**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. 6381 |  | Попов Н.В. |
| Преподаватель |  | Губкин А.Ф. |

Санкт-Петербург

2018

**Постановка задачи.**

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

Таблица переменных

|  |  |  |
| --- | --- | --- |
| Переменные | Тип хранимых данных | Назначение |
| TypePC\_FF | DB | Вывод строки 'Тип IBM PC - PC',10,13,'$' |
| TypePC\_FEFB | DB | Вывод строки 'Тип IBM PC - PC/XT',10,13,'$' |
| TypePC\_AT | DB | Вывод строки 'Тип IBM PC - AT',10,13,'$' |
| TypePC\_FA | DB | Вывод строки 'Тип IBM PC - PS2 модель 30',10,13,'$' |
| TypePC\_FC | DB | Вывод строки 'Тип IBM PC - PS2 модель 50 или 60',10,13,'$' |
| TypePC\_F8 | DB | Вывод строки 'Тип IBM PC - PS2 модель 80',10,13,'$' |
| TypePC\_FD | DB | Вывод строки 'Тип IBM PC - PCjr',10,13,'$' |
| TypePC\_F9 | DB | Вывод строки 'Тип IBM PC - PC Convertible',10,13,'$' |
| ErTypePC | DB | Вывод строки ' $'. Вместо пробелов помещается тип ПС в 16 с/с |
| ErrorType | DB | Вывод строки 'Код не совпадает ни с одним значением',10,13,'Код = $' |
| VersionSyst | DB | Вывод строки 'Версия системы MS DOS - . $' |
| SerNumOEM | DB | Вывод строки 10,13,'Серийный номер ОЕМ - $' |
| SerNumUser | DB | Вывод строки 10,13,'Серийный номер пользователя - $' |
| SerNumUserStr | DB | Вывод строки ' $'. Вместо пробелов помещается номер пользователя. |

Описание функций и структур данных:

1. FIND\_TYPE\_PC: печатает тип ОС. Если определить не удалось, выводит соответствующее сообщение и полученное значение в 16ричной с/с из предпоследнего байта ROM BIOS.
2. FIND\_VERSION\_MS\_DOS: печатает версию ОС, серийный номер OEM и серийный номер пользователя
3. TETR\_TO\_HEX: вспомогательная функция для работы функции BYTE\_TO\_HEX.
4. BYTE\_TO\_HEX: переводит число AL в коды символов 16-ой с/с, записывая получившееся в bl и bh
5. WRD\_TO\_HEX: переводит число AX в строку в 16-ой с/с, записывая получившееся в di, начиная с младшей цифры
6. BYTE\_TO\_DEC: переводит байт из AL в десятичную с/с и записывает получившееся число по адресу si, начиная с младшей цифры

L1.exe – «плохой» EXE, L1exe.exe – «хороший»

Последовательность действий, выполняемых утилитой:

1. Определение и печать типа ОС с помощью функции FIND\_TYPE\_PC
2. Определение и печать версии ОС, серийного номера OEM и серийного номера пользователя с помощью функции FIND\_VERSION\_MS\_DOS

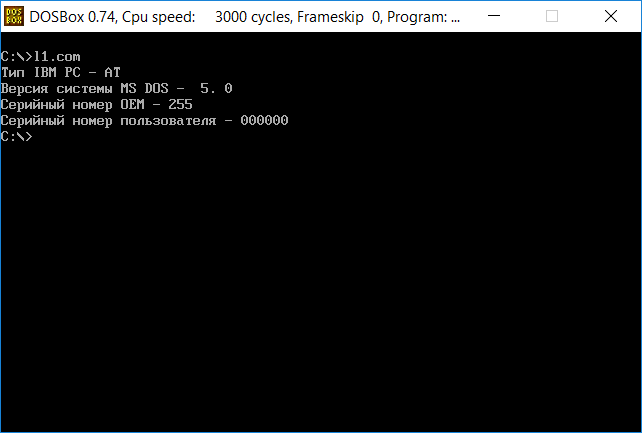


Рисунок 1. Работа программы L1.COM.

**Заключение.**

В процессе выполнения данной 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?

Com-программа содержит один сегмент, поэтому ее размер составляет 64 Кб.

1. EXE-программа?

EXE-программа содержит не менее 1 сегмента, поэтому ее размер не ограничен 64 КБ.

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

В тексте COM-программы обязательно должна быть директива ORG 100h, сдвигающая адресацию в программе на 256 байт, т.к. в первых 256 байтах программы находится PSP.

Также должна присутствовать директива ASSUME, так как ассемблеру необходимо знать о сегменте кода для того, чтобы установить выполняемую программу.

Если директива ASSUME будет закомментирована, то при компиляции выявится ошибка о том, что регистр CS не указывает на начало сегмента кода. (Рис. 2)

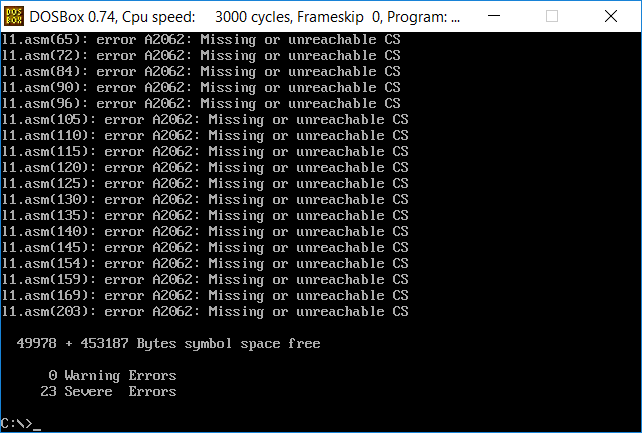


Рисунок 2. Ошибки при компиляции с закомментированной директивой ASSUME.

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

В COM-программе нельзя использовать команды вида mov register,segment и команды, содержащие дальнюю(far) адресацию, т.к. в этих командах используется relocation table, в которой содержатся адреса сегментов. Такой таблицы в COM-программах нет, поэтому и эти команды не действительны.

