QV068–GZ

# Rainbow™

# MS™-DOS V2.05 Technical Documentation

digital equipment corporation

**First Printing, November 1984**

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

CP/M and CP/M–86 are registered trademarks of Digital Research Inc.
CP/M–80 is a trademark of Digital Research Inc.

MS is a trademark of Microsoft Corporation.

Z80 is a registered trademark of Zilog, Inc.

8088 is a registered trademark of Intel Corporation.

The following are trademarks of Digital Equipment Corporation:

digital™

DEC
DECmate
DECsystem–10
DECSYSTEM–20
DECUS
DECwriter
DIBOL
MASSBUS
PDP
P/OS
Professional
Rainbow
RSTS
RSX
UNIBUS
VAX
VMS
VT
Work Processor

Printed in U.S.A.

# Rainbow™ MS™–DOS V2.05 Technical Documentation

## Recommended Documents

Contains a list of additional reading materials.

## Rainbow MS™–DOS V2.11 Update Notes

These update notes contain information about the MS–DOS Version 2.11 operating system. Use them in conjunction with the *Rainbow MS–DOS V2.05 Programmer's Guide* if you are using the MS–DOS Version 2.11 operating system.

## Rainbow MS™–DOS V2.05 Programmer's Guide

This programmer's guide describes the Rainbow computer hardware and firmware. It also describes the MS–DOS Version 2.05 operating system, and differences between Version 2.01 and Version 2.05. The guide describes the BIOS, and the additional functions under the MS–DOS Version 2.05 operating system.

The appendix lists the Rainbow specifications for on-disk structures, plus additional information about the MS–DOS Version 2.05 operating system functions.

## Rainbow MS™–DOS V2.05 BIOS Listings

These listings contain the MS–DOS V2.05 Basic Input/Output System (BIOS).

## Microsoft MS™–DOS Operating System Programmer's Reference Manual

This manual (for system programmers) contains descriptions and examples of MS–DOS system calls and interrupts. It covers installation information for device drivers. There is also technical information about disk allocation, control blocks, work areas, EXE file structure and loading.

## Microsoft MS™–DOS Operating System Macro Assembler Manual

This manual describes Microsoft's utility programs used for developing assembly language programs. They include the:

- Macro Assembler
- LINK Linker
- LIB Library Manager
- CREF Cross Reference
- DEBUG

## Rainbow Guidelines for Producing Translatable Products

This document explains how to design and build software and write text so that they can be easily translated into other languages.

# Rainbow™ Recommended Documents

## Other Technical Documentation Kits

1. Rainbow CP/M–86/80 V2.0 Technical Documentation (QV067–GZ)

2. Rainbow 100+/100B Technical Documentation (QV069–GZ)

## Additional Documents

1. Letterprinter 100 User Documentation Package (EK–LP100–UG–001)

   Includes:

   Letterprinter 100 Operator Guide
   LA100-Series Programmer Reference Manual
   Letterprinter 100 Installation Guide
   Letterprinter 100 Operator and Programmer Reference Card

2. Letterwriter 100 User Documentation Package (EK–LW100–UG–001)

   Includes:

   Letterwriter 100 Operator Guide
   Letterwriter 100 Installation Guide
   LA100-Series Programmer Reference Manual

3. Installing and Using the LQP02 Printer (AA–L662B–TK)

4. Installing and Using the LA50 Printer (EK–0LA50–UG–001)

   Includes:

   LA50 Printer Programmer Reference Manual

5. Rainbow 100 Extended Communications Option Programmer's Reference Guide (AA–V172A–TV)

6. PC100 Rainbow 100B System Unit IPB (EK–SB100–IP)

   Includes:

   EK–ORX50–IP
   EK–LK201–IP
   EK–VR201–IP

7. VT102 Video Terminal User Guide (EK–VT102–UG–003)

8. CP/M Operating System Manual (AA–X637A–TV)

## HOW TO ORDER ADDITIONAL DOCUMENTATION

**If you want to order additional documentation by phone:**

| **And you live in:** | **Call:** | **Between the hours of:** |
|---|---|---|
| New Hampshire, Alaska or Hawaii | 603-884-6660 | 8:30 AM and 6:00 PM Eastern Time |
| Continental USA or Puerto Rico | 1-800-258-1710 | 8:30 AM and 6:00PM Eastern Time |
| Canada (Ottawa-Hull) | 613-234-7726 | 8:00 AM and 5:00 PM Eastern Time |
| Canada (British Columbia) | 1-800-267-6146 | 8:00 AM and 5:00 PM Eastern Time |
| Canada (all other) | 112-800-267-6146 | 8:00 AM and 5:00 PM Eastern Time |

**If you want to order additional documentation by direct mail:**

**And you live in:** **Write to:**

USA or Puerto Rico

DIGITAL EQUIPMENT CORPORATION
ATTN: Peripherals and Supplies Group
P.O. Box CS2008
Nashua, NH 03061

NOTE: Prepaid orders from Puerto Rico must be placed with the local DIGITAL subsidiary (Phone 809-754-7575)

Canada

DIGITAL EQUIPMENT OF CANADA LTD.
940 Belfast Road
Ottawa, Ontario K1G 4C2
Attn: P&SG Business Manager

Other than USA, Puerto Rico or Canada

DIGITAL EQUIPMENT CORPORATION
Peripherals and Supplies Group
P&SG Business Manager
c/o Digital's local subsidiary or approved distributor

## TO ORDER MANUALS WITH EK PART NUMBERS WRITE OR CALL

P&CS PUBLICATIONS
Circulation Services
10 Forbes Road
NR03/W3
Northboro, Massachusetts 01532
(617)351-4325

Rainbow™

# MS™–DOS Version 2.11

## Update Notes

November 1984

This update note contains some additional information for those users of the MS–DOS Version 2.11 operating system.

