

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. 8382

Рочева А.К.

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 их загрузки в основную память.

Ход выполнения.

1. Для определения типа PC и версии системы были написаны тексты .COM и .EXE модулей (см. Приложение А и В). Тип PC определяется предпоследним байтом ROM BIOS (табл. 1).

Таблица 1 – Соответствие кода и типа

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

Версия системы определяется значением регистров AL, AH, BH, BL:CH, полученных после выполнения функции 30H прерывания 21H. (в AL – номер основной версии, в AH – номер модификации, в 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:\>TYPECOM.COM
AT
Version:
5.0
OEM number:
0
User serial number:
000000

```

рис. 1 – результат выполнения .COM модуля

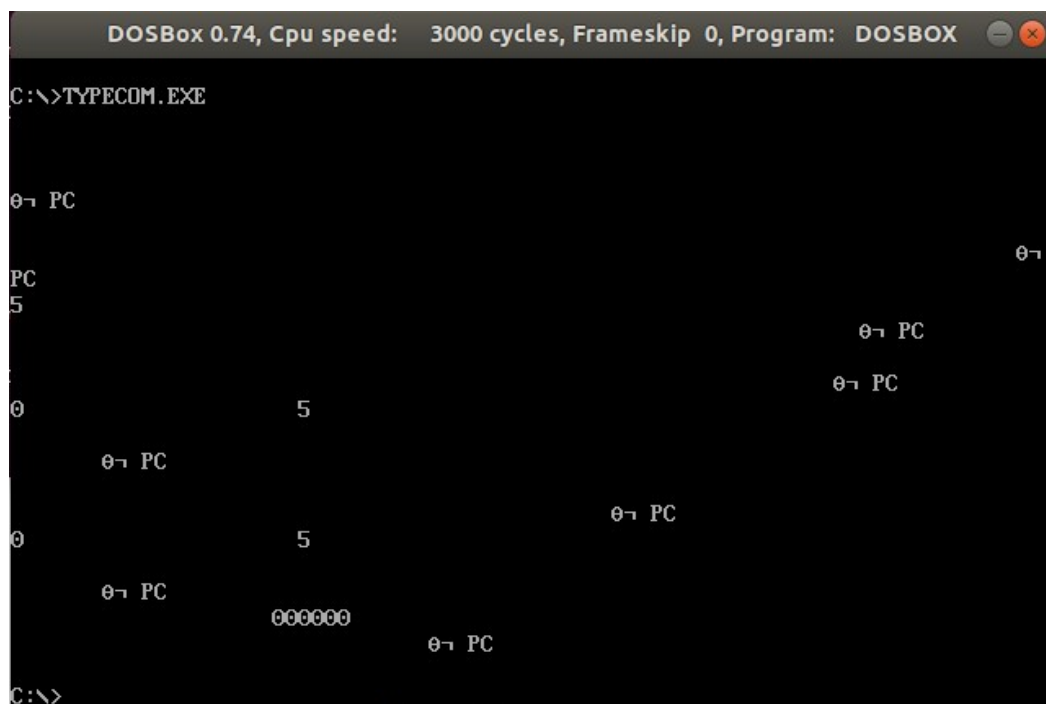


рис. 2 – результат выполнения «плохого» .EXE модуля

