несколько модулей:

Рис. 1 - «хороший» .COM модуль.

```
F:\>1b1com
IBM PC type: AT
MS DOS version: 05.0
OEM:240
User: 000000H
```

Рис. 2 - «плохой» .EXE модуль.

```
F:\>1b1com.exe

               щ+⊕IBM PC type: PC
          5 0
    щ+⊕IBM PC type: PC
240
e: PC                щ+⊕IBM PC typ
    000000           щ+⊕IBM PC type: PC
F:\>
```

Рис. 3 - «хороший» .EXE
модуль.

```
F:\>1b1exe
IBM PC type: AT
MS DOS version: 05.0
OEM:240
User: 000000H
```

Вывод.

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

Ответы на контрольные вопросы.

Отличия исходных текстов .COM и .EXE программ

1) Сколько сегментов должна содержать .COM программа?

1) Один сегмент, содержащий в себе и код, и данные. Стек генерируется автоматически.

2) .EXE программа?

2) Три сегмента: сегмент кода, сегмент данных и сегмент стека. Иногда сегменты можно объединять, например, выделить память для стека в сегменте кода. Стек должен быть объявлен только в основной модуле (при сборке нескольких модулей).

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

3) Необходимо обеспечить смещение в 256 байт от нулевого адреса (ORG 100h), чтобы не попасть в область PSP. Также необходимо использовать ASSUME (CS:TESTPC, DS:TESTPC, etc), чтобы сегментные регистры указывали на один сегмент.

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

4) Нет. В .COM программах отсутствует таблица настройки (relocation table), поэтому команды с указанием сегментов не могут быть

выполнены. Таблица настройки используется для того, чтобы при загрузке настроить адреса. Но это лишь в том случае, если в программе используются адреса сегментов. В противном случае таблица перемещения не содержит элементов.

Отличия форматов файлов . COM и . EXE модулей

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

1) Модуль .COM состоит из одного сегмента, состоящего из сегмента кода и сегмента данных, сегмент стека генерируется автоматически при создании программы. Модуль ограничен размером одного сегмента и не превышает 64 Кб. Код начинается с адреса 0h, но при загрузке модуля устанавливается смещение в 100h.

Address	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	Dump
00000000	e9	2b	01	49	42	4d	20	50	43	20	74	79	70	65	3a	20	é+.IBM PC type:
00000010	50	43	0d	0a	24	49	42	4d	20	50	43	20	74	79	70	65	PC..\$IBM PC type
00000020	3a	20	50	43	2f	58	54	0d	0a	24	49	42	4d	20	50	43	: PC/XT..\$IBM PC
00000030	20	74	79	70	65	3a	20	50	43	6a	72	0d	0a	24	49	42	type: PCjr..\$IB
00000040	4d	20	50	43	20	74	79	70	65	3a	20	41	54	0d	0a	24	M PC type: AT..\$
00000050	49	42	4d	20	50	43	20	74	79	70	65	3a	20	50	53	20	IBM PC type: PS
00000060	6d	6f	64	65	6c	20	33	30	0d	0a	24	49	42	4d	20	50	model 30..\$IBM P
00000070	43	20	74	79	70	65	3a	20	50	43	20	43	6f	6e	76	65	C type: PC Conve
00000080	72	74	69	62	6c	65	0d	0a	24	49	42	4d	20	50	43	20	rtible..\$IBM PC
00000090	74	79	70	65	3a	20	50	43	20	6d	6f	64	65	6c	20	38	type: PC model 8
000000a0	30	0d	0a	24	4d	53	20	44	4f	53	20	76	65	72	73	69	0..\$MS DOS versi
000000b0	6f	6e	3a	20	30	31	2e	20	20	20	0d	0a	24	4f	45	4d	on: 01. ..\$OEM
000000c0	3a	20	20	20	0d	0a	24	55	73	65	72	3a	20	20	20	20	: ..\$User:
000000d0	20	20	20	48	0d	0a	24	24	0f	3c	09	76	02	04	07	04	H..\$\$.<.v....
000000e0	30	c3	51	8a	e0	e8	ef	ff	86	c4	b1	04	d2	e8	e8	e6	0AQŠàèiy†À±.Òèèæ
000000f0	ff	59	c3	53	8a	fc	e8	e9	ff	88	24	4e	88	04	4e	8a	yyĀSŠuèéy^\$N^..NŠ
00000100	c7	e8	de	ff	88	24	4e	88	04	5b	c3	51	52	32	e4	33	ÇèBy^\$N^.[ĀQR2a3
00000110	d2	b9	0a	00	f7	f1	80	ca	30	88	14	4e	33	d2	3d	0a	Ò¹...+ñeĒ0^..N3Ò=.
00000120	00	73	f1	3c	00	74	04	0c	30	88	04	5a	59	c3	bb	00	.sñ<.t..0^..ZYĀ».
00000130	f0	8e	c3	26	a0	fe	ff	3c	f8	72	70	3c	ff	75	0a	ba	ŠŽĀ&.py<ørp<yü.°
00000140	03	01	b4	09	cd	21	eb	72	90	3c	fe	75	0a	ba	15	01	..'.í!èr.<pu.°..
00000150	b4	09	cd	21	eb	64	90	3c	fb	75	0a	ba	15	01	b4	09	'.í!èd.<úu.°...'
00000160	cd	21	eb	56	90	3c	fd	75	0a	ba	2a	01	b4	09	cd	21	í!èv.<yü.°*..'.í!
00000170	eb	48	90	3c	fc	75	0a	ba	3e	01	b4	09	cd	21	eb	3a	èH.<úu.°>..'.í!è:
00000180	90	3c	fa	75	0a	ba	50	01	b4	09	cd	21	eb	2c	90	3c	.<úu.°P..'.í!è,.<
00000190	f8	75	0a	ba	89	01	b4	09	cd	21	eb	1e	90	3c	f9	75	ou.°%..'.í!è...<úu
000001a0	0a	ba	b6	01	b4	09	cd	21	eb	10	90	e8	34	ff	8a	fc	.°k..'.í!è...è4yŠu
000001b0	8a	d0	b4	06	cd	21	8a	d7	cd	21	b4	30	cd	21	be	a4	ŠĐ'.í!Š×í!'0í!¼a
000001c0	01	83	c6	11	3c	00	74	07	8a	f4	e8	3e	ff	8a	c6	83	.fĒ.<.t..Šðè>yŠĒf
000001d0	c6	03	e8	36	ff	ba	a4	01	b4	09	cd	21	8a	c7	be	bd	Ē.è6y°a..'.í!ŠÇ¼s
000001e0	01	83	c6	06	e8	24	ff	ba	bd	01	b4	09	cd	21	be	c7	.fĒ.èŠy°%s..'.í!¼Ç
000001f0	01	83	c6	0b	8b	c1	e8	fa	fe	8a	c3	e8	e4	fe	83	ee	.fĒ.<ĀèùpŠĀèàpfí
00000200	02	89	04	ba	c7	01	b4	09	cd	21	32	c0	b4	4c	cd	21	.%..°Ç..'.í!2Ā'Lí!