If you are currently using the MS–DOS Version 2.05 operating system disregard the information contained in this update note. However, we recommend you save this update note in case you ever purchase the MS–DOS Version 2.11 operating system.

## The PRINT Command

The first time you use the PRINT command after starting the MS-DOS operating system, it prompts you for the name of the list device. Because the default is "PRN:", (which is the normal printer port), you just need to press the "Return" key to continue.

## The FORMAT Utility

If you use the FORMAT utility to format an MS-DOS disk and specify a volume identification, you can use the SYS command to create a new MS-DOS system volume.

## Directory Paths

The MS-DOS Version 2.11 operating system supports a feature known as "directory paths", which allows you to create and maintain private areas or "sub-directories" on the same volume. The following are some more "hints" on how to use sub-directories and paths:

1. If you specify a directory path in a command line other than COPY, TYPE or DIR, it is ignored, because COMMAND.COM does not process directory paths. COPY, DIR and TYPE process paths themselves.

2. If you mistakenly type a "\" in front of an otherwise legitimate command (for example, \bin\masm), the MS-DOS operating system returns immediately to the prompt without any message being displayed.

## Using the EDLIN Editor

When using the EDLIN editor, use the "Interrupt" key to generate an "escape" character when required instead of the Escape key (F11 (ESC)).

## Using the CTTY Command

If you use the "CTTY" command to change the console from device "CON:" to device "AUX:", for example, explicit references to device "CON:" in subsequent commands continue to be honored. For example, the command "COPY CON:TEXT.TXT" takes input from device "CON:" rather than device "AUX:", as expected.

Using the RECOVER Utility Program

The MS-DOS Version 2.11 operating system includes a utility program called "RECOVER" for recovering "bad spots" on your diskettes. It is described in the Chapter 5 of the Rainbow MS-DOS Version 2.11 Advanced User's Guide contained in your operating system kit. DIGITAL recommends, however, that you use this command only as a "last resort" to recover files from a bad diskette. This is because RECOVER could misinterpret data that may NOT be corrupted, which could lead to unpredictable results.

Using the PROMPT Command

When you use the PROMPT command with the $P option, be sure that you specify an existing drive. Using this option with a non-existent drive requires that you reset the system.

Interrupt Vectors

The MS-DOS Version 2.11 operating system allows you to get or set all the MS-DOS interrupt vectors, 20H through 27H, using DOS function request 25H and 35H.

Serial I/O Functions

The MS-DOS Version 2.11 operating system implements:

1. The serial I/O function 14 "Set/Clear Modem Signals."

2. The serial I/O function 21 "Program Device Interrupt."

Using Ports

The MS-DOS Version 2.11 operating system supports 7M for the communications port, printer port, and the extended communications port.

Buffer Overflows

When you use the communication IOCTL functions and the serial receive buffer overflows, the last character is SUB (1Ah). The MS-DOS Version 2.11 operating system uses a bit to flag whether or not a real SUB character has been received or an actual overflow has occurred. The most significant bit of the character status byte (CHAR_STAT) is set when the serial receive buffer overflows.

## Device Number

With the MS-DOS Version 2.11 operating system, the device number for the communication IOCTL functions 0, 3, 17, 19, 21, and 23 is placed into the communication control blocks or the interrupt service routine description.

## IOCTL Communication Driver

The MS-DOS Version 2.11 operating system contains a new IOCTL communication driver, function 26. Function 26 returns a double word pointer to a direct high performance entrance into the communications drivers. It bypasses the normal front end of MS-DOS. This direct entrance avoids the overhead of MS-DOS and the lack of MS-DOS re-entry.

NOTE

> This function is not available in previous versions of the MS-DOS operating system. Therefore, any application programs using function 26 will not work under previous versions of DIGITAL's MS-DOS operating system.

Use function 26 only in situations that require faster re-entry access to the communications drivers. To use function 26:

```
ENTRY
AX     = 4402H
BX     = File handle
DS:DX  = Packet address
         FUNCTION in packet = 26
         (no other entries used)

Invoke with INT 21H

EXIT
FUNCRET = FFH
BUFFER  = Double word pointer to the
          entry of the communications driver
```

NOTE

> Call function 26 only once when initializing your application.

After function 26 gives you the entry address, you use all the other IOCTL functions by calling the drivers as follows:

```
ES:DI = Packet
DL    = Device number
      1 = Communications port
      2 = Printer port
      2 = Extended communications port
```

Invoke the function by performing a:

```
CALL DWORD PTR (buffer)
```

Where buffer = buffer of the packet returned by
               your previous function 26 call

All returns in the packet and actions performed are identical to those functions invoked through MS-DOS INT 21H.

Rainbow™

# MS™-DOS V2.05 Programmer's Guide

digital equipment corporation

**First Printing, November 1984**

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

MS is a trademark of Microsoft Corporation.

Lotus 1–2–3 is a trademark of Lotus Development Corp.

UNIX is a trademark of Bell Laboratories.

Concurrent CP/M is a trademark of Digital Research inc.

The following are trademarks of Digital Equipment Corporation:

digital™

DEC
DECmate
DECsystem–10
DECSYSTEM–20
DECUS
DECwriter
DIBOL
GIGI
MASSBUS
PDP
P/OS
Professional
Rainbow
RSTS
RSX
UNIBUS
VAX
VMS
VT
Work Processor

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

Printed in U.S.A.