```

C:\>TYPEEXE.EXE
AT
Version:
5.0
OEM number:
0
User serial number:
000000

```

рис. 3 -результат выполнения «хорошего» .EXE модуля

Отличия исходных текстов COM и EXE программ

1. Сколько сегментов должна содержать COM-программа?

- COM-программа должна содержать один сегмент.

2. EXE-программа?

- EXE-программа может содержать разное кол-во сегментов в зависимости от используемой модели памяти (пр: для small может быть только один сегмент кода и один сегмент данных, для large — несколько сегментов кода и несколько сегментов данных).

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

- ORG 100h — устанавливает относительный адрес для начала выполнения программы (т.е резервирует 100h байт от начального адреса для PSP).

- ASSUME — устанавливает соответствие сегментных регистров сегментам (все они будут указывать на один сегмент, потому что он единственный).

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

- в COM-программе нельзя использовать команды, использующие адрес сегмента, т.к этот адрес становится известным только после загрузки сегмента в память.

2. Файл загрузочного модуля .COM, «плохого» .EXE и «хорошего» в шестнадцатеричном виде представлены на рис. 4, 5, 6, 7.

0000:0000	E9 AA 00 50	43 0D 0A 24	50 43 2F 58	54 0D 0A 24	éª.PC..\$PC/XT..\$
0000:0010	41 54 0D 0A	24 50 53 32	20 6D 6F 64	65 6C 20 33	AT..\$PS2 model 3
0000:0020	30 0D 0A 24	50 53 32 20	6D 6F 64 65	6C 20 38 30	0..\$PS2 model 80
0000:0030	0D 0A 24 50	43 6A 72 0D	0A 24 50 43	20 43 6F 6E	..\$PCjr..\$PC Con
0000:0040	76 65 72 74	69 62 6C 65	0D 0A 24 4F	74 68 65 72	vertible..\$0ther
0000:0050	20 74 79 70	65 3A 0D 0A	24 20 20 20	20 20 20 20	type:..\$
0000:0060	0D 0A 24 56	65 72 73 69	6F 6E 3A 20	0D 0A 24 20	..\$Version: ..\$
0000:0070	24 2E 24 56	65 72 73 69	6F 6E 20 3C	32 2E 30 0D	\$. \$Version <2.0.
0000:0080	0A 24 4F 45	4D 20 6E 75	6D 62 65 72	3A 0D 0A 24	.\$OEM number:..\$
0000:0090	55 73 65 72	20 73 65 72	69 61 6C 20	6E 75 6D 62	User serial numb
0000:00A0	65 72 3A 0A	20 20 20 20	20 20 0D 0A	24 B8 00 F0	er:..\$. .đ
0000:00B0	8E C0 26 A0	FE FF 3C FF	74 20 3C FE	74 22 3C FB	.À& py<ýt <pt"<û
0000:00C0	74 1E 3C FC	74 20 3C FA	74 22 3C F8	74 24 3C FD	t.<üt <út"<øt\$<y
0000:00D0	74 26 3C F9	74 28 3C F9	75 2A BA 03	01 EB 3D 90	t&<üt (<uu*º..ë=.
0000:00E0	BA 08 01 EB	37 90 BA 10	01 EB 31 90	BA 15 01 EB	º..ë7.º..ë1.º..ë
0000:00F0	2B 90 BA 24	01 EB 25 90	BA 33 01 EB	1F 90 BA 3A	+..º\$.ë%.º3.ë..º:
0000:0100	01 EB 19 90	BA 4B 01 B4	09 CD 21 E8	BA 00 E8 A3	.ë..ºK.´.Í!èº.èf
0000:0110	00 8A C4 E8	9E 00 E8 92	00 EB 08 90	B4 09 CD 21	..Äè..è..ë..´.Í!
0000:0120	EB 01 90 B4	30 CD 21 51	53 50 3C 00	74 1D BA 63	ë..´0Í!QSP<.t.ºc
0000:0130	01 50 B4 09	CD 21 58 BE	6F 01 E8 B4	00 83 C6 01	.P´.Í!X¾o.è´.Æ.
0000:0140	BA 6F 01 B4	09 CD 21 58	EB 09 90 BA	73 01 B4 09	ºo.´.Í!Xë..ºs.´.
0000:0150	CD 21 58 BA	71 01 50 B4	09 CD 21 58	BE 59 01 8A	Í!Xºq.P´.Í!X¾Y..
0000:0160	C4 E8 8D 00	83 C6 01 BA	59 01 B4 09	CD 21 BA 82	Äè...Æ.ºY.´.Í!º.
0000:0170	01 50 B4 09	CD 21 58 BE	59 01 8A C7	E8 72 00 83	.P´.Í!X¾Y...Çèr..
0000:0180	C6 01 BA 59	01 B4 09 CD	21 BF 90 01	83 C7 19 8B	Æ.ºY.´.Í!¿...Ç..
0000:0190	C1 E8 45 00	8A C3 E8 2F	00 83 EF 02	89 05 BA 90	ÄèE..Äè/..i...º.
0000:01A0	01 B4 09 CD	21 32 C0 B4	4C CD 21 50	BA 59 01 B4	..´.Í!2À´LÍ!PºY.´
0000:01B0	09 CD 21 58	50 8B D0 B4	02 CD 21 58	C3 24 0F 3C	.Í!XP.Đ´.Í!XÅ\$.<
0000:01C0	09 76 02 04	07 04 30 C3	51 8A E0 E8	EF FF 86 C4	.v....0ÃQ.àèiÿ.Ã
0000:01D0	B1 04 D2 E8	E8 E6 FF 59	C3 53 8A FC	E8 E9 FF 88	±.0èèæÿYÅS.üèéÿ.
0000:01E0	25 4F 88 05	4F 8A C7 E8	DE FF 88 25	4F 88 05 5B	%0..0.Çèÿÿ.%0..[
0000:01F0	C3 51 52 32	E4 33 D2 B9	0A 00 F7 F1	80 CA 30 88	ÅQR2ä30¹..÷ñ.Ê0.