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

2) В «плохом» .EXE данные и код располагаются в одном сегменте, что для .EXE некорректно, так как код и данные должны быть разделены на отдельные сегменты. Код располагается с адреса 300h, а с адреса 0h - заголовок .EXE модуля.

Address	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	Dump
00000270	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000280	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000290	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000002a0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000002b0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000002c0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000002d0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
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	2b	01	49	42	4d	20	50	43	20	74	79	70	65	3a	20	é+.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	50	43	6a	72	0d	0a	24	49	42	type: PCjr..\$IB
00000340	4d	20	50	43	20	74	79	70	65	3a	20	41	54	0d	0a	24	M PC type: AT..\$
00000350	49	42	4d	20	50	43	20	74	79	70	65	3a	20	50	53	20	IBM PC type: PS
00000360	6d	6f	64	65	6c	20	33	30	0d	0a	24	49	42	4d	20	50	model 30..\$IBM P
00000370	43	20	74	79	70	65	3a	20	50	43	20	43	6f	6e	76	65	C type: PC Conve
00000380	72	74	69	62	6c	65	0d	0a	24	49	42	4d	20	50	43	20	rtible..\$IBM PC
00000390	74	79	70	65	3a	20	50	43	20	6d	6f	64	65	6c	20	38	type: PC model 8
000003a0	30	0d	0a	24	4d	53	20	44	4f	53	20	76	65	72	73	69	0..\$MS DOS versi
000003b0	6f	6e	3a	20	30	31	2e	20	20	20	0d	0a	24	4f	45	4d	on: 01. ..\$OEM
000003c0	3a	20	20	20	0d	0a	24	55	73	65	72	3a	20	20	20	20	: ..\$User:
000003d0	20	20	20	48	0d	0a	24	24	0f	3c	09	76	02	04	07	04	H..\$.<.v....
000003e0	30	c3	51	8a	e0	e8	ef	ff	86	c4	b1	04	d2	e8	e8	e6	0ÄQ5aëiy+Ä±.0ëëæ
000003f0	ff	59	c3	53	8a	fc	88	89	ff	88	24	4a	88	04	4a	8a	0vÏçëñäëö-äN- Në

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

3) В .EXE код, данные и стек поделены на сегменты. Программа в формате .EXE может иметь любой размер. Файл имеет заголовок, который используется при его загрузке. Заголовок состоит из форматированной части, содержащей сигнатуру (4D5Ah) и данные, необходимые для загрузки EXE-файла: длина образа задачи, число элементов таблицы настройки адресов, значение IP при входе в задачу и т.д. В отличие от «плохого» EXE в «хорошем» EXE присутствуют три сегмента: сегмент кода, сегмент данных и сегмент стека, а «плохой» EXE содержит один сегмент, совмещающий код и данные. Также в «плохом» EXE адресация кода начинается с 300h, так как он

получается из .COM файла, в котором изначально сегмент кода смещён на 100h, а при создании «плохого» EXE к этому смещению добавляется размер заголовка. В данной случае смещение кода 2f0h, так как выделяется память под стек (200h, 16 байт) и данные (210h).