# CONTENTS

CHAPTER 1 RAINBOW HARDWARE

CHAPTER 2 FIRMWARE

CHAPTER 4 MS-DOS VERSION 2.05 EXTENDED DOS FUNCTIONS

CHAPTER 5 MS-DOS PROGRAMMING NOTES

TABLES

# PREFACE

## INTENDED READERS

This guide is intended for experienced programmers who wish to write applications and programs for the Rainbow 100, 100+, and 100B computers. It provides an overview of the various documents that form the Rainbow MS-DOS Version 2.05 Technical Documentation Kit.

Readers who wish to learn how to use Microsoft's Macro Assembler should study the Microsoft MS-DOS Operating System Macro Assembler Manual contained in this kit.

### Guide Organization

Chapter 1 describes the Rainbow's hardware.

Chapter 2 discusses the Rainbow's firmware, (the features provided in its Read Only Memory (ROM)).

Chapter 3 introduces the MS-DOS operating system: its components, and how it stores and retrieves disk files. Chapter 3 also mentions some differences between MS-DOS Version 2.05 and Version 2.01.

Chapter 4 describes the MS-DOS BIOS, and the additional functions provided in the BIOS of MS-DOS Version 2.05.

Chapter 5 describes some of the differences between the Rainbow computer and the IBM PC. Use this chapter if you want to convert programs from one system to the other, or if you want to write programs that will run on either system. This chapter also includes a list of caveats and some useful programming examples.

Appendix A contains Rainbow specifications for on-disk structures.

Appendix B contains several Microsoft articles relating to MS-DOS functions.

# CHAPTER 1

# RAINBOW HARDWARE

## 1.1 INTRODUCTION

This chapter describes the Rainbow computer hardware, emphasizing those features of interest to programmers. This chapter assumes that you know the operation and characteristics of the Rainbow hardware by studying the manuals shipped with every Rainbow computer.

The Installation Guide provided with each Rainbow computer describes the system components. The guide illustrates the system unit, monitor unit and keyboard unit.

This chapter describes the hardware in general. Detailed descriptions are given for items relevant to programmers. The definitive hardware specification, however, is the system specification for the Rainbow computer you are using.

## 1.2 GENERAL DESCRIPTION

Figure 1-1 shows the Rainbow computer. The base system consists of three parts:

- System unit
- Keyboard
- Monitor

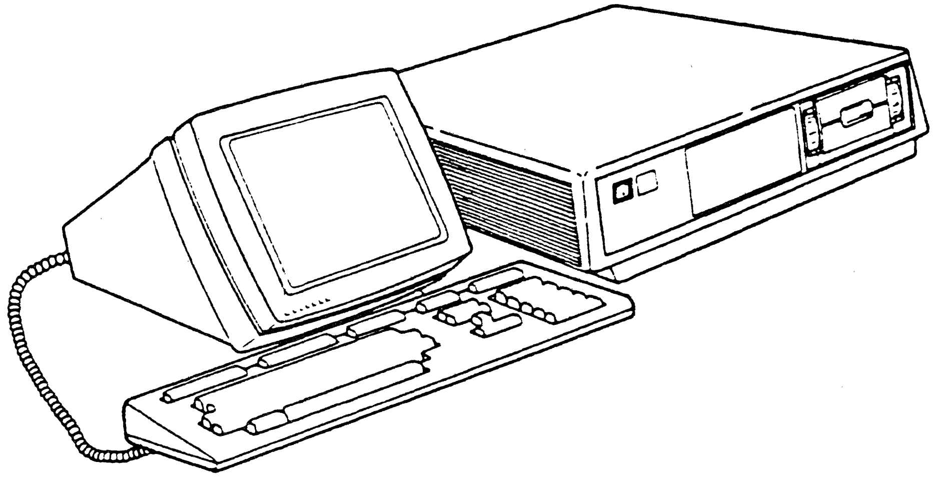


Figure 1-1: Rainbow Computer

The system unit contains the system module, a large printed circuit board where the logic circuits of the system are located. The Rainbow 100+ computer includes a Winchester hard disk drive. Figures 1-2 and 1-3 show the system block diagrams for the Rainbow 100 and Rainbow 100+/100B computers.