0000:0200	14 4E 33 D2	3D 0A 00 73	F1 3C 00 74	04 0C 30 88	.N30=..sñ<.t..0
0000:0210	04 5A 59 C3				.ZYÃ

рис. 4 - Файл загрузочного модуля .COM в 16-м виде

0000:0000	4D 5A 14 01	03 00 00 00	20 00 00 00	FF FF 00 00	MZ.....ÿÿ..
0000:0010	00 00 1B 52	00 01 00 00	1E 00 00 00	01 00 00 00	...R.....
0000:0020	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0030	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0040	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0050	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0060	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0070	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0080	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0090	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00A0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00B0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00C0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00D0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00E0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00F0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0100	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0110	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0120	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0130	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0140	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0150	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0160	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0170	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0180	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0190	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01A0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01B0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01C0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01D0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01E0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01F0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0200	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0210	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0220	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0230	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0240	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0250	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0260	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0270	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0280	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0290	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02A0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02B0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02C0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02D0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02E0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02F0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0300	E9 AA 00 50	43 0D 0A 24	50 43 2F 58	54 0D 0A 24	éa.PC..\$PC/XT..\$
0000:0310	41 54 0D 0A	24 50 53 32	20 6D 6F 64	65 6C 20 33	AT..\$PS2 model 3
0000:0320	30 0D 0A 24	50 53 32 20	6D 6F 64 65	6C 20 38 30	0..\$PS2 model 80
0000:0330	0D 0A 24 50	43 6A 72 0D	0A 24 50 43	20 43 6F 6E	..\$PCjr..\$PC Con
0000:0340	76 65 72 74	69 62 6C 65	0D 0A 24 4F	74 68 65 72	vertible..\$0ther
0000:0350	20 74 79 70	65 3A 0D 0A	24 20 20 20	20 20 20 20	type:..\$
0000:0360	0D 0A 24 56	65 72 73 69	6F 6E 3A 20	0D 0A 24 20	..\$Version: ..\$
0000:0370	24 2E 24 56	65 72 73 69	6F 6E 20 3C	32 2E 30 0D	\$. \$Version <2.0.
0000:0380	0A 24 4F 45	4D 20 6E 75	6D 62 65 72	3A 0D 0A 24	..\$OEM number:..\$
