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

Студентка гр. 0381 _____ Короткина Е.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 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 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 взят предоставленный шаблон, содержащий процедуры: TETR_TO_HEX, BYTE_TO_HEX, WRD_TO_HEX, BYTE_TO_DEX.

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.

Для вывода сообщений написана процедура WRITEMESSAGE.

В файле lab1com.asm написан код исходного .COM модуля. 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 TYPEDETECTION, определяющая тип PC. В этой процедуре в AL сохраняется значение байта, в котором записан код системы, а затем проводит сравнение полученного значения с кодами из табл. 1. При обнаружении совпадения происходит переход к метке, в которой в DX заносится смещение соответствующего сообщения, после чего вызывается процедура WRITESTRING для печати сообщения.

Написана процедура VERSIONDETECTION, определяющая версию системы, серийный номер OEM и номер пользователя. Функцией 30h прерывания 21h получаются необходимые данные.

Командами MASM lab1com.asm получен объектный файл lab1com.obj, из которого затем командой LINK lab1com.obj собирается «плохой» .EXE-модуль. При попытке запуска lab1com.exe программа выводит следующее сообщение:

Рисунок 1 - вывод модуля lab1com.exe

Командой EXE2BIN lab1com.exe lab1com.com получен .COM-модуль. При попытке его запуска выводятся корректные сообщения:

Рисунок 2 - вывод модуля lab1com.com

В файле lab1exe.asm написан код «хорошего» .EXE модуля. Для этого был скопирован код из файла lab1com.asm, после чего в него был внесен ряд изменений: добавлены определения сегмента стека и данных, строки сообщений вынесены в сегмент данных; код, из которого вызывались процедуры TYPEDETECTION и VERSIONDETECTION вынесен в добавленную дальнюю процедуру MAIN, в ней также присутствует загрузка адреса сегмента данных.

После сборки и запуска lab1exe.exe выводятся корректные сообщения:

Рисунок 3 - вывод модуля lab1exe.exe

Для ответов на контрольные вопросы раздела **«Отличия исходных текстов COM и EXE программ»** было проведено сравнение исходных текстов .COM и .EXE модулей.

1. Сколько сегментов должна содержать COM-программа?

COM-программа должна содержать один сегмент - сегмент кода, в нем же определяются все данные. Стек для COM-программ генерируется автоматически.

2. Сколько сегментов должна содержать EXE-программа?

EXE-программа должна содержать не менее одного сегмента - сегмент кода. Также она может содержать сегменты данных, сегмент стека. Если сегмент стека не был задан, то будет использован стек DOS. Данные обязательно должны быть вынесены в отдельный сегмент.

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

В тексте COM-программы обязательно должны быть директива ASSUME, которая расставляет соответствия между сегментными регистрами и единственным сегментом; директива ORG 100h - смещение кода от начала PSP (без этой директивы модуль можно собрать и запустить, но вывод будет некорректным).

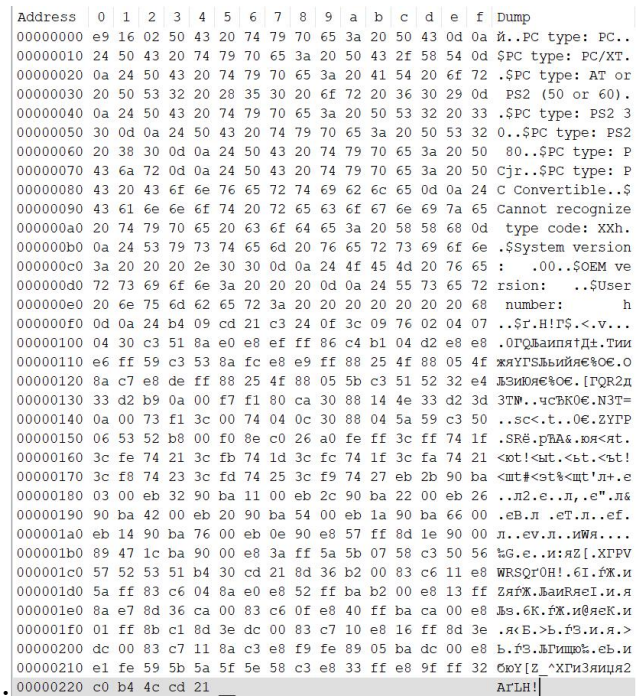
4. Все ли форматы команд можно использовать в COM-программе?