**Отличия форматов файлов COM и EXE модулей**

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

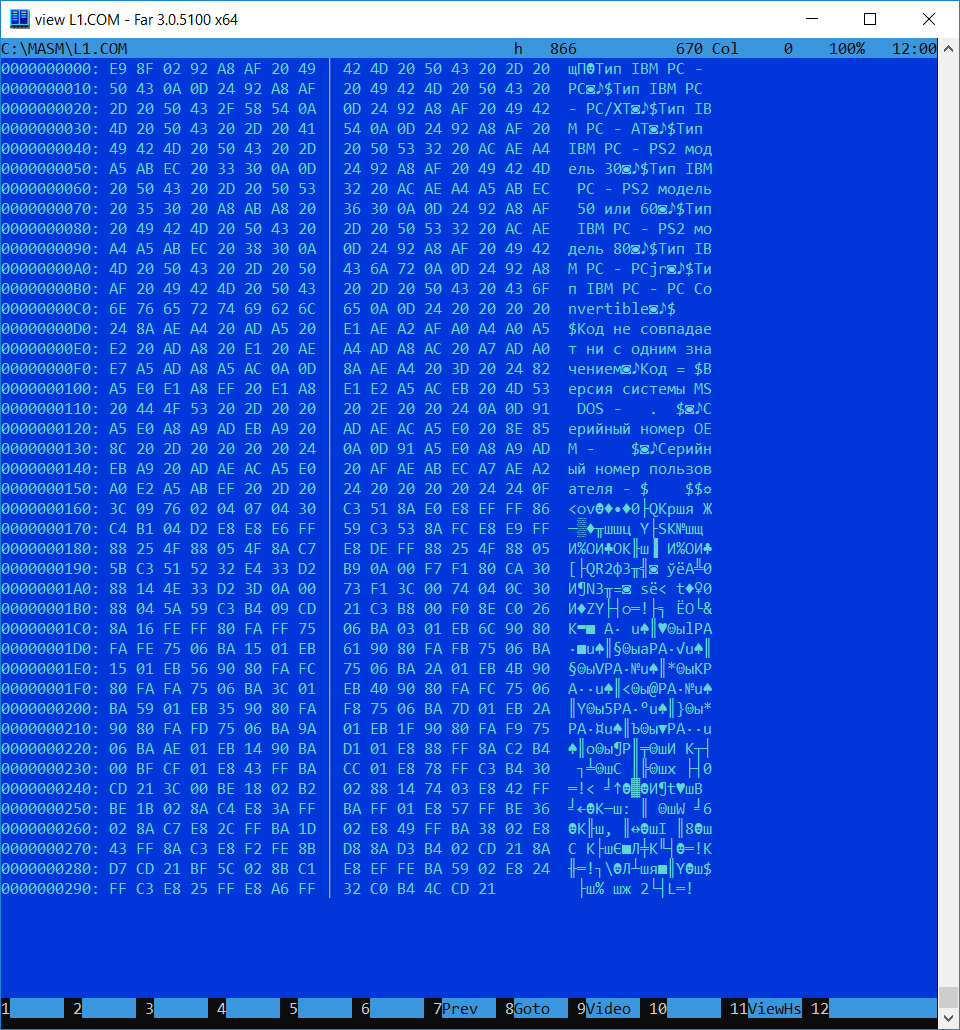


Рисунок 2. Файл загрузочного модуля L1.COM в 16 с/с.

COM-файл содержит только данные и код. В файле код располагается с нулевого адреса.

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

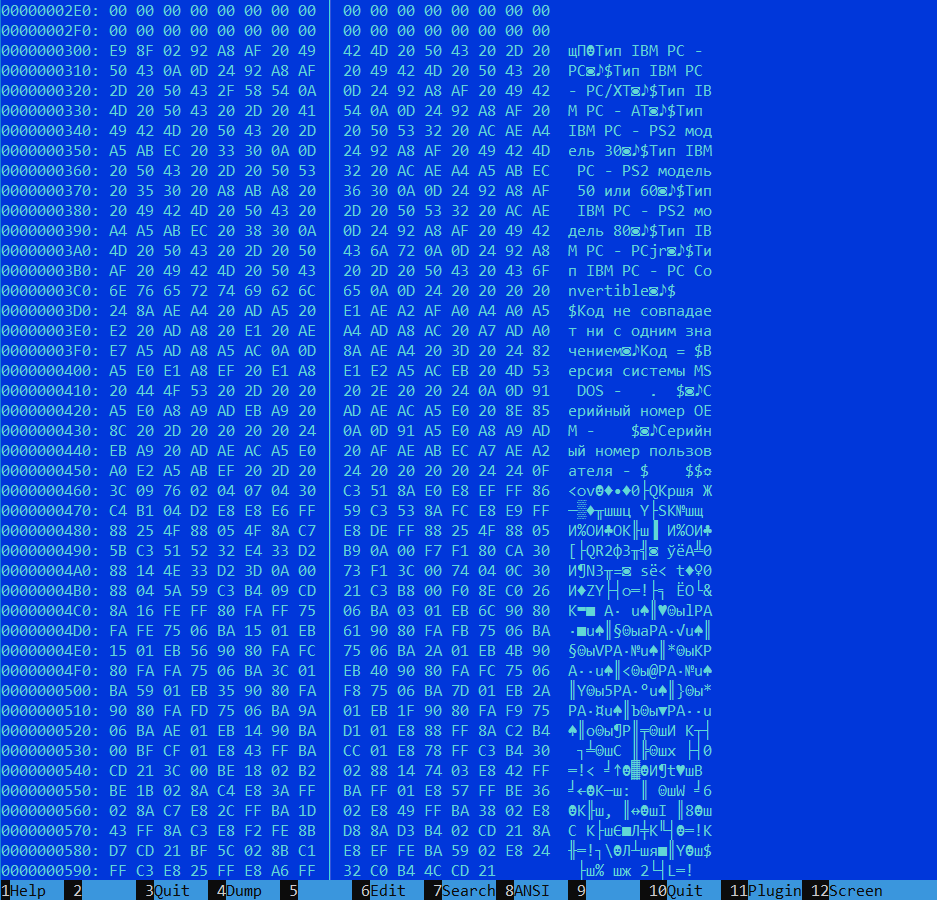
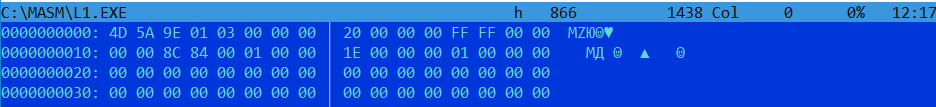
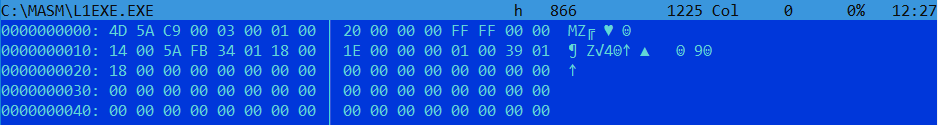


Рисунок 3. Файл загрузочного модуля плохого L1.EXE в 16 с/с.

В «плохом» EXE-файле код и данные не разделены по сегментам, а перемешаны: перед данными по адресу 300h находится метка перехода E9 8F 02. Код располагается с адреса 300h, т.к. заголовок занимает 200h байт и команда ORG 100h «сдвигает» код на дополнительные 100h. С нулевого адреса располагается заголовок. В первых двух байтах можно увидеть символы MZ, означающие, что формат файла – 16-битный и его следует запускать в соответствии со структурой EXE-файлов. За заголовком следует relocation table.