0000:0390	55 73 65 72	20 73 65 72	69 61 6C 20	6E 75 6D 62	User serial numb
0000:03A0	65 72 3A 0A	20 20 20 20	20 20 0D 0A	24 B8 00 F0	er:..\$. ð
0000:03B0	8E C0 26 A0	FE FF 3C FF	74 20 3C FE	74 22 3C FB	.Å& pÿ<ÿt <pt"<ù
0000:03C0	74 1E 3C FC	74 20 3C FA	74 22 3C F8	74 24 3C FD	t.<ùt <ùt"<øt\$<ÿ
0000:03D0	74 26 3C F9	74 28 3C F9	75 2A BA 03	01 EB 3D 90	t&<ùt (<ùu*ø..è=.
0000:03E0	BA 08 01 EB	37 90 BA 10	01 EB 31 90	BA 15 01 EB	ø..è7.ø..è1.ø..è
0000:03F0	2B 90 BA 24	01 EB 25 90	BA 33 01 EB	1F 90 BA 3A	+..ø\$.è%.ø3.è..ø:
0000:0400	01 EB 19 90	BA 4B 01 B4	09 CD 21 E8	BA 00 E8 A3	..è..øK..Í!èø.èf
0000:0410	00 8A C4 E8	9E 00 E8 92	00 EB 08 90	B4 09 CD 21	..Àè..è..è..Í!
0000:0420	EB 01 90 B4	30 CD 21 51	53 50 3C 00	74 1D BA 63	è..`ØÍ!QSP<.t.øc
0000:0430	01 50 B4 09	CD 21 58 BE	6F 01 E8 B4	00 83 C6 01	.P'.Í!X&ø.è'..Æ.
0000:0440	BA 6F 01 B4	09 CD 21 58	EB 09 90 BA	73 01 B4 09	øø..Í!Xè..ø\$.Í!
0000:0450	CD 21 58 BA	71 01 50 B4	09 CD 21 58	BE 59 01 8A	Í!Xøq.P'.Í!X&Y..
0000:0460	C4 E8 8D 00	83 C6 01 BA	59 01 B4 09	CD 21 BA 82	Àè..Æ.øY..Í!ø.
0000:0470	01 50 B4 09	CD 21 58 BE	59 01 8A C7	E8 72 00 83	.P'.Í!X&Y..Çèr..
0000:0480	C6 01 BA 59	01 B4 09 CD	21 BF 90 01	83 C7 19 8B	Æ.øY..Í!è...Ç..
0000:0490	C1 E8 45 00	8A C3 E8 2F	00 83 EF 02	89 05 BA 90	ÀèE..Àè/..Í...ø.
0000:04A0	01 B4 09 CD	21 32 C0 B4	4C CD 21 50	BA 59 01 B4	..Í!2A`LÍ!PøY..
0000:04B0	09 CD 21 58	50 88 D0 B4	02 CD 21 58	C3 24 0F 3C	..Í!XP.Ø'.Í!X&\$<
0000:04C0	09 76 02 04	07 04 30 C3	51 8A E0 E8	EF FF 86 C4	.v...øAQ.àèiÿ.Å
0000:04D0	B1 04 D2 E8	E8 E6 FF 59	C3 53 8A FC	E8 E9 FF 88	±.ØèæÿY&S.üèéÿ.
0000:04E0	25 4F 88 05	4F 8A C7 E8	DE FF 88 25	4F 88 05 5B	%0..0.ÇèPÿ.%0..[
0000:04F0	C3 51 52 32	E4 33 D2 B9	0A 00 F7 F1	80 CA 30 88	AQR2&30¹..±ñ.Èø.
0000:0500	14 4E 33 D2	3D 0A 00 73	F1 3C 00 74	04 0C 30 88	.N30=...sñ<.t..0.
0000:0510	04 5A 59 C3				.ZYÅ

рис. 5 - Файл «плохого» .EXE модуля в 16-м виде

0000:0000	4D 5A 1A 00	04 00 01 00	20 00 00 00	FF FF 00 00	MZ..... ħħ..
0000:0010	00 02 D6 6F	69 00 2B 00	1E 00 00 00	01 00 6A 00	..Ńoi.+.....j.
0000:0020	2B 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	+.....
0000:0030	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0040	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0050	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0060	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0070	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0080	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0090	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00A0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00B0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00C0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00D0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00E0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:00F0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0100	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0110	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0120	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0130	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0140	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0150	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0160	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0170	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0180	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0190	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01A0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01B0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01C0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01D0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01E0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:01F0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0200	