Нет - команды, операндами которых являются сегменты, быть выполнены не могут, т.к. в COM-модулях отсутствует заголовок, в котором содержится таблица настройки - таблица, по которой осуществляется поиск абсолютных адресов сегментов.

Для ответа на вопросы раздела **«Отличия форматов файлов COM и EXE модулей»** файлы lab1com.com, lab1com.exe и lab1exe.exe были открыты в редакторе в шестнадцатеричном виде.

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

В файле COM имеется только один сегмент - сегмент кода, в котором располагается код и данные. Код располагается с адреса 0 (в этой работе это команда jmp BEGIN, после неё размещены данные-строки, затем - снова код). При запуске модуля ко всем адресам будет добавлено смещение 100h, поскольку в COM модулях используется директива ORG 100h для выделения 256 байт под PSP.

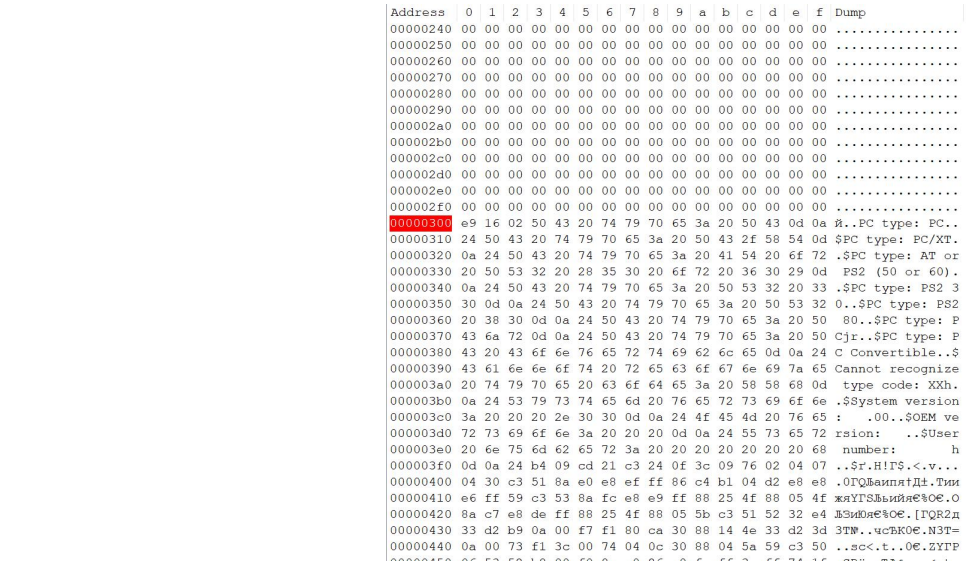


```
Address  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f Dump
00000000 e9 16 02 50 43 20 74 79 70 65 3a 20 50 43 0d 0a й..PC type: PC..
00000010 24 50 43 20 74 79 70 65 3a 20 50 43 2f 58 54 0d $PC type: PC/XT.
00000020 0a 24 50 43 20 74 79 70 65 3a 20 41 54 20 6f 72 .$PC type: AT or
00000030 20 50 53 32 20 28 35 30 20 6f 72 20 36 30 29 0d  PS2 (50 or 60).
00000040 0a 24 50 43 20 74 79 70 65 3a 20 50 53 32 20 33 .$PC type: PS2 3
00000050 30 0d 0a 24 50 43 20 74 79 70 65 3a 20 50 53 32 0..$PC type: PS2
00000060 20 38 30 0d 0a 24 50 43 20 74 79 70 65 3a 20 50  80..$PC type: P
00000070 43 6a 72 0d 0a 24 50 43 20 74 79 70 65 3a 20 50 Cjr..$PC type: P
00000080 43 20 43 6f 6e 76 65 72 74 69 62 6c 65 0d 0a 24 C Convertible..$
00000090 43 61 6e 6e 6f 74 20 72 65 63 6f 67 6e 69 7a 65 Cannot recognize
000000a0 20 74 79 70 65 20 63 6f 64 65 3a 20 58 58 68 0d  type code: XXh.