Address	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	Dump
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	49	42	4d	20	50	43	20	74	79	70	65	3a	20	50	43	0d	IBM PC type: PC.
00000220	0a	24	49	42	4d	20	50	43	20	74	79	70	65	3a	20	50	..\$IBM PC type: P
00000230	43	2f	58	54	0d	0a	24	49	42	4d	20	50	43	20	74	79	C/XT..\$IBM PC ty
00000240	70	65	3a	20	50	43	6a	72	0d	0a	24	49	42	4d	20	50	pe: PCjr..\$IBM P
00000250	43	20	74	79	70	65	3a	20	41	54	0d	0a	24	49	42	4d	C type: AT..\$IBM
00000260	20	50	43	20	74	79	70	65	3a	20	50	53	20	6d	6f	64	PC type: PS mod
00000270	65	6c	20	33	30	0d	0a	24	49	42	4d	20	50	43	20	74	el 30..\$IBM PC t
00000280	79	70	65	3a	20	50	43	20	43	6f	6e	76	65	72	74	69	type: PC Converti
00000290	62	6c	65	0d	0a	24	49	42	4d	20	50	43	20	74	79	70	ble..\$IBM PC typ
000002a0	65	3a	20	50	43	20	6d	6f	64	65	6c	20	38	30	0d	0a	pe: PC model 80..
000002b0	24	4d	53	20	44	4f	53	20	76	65	72	73	69	6f	6e	3a	\$MS DOS version:
000002c0	20	30	31	2e	20	20	20	0d	0a	24	4f	45	4d	3a	20	20	01. ..\$OEM:
000002d0	20	0d	0a	24	55	73	65	72	3a	20	20	20	20	20	20	20	..\$User:
000002e0	48	0d	0a	24	00	00	00	00	00	00	00	00	00	00	00	00	H..\$.
000002f0	1e	33	c0	50	b8	01	00	8e	d8	8e	c0	bb	00	f0	8e	c3	.3ÄP,...žžžÄ».ožžÄ
00000300	26	a0	fe	ff	3c	f8	72	70	3c	ff	75	0a	ba	00	00	b4	&.bŸ<ørp<Ÿu.°..'
00000310	09	cd	21	eb	72	90	3c	fe	75	0a	ba	12	00	b4	09	cd	.f!ër.<pu.°...'.f

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

4) Стек находится между PSP и данными и занимает 1 параграф (в данном случае. При количестве байт, не кратном 16, добавляется отступ).

Загрузка .COM модуля в основную память

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

1) Определяется сегментный адрес участка ОП, у которого достаточно места для загрузки программы, файл считывается с диска и помещается в память, начиная с PSP:0100h. После загрузки сегментные регистры CS, DS, ES и SS указывают на PSP, SP указывает на конец сегмента PSP, слово 00H помещено в стек, IP содержит 100H в результате команды JMP PSP:100H.

2) Что располагается с адреса 0?

2) PSP.(Program Segment Prefix)

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

3) Сегментные регистры CS, DS, ES и SS указывают на сегмент кода, сегмент данных, дополнительные данные и стек соответственно. В начале выполнения программы регистры указывают на начало PSP.

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

4).Стек генерируется автоматически при создании .COM программы и находится перед PSP. SS указывает на начало PSP, регистр SP указывает на конец стека. Адреса стека расположены в диапазоне FFFh – 0h.

Загрузка «хорошего» EXE модуля в основную память

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

1)

1.В области памяти после резидентной части выполняющей загрузку программы строится PSP;

2.Стандартная часть заголовка считывается в память;

3.Определяется длина тела загрузочного модуля (разность длины файла 04-07 и длины заголовка 08-09 плюс число байт в последнем блоке 02-03). В зависимости от признака, указывающего загружать задачу в конец памяти или в начало, определяется сегментный адрес для загрузки. Этот сегмент называется *начальным сегментом*;

4.Загрузочный модуль считывается в начальный сегмент;

5.Таблица настройки порциями считывается в рабочую память;

6.Для каждого элемента таблицы настройки к полю сегмента прибавляется сегментный адрес начального сегмента. В результате элемент таблицы указывает на слово в памяти, к которому прибавляется сегментный адрес начального сегмента;

7.Когда таблица настройки адресов обработана, в регистры SS и SP записываются значения, указанные в заголовке, а к SS прибавляется сегментный адрес начального сегмента. В ES и DS записывается сегментный адрес начала PSP. Управление передается по адресу, указанному в заголовке (байты 14-17).

2) На что указывают регистры DS и ES??

2) В начале выполнения программы они указывают на PSP.

3) Как определяется стек?

3) Стек определяется с помощью директивы `.stack`, после которой задаётся размер стека, либо с помощью конструкции: `name`

`Segment[READONLY] [align][combine][use][character] alias(string)[class]`
`statements`

`name ENDS`

4) Как определяется точка входа?

4) Директивой `END`.