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0210	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0220	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0230	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0240	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0250	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0260	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0270	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0280	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0290	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02A0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02B0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02C0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02D0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02E0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:02F0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0300	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0310	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00

рис. 6 - Файл «хорошего» .EXE модуля в 16-м виде (начало)

0000:0320	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0330	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0340	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0350	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0360	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0370	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0380	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0390	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:03A0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:03B0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:03C0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:03D0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:03E0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:03F0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
0000:0400	50 43 0D 0A	24 50 43 2F	58 54 0D 0A	24 41 54 0D	PC..\$PC/XT..\$AT.
0000:0410	0A 24 50 53	32 20 6D 6F	64 65 6C 20	33 30 0D 0A	.\$PS2 model 30..
0000:0420	24 50 53 32	20 6D 6F 64	65 6C 20 38	30 0D 0A 24	\$PS2 model 80..\$
0000:0430	50 43 6A 72	0D 0A 24 50	43 20 43 6F	6E 76 65 72	PCjr..\$PC Conver
0000:0440	74 69 62 6C	65 0D 0A 24	4F 74 68 65	72 20 74 79	tible..\$0ther ty
0000:0450	70 65 3A 0D	0A 24 20 20	20 20 20 20	20 0D 0A 24	pe:..\$..\$
0000:0460	56 65 72 73	69 6F 6E 3A	20 0D 0A 24	20 24 2E 24	Version: ..\$ \$.\$
0000:0470	56 65 72 73	69 6F 6E 20	3C 32 2E 30	0D 0A 24 4F	Version <2.0..\$0
0000:0480	45 4D 20 6E	75 6D 62 65	72 3A 0D 0A	24 55 73 65	EM number:..\$Use
0000:0490	72 20 73 65	72 69 61 6C	20 6E 75 6D	62 65 72 3A	r serial number:
0000:04A0	0A 20 20 20	20 20 20 0D	0A 24 00 00	00 00 00 00	..\$.....
0000:04B0	50 BA 56 00	B4 09 CD 21	58 50 8B D0	B4 02 CD 21	P°V.'.Í!XP.Đ'.Í!
0000:04C0	58 C3 24 0F	3C 09 76 02	04 07 04 30	C3 51 8A E0	XÃ\$.<.v....0ÃQ.à
0000:04D0	E8 EF FF 86	C4 B1 04 D2	E8 E8 E6 FF	59 C3 53 8A	èiÿ.Ã±.0èèæÿYÃS.
0000:04E0	FC E8 E9 FF	88 25 4F 88	05 4F 8A C7	E8 DE FF 88	üèéÿ.%0..0.ÇèPÿ.
0000:04F0	25 4F 88 05	5B C3 51 52	32 E4 33 D2	B9 0A 00 F7	%0..[ÃQR2ã30¹..÷
0000:0500	F1 80 CA 30	88 14 4E 33	D2 3D 0A 00	73 F1 3C 00	ñ.Ê0..N30=..sñ<.
0000:0510	74 04 0C 30	88 04 5A 59	C3 B8 20 00	8E D8 B8 00	t..0..ZYÃ..0..
0000:0520	F0 8E C0 26	A0 FE FF 3C	FF 74 20 3C	FE 74 22 3C	ð.Ã& pÿ<ÿt <pt"<
0000:0530	FB 74 1E 3C	FC 74 20 3C	FA 74 22 3C	F8 74 24 3C	ût.<ût <ût"<øt\$<
0000:0540	FD 74 26 3C	F9 74 28 3C	F9 75 2A BA	00 00 EB 3D	ýt&<ût(<ûu*°..ë=