000000b0 0a 24 53 79 73 74 65 6d 20 76 65 72 73 69 6f 6e .$System version
000000c0 3a 20 20 20 2e 30 30 0d 0a 24 4f 45 4d 20 76 65 :   .00..$OEM ve
000000d0 72 73 69 6f 6e 3a 20 20 20 0d 0a 24 55 73 65 72 rsion:   ..$User
000000e0 20 6e 75 6d 62 65 72 3a 20 20 20 20 20 20 20 68  number:       h
000000f0 0d 0a 24 b4 09 cd 21 c3 24 0f 3c 09 76 02 04 07 ..$r.H!Г$.<.v...
00000100 04 30 c3 51 8a e0 e8 ef ff 86 c4 b1 04 d2 e8 e8 .0ГQЉаипя†Д±.Тии
00000110 e6 ff 59 c3 53 8a fc e8 e9 ff 88 25 4f 88 05 4f жяYГSЉьийяє%Оє.O
00000120 8a c7 e8 de ff 88 25 4f 88 05 5b c3 51 52 32 e4 ЉЗиЮяє%Оє.[ГQR2д
00000130 33 d2 b9 0a 00 f7 f1 80 ca 30 88 14 4e 33 d2 3d 3TN..чсЂK0€.N3T=
00000140 0a 00 73 f1 3c 00 74 04 0c 30 88 04 5a 59 c3 50 ..sc<.t..0€.ZYГP
00000150 06 53 52 b8 00 f0 8e c0 26 a0 fe ff 3c ff 74 1f .SRё.рЋА&.юя<яt.
00000160 3c fe 74 21 3c fb 74 1d 3c fc 74 1f 3c fa 74 21 <юt!<ыt.<ьt.<ъt!
00000170 3c f8 74 23 3c fd 74 25 3c f9 74 27 eb 2b 90 ba <шt#<эt%<щt'л+.e
00000180 03 00 eb 32 90 ba 11 00 eb 2c 90 ba 22 00 eb 26 ..л2.е..л,.е".л&
00000190 90 ba 42 00 eb 20 90 ba 54 00 eb 1a 90 ba 66 00 .eB.л .eT.л..ef.
000001a0 eb 14 90 ba 76 00 eb 0e 90 e8 57 ff 8d 1e 90 00 л..ev.л..иWя....
000001b0 89 47 1c ba 90 00 e8 3a ff 5a 5b 07 58 c3 50 56 %G.е..и:яZ[.XГPV
000001c0 57 52 53 51 b4 30 cd 21 8d 36 b2 00 83 c6 11 e8 WRSQг0H!.6I.ѓЖ.и
000001d0 5a ff 83 c6 04 8a e0 e8 52 ff ba b2 00 e8 13 ff ZяѓЖ.ЉаиRяеI.и.я
000001e0 8a e7 8d 36 ca 00 83 c6 0f e8 40 ff ba ca 00 e8 Љз.6К.ѓЖ.и@яеК.и
000001f0 01 ff 8b c1 8d 3e dc 00 83 c7 10 e8 16 ff 8d 3e .я‹Б.>Ь.ѓЗ.и.я.>
00000200 dc 00 83 c7 11 8a c3 e8 f9 fe 89 05 ba dc 00 e8 Ь.ѓЗ.ЉГишщ%.еЬ.и
00000210 e1 fe 59 5b 5a 5f 5e 58 c3 e8 33 ff e8 9f ff 32 бюY[Z_^XГиЗяиџя2
00000220 c0 b4 4c cd 21                                  АґLH!
```