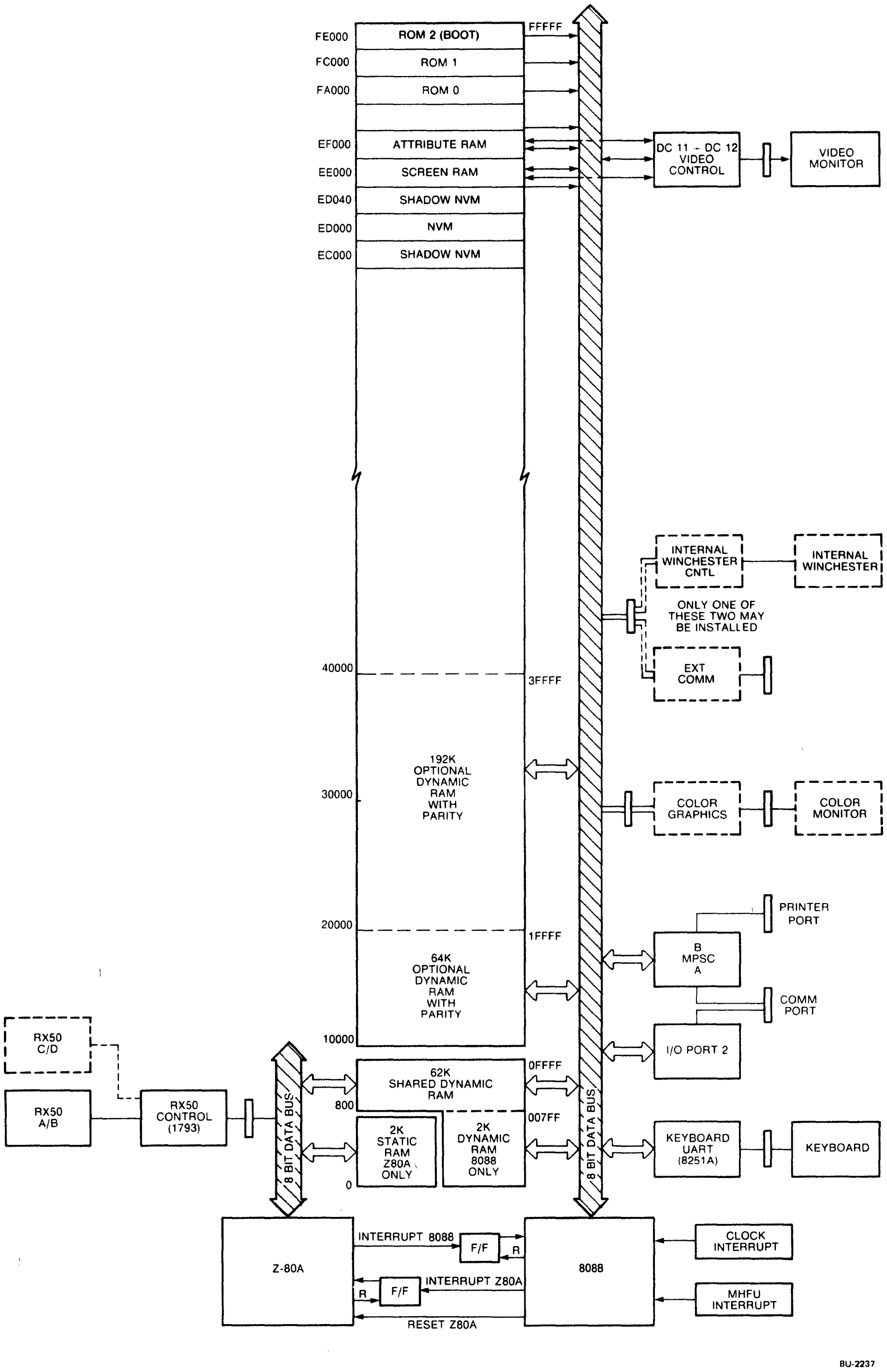


Figure 1-2: Rainbow 100 System Block Diagram

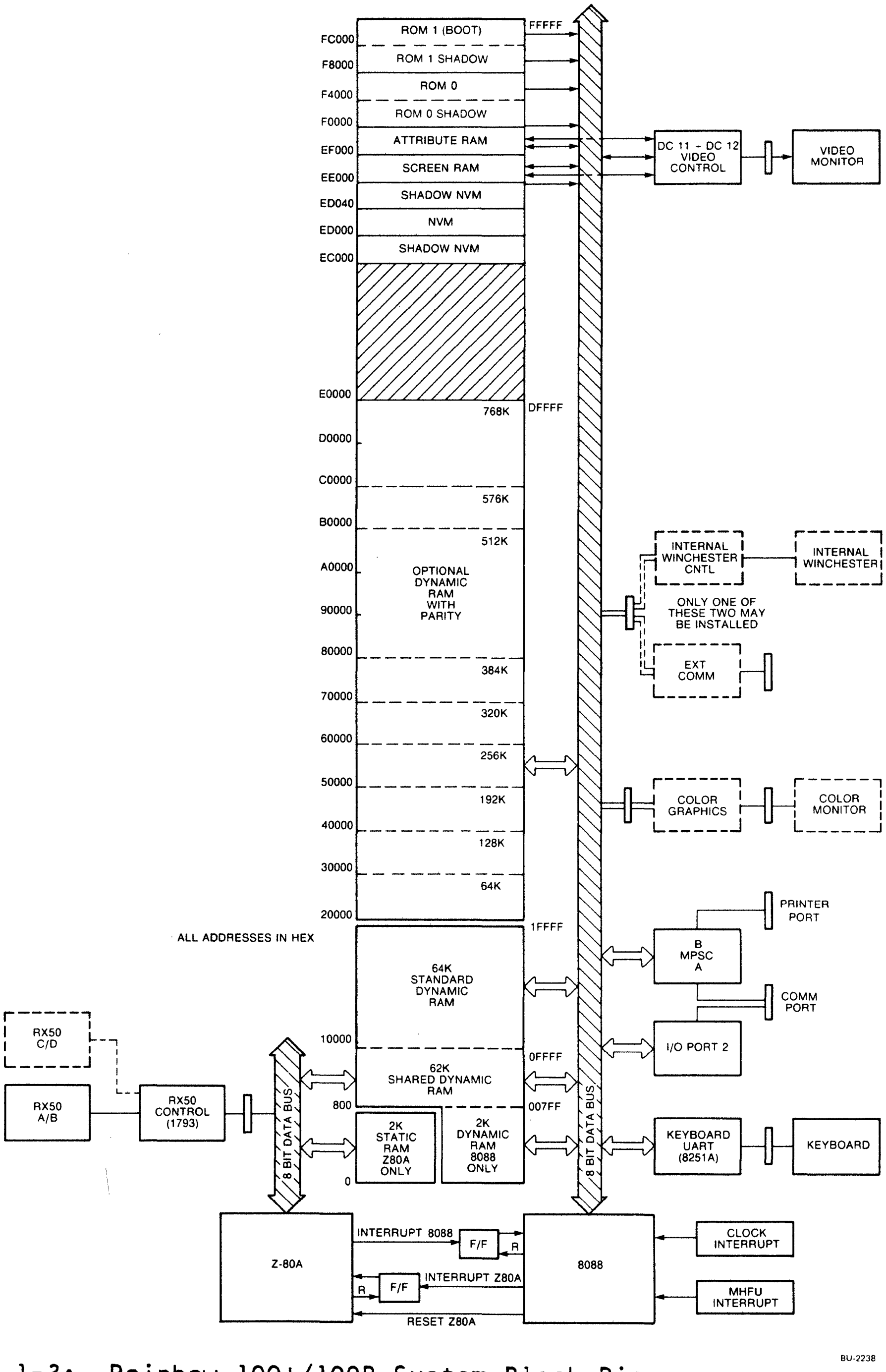


Figure 1-3: Rainbow 100+/100B System Block Diagram

## 1.3 PROCESSORS

The Rainbow computer uses dual microprocessors, an 8088 and a Z80A. These work together to provide the Rainbow computer's functionality.

System functions are divided between the 8088 and Z80A. The Z80A controls the disk drives, while the 8088 controls the video output, keyboard, communications input/output (I/O), printer I/O, and all optional devices.

Both microprocessors can access certain shared random access memory (RAM), and each can also access private memory. The 8088 clock rate is 4.815 Mhz, and the Z80A clock rate is 4.012 Mhz.

## 1.4 RANDOM ACCESS MEMORY

The Rainbow 100A contains 64K bytes of RAM. When DIGITAL first shipped the Rainbow computer you could increase this RAM by inserting an optional memory expansion card. This optional RAM card was available in two sizes: 64K bytes and 192K bytes for a system total of either 128K bytes or 256K bytes. A 64K byte card is not upgradable to a larger size.

DIGITAL has since announced an adapter card for Rainbow computers that permits it to use the expansion RAM option. This makes it possible for all Rainbow computers to add a total of 768K bytes of memory.

The basic Rainbow 100B contains 128K bytes of RAM. You can expand the memory from 192K bytes to 896K bytes in multiples of 64K bytes.

The expansion memory is contained on a printed circuit card that holds from one to three "banks" of nine memory chips each. All chip positions are socketed so that the chips can be field installed. Each bank can be filled with nine memory chips yielding either of two capacities:

- 64K bytes
- 256K bytes

As a result, the card can have any of nine possible configurations - from 64K bytes to 768K bytes. Table 1-1 shows various configurations. You can purchase the option in 128K byte or 256K byte configuration. You can also purchase expansion kits containing nine 64K-bit or nine 256K-bit RAM chips (sufficient for one bank).

Table 1-1: Rainbow 100B Optional Memory

| SIZE (K) | CHIP SIZES IN | | |
|---|---|---|---|
| | BANK 1 | BANK 2 | BANK 3 |
| 128 (Base Configuration) | 64K | 64K | - |
| 192 | 64K | 64K | 64K |
| 256 (Base Configuration) | 256K | - | - |
| 320 | 256K | 64K | - |
| 384 | 256K | 64K | 64K |
| 512 | 256K | 256K | - |