0000:0550	90 BA 05 00	EB 37 90 BA	0D 00 EB 31	90 BA 12 00	..°..ë7.°..ë1.°..
0000:0560	EB 2B 90 BA	21 00 EB 25	90 BA 30 00	EB 1F 90 BA	ë+.°!.ë%.°0.ë...°
0000:0570	37 00 EB 19	90 BA 48 00	B4 09 CD 21	E8 4E FF E8	7.ë...°H.'.Í!èNÿè
0000:0580	37 FF 8A C4	E8 32 FF E8	26 FF EB 08	90 B4 09 CD	7ÿ.Ãè2ÿè&ÿë...'.Í
0000:0590	21 EB 01 90	B4 30 CD 21	50 3C 00 74	1D BA 60 00	!ë...°0Í!P<.t.°..
0000:05A0	50 B4 09 CD	21 58 BE 6C	00 E8 4A FF	83 C6 01 BA	P'.Í!X¼l.èJÿ.Æ.°
0000:05B0	6C 00 B4 09	CD 21 58 EB	09 90 BA 70	00 B4 09 CD	l.'.Í!Xë...°p.'.Í
0000:05C0	21 58 BA 6E	00 50 B4 09	CD 21 58 BE	56 00 8A C4	!X°n.P'.Í!X¼V..Ã
0000:05D0	E8 23 FF 83	C6 01 BA 56	00 B4 09 CD	21 BA 7F 00	è#ÿ.Æ.°V.'.Í!°..
0000:05E0	50 B4 09 CD	21 58 BE 56	00 8A C7 E8	08 FF 83 C6	P'.Í!X¼V..Çè.ÿ.Æ
0000:05F0	01 BA 56 00	B4 09 CD 21	BF 8D 00 83	C7 19 8B C1	..°V.'.Í!¿...Ç..Ã
0000:0600	E8 DB FE 8A	C3 E8 C5 FE	83 EF 02 89	05 BA 8D 00	èÛp.ÃèÃp.ĩ...°..
0000:0610	B4 09 CD 21	32 C0 B4 4C	CD 21		'.Í!2Ã'LI!

рис. 7 - Файл «хорошего» .EXE модуля в 16-м виде (конец)

Отличия форматов файлов COM и EXE модулей

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

- COM-файл не превышает 64Кб и содержит один сегмент. Код располагается с адреса 0h, при этом в первых 100h байт размещается PSP.

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

- «Плохой» EXE файл, так же, как и COM файла, имеет один сегмент. Код располагается с адреса 300h. С адреса 0 располагается информация для загрузчика, которая образует заголовок.

MZ показывает, что это EXE файл и нужно генерировать 16-битный код (а не 32-битный).

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

- В «хорошем» EXE файле есть сегменты данных, кода и стека, в отличии от «плохого», в котором только один сегмент. С нулевого адреса, так же, как и в «плохом» EXE, располагается заголовок, затем таблица настроек.

В «плохом» файле адресация всегда начинается с адреса 300h.

3. Запуск файла .COM в отладчике TD.EXE показан на рис. 8, запуск «хорошего» .EXE на рис. 9.

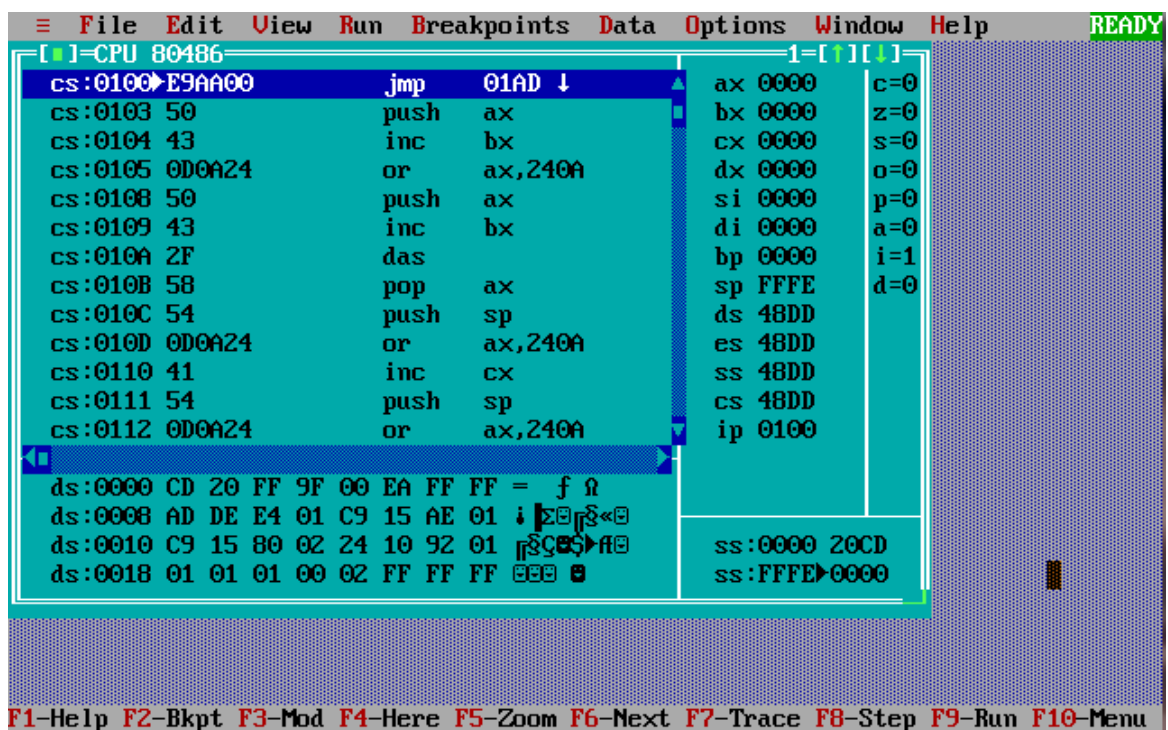


рис. 8 - файл .COM в отладчике

Загрузка COM модуля в основную память

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

- Смещение в сегменте команд равно 100h, сегментные регистры указывают на PSP.

2. Что располагается с адреса 0?

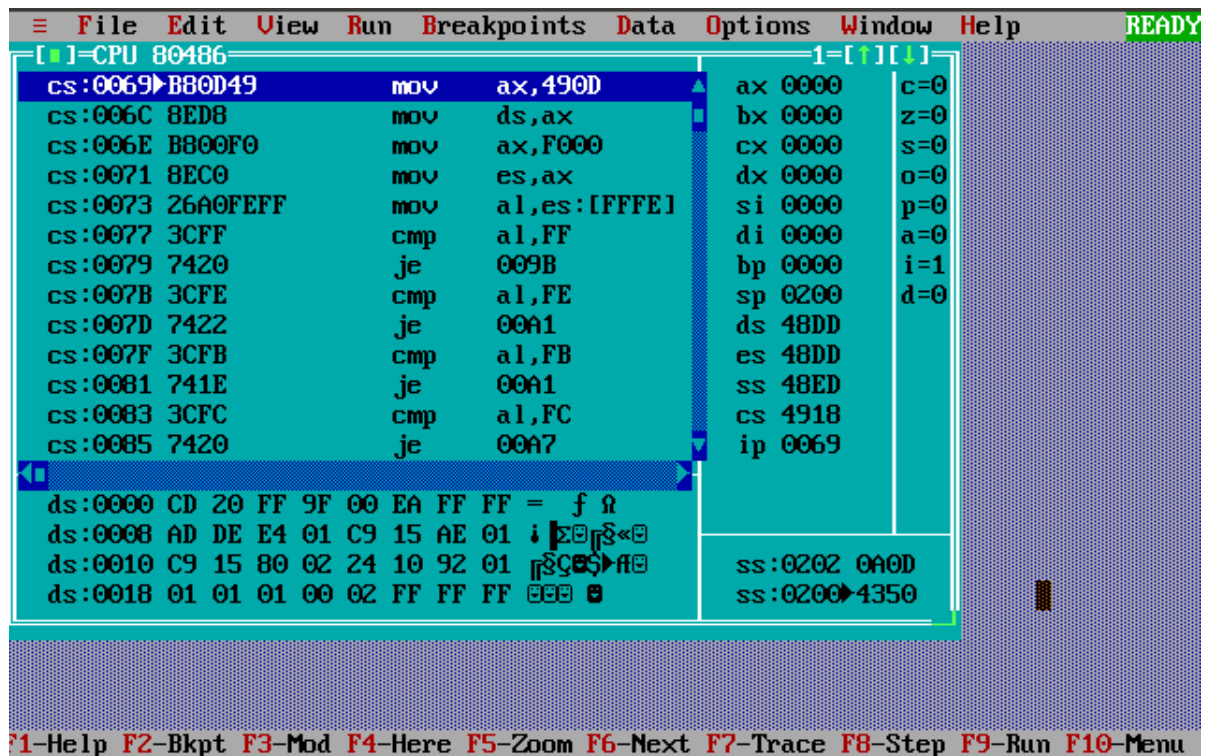
- С адреса 0 до адреса 100h располагается PSP.

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

- Сегментные регистры имеют значения 48DD и указывают на PSP.

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

- Значение регистра SP устанавливается (автоматически) так, чтобы он указывал на последнюю доступную в сегменте ячейку памяти (SP указывает на FFFE). Таким образом программа занимает начало, а стек - конец сегмента.



Загрузка «хорошего» EXE модуля в основную память

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

- Значения DS, ES (48DD) устанавливаются на начало PSP, SS (48ED) – на начало сегмента стека, CS (4918) – на начало сегмента команд.

2. На что указывают регистры DS и ES?

- Регистры DS и ES указывают на начало PSP.

3. Как определяется стек?

- Стек определяется при помощи директивы .STACK или при помощи директивы ASSUME, которая установит сегментный регистр SS на начало сегмента стека.

4. Как определяется точка входа?

- Точка входа определяется при помощи директивы END (за ней идет название функции или метки, с которой нужно начать выполнение программы).

Выводы

В ходе выполнения работы были изучены COM и EXE файлы и их различия. Так же были получены две программы `tyresom.com` и `tyreexe.exe`.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД TYPECOM.ASM