Рисунок 4 - содержимое lab1com.com

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

«Плохой» EXE имеет следующую структуру: сначала идет заголовок с технической информацией, затем - единственный сегмент, в котором расположены и код, и данные. С адреса 0 располагается заголовок, в котором содержатся сведения о размере модуля, адресе стека, относительных смещениях и др., сам код начинается с адреса 300h.

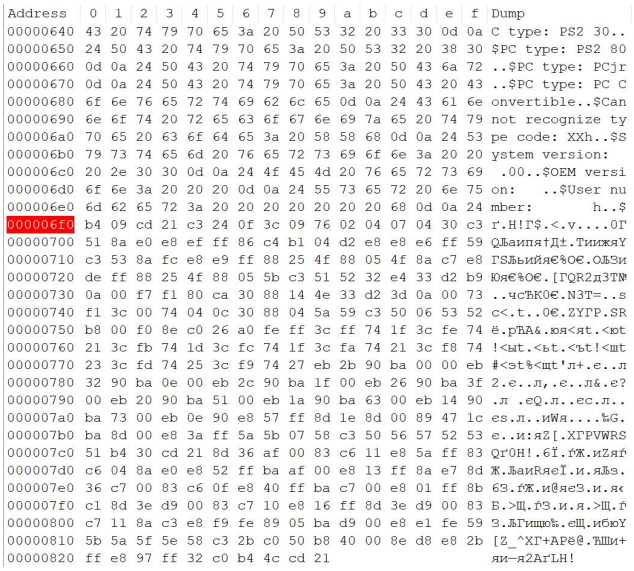


```
Address  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f Dump
00000240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000260 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000270 00 00 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 16 02 50 43 20 74 79 70 65 3a 20 50 43 0d 0a й..PC type: PC..
00000310 24 50 43 20 74 79 70 65 3a 20 50 43 2f 58 54 0d $PC type: PC/XT.
00000320 0a 24 50 43 20 74 79 70 65 3a 20 41 54 20 6f 72 .$PC type: AT or
00000330 20 50 53 32 20 28 35 30 20 6f 72 20 36 30 29 0d  PS2 (50 or 60).
00000340 0a 24 50 43 20 74 79 70 65 3a 20 50 53 32 20 33 .$PC type: PS2 3
00000350 30 0d 0a 24 50 43 20 74 79 70 65 3a 20 50 53 32 0..$PC type: PS2
00000360 20 38 30 0d 0a 24 50 43 20 74 79 70 65 3a 20 50  80..$PC type: P
00000370 43 6a 72 0d 0a 24 50 43 20 74 79 70 65 3a 20 50 Cjr..$PC type: P
00000380 43 20 43 6f 6e 76 65 72 74 69 62 6c 65 0d 0a 24 C Convertible..$
00000390 43 61 6e 6e 6f 74 20 72 65 63 6f 67 6e 69 7a 65 Cannot recognize
000003a0 20 74 79 70 65 20 63 6f 64 65 3a 20 58 58 68 0d  type code: XXh.
000003b0 0a 24 53 79 73 74 65 6d 20 76 65 72 73 69 6f 6e .$System version
000003c0 3a 20 20 20 2e 30 30 0d 0a 24 4f 45 4d 20 76 65 :   .00..$OEM ve
000003d0 72 73 69 6f 6e 3a 20 20 20 0d 0a 24 55 73 65 72 rsion:   ..$User
000003e0 20 6e 75 6d 62 65 72 3a 20 20 20 20 20 20 20 68  number:       h
000003f0 0d 0a 24 b4 09 cd 21 c3 24 0f 3c 09 76 02 04 07 ..$r.H!Г$.<.v...
00000400 04 30 c3 51 8a e0 e8 ef ff 86 c4 b1 04 d2 e8 e8 .0ГQЉаипя†Д±.Тии
00000410 e6 ff 59 c3 53 8a fc e8 e9 ff 88 25 4f 88 05 4f жяYГSЉьийяє%Оє.O
00000420 8a c7 e8 de ff 88 25 4f 88 05 5b c3 51 52 32 e4 ЉЗиЮяє%Оє.[ГQR2д
00000430 33 d2 b9 0a 00 f7 f1 80 ca 30 88 14 4e 33 d2 3d 3TN..чсЂK0€.N3T=
00000440 0a 00 73 f1 3c 00 74 04 0c 30 88 04 5a 59 c3 50 ..sc<.t..0€.ZYГP
```

Рисунок 5 - фрагмент содержимого файла lab1com.exe. Выделен адрес начала кода.