| 576 | 256K | 256K | 64K |
| 768 | 256K | 256K | 256K |

The optional expansion memory contains parity checking circuitry. A non-maskable interrupt is generated if a parity error occurs. Its handler (which is in the firmware) normally causes the system to display an error message, then halt. Should an application program perform some other form of error trapping, it must "take over" the interrupt and provide its own handler.

All of the memory except for the Z80A's 2K-Private RAM (see next paragraph) is of the dynamic type. Such memory must be "refreshed" every few milliseconds to insure that its data is not lost. Refreshing is done by special circuits that use some of the available memory access cycles. The 8088, Z80A, direct memory access (DMA) and refresh circuitry all access this memory independently of one another. When one device attempts to access memory while another is in the process of doing so, the attempting device must wait until the memory is available. This is known as "contention." Notice that such contention usually makes it impossible to program precise timing loops.

The Z80A can address a total of only 64K bytes, and therefore cannot access memory locations at addresses greater than OFFFFH.

Both microprocessors can address the 62K bytes of RAM in their address space of 00800H through OFFFFH. This shared RAM allows the two microprocessors to pass data to one another, a necessary requirement of the Rainbow computer's architecture.

Both the 8088 and Z80A have 2K bytes of private RAM in the address space of 00000 to 007FFH. The 8088 private RAM is part of the 64K dynamic RAM, of which the top 62K is shared. The Z80A private RAM is of the static type. As such, it neither needs refreshing nor experiences contention for access cycles with the 8088 or other devices. This memory should be used for time-critical Z80A routines, such as programmed diskette reading and writing.

The extended communications option contains Direct Memory Access capabilities. Such DMA operations can only be performed to and from the dynamic RAM in Address space 00000 through 0FFFFH.

## 1.5 NON-VOLATILE MEMORY (NVM)

The Rainbow computer has 256 nibbles of non-volatile memory, which retains stored values even when you turn the power off. The system uses this memory to store its Set-Up values. It is located at address ED000H through ED0FFH, but should not be directly accessed by application programs.