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



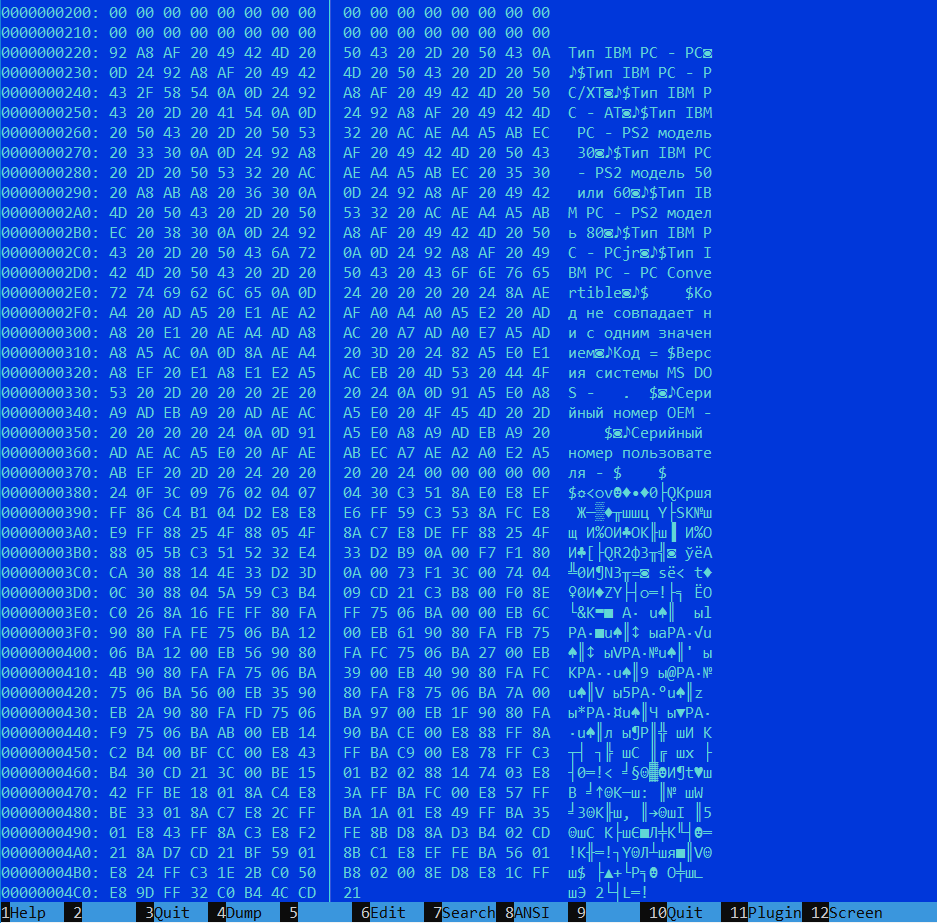


Рисунок 4. Файл загрузочного модуля хорошего L1.EXE в 16 с/с.

В отличие от «плохого» EXE-файла, в «хорошем» код, стек и данные выделены в отдельные сегменты. Код начинается с 220h байта, т.к. первые 200 байт содержат заголовк, а 200h-220h байты занимает сегмент стека.

Для «хорошего» EXE в директиве org 100h нет необходимости, т.к. загрузчик автоматически расположит программу после PSP.

**Загрузка COM модуля в основную память**

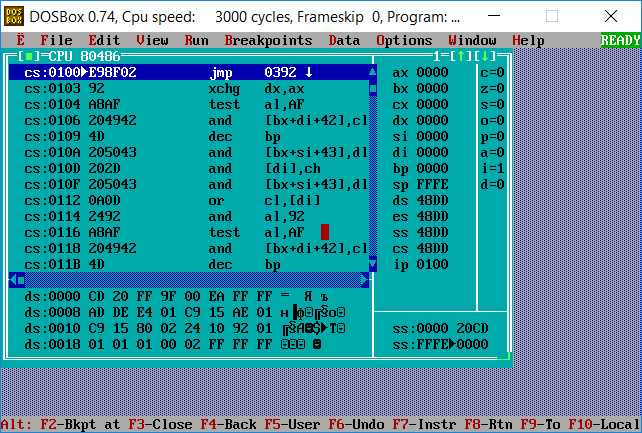


Рисунок 5. Отладчик TD.EXE и загруженный в него файл L1.COM.

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

Формат загрузки модуля COM:

1. Выделение сегмента памяти для модуля
2. Установка всех сегментных регистров на начало выделенного сегмента памяти
3. Построение в первых 100h байтах памяти PSP
4. Загрузка содержимого COM-файла и присваивание регистру IP значения 100h.
5. Регистр SP устанавливается в конец сегмента
6. Что располагается с адреса 0?

С нулевого адреса располагается PSP.

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

Все сегментные регистры равны в данном случае 48DD и указывают на начало PSP.

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

Стек занимает весь сегмент COM-программы, его начало находится в конце сегмента. SS указывает на начало сегмента, а SP=FFFEh – на его конец. Стек может дойти до кода/данных программы при достаточном количестве элементов.

**Загрузка «хорошего» EXE модуля в основную память**

1. Как загружается «хороший» EXE? Какие значения имеют сегментные регистры? На что указывают регистры DS и ES?

DS и ES имеют значение 48DD и указывают на начало PSP, SS=48ED – начало сегмента стека, CS=4905 – начало сегмента команд.

1. Как определяется стек?

Стек выделяется в программе явным образом в сегменте стека.

1. Как определяется точка входа?

Последняя строка программы содержит директиву END, которая говорит программе ассемблера, что закончился вообще весь текст программы, и больше ничего транслировать не нужно. В качестве операнда этой директивы, указывается точка входа в программу, т.е. адрес первой выполнимой программной строки, т е Main.

**Приложение А**

**Исходный файл L1.asm**

TESTPC SEGMENT

ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING

ORG 100H

START:

JMP BEGIN

;Данные

TypePC\_FF DB 'Тип IBM PC - PC',10,13,'$'

TypePC\_FEFB DB 'Тип IBM PC - PC/XT',10,13,'$'

TypePC\_AT DB 'Тип IBM PC - AT',10,13,'$'

TypePC\_FA DB 'Тип IBM PC - PS2 модель 30',10,13,'$'

TypePC\_FC DB 'Тип IBM PC - PS2 модель 50 или 60',10,13,'$'

TypePC\_F8 DB 'Тип IBM PC - PS2 модель 80',10,13,'$'

TypePC\_FD DB 'Тип IBM PC - PCjr',10,13,'$'

TypePC\_F9 DB 'Тип IBM PC - PC Convertible',10,13,'$'

ErTypePC DB ' $'

ErrorType DB 'Код не совпадает ни с одним значением',10,13,'Код = $'

VersionSyst DB 'Версия системы MS DOS - . $'

SerNumOEM DB 10,13,'Серийный номер ОЕМ - $'

SerNumUser DB 10,13,'Серийный номер пользователя - $'

SerNumUserStr DB ' $'

;Процедуры

;--------------------------------------------------

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 шестн. 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

;AX - число, DI - адрес последего 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

;-------------------------------

PRINT\_STR PROC near ;Печать строки, помещенной в DX

mov ah,09h

int 21h

ret

PRINT\_STR ENDP

;-------------------------------

FIND\_TYPE\_PC PROC near ;Нахождение типа ПК

mov ax, 0f000h

mov es,ax

mov dl,es:[0fffeh]

cmp dl, 0FFh

jne x1

mov DX,offset TypePC\_FF

jmp wrType

x1:

cmp dl, 0feh

jne x2

mov DX,offset TypePC\_FEFB

jmp wrType

x2:

cmp dl, 0fbh

jne x3

mov DX,offset TypePC\_FEFB

jmp wrType

x3:

cmp dl, 0fch

jne x4

mov DX,offset TypePC\_AT

jmp wrType

x4:

cmp dl, 0fah

jne x5

mov DX,offset TypePC\_FA

jmp wrType

x5:

cmp dl, 0fch

jne x6

mov DX,offset TypePC\_FC

jmp wrType

x6:

cmp dl, 0f8h

jne x7

mov DX,offset TypePC\_F8

jmp wrType

x7:

cmp dl, 0fdh

jne x8

mov DX,offset TypePC\_FD

jmp wrType

x8:

cmp dl, 0f9h

jne x9

mov DX,offset TypePC\_F9

jmp wrType

x9:

mov DX,offset ErrorType

call PRINT\_STR

mov al, dl

mov ah, 0

mov di,offset ErTypePC+3

call WRD\_TO\_HEX

mov DX,offset ErTypePC

wrType:

call PRINT\_STR

ret

FIND\_TYPE\_PC ENDP

;-------------------------------

FIND\_VERSION\_MS\_DOS PROC near ;Нахождение версии MS DOC

mov ah,30h

int 21h

cmp al,0

mov si,offset VersionSyst+25

mov dl,02h

mov [si],dl

je numBasVer0

call BYTE\_TO\_DEC

numBasVer0:

mov si,offset VersionSyst+28

mov al,ah

call BYTE\_TO\_DEC

mov DX,offset VersionSyst

call PRINT\_STR

mov si,offset SerNumOEM+25

mov al,bh

call BYTE\_TO\_DEC

mov dx,offset SerNumOEM

call PRINT\_STR

mov dx,offset SerNumUser

call PRINT\_STR

mov al,bl

call BYTE\_TO\_HEX

mov bx,ax

mov dl,bl

mov ah,02h

int 21h

mov dl,bh

int 21h

mov di,offset SerNumUserStr+3

mov ax,cx

call WRD\_TO\_HEX

mov dx,offset SerNumUserStr

call PRINT\_STR

ret

FIND\_VERSION\_MS\_DOS ENDP

;-------------------------------

; КОД

BEGIN:

call FIND\_TYPE\_PC

call FIND\_VERSION\_MS\_DOS

;Выход в DOS

xor AL,AL

mov AH,4Ch

int 21H

TESTPC ENDS

END START

**Исходный код L1exe.asm**

AStack SEGMENT STACK

DW 10 DUP(?)