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

«Хороший» EXE имеет следующую структуру: сначала идет заголовок с технической информацией, затем - сегмент стека (адреса 200h-600h, т.к. на стек было выделено 512 слов по 2 байта), сегмент данных (адреса 600h-6F0h) и сегмент кода (с адреса 6F0h и до конца файла). От «плохого» EXE данный модуль отличает деление на сегменты - код отдельно, данные отдельно.



```
Address  0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f  Dump
00000640 43 20 74 79 70 65 3a 20 50 53 32 20 33 30 0d 0a C type: PS2 30..
00000650 24 50 43 20 74 79 70 65 3a 20 50 53 32 20 38 30 $PC type: PS2 80
00000660 0d 0a 24 50 43 20 74 79 70 65 3a 20 50 43 6a 72 ..$PC type: PCjr
00000670 0d 0a 24 50 43 20 74 79 70 65 3a 20 50 43 20 43 ..$PC type: PC C
00000680 6f 6e 76 65 72 74 69 62 6c 65 0d 0a 24 43 61 6e onvertible..$Can
00000690 6e 6f 74 20 72 65 63 6f 67 6e 69 7a 65 20 74 79 not recognize ty
000006a0 70 65 20 63 6f 64 65 3a 20 58 58 68 0d 0a 24 53 pe code: XXh..$S
000006b0 79 73 74 65 6d 20 76 65 72 73 69 6f 6e 3a 20 20 ystem version:
000006c0 20 2e 30 30 0d 0a 24 4f 45 4d 20 76 65 72 73 69  .00..$OEM versi
000006d0 6f 6e 3a 20 20 20 0d 0a 24 55 73 65 72 20 6e 75 on:   ..$User nu
000006e0 6d 62 65 72 3a 20 20 20 20 20 20 20 68 0d 0a 24 mber:       h..$
000006f0 b4 09 cd 21 c3 24 0f 3c 09 76 02 04 07 04 30 c3 r.H!Г$.<.v....0Г
00000700 51 8a e0 e8 ef ff 86 c4 b1 04 d2 e8 e8 e6 ff 59 QЉаипя†Д±.ТиижяY
00000710 c3 53 8a fc e8 e9 ff 88 25 4f 88 05 4f 8a c7 e8 ГSЊьийяЄ%ОЄ.ОЉЗи
00000720 de ff 88 25 4f 88 05 5b c3 51 52 32 e4 33 d2 b9 Юяє%ОЄ.[ГQR2д3Т№
00000730 0a 00 f7 f1 80 ca 30 88 14 4e 33 d2 3d 0a 00 73 ..чсЂК0Є.N3Т=..s
00000740 f1 3c 00 74 04 0c 30 88 04 5a 59 c3 50 06 53 52 c<.t..0Є.ZYГP.SR
00000750 b8 00 f0 8e c0 26 a0 fe ff 3c ff 74 1f 3c fe 74 ё.рЋА&.юя<яt.<юt
00000760 21 3c fb 74 1d 3c fc 74 1f 3c fa 74 21 3c f8 74 !<ыt.<ьt.<ъt!<шt
00000770 23 3c fd 74 25 3c f9 74 27 eb 2b 90 ba 00 00 eb #<эt%<щt'л+.е..л
00000780 32 90 ba 0e 00 eb 2c 90 ba 1f 00 eb 26 90 ba 3f 2.е..л,.е..л&.е?
00000790 00 eb 20 90 ba 51 00 eb 1a 90 ba 63 00 eb 14 90 .л .еQ.л..ес.л..
000007a0 ba 73 00 eb 0e 90 e8 57 ff 8d 1e 8d 00 89 47 1c es.л..иWя....%G.
000007b0 ba 8d 00 e8 3a ff 5a 5b 07 58 c3 50 56 57 52 53 е..и:яZ[.XГPVWRS
000007c0 51 b4 30 cd 21 8d 36 af 00 83 c6 11 e8 5a ff 83 Qґ0Н!.6Ї.ѓЖ.иZяѓ
000007d0 c6 04 8a e0 e8 52 ff ba af 00 e8 13 ff 8a e7 8d Ж.ЊаиRяєЇ.и.яЊз.
000007e0 36 c7 00 83 c6 0f e8 40 ff ba c7 00 e8 01 ff 8b 6З.ѓЖ.и@яєЗ.и.я‹
000007f0 c1 8d 3e d9 00 83 c7 10 e8 16 ff 8d 3e d9 00 83 Б.>Щ.ѓЗ.и.я.>Щ.ѓ
00000800 c7 11 8a c3 e8 f9 fe 89 05 ba d9 00 e8 e1 fe 59 З.ЊГищю‰.єЩ.иЮюY
00000810 5b 5a 5f 5e 58 c3 2b c0 50 b8 40 00 8e d8 e8 2b [Z_^XГ+АР@.ЋШи+
00000820 ff e8 97 ff 32 c0 b4 4c cd 21                   яи—я2АґLН!
```

Рисунок 6 - фрагмент содержимого файла lab1exe.exe. Выделен адрес начала кода и сегмента кода, выше - сегмент данных.

Для ответа на вопросы раздела **«Загрузка COM модуля в основную память»** был открыт отладчик TD.EXE и загружен файл lab1com.com.