The Rainbow 100B firmware provides a slightly different set of NVM parameters than the Rainbow 100's NVM parameters. Also, the Rainbow 100B's Set-Up automatically determines and displays the size of installed RAM, whereas the Rainbow 100's Set-Up simply stores whatever value you enter there, whether it accurately reflects the size of installed RAM or not.

Although the NVM is addressed as bytes of data, only the low four bits of the data have meaning. This is due to the 256 x 4 organization of the NVM chip. The four most significant bits of each NVM data byte are ignored when written, and unpredictable when read.

## 1.6 VIDEO DISPLAY MEMORY

The video logic can use up to 4K bytes of screen RAM and 4K bytes of attribute RAM. Data to be displayed is placed in this RAM by firmware routines. This memory should not be directly accessed by application programs. The firmware provides functions for modifying the display memory. Portions of the Screen and Attribute RAM are also used for firmware variables, flags, pointers, stack, buffer and so forth. Accessing these RAM locations can cause a system reset or cause the system to stop.

## 1.7 READ ONLY MEMORY

The Rainbow 100 contains 24K bytes, and the Rainbow 100B 32K bytes, of Read Only Memory (ROM). This ROM contains diagnostic, VT102 terminal emulation, and VT102 console routines. These routines are also known as the system firmware.

The diagnostics are described in the manuals you received with your computer. The VT102 terminal emulation is described in the Terminal Emulation Manual. Read the "Functional Anomalies" found in the system specification if you want to program a remote host to work with the Rainbow computer as a terminal.

## 1.8 DISKETTES

The basic Rainbow computers include a dual-disk drive that can store 400K bytes of data on each of two 5 1/4 inch diskettes. You can add a second dual-disk drive, bringing the total on-line diskette storage capacity to 1600K bytes, or 1.6 megabytes. A Rainbow computer can contain either a Winchester type hard disk or the second RX50 drive, not both. (See "Hard Disk Option".) You install the optional disk drive in the system module housing.

A separate card containing a Western Digital 1793 Disk Controller chip, and related logic circuits provides control of the diskettes. This diskette controller card is part of the basic system.

You must format diskettes before can write on them. DIGITAL-type RX50K diskettes have been preformatted for ten 512-byte sectors per track and can be used immediately. Non-DIGITAL diskettes (and those that have lost format data) must be formatted for the Rainbow computer before you can use them.

The MS-DOS operating system also requires that diskettes be "soft" formatted for its use. This is a different operation than the physical formatting required by the hardware. The MS-DOS Version 2.05 operating system contains a special format utility (FORMAT) that you can use to physically and "soft" format your diskettes.

The 1793 Disk Controller chip has been hard-wired to read and write only in double-density mode. The RX50 accesses 80 tracks spaced at 96 tracks per inch (96 TPI). The RX50 can also read 48 TPI diskettes with proper software, as can diskettes having different numbers of sectors (for example, IBM 8 and 9 sector diskettes).

## 1.9 KEYBOARD

The keyboard communicates with the system module through an 8251A Universal Asynchronous Receiver Transmitter (UART) by means of a cable. The cable attaches to the monitor module, using a modular telephone connector. The keyboard signals then pass through the monitor's cable to the system unit.

Pressing a key causes a single uniquely coded byte to be serially transmitted to the system unit where it generates an interrupt 26H. The normal interrupt handler is contained in the firmware. It places the incoming keycode into a buffer, from which it can be obtained by the application software. This routine is complex and is used by both the Terminal mode and Console mode firmware. Applications should NOT attempt to do keyboard processing, but should use the functions provided in the firmware for accessing keyboard characters.

## 1.10 INTERRUPTS

Both the 8088 and Z80A microprocessors have interrupt capabilities. Hardware signals or software instructions can generate 8088 interrupts. You can program each processor to interrupt the other.

The Rainbow computer's operating system always treats one of the processors as a "master" and the other as a "slave." When the master requires the slave to perform a function, it notifies the slave by generating an interrupt to it. The original slave then becomes a master and the original master becomes a slave.

The Rainbow 100B hardware interrupts can be assigned two sets of interrupt vectors under software control. The Rainbow 100B's firmware provides routines to move the interrupt vectors from one location to another. This is necessary because the Rainbow computer's hardware and MS-DOS operating system both use some of the same interrupts. The firmware relocation function is intended only for use by the MS-DOS operating system. If you try to change vectors from within an application you cannot use MS-DOS. If you access the hardware directly instead of by the firmware the keyboard-related access capabilities are destroyed.

The system module generates a vertical frequency interrupt either 50 or 60 times per second, determined by a Set-Up parameter. The firmware video display refresh routines use this interrupt. These refresh routines include a software interrupt 44H call that the operating systems use. They, in turn, generate an interrupt 64H, as a source of real-time signals for application use.

The MS-DOS operating system moves many interrupt vectors to alternate locations upon being loaded. Applications must always use the switched values. Tables 1-2 and 1-3 show the hardware-generated interrupts and addresses.