AStack ENDS

;//////////////////////////////////////////////////

DATA SEGMENT

TypePC\_FF DB 'Тип IBM PC - PC',10,13,'$'

TypePC\_FEFB DB 'Тип IBM PC - PC/XT',10,13,'$'

TypePC\_AT DB 'Тип IBM PC - AT',10,13,'$'

TypePC\_FA DB 'Тип IBM PC - PS2 модель 30',10,13,'$'

TypePC\_FC DB 'Тип IBM PC - PS2 модель 50 или 60',10,13,'$'

TypePC\_F8 DB 'Тип IBM PC - PS2 модель 80',10,13,'$'

TypePC\_FD DB 'Тип IBM PC - PCjr',10,13,'$'

TypePC\_F9 DB 'Тип IBM PC - PC Convertible',10,13,'$'

ErTypePC DB ' $'

ErrorType DB 'Код не совпадает ни с одним значением',10,13,'Код = $'

VersionSyst DB 'Версия системы MS DOS - . $'

SerNumOEM DB 10,13,'Серийный номер OEM - $'

SerNumUser DB 10,13,'Серийный номер пользователя - $'

SerNumUserStr DB ' $'

DATA ENDS

;//////////////////////////////////////////////////

CODE SEGMENT

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

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

push CX

mov AH,AL

call TETR\_TO\_HEX

xchg AL,AH

mov CL,4

shr AL,CL

call TETR\_TO\_HEX

pop CX

ret

BYTE\_TO\_HEX ENDP

;-------------------------------

WRD\_TO\_HEX PROC near

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

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

;-------------------------------

PRINT\_STR PROC near

mov ah,09h

int 21h

ret

PRINT\_STR ENDP

;-------------------------------

FIND\_TYPE\_PC PROC near

mov ax, 0f000h

mov es,ax

mov dl,es:[0fffeh]

cmp dl, 0FFh

jne x1

mov DX,offset TypePC\_FF

jmp wrType

x1:

cmp dl, 0feh

jne x2

mov DX,offset TypePC\_FEFB

jmp wrType

x2:

cmp dl, 0fbh

jne x3

mov DX,offset TypePC\_FEFB

jmp wrType

x3:

cmp dl, 0fch

jne x4

mov DX,offset TypePC\_AT

jmp wrType

x4:

cmp dl, 0fah

jne x5

mov DX,offset TypePC\_FA

jmp wrType

x5:

cmp dl, 0fch

jne x6

mov DX,offset TypePC\_FC

jmp wrType

x6:

cmp dl, 0f8h

jne x7

mov DX,offset TypePC\_F8

jmp wrType

x7:

cmp dl, 0fdh

jne x8

mov DX,offset TypePC\_FD

jmp wrType

x8:

cmp dl, 0f9h

jne x9

mov DX,offset TypePC\_F9

jmp wrType

x9:

mov DX,offset ErrorType

call PRINT\_STR

mov al, dl

mov ah, 0

mov di,offset ErTypePC+3

call WRD\_TO\_HEX

mov DX,offset ErTypePC

wrType:

call PRINT\_STR

ret

FIND\_TYPE\_PC ENDP

;-------------------------------

FIND\_VERSION\_MS\_DOS PROC near

mov ah,30h

int 21h

cmp al,0

mov si,offset VersionSyst+25

mov dl,02h

mov [si],dl

je numBasVer0

call BYTE\_TO\_DEC

numBasVer0:

mov si,offset VersionSyst+28

mov al,ah

call BYTE\_TO\_DEC

mov DX,offset VersionSyst

call PRINT\_STR

mov si,offset SerNumOEM+25

mov al,bh

call BYTE\_TO\_DEC

mov dx,offset SerNumOEM

call PRINT\_STR

mov dx,offset SerNumUser

call PRINT\_STR

mov al,bl

call BYTE\_TO\_HEX

mov bx,ax

mov dl,bl

mov ah,02h

int 21h

mov dl,bh

int 21h

mov di,offset SerNumUserStr+3

mov ax,cx

call WRD\_TO\_HEX

mov dx,offset SerNumUserStr

call PRINT\_STR

ret

FIND\_VERSION\_MS\_DOS ENDP

;-------------------------------

Main PROC near

push DS

sub AX, AX

push AX

mov AX, DATA

mov DS,AX

call FIND\_TYPE\_PC

call FIND\_VERSION\_MS\_DOS

xor AL,AL

mov AH,4Ch

int 21H

Main ENDP

CODE ENDS

END Main

**Приложение Б**

**Файл листинга L1.asm**

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:37:02

Page 1-1

0000 TESTPC SEGMENT

ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING

, SS:NOTHING

0100 ORG 100H

0100 START:

0100 E9 0392 R JMP BEGIN

;„ ­­лҐ

0103 92 A8 AF 20 49 42 TypePC\_FF DB '’ЁЇ IBM PC - PC',10,13,'$

'

4D 20 50 43 20 2D

20 50 43 0A 0D 24

0115 92 A8 AF 20 49 42 TypePC\_FEFB DB '’ЁЇ IBM PC - PC/XT',10,

13,'$'

4D 20 50 43 20 2D

20 50 43 2F 58 54

0A 0D 24

012A 92 A8 AF 20 49 42 TypePC\_AT DB '’ЁЇ IBM PC - AT',10,13,'$

'

4D 20 50 43 20 2D

20 41 54 0A 0D 24

013C 92 A8 AF 20 49 42 TypePC\_FA DB '’ЁЇ IBM PC - PS2 ¬®¤Ґ«м 3

0',10,13,'$'

4D 20 50 43 20 2D

20 50 53 32 20 AC

AE A4 A5 AB EC 20

33 30 0A 0D 24

0159 92 A8 AF 20 49 42 TypePC\_FC DB '’ЁЇ IBM PC - PS2 ¬®¤Ґ«м 5

0 Ё«Ё 60',10,13,'$'

4D 20 50 43 20 2D

20 50 53 32 20 AC

AE A4 A5 AB EC 20

35 30 20 A8 AB A8

20 36 30 0A 0D 24

017D 92 A8 AF 20 49 42 TypePC\_F8 DB '’ЁЇ IBM PC - PS2 ¬®¤Ґ«м 8

0',10,13,'$'

4D 20 50 43 20 2D

20 50 53 32 20 AC

AE A4 A5 AB EC 20

38 30 0A 0D 24

019A 92 A8 AF 20 49 42 TypePC\_FD DB '’ЁЇ IBM PC - PCjr',10,13,

'$'