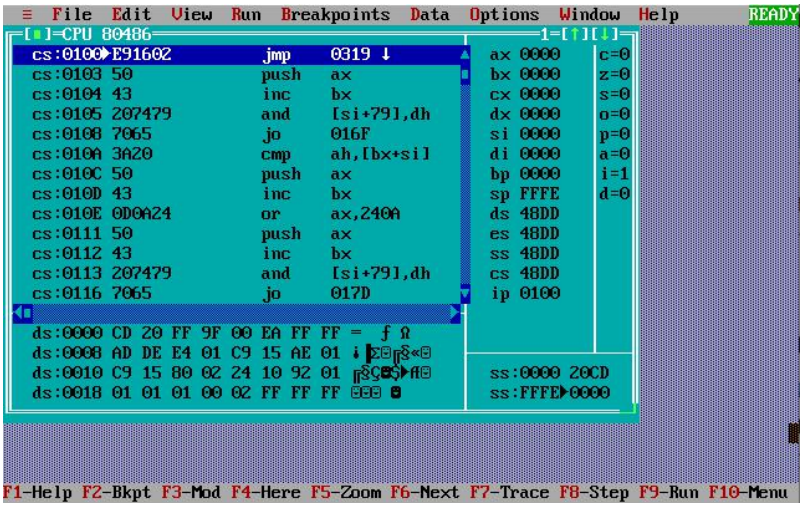


Рисунок 7 - отладчик TD.EXE с открытым COM-файлом.

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

Определяется сегментный адрес свободного участка памяти для загрузки программы, 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. Выполняется чтение программы с ее записью по адресу PSP:0100h. Код располагается с адреса 0100h.

2. Что располагается с адреса 0?

С адреса 0 располагается PSP - префикс программного сегмента.

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.

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

Стек генерируется автоматически и располагается в сегменте кода, в него автоматически записывается значение 0000. Стек имеет адреса от FFFE до 0000 и растет в меньшую сторону, т.е. если стек будет слишком заполнен, он может «затереть» код.

Для ответа на вопросы раздела **«Загрузка «хорошего» EXE модуля в основную память»** был открыт отладчик TD.EXE и загружен «хороший» EXE-модуль lab1exe.exe.

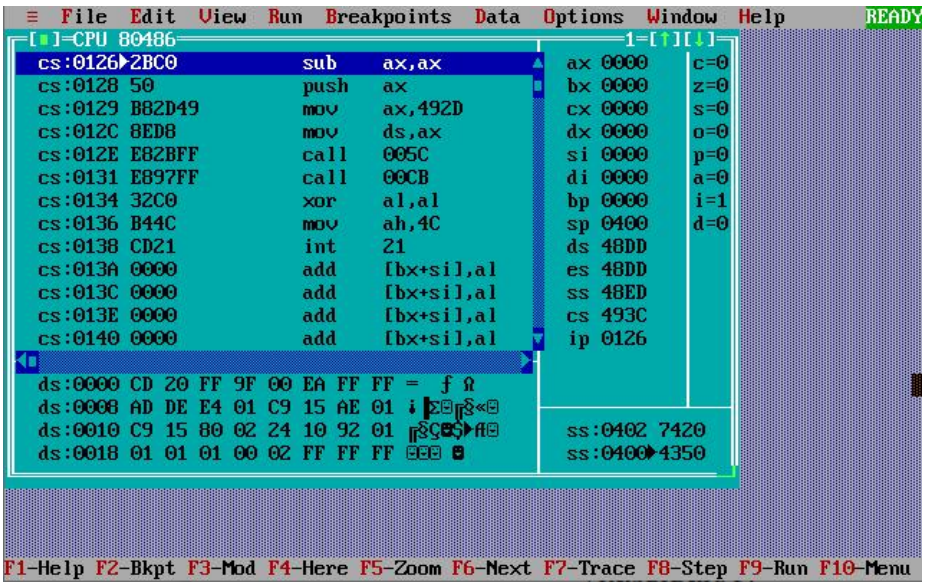


Рисунок 8 - отладчик TD.EXE с открытым EXE-файлом.

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

Определяется сегментный адрес свободного участка памяти для загрузки программы, 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. Стек также можно определить с помощью стандартной директивы SEGMENT, использовав команду

Имя_Сегмента SEGMENT STACK

…

Имя_Сегмента ENDS

4. Как определяется точка входа?

Точка входа определяется в сегменте кода после директивы END:

END ИмяТочкиВхода

Вывод.

Были исследованы различия в структурах исходных текстов модулей типов .COM и .EXE, структуры файлов загрузочных модулей и способы их загрузки в основную память.