```
TESTPC      SEGMENT
              ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING
              ORG 100H ; резервирование места для PSP
START:      JMP BEGIN

PC db  'PC',0DH,0AH,'$'
XT db  'PC/XT',0DH,0AH,'$'
tAT db  'AT',0DH,0AH,'$'
PS2_30 db  'PS2 model 30',0DH,0AH,'$'
PS2_80 db  'PS2 model 80',0DH,0AH,'$'
PCJR db  'PCjr',0DH,0AH,'$'
PC_CONVERTIBLE db  'PC Convertible',0DH,0AH,'$'
OTHER_TYPE db  'Other type:',0DH,0AH,'$'
END_LINE db  '          ', 0DH,0AH,'$'

VERSION db  'Version: ',0DH,0AH,'$'
VERS db  ' $'
MODIFICATION db  '.$'
VERSION2 db  'Version <2.0',0DH,0AH,'$'
OEM db  'OEM number:',0DH,0AH,'$'
SERIAL_NUMBER db  'User serial number:', 0AH, '          ', 0DH,0AH,'$'

BEGIN:
    mov ax,0F000H
    mov es,ax
    mov al,es:[0FFFEH]

    cmp al, 0FFH
    je itIsPC
    cmp al, 0FEH
    je itIsPC_XT
    cmp al, 0FBH
    je itIsPC_XT
    cmp al, 0FCH
    je itIsAT
    cmp al, 0FAH
    je itIsPS2_30
    cmp al, 0F8H
    je itIsPS2_80
    cmp al, 0FDH
    je itIsPCjr
    cmp al, 0F9H
```



```

        je itIsPCconvertible

        cmp al, 0F9H
        jne itIsOther

;-----
; Для вывода типа
itIsPC:
        mov dx, offset PC
        jmp writeType

itIsPC_XT:
        mov dx, offset XT
        jmp writeType

itIsAT:
        mov dx, offset tAT
        jmp writeType

itIsPS2_30:
        mov dx, offset PS2_30
        jmp writeType

itIsPS2_80:
        mov dx, offset PS2_80
        jmp writeType

itIsPCjr:
        mov dx, offset PCJR
        jmp writeType

itIsPCconvertible:
        mov dx, offset PC_CONVERTIBLE
        jmp writeType

itIsOther:
        mov dx, offset OTHER_TYPE
        mov ah, 09h
        int 21h
        call BYTE_TO_HEX
        call PRINT_NUM
        mov al, ah
        call PRINT_NUM
        call PRINT_END_LINE
        jmp OS_VERSION

writeType:
        mov ah, 09h

```

```

        int 21h
        jmp OS_VERSION
;-----

;-----
; Для вывода версии системы

OS_VERSION:
        mov ah, 30h
        int 21h

printVer:
        push ax
        cmp al, 0
        je ver2

        mov dx, offset VERSION
        push ax
        mov ah, 09h
        int 21h
        pop ax

        mov si, offset VERS
        call BYTE_TO_DEC
        add si, 1
        mov dx, offset VERS
        mov ah, 09h
        int 21h
        pop ax
        jmp numMod

ver2:
        mov dx, offset VERSION2
        mov ah, 09h
        int 21h
        pop ax

numMod:
        mov dx, offset MODIFICATION
        push ax
        mov ah, 09h
        int 21h
        pop ax
        mov si, offset END_LINE
        mov al, ah
        call BYTE_TO_DEC
        add si, 1

```

```

        mov dx, offset END_LINE
        mov ah, 09h
        int 21h

numOEM:
        mov dx, offset OEM
        push ax
        mov ah, 09h
        int 21h
        pop ax
        mov si, offset END_LINE
        mov al, bh
        call BYTE_TO_DEC
        add si, 1
        mov dx, offset END_LINE
        mov ah, 09h
        int 21h

serialNumb:

        mov di, offset SERIAL_NUMBER
        add di, 25
        mov ax, cx
        call WRD_TO_HEX
        mov al, bl
        call BYTE_TO_HEX
        sub di, 2
        mov [di], ax
        mov dx, offset SERIAL_NUMBER
        mov ah, 09h
        int 21h

        xor al, al
        mov ah, 4Ch
        int 21h

;-----

PRINT_END_LINE PROC near
        push ax
        mov dx, offset END_LINE
        mov ah, 09h
        int 21h
        pop ax
PRINT_END_LINE ENDP

```

```

PRINT_NUM PROC near
    ; вывод al
    push ax
    mov dx, ax
    mov ah, 02h
    int 21h
    pop ax
    ret
PRINT_NUM ENDP

TETR_TO_HEX PROC near
    and AL, 0Fh
    cmp AL, 09
    jbe next
    add AL, 07
next:
    add AL, 30h
    ret
TETR_TO_HEX ENDP
;-----
BYTE_TO_HEX PROC near
;байт в AL переводится в два символа шест. числа в AX
    push CX
    mov AH, AL
    call TETR_TO_HEX
    xchg AL, AH
    mov CL, 4
    shr AL, CL
    call TETR_TO_HEX ; в AL старшая цифра
    pop CX ; в AH младшая
    ret
BYTE_TO_HEX ENDP
;-----
WRD_TO_HEX PROC near
;перевод в 16 с/с 16-ти разрядного числа
; в AX - число, DI - адрес последнего символа
    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
; перевод в 10с/с, SI - адрес поля младшей цифры
    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_l
    or AL,30h
    mov [SI],AL
end_l:
    pop DX
    pop CX
    ret
BYTE_TO_DEC ENDP

TESTPC      ENDS
            END START

```


ПРИЛОЖЕНИЕ В

ИСХОДНЫЙ КОД TYPEEXE.ASM

```

AStack    SEGMENT    STACK
                DW 100h DUP(?)