4D 20 50 43 20 2D

20 50 43 6A 72 0A

0D 24

01AE 92 A8 AF 20 49 42 TypePC\_F9 DB '’ЁЇ IBM PC - PC Convertib

le',10,13,'$'

4D 20 50 43 20 2D

20 50 43 20 43 6F

6E 76 65 72 74 69

62 6C 65 0A 0D 24

01CC 20 20 20 20 24 ErTypePC DB ' $'

01D1 8A AE A4 20 AD A5 ErrorType DB 'Љ®¤ ­Ґ б®ўЇ ¤ Ґв ­Ё б ®¤­

Ё¬ §­ зҐ­ЁҐ¬',10,13,'Љ®¤ = $'

20 E1 AE A2 AF A0

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:37:02

Page 1-2

A4 A0 A5 E2 20 AD

A8 20 E1 20 AE A4

AD A8 AC 20 A7 AD

A0 E7 A5 AD A8 A5

AC 0A 0D 8A AE A4

20 3D 20 24

01FF 82 A5 E0 E1 A8 EF VersionSyst DB '‚ҐабЁп бЁбвҐ¬л MS DOS -

. $'

20 E1 A8 E1 E2 A5

AC EB 20 4D 53 20

44 4F 53 20 2D 20

20 20 2E 20 20 24

021D 0A 0D 91 A5 E0 A8 SerNumOEM DB 10,13,'‘ҐаЁ©­л© ­®¬Ґа Ћ…Њ

- $'

A9 AD EB A9 20 AD

AE AC A5 E0 20 8E

85 8C 20 2D 20 20

20 20 24

0238 0A 0D 91 A5 E0 A8 SerNumUser DB 10,13,'‘ҐаЁ©­л© ­®¬Ґа Ї®«

м§®ў вҐ«п - $'

A9 AD EB A9 20 AD

AE AC A5 E0 20 AF

AE AB EC A7 AE A2

A0 E2 A5 AB EF 20

2D 20 24

0259 20 20 20 20 24 SerNumUserStr DB ' $'

;Џа®жҐ¤гал

;----------------------------------------------

----

025E TETR\_TO\_HEX PROC NEAR

025E 24 0F and al,0fh

0260 3C 09 cmp AL,09

0262 76 02 jbe NEXT

0264 04 07 add AL,07

0266 NEXT:

0266 04 30 add AL,30h

0268 C3 ret

0269 TETR\_TO\_HEX ENDP

;----------------------------------------------

----

0269 BYTE\_TO\_HEX PROC near

;Ў ©в ў AL ЇҐаҐў®¤Ёвбп ў ¤ў бЁ¬ў®« иҐбв­. зЁб

« ў AX

0269 51 push CX

026A 8A E0 mov AH,AL

026C E8 025E R call TETR\_TO\_HEX

026F 86 C4 xchg AL,AH

0271 B1 04 mov CL,4

0273 D2 E8 shr AL,CL

0275 E8 025E R call TETR\_TO\_HEX ; ў AL бв аи п жЁда

0278 59 pop CX ; ў AH ¬« ¤и п

0279 C3 ret

027A BYTE\_TO\_HEX ENDP

;-------------------------------

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:37:02

Page 1-3

027A WRD\_TO\_HEX PROC near

;ЇҐаҐў®¤ ў 16 б/б 16-вЁ а §ап¤­®Ј® зЁб«

;AX - зЁб«®, DI - ¤аҐб Ї®б«Ґ¤ҐЈ® бЁ¬ў®«

027A 53 push BX

027B 8A FC mov BH,AH

027D E8 0269 R call BYTE\_TO\_HEX

0280 88 25 mov [DI],AH

0282 4F dec DI

0283 88 05 mov [DI],AL

0285 4F dec DI

0286 8A C7 mov AL,BH

0288 E8 0269 R call BYTE\_TO\_HEX

028B 88 25 mov [DI],AH

028D 4F dec DI

028E 88 05 mov [DI],AL

0290 5B pop BX

0291 C3 ret

0292 WRD\_TO\_HEX ENDP

;----------------------------------------------

----

0292 BYTE\_TO\_DEC PROC near

; ЇҐаҐў®¤ ў 10б/б, SI - ¤аҐб Ї®«п ¬« ¤иҐ© жЁда

л

0292 51 push CX

0293 52 push DX

0294 32 E4 xor AH,AH

0296 33 D2 xor DX,DX

0298 B9 000A mov CX,10

029B loop\_bd:

029B F7 F1 div CX

029D 80 CA 30 or DL,30h

02A0 88 14 mov [SI],DL

02A2 4E dec SI

02A3 33 D2 xor DX,DX

02A5 3D 000A cmp AX,10

02A8 73 F1 jae loop\_bd

02AA 3C 00 cmp AL,00h

02AC 74 04 je end\_l

02AE 0C 30 or AL,30h

02B0 88 04 mov [SI],AL

02B2 end\_l:

02B2 5A pop DX

02B3 59 pop CX

02B4 C3 ret

02B5 BYTE\_TO\_DEC ENDP

;-------------------------------

02B5 PRINT\_STR PROC near ;ЏҐз вм бва®ЄЁ, Ї®¬ҐйҐ­­®©

ў DX

02B5 B4 09 mov ah,09h

02B7 CD 21 int 21h

02B9 C3 ret

02BA PRINT\_STR ENDP

;-------------------------------

02BA FIND\_TYPE\_PC PROC near ;Ќ е®¦¤Ґ­ЁҐ вЁЇ ЏЉ

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:37:02

Page 1-4

02BA B8 F000 mov ax, 0f000h

02BD 8E C0 mov es,ax

02BF 26: 8A 16 FFFE mov dl,es:[0fffeh]

02C4 80 FA FF cmp dl, 0FFh

02C7 75 06 jne x1

02C9 BA 0103 R mov DX,offset TypePC\_FF

02CC EB 6C 90 jmp wrType

02CF x1:

02CF 80 FA FE cmp dl, 0feh

02D2 75 06 jne x2

02D4 BA 0115 R mov DX,offset TypePC\_FEFB

02D7 EB 61 90 jmp wrType

02DA x2:

02DA 80 FA FB cmp dl, 0fbh

02DD 75 06 jne x3

02DF BA 0115 R mov DX,offset TypePC\_FEFB

02E2 EB 56 90 jmp wrType

02E5 x3:

02E5 80 FA FC cmp dl, 0fch

02E8 75 06 jne x4

02EA BA 012A R mov DX,offset TypePC\_AT

02ED EB 4B 90 jmp wrType

02F0 x4:

02F0 80 FA FA cmp dl, 0fah

02F3 75 06 jne x5

02F5 BA 013C R mov DX,offset TypePC\_FA

02F8 EB 40 90 jmp wrType

02FB x5:

02FB 80 FA FC cmp dl, 0fch

02FE 75 06 jne x6

0300 BA 0159 R mov DX,offset TypePC\_FC

0303 EB 35 90 jmp wrType

0306 x6:

0306 80 FA F8 cmp dl, 0f8h

0309 75 06 jne x7

030B BA 017D R mov DX,offset TypePC\_F8

030E EB 2A 90 jmp wrType

0311 x7:

0311 80 FA FD cmp dl, 0fdh

0314 75 06 jne x8

0316 BA 019A R mov DX,offset TypePC\_FD

0319 EB 1F 90 jmp wrType

031C x8:

031C 80 FA F9 cmp dl, 0f9h

031F 75 06 jne x9

0321 BA 01AE R mov DX,offset TypePC\_F9

0324 EB 14 90 jmp wrType

0327 x9:

0327 BA 01D1 R mov DX,offset ErrorType

032A E8 02B5 R call PRINT\_STR

032D 8A C2 mov al, dl

032F B4 00 mov ah, 0

0331 BF 01CF R mov di,offset ErTypePC+3

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:37:02

Page 1-5

0334 E8 027A R call WRD\_TO\_HEX

0337 BA 01CC R mov DX,offset ErTypePC

033A wrType:

033A E8 02B5 R call PRINT\_STR

033D C3 ret

033E FIND\_TYPE\_PC ENDP

;-------------------------------

033E FIND\_VERSION\_MS\_DOS PROC near ;Ќ е®¦¤Ґ­ЁҐ ўҐабЁ

Ё MS DOC

033E B4 30 mov ah,30h

0340 CD 21 int 21h

0342 3C 00 cmp al,0

0344 BE 0218 R mov si,offset VersionSyst+25

0347 B2 02 mov dl,02h

0349 88 14 mov [si],dl

034B 74 03 je numBasVer0

034D E8 0292 R call BYTE\_TO\_DEC

0350 numBasVer0:

0350 BE 021B R mov si,offset VersionSyst+28

0353 8A C4 mov al,ah

0355 E8 0292 R call BYTE\_TO\_DEC

0358 BA 01FF R mov DX,offset VersionSyst

035B E8 02B5 R call PRINT\_STR

035E BE 0236 R mov si,offset SerNumOEM+25

0361 8A C7 mov al,bh

0363 E8 0292 R call BYTE\_TO\_DEC

0366 BA 021D R mov dx,offset SerNumOEM

0369 E8 02B5 R call PRINT\_STR

036C BA 0238 R mov dx,offset SerNumUser

036F E8 02B5 R call PRINT\_STR

0372 8A C3 mov al,bl

0374 E8 0269 R call BYTE\_TO\_HEX

0377 8B D8 mov bx,ax

0379 8A D3 mov dl,bl

037B B4 02 mov ah,02h

037D CD 21 int 21h

037F 8A D7 mov dl,bh

0381 CD 21 int 21h

0383 BF 025C R mov di,offset SerNumUserStr+3

0386 8B C1 mov ax,cx

0388 E8 027A R call WRD\_TO\_HEX

038B BA 0259 R mov dx,offset SerNumUserStr

038E E8 02B5 R call PRINT\_STR

0391 C3 ret

0392 FIND\_VERSION\_MS\_DOS ENDP

;-------------------------------

; ЉЋ„

0392 BEGIN:

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:37:02

Page 1-6

0392 E8 02BA R call FIND\_TYPE\_PC

0395 E8 033E R call FIND\_VERSION\_MS\_DOS

;‚ле®¤ ў DOS

0398 32 C0 xor AL,AL

039A B4 4C mov AH,4Ch

039C CD 21 int 21H

039E TESTPC ENDS

END START

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:37:02

Symbols-1

Segments and Groups:

N a m e Length Align Combine Class

TESTPC . . . . . . . . . . . . . 039E PARA NONE

Symbols:

N a m e Type Value Attr

BEGIN . . . . . . . . . . . . . L NEAR 0392 TESTPC

BYTE\_TO\_DEC . . . . . . . . . . N PROC 0292 TESTPC Length = 0023

BYTE\_TO\_HEX . . . . . . . . . . N PROC 0269 TESTPC Length = 0011

END\_L . . . . . . . . . . . . . L NEAR 02B2 TESTPC

ERRORTYPE . . . . . . . . . . . L BYTE 01D1 TESTPC

ERTYPEPC . . . . . . . . . . . . L BYTE 01CC TESTPC

FIND\_TYPE\_PC . . . . . . . . . . N PROC 02BA TESTPC Length = 0084

FIND\_VERSION\_MS\_DOS . . . . . . N PROC 033E TESTPC Length = 0054

LOOP\_BD . . . . . . . . . . . . L NEAR 029B TESTPC

NEXT . . . . . . . . . . . . . . L NEAR 0266 TESTPC

NUMBASVER0 . . . . . . . . . . . L NEAR 0350 TESTPC

PRINT\_STR . . . . . . . . . . . N PROC 02B5 TESTPC Length = 0005

SERNUMOEM . . . . . . . . . . . L BYTE 021D TESTPC

SERNUMUSER . . . . . . . . . . . L BYTE 0238 TESTPC

SERNUMUSERSTR . . . . . . . . . L BYTE 0259 TESTPC

START . . . . . . . . . . . . . L NEAR 0100 TESTPC

TETR\_TO\_HEX . . . . . . . . . . N PROC 025E TESTPC Length = 000B

TYPEPC\_AT . . . . . . . . . . . L BYTE 012A TESTPC

TYPEPC\_F8 . . . . . . . . . . . L BYTE 017D TESTPC

TYPEPC\_F9 . . . . . . . . . . . L BYTE 01AE TESTPC

TYPEPC\_FA . . . . . . . . . . . L BYTE 013C TESTPC

TYPEPC\_FC . . . . . . . . . . . L BYTE 0159 TESTPC

TYPEPC\_FD . . . . . . . . . . . L BYTE 019A TESTPC

TYPEPC\_FEFB . . . . . . . . . . L BYTE 0115 TESTPC

TYPEPC\_FF . . . . . . . . . . . L BYTE 0103 TESTPC

VERSIONSYST . . . . . . . . . . L BYTE 01FF TESTPC

WRD\_TO\_HEX . . . . . . . . . . . N PROC 027A TESTPC Length = 0018

WRTYPE . . . . . . . . . . . . . L NEAR 033A TESTPC

X1 . . . . . . . . . . . . . . . L NEAR 02CF TESTPC

X2 . . . . . . . . . . . . . . . L NEAR 02DA TESTPC

X3 . . . . . . . . . . . . . . . L NEAR 02E5 TESTPC

X4 . . . . . . . . . . . . . . . L NEAR 02F0 TESTPC

X5 . . . . . . . . . . . . . . . L NEAR 02FB TESTPC

X6 . . . . . . . . . . . . . . . L NEAR 0306 TESTPC

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:37:02

Symbols-2

X7 . . . . . . . . . . . . . . . L NEAR 0311 TESTPC

X8 . . . . . . . . . . . . . . . L NEAR 031C TESTPC

X9 . . . . . . . . . . . . . . . L NEAR 0327 TESTPC

@CPU . . . . . . . . . . . . . . TEXT 0101h

@FILENAME . . . . . . . . . . . TEXT l1

@VERSION . . . . . . . . . . . . TEXT 510

211 Source Lines

211 Total Lines

43 Symbols

47916 + 451152 Bytes symbol space free

0 Warning Errors

0 Severe Errors

**Map-файл L1.asm**

LINK : warning L4021: no stack segment

Start Stop Length Name Class

00000H 0039DH 0039EH TESTPC

Program entry point at 0000:0100

**Файл листинга L1exe.exe**

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:43:41

Page 1-1

0000 AStack SEGMENT STACK

0000 000A[ DW 10 DUP(?)

????