Table 1-2: 8088 Interrupt Types

Hardware Generated

| Priority | Interrupt Source | Type No. (Hex) | Address (Hex) |
|---|---|---|---|
| | | Norm/Relocated | Norm/Relocated |
| Highest 2 | Parity error | 02 | 08 |
| 2 | Vert Freq Interrupt | 20/A0 | 80/280 |
| 6 | 7201 (Ext Comm Int 1) | 21/A1 | 84/284 |
| 2 | Graphics Option | 22/A2 | 88/288 |
| 2,6 | DMA (Ext Comm Opt) | 23/A3 | 8C/28C |
| 3 | 7201 (Comm/Printer) | 24/A4 | 90/290 |
| 2,6 | 7201 (Ext Comm Int 0) (or Winchester controlled) | 25/A5 | 94/294 |
| 2 | 8251A (Keyboard) | 26/A6 | 98/298 |
| Lowest | Z80A | 27/A7 | 9C/29C |

Software Generated

| | | | |
|---|---|---|---|
| 5 | Firmware Functions | 18 | 60 |
| 2 | O/S Clock Tic | 2C/AC | B0/2B0 |
| 4 | MS-DOS | 20 | 80 |
| 4 | MS-DOS System Functions | 21 | 84 |
| 4 | MS-DOS | 22 | 88 |
| 4 | MS-DOS | 23 | 8C |
| 4 | MS-DOS | 24 | 90 |
| 4 | MS-DOS | 25 | 94 |
| 4 | MS-DOS | 26 | 98 |
| 4 | MS-DOS | 27 | 9C |
| | Appl. Clock Tic | 64 | 190 |

NOTES

1. Relocated interrupt codes and addresses are only applicable to the Rainbow 100+ and 100B computers.
2. Initialized by INT 18H Function OCH.
3. Reset/disabled by INT 18H Function 16H.
4. MS-DOS reserves all interrupts 20H through 3FH for its own use.
5. MS-DOS remaps Int 28H (normal Firmware entry) to 18H for a applications use during its startup operation.
6. There are signal path names to the Extended Communications Option connector. The Extended Communications Option only uses 23H and 25H, but other devices can use all three types.

Table 1-3: Z80A Interrupt Type

| | | | |
|---|---|---|---|
| Only | 8088 | RST 6 | 30H |

## 1.11 VIDEO MONITOR

The video control logic controls the data displayed on the video monitor. The DC011 and DC012 DIGITAL chips are used for the control logic. The system specification explains the operation of the video control logic.

## 1.12 I/O CONTROL PORTS

The 8088 and Z80A processors each send and receive data and control information to I/O devices through ports. Special In and Out instructions of the microprocessor access these ports (Addresses). Figures 1-4 and 1-5 show the addresses (port numbers) at which the several I/O devices are accessed. These figures also show bit use. Most of the ports are latches; that is, they hold the data placed in them until it is changed. Input and output ports are independent of one another. Output ports can only be written into, while input ports can only be read.

NOTE

> Data sent to an output port is generally NOT readable by an Input instruction to the same port address.

## Z80 PORTS

| | INPUT (READ) | | | | | | | | HEX PORT ADDRESS: * | OUTPUT (WRITE) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CLEAR INT. OF Z80 BY 8088<br>(ANY VALUE) | | | | | | | | 0X | INTERRUPT 8088<br>(ANY VALUE) | | | | | | | |
| | FLOPPY DIAGNOSTIC READ REG. | | | | | | | | | FLOPPY DIAGNOSTIC WRITE REG | | | | | | | |
| 21 | $\overline{\text{STEP}}$<br>(1793) | $\overline{\text{WG}}$<br>(1793) | $\overline{\text{TRK0}}$ | $\overline{\text{DIRC}}$ | $\overline{\text{RDY}}$<br>(RX50) | $\overline{\text{INT 88}}$ | $\overline{\text{INT Z80}}$ | $\overline{\text{ZFLIP}}$ | (20H)<br>21H | $\overline{\text{ZFLIP}}$ | $\overline{\text{LED}}$ | $\overline{\text{LED}}$ | $\overline{\text{LED}}$ | ← NOT USED → | | | |
| | | | | | | | | | | 0 = RESET = OUT 20 = INVERTED = Z80 RESET<br>1 = SET = OUT 21 = NORMAL | | | | | | | |
| | GENERAL FLOPPY STATUS REG | | | | | | | | | GENERAL FLOPPY COMMAND REG. | | | | | | | |
| | DRQ | IRQ | SIDE SELECT (0) | $\overline{\text{MOTOR}}$ ON C/D | $\overline{\text{MOTOR}}$ ON A/B | $\overline{\text{TG}}$ 43 | DRIVE SELECTED 00-11 | | 40H | PRE COMP 1 | PRE COMP 0 | SIDE 0 | MOTOR ON C/D | MOTOR ON A/B | FORCE RDY | DRIVE SELECT 00-11 | |
| | 1793 STATUS REG | | | | | | | | | 1793 COMMAND REG | | | | | | | |
| | SEE 1793 DATA SHEET | | | | | | | | 60H | SEE 1793 DATA SHEET | | | | | | | |
| | 1793 TRACK REG. | | | | | | | | | 1793 TRACK REG | | | | | | | |
| | CURRENT TRACK # | | | | | | | | 61H | (NOT NORMALLY USED, BUT IS<br>USED TO FOOL 1793 FOR VT-180 DISKS) | | | | | | | |
| | 1793 SECTOR REG | | | | | | | | | 1793 SECTOR REG. | | | | | | | |
| | LAST SECTOR LOADED,<br>TRACK # FROM READ HEADER OP. | | | | | | | | 62H | SECTOR FOR NEXT R/W | | | | | | | |
| | 1793 DATA REGISTER | | | | | | | | | 1793 DATA REG. | | | | | | | |
| | DATA IN | | | | | | | | 63H | DATA OUT,<br>TRACK # FOR SEEK | | | | | | | |

* EACH PORT ADDRESS SHOWN IS ALSO ACTIVATED BY THREE SHADOW ADDRESSES; E.G. 0X, 1X, 8X & 9X ALL INTERRUPT THE 8088.