AStack    ENDS


DATA    SEGMENT

        PC db  'PC',0DH,0AH,'$'
        XT db  'PC/XT',0DH,0AH,'$'
        tAT db  'AT',0DH,0AH,'$'
        PS2_30 db  'PS2 model 30',0DH,0AH,'$'
        PS2_80 db  'PS2 model 80',0DH,0AH,'$'
        PCJR db  'PCjr',0DH,0AH,'$'
        PC_CONVERTIBLE db  'PC Convertible',0DH,0AH,'$'
        OTHER_TYPE db  'Other type:',0DH,0AH,'$'
        END_LINE db  '          ',0DH,0AH,'$'


        VERSION db  'Version: ',0DH,0AH,'$'
        VERS db  ' $'
        MODIFICATION db  '.$'
        VERSION2 db  'Version <2.0',0DH,0AH,'$'
        OEM db  'OEM number:',0DH,0AH,'$'
        SERIAL_NUMBER db  'User serial number:', 0AH, '          ',
0DH,0AH,'$'
DATA ENDS


CODE SEGMENT
        ASSUME CS:CODE,DS:DATA,SS:AStack


;-----
PRINT_END_LINE PROC near
        push ax
        mov dx, offset END_LINE
        mov ah, 09h
        int 21h
        pop ax
PRINT_END_LINE ENDP


PRINT_NUM PROC near
        ; вывод al
        push ax
        mov dx, ax
        mov ah, 02h
        int 21h

```

```

        pop ax
        ret
PRINT_NUM ENDP
;-----
TETR_TO_HEX PROC near
    and AL,0Fh
    cmp AL,09
    jbe next
    add AL,07
next:
    add AL,30h
    ret
TETR_TO_HEX ENDP
;-----
BYTE_TO_HEX PROC near
;байт в AL переводится в два символа шест. числа в AX
    push CX
    mov AH,AL
    call TETR_TO_HEX
    xchg AL,AH
    mov CL,4
    shr AL,CL
    call TETR_TO_HEX ;в AL старшая цифра
    pop CX ;в AH младшая
    ret
BYTE_TO_HEX ENDP
;-----
WRD_TO_HEX PROC near
;перевод в 16 с/с 16-ти разрядного числа
; в AX - число, DI - адрес последнего символа
    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
; перевод в 10с/с, SI - адрес поля младшей цифры

```

```

    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_l
    or AL,30h
    mov [SI],AL
end_l:
    pop DX
    pop CX
    ret
BYTE_TO_DEC ENDP
;-----

```

```

Main PROC FAR
    mov ax, DATA
    mov ds, ax

    mov ax,0F000H
    mov es,ax
    mov al,es:[0FFFEH]

    cmp al, 0FFH
    je itIsPC
    cmp al, 0FEH
    je itIsPC_XT
    cmp al, 0FBH
    je itIsPC_XT
    cmp al, 0FCH
    je itIsAT
    cmp al, 0FAH
    je itIsPS2_30
    cmp al, 0F8H
    je itIsPS2_80
    cmp al, 0FDH
    je itIsPCjr
    cmp al, 0F9H

```

```

        je itIsPCconvertible

        cmp al, 0F9H
        jne itIsOther

;-----
; Для вывода типа
itIsPC:
        mov dx, offset PC
        jmp writeType

itIsPC_XT:
        mov dx, offset XT
        jmp writeType

itIsAT:
        mov dx, offset tAT
        jmp writeType

itIsPS2_30:
        mov dx, offset PS2_30
        jmp writeType

itIsPS2_80:
        mov dx, offset PS2_80
        jmp writeType

itIsPCjr:
        mov dx, offset PCJR
        jmp writeType

itIsPCconvertible:
        mov dx, offset PC_CONVERTIBLE
        jmp writeType

itIsOther:
        mov dx, offset OTHER_TYPE
        mov ah, 09h
        int 21h
        call BYTE_TO_HEX
        call PRINT_NUM
        mov al, ah
        call PRINT_NUM
        call PRINT_END_LINE
        jmp OS_VERSION

writeType:
        mov ah, 09h

```

```

        int 21h
        jmp OS_VERSION
;-----

;-----
; Для вывода версии системы

OS_VERSION:
        mov ah, 30h
        int 21h

printVer:
        push ax
        cmp al, 0
        je ver2

        mov dx, offset VERSION
        push ax
        mov ah, 09h
        int 21h
        pop ax

        mov si, offset VERS
        call BYTE_TO_DEC
        add si, 1
        mov dx, offset VERS
        mov ah, 09h
        int 21h
        pop ax
        jmp numMod

ver2:
        mov dx, offset VERSION2
        mov ah, 09h
        int 21h
        pop ax

numMod:
        mov dx, offset MODIFICATION
        push ax
        mov ah, 09h
        int 21h
        pop ax
        mov si, offset END_LINE
        mov al, ah
        call BYTE_TO_DEC
        add si, 1

```



```

        mov dx, offset END_LINE
        mov ah, 09h
        int 21h

numOEM:
        mov dx, offset OEM
        push ax
        mov ah, 09h
        int 21h
        pop ax
        mov si, offset END_LINE
        mov al, bh
        call BYTE_TO_DEC
        add si, 1
        mov dx, offset END_LINE
        mov ah, 09h
        int 21h

serialNumb:

        mov di, offset SERIAL_NUMBER
        add di, 25
        mov ax, cx
        call WRD_TO_HEX
        mov al, bl
        call BYTE_TO_HEX
        sub di, 2
        mov [di], ax
        mov dx, offset SERIAL_NUMBER
        mov ah, 09h
        int 21h

        xor al, al
        mov ah, 4Ch
        int 21h

Main ENDP
CODE ENDS
        END Main

```