]

0014 AStack ENDS

;//////////////////////////////////////////////

////

0000 DATA SEGMENT

0000 92 A8 AF 20 49 42 TypePC\_FF DB '’ЁЇ IBM PC - PC',10,13,'$

'

4D 20 50 43 20 2D

20 50 43 0A 0D 24

0012 92 A8 AF 20 49 42 TypePC\_FEFB DB '’ЁЇ IBM PC - PC/XT',10,

13,'$'

4D 20 50 43 20 2D

20 50 43 2F 58 54

0A 0D 24

0027 92 A8 AF 20 49 42 TypePC\_AT DB '’ЁЇ IBM PC - AT',10,13,'$

'

4D 20 50 43 20 2D

20 41 54 0A 0D 24

0039 92 A8 AF 20 49 42 TypePC\_FA DB '’ЁЇ IBM PC - PS2 ¬®¤Ґ«м 3

0',10,13,'$'

4D 20 50 43 20 2D

20 50 53 32 20 AC

AE A4 A5 AB EC 20

33 30 0A 0D 24

0056 92 A8 AF 20 49 42 TypePC\_FC DB '’ЁЇ IBM PC - PS2 ¬®¤Ґ«м 5

0 Ё«Ё 60',10,13,'$'

4D 20 50 43 20 2D

20 50 53 32 20 AC

AE A4 A5 AB EC 20

35 30 20 A8 AB A8

20 36 30 0A 0D 24

007A 92 A8 AF 20 49 42 TypePC\_F8 DB '’ЁЇ IBM PC - PS2 ¬®¤Ґ«м 8

0',10,13,'$'

4D 20 50 43 20 2D

20 50 53 32 20 AC

AE A4 A5 AB EC 20

38 30 0A 0D 24

0097 92 A8 AF 20 49 42 TypePC\_FD DB '’ЁЇ IBM PC - PCjr',10,13,

'$'

4D 20 50 43 20 2D

20 50 43 6A 72 0A

0D 24

00AB 92 A8 AF 20 49 42 TypePC\_F9 DB '’ЁЇ IBM PC - PC Convertib

le',10,13,'$'

4D 20 50 43 20 2D

20 50 43 20 43 6F

6E 76 65 72 74 69

62 6C 65 0A 0D 24

00C9 20 20 20 20 24 ErTypePC DB ' $'

00CE 8A AE A4 20 AD A5 ErrorType DB 'Љ®¤ ­Ґ б®ўЇ ¤ Ґв ­Ё б ®¤­

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:43:41

Page 1-2

Ё¬ §­ зҐ­ЁҐ¬',10,13,'Љ®¤ = $'

20 E1 AE A2 AF A0

A4 A0 A5 E2 20 AD

A8 20 E1 20 AE A4

AD A8 AC 20 A7 AD

A0 E7 A5 AD A8 A5

AC 0A 0D 8A AE A4

20 3D 20 24

00FC 82 A5 E0 E1 A8 EF VersionSyst DB '‚ҐабЁп бЁбвҐ¬л MS DOS -

. $'

20 E1 A8 E1 E2 A5

AC EB 20 4D 53 20

44 4F 53 20 2D 20

20 20 2E 20 20 24

011A 0A 0D 91 A5 E0 A8 SerNumOEM DB 10,13,'‘ҐаЁ©­л© ­®¬Ґа OEM

- $'

A9 AD EB A9 20 AD

AE AC A5 E0 20 4F

45 4D 20 2D 20 20

20 20 24

0135 0A 0D 91 A5 E0 A8 SerNumUser DB 10,13,'‘ҐаЁ©­л© ­®¬Ґа Ї®«

м§®ў вҐ«п - $'

A9 AD EB A9 20 AD

AE AC A5 E0 20 AF

AE AB EC A7 AE A2

A0 E2 A5 AB EF 20

2D 20 24

0156 20 20 20 20 24 SerNumUserStr DB ' $'

015B DATA ENDS

;//////////////////////////////////////////////

////

0000 CODE SEGMENT

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

0000 TETR\_TO\_HEX PROC NEAR

0000 24 0F and al,0fh

0002 3C 09 cmp AL,09

0004 76 02 jbe NEXT

0006 04 07 add AL,07

0008 NEXT:

0008 04 30 add AL,30h

000A C3 ret

000B TETR\_TO\_HEX ENDP

;----------------------------------------------

----

000B BYTE\_TO\_HEX PROC near

000B 51 push CX

000C 8A E0 mov AH,AL

000E E8 0000 R call TETR\_TO\_HEX

0011 86 C4 xchg AL,AH

0013 B1 04 mov CL,4

0015 D2 E8 shr AL,CL

0017 E8 0000 R call TETR\_TO\_HEX

001A 59 pop CX

001B C3 ret

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:43:41

Page 1-3

001C BYTE\_TO\_HEX ENDP

;-------------------------------

001C WRD\_TO\_HEX PROC near

001C 53 push BX

001D 8A FC mov BH,AH

001F E8 000B R call BYTE\_TO\_HEX

0022 88 25 mov [DI],AH

0024 4F dec DI

0025 88 05 mov [DI],AL

0027 4F dec DI

0028 8A C7 mov AL,BH

002A E8 000B R call BYTE\_TO\_HEX

002D 88 25 mov [DI],AH

002F 4F dec DI

0030 88 05 mov [DI],AL

0032 5B pop BX

0033 C3 ret

0034 WRD\_TO\_HEX ENDP

;----------------------------------------------

----

0034 BYTE\_TO\_DEC PROC near

0034 51 push CX

0035 52 push DX

0036 32 E4 xor AH,AH

0038 33 D2 xor DX,DX

003A B9 000A mov CX,10

003D loop\_bd:

003D F7 F1 div CX

003F 80 CA 30 or DL,30h

0042 88 14 mov [SI],DL

0044 4E dec SI

0045 33 D2 xor DX,DX

0047 3D 000A cmp AX,10

004A 73 F1 jae loop\_bd

004C 3C 00 cmp AL,00h

004E 74 04 je end\_l

0050 0C 30 or AL,30h

0052 88 04 mov [SI],AL

0054 end\_l:

0054 5A pop DX

0055 59 pop CX

0056 C3 ret

0057 BYTE\_TO\_DEC ENDP

;-------------------------------

0057 PRINT\_STR PROC near

0057 B4 09 mov ah,09h

0059 CD 21 int 21h

005B C3 ret

005C PRINT\_STR ENDP

;-------------------------------

005C FIND\_TYPE\_PC PROC near

005C B8 F000 mov ax, 0f000h

005F 8E C0 mov es,ax

0061 26: 8A 16 FFFE mov dl,es:[0fffeh]

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:43:41

Page 1-4

0066 80 FA FF cmp dl, 0FFh

0069 75 06 jne x1

006B BA 0000 R mov DX,offset TypePC\_FF

006E EB 6C 90 jmp wrType

0071 x1:

0071 80 FA FE cmp dl, 0feh

0074 75 06 jne x2

0076 BA 0012 R mov DX,offset TypePC\_FEFB

0079 EB 61 90 jmp wrType

007C x2:

007C 80 FA FB cmp dl, 0fbh

007F 75 06 jne x3

0081 BA 0012 R mov DX,offset TypePC\_FEFB

0084 EB 56 90 jmp wrType

0087 x3:

0087 80 FA FC cmp dl, 0fch

008A 75 06 jne x4

008C BA 0027 R mov DX,offset TypePC\_AT

008F EB 4B 90 jmp wrType

0092 x4:

0092 80 FA FA cmp dl, 0fah

0095 75 06 jne x5

0097 BA 0039 R mov DX,offset TypePC\_FA

009A EB 40 90 jmp wrType

009D x5:

009D 80 FA FC cmp dl, 0fch

00A0 75 06 jne x6

00A2 BA 0056 R mov DX,offset TypePC\_FC

00A5 EB 35 90 jmp wrType

00A8 x6:

00A8 80 FA F8 cmp dl, 0f8h

00AB 75 06 jne x7

00AD BA 007A R mov DX,offset TypePC\_F8

00B0 EB 2A 90 jmp wrType

00B3 x7:

00B3 80 FA FD cmp dl, 0fdh

00B6 75 06 jne x8

00B8 BA 0097 R mov DX,offset TypePC\_FD

00BB EB 1F 90 jmp wrType

00BE x8:

00BE 80 FA F9 cmp dl, 0f9h

00C1 75 06 jne x9

00C3 BA 00AB R mov DX,offset TypePC\_F9

00C6 EB 14 90 jmp wrType

00C9 x9:

00C9 BA 00CE R mov DX,offset ErrorType

00CC E8 0057 R call PRINT\_STR

00CF 8A C2 mov al, dl

00D1 B4 00 mov ah, 0

00D3 BF 00CC R mov di,offset ErTypePC+3

00D6 E8 001C R call WRD\_TO\_HEX

00D9 BA 00C9 R mov DX,offset ErTypePC

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:43:41

Page 1-5

00DC wrType:

00DC E8 0057 R call PRINT\_STR

00DF C3 ret

00E0 FIND\_TYPE\_PC ENDP

;-------------------------------

00E0 FIND\_VERSION\_MS\_DOS PROC near

00E0 B4 30 mov ah,30h

00E2 CD 21 int 21h

00E4 3C 00 cmp al,0

00E6 BE 0115 R mov si,offset VersionSyst+25

00E9 B2 02 mov dl,02h

00EB 88 14 mov [si],dl

00ED 74 03 je numBasVer0

00EF E8 0034 R call BYTE\_TO\_DEC

00F2 numBasVer0:

00F2 BE 0118 R mov si,offset VersionSyst+28

00F5 8A C4 mov al,ah

00F7 E8 0034 R call BYTE\_TO\_DEC

00FA BA 00FC R mov DX,offset VersionSyst

00FD E8 0057 R call PRINT\_STR

0100 BE 0133 R mov si,offset SerNumOEM+25

0103 8A C7 mov al,bh

0105 E8 0034 R call BYTE\_TO\_DEC

0108 BA 011A R mov dx,offset SerNumOEM

010B E8 0057 R call PRINT\_STR

010E BA 0135 R mov dx,offset SerNumUser

0111 E8 0057 R call PRINT\_STR

0114 8A C3 mov al,bl

0116 E8 000B R call BYTE\_TO\_HEX

0119 8B D8 mov bx,ax

011B 8A D3 mov dl,bl

011D B4 02 mov ah,02h

011F CD 21 int 21h

0121 8A D7 mov dl,bh

0123 CD 21 int 21h

0125 BF 0159 R mov di,offset SerNumUserStr+3

0128 8B C1 mov ax,cx

012A E8 001C R call WRD\_TO\_HEX

012D BA 0156 R mov dx,offset SerNumUserStr

0130 E8 0057 R call PRINT\_STR

0133 C3 ret

0134 FIND\_VERSION\_MS\_DOS ENDP

;-------------------------------

0134 Main PROC near

0134 1E push DS

0135 2B C0 sub AX, AX

0137 50 push AX

0138 B8 ---- R mov AX, DATA

013B 8E D8 mov DS,AX

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:43:41

Page 1-6

013D E8 005C R call FIND\_TYPE\_PC

0140 E8 00E0 R call FIND\_VERSION\_MS\_DOS

0143 32 C0 xor AL,AL

0145 B4 4C mov AH,4Ch

0147 CD 21 int 21H

0149 Main ENDP

0149 CODE ENDS

END Main

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:43:41

Symbols-1

Segments and Groups:

N a m e Length Align Combine Class

ASTACK . . . . . . . . . . . . . 0014 PARA STACK

CODE . . . . . . . . . . . . . . 0149 PARA NONE

DATA . . . . . . . . . . . . . . 015B PARA NONE

Symbols:

N a m e Type Value Attr

BYTE\_TO\_DEC . . . . . . . . . . N PROC 0034 CODE Length = 0023

BYTE\_TO\_HEX . . . . . . . . . . N PROC 000B CODE Length = 0011

END\_L . . . . . . . . . . . . . L NEAR 0054 CODE

ERRORTYPE . . . . . . . . . . . L BYTE 00CE DATA

ERTYPEPC . . . . . . . . . . . . L BYTE 00C9 DATA

FIND\_TYPE\_PC . . . . . . . . . . N PROC 005C CODE Length = 0084

FIND\_VERSION\_MS\_DOS . . . . . . N PROC 00E0 CODE Length = 0054

LOOP\_BD . . . . . . . . . . . . L NEAR 003D CODE

MAIN . . . . . . . . . . . . . . N PROC 0134 CODE Length = 0015

NEXT . . . . . . . . . . . . . . L NEAR 0008 CODE

NUMBASVER0 . . . . . . . . . . . L NEAR 00F2 CODE

PRINT\_STR . . . . . . . . . . . N PROC 0057 CODE Length = 0005

SERNUMOEM . . . . . . . . . . . L BYTE 011A DATA

SERNUMUSER . . . . . . . . . . . L BYTE 0135 DATA

SERNUMUSERSTR . . . . . . . . . L BYTE 0156 DATA

TETR\_TO\_HEX . . . . . . . . . . N PROC 0000 CODE Length = 000B

TYPEPC\_AT . . . . . . . . . . . L BYTE 0027 DATA

TYPEPC\_F8 . . . . . . . . . . . L BYTE 007A DATA

TYPEPC\_F9 . . . . . . . . . . . L BYTE 00AB DATA

TYPEPC\_FA . . . . . . . . . . . L BYTE 0039 DATA

TYPEPC\_FC . . . . . . . . . . . L BYTE 0056 DATA

TYPEPC\_FD . . . . . . . . . . . L BYTE 0097 DATA

TYPEPC\_FEFB . . . . . . . . . . L BYTE 0012 DATA

TYPEPC\_FF . . . . . . . . . . . L BYTE 0000 DATA

VERSIONSYST . . . . . . . . . . L BYTE 00FC DATA

WRD\_TO\_HEX . . . . . . . . . . . N PROC 001C CODE Length = 0018

WRTYPE . . . . . . . . . . . . . L NEAR 00DC CODE

X1 . . . . . . . . . . . . . . . L NEAR 0071 CODE

X2 . . . . . . . . . . . . . . . L NEAR 007C CODE

X3 . . . . . . . . . . . . . . . L NEAR 0087 CODE

X4 . . . . . . . . . . . . . . . L NEAR 0092 CODE

Microsoft (R) Macro Assembler Version 5.10 2/26/18 20:43:41

Symbols-2

X5 . . . . . . . . . . . . . . . L NEAR 009D CODE

X6 . . . . . . . . . . . . . . . L NEAR 00A8 CODE

X7 . . . . . . . . . . . . . . . L NEAR 00B3 CODE

X8 . . . . . . . . . . . . . . . L NEAR 00BE CODE

X9 . . . . . . . . . . . . . . . L NEAR 00C9 CODE

@CPU . . . . . . . . . . . . . . TEXT 0101h

@FILENAME . . . . . . . . . . . TEXT l1exe

@VERSION . . . . . . . . . . . . TEXT 510

213 Source Lines

213 Total Lines

44 Symbols

47902 + 451166 Bytes symbol space free

0 Warning Errors

0 Severe Errors

**Map-файл L1exe.exe**

Start Stop Length Name Class

00000H 00013H 00014H ASTACK

00020H 0017AH 0015BH DATA

00180H 002C8H 00149H CODE

Program entry point at 0018:0134