BU-2239

Figure 1-4: Z80 Ports

8088 PORTS

INPUT

| # (HEX) | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| 00 | | CLEAR 8088 INTERRUPT (ANY VALUE) | | | | | | | |
| 02 | GEN. COMM. STATUS REG | $\overline{\mathrm{INT}}$<br>$\overline{\mathrm{Z80}}$ | $\overline{\mathrm{INT}}$<br>$\overline{\mathrm{8088}}$ | $\overline{\mathrm{MHFU}}$<br>$\overline{\mathrm{ENBL}}$ | RLSD | CTS | DSR | SI<br>—<br>SCF | COMM RI |
| 04 | | | | | | | | | |
| 06 | | | | | | | | | |
| 0A | MAINTENANCE PORT | RECALL NVM | PROG NVM | PORT LOOP-BACK | DIAG LOOP-BACK | 0<br>(STRAPPED TO GND) | 0 | 0 | Z80 RESET<br>• |
| 0C | | | | | | | | | |
| 0E | | | | | | | | | |
| 10 | KB DATA REG | | | | | | | | |
| 11 | KB STATUS REG | DSR | SYN<br>—<br>BRK | FE | OE | PE | TX EMPTY | RX RDY | TX RDY |
| 20 - 2F | EXTENDED COMM OPTION: OPTION SELECT 1 | | | | | | | | |
| 40<br>41 | | DATA IN REG. | | | | | | | |
| 42<br>43 | STATUS REG | | | | | | | | |

OUTPUT

| # (HEX) | - BIT - | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 00 | | | INTERRUPT THE Z80 (ANY VALUE) | | | | | | | |
| 02 | | COMM AND LED REGISTER | $\overline{\mathrm{LED}}$ | $\overline{\mathrm{LED}}$ | $\overline{\mathrm{LED}}$ | $\overline{\mathrm{LED}}$ | $\overline{\mathrm{RTS}}$ | $\overline{\mathrm{DTR}}$ | $\overline{\mathrm{SRTS}}$ | SPD SEL<br>• REV |
| 04 | | DC-011 WRITE REG. | | | | | | | | |
| 06 | | COMM BIT RATE REGISTER | TRANSMIT BIT RATE (SEE TABLE) | | | | REC BIT RATE (SEE TABLE) | | | |
| 0A | | MAINTENANCE PORT | RECALL NVM | PROG NVM | COMM LOOP-BACK | DIAG LOOP-BACK | PAR TEST | GRAF. VIDEO SEC | $\overline{\mathrm{DISP.}}$<br>$\overline{\mathrm{BLNK}}$ | Z80 RESET<br>• |
| 0C | | DC-012 WRITE REGISTER | | | | | | | | |
| 0E | | PRINTER BIT RATE REGISTER | | | | | COMM PORT CLOCK: 0=INT 1=EXT | BIT RATE (SEE TABLE) | | |
| 10 | 8251A | KB DATA REG | | | | | | | | |
| 11 | 8251A | KB COMMAND REG | | | SEE | 8251A | DATA | SHEET | | |
| 40<br>41 | 7201 COMM PRINTER | | DATA OUT REG | | | | | | | |
| 42<br>43 | 7201 COMM PRINTER | COMMAND REGISTER | | | | | | | | |

50 - 5F EXTENDED GRAPHICS OPTION: OPTION SELECT

60 - 6F EXTENDED COMM OPTION: OPTION SELECT 2

BU-2240

Figure 1-5: 8088 Ports

## 1.13 COMMUNICATIONS AND PRINTER INTERFACES

The basic Rainbow computer communicates to external devices through two interfaces - the communications interface and the printer interface . These interfaces share a single Intel 8274 (or Nec 7201) Universal Synchronous Asynchronous Receiver Transmitter (USART) Multi-Protocol Serial Controller (MPSC) chip mounted on the system motherboard.

The Intel data sheet and application notes contained in the Rainbow 100+/100B Technical Documentation Kit, (order number QV069-GZ), describe the operation of the 8274 chip. Both interfaces use PB25 25-pin connectors mounted on the rear of the system module. The Communication interface connector is a male (DB-25P), while the printer connector is a female (DB-25S). The communications port provides full modem support (in conjunction with I/O port 2) for both asynchronous and bisynchronous modes. It has an RS-423 compatible interface conforming to CCITT V.21, V.22, and V.23 specifications. This port supports full- and half-duplex modems and break detection. The firmware uses the communications port to attach to a host computer when in terminal mode. The MS-DOS Version 2.05 operating system provides an extensive set of system functions for controlling the communications, printer, and optional communications ports.

The printer port also has an RS-423 interface, which is compatible with both DIGITAL and non-DIGITAL printers. Both the firmware and the MS-DOS operating system use the printer port to send asynchronous data to serial printers such as the DIGITAL LA50, LA100, or LQP02 printers. Data Terminal Ready (DTR) is supported for this port and can be used by software to control printers or other devices that do not support the XON/XOFF protocol. The Rainbow 100B's terminal mode supports both XON/XOFF and DTR protocols simultaneously. This means that the device attached to the Rainbow 100B's printer port must assert DTR, or no data is sent to the printer, and the system stops.

In console mode under the MS-DOS operating system, two protocols can be selected for the printer port. The Rainbow computer and the MS-DOS operating system default to the XON/XOFF protocol. To enable DTR, the printer port must be specifically programmed by the application or utility.

Figure 1-6 shows the signal paths to the communications and printer ports. Table 1-4 shows baud rates available for both the communications and printer ports. You initially select these ports by the Set-Up parameters, but they can also be selected as shown in Table 1-4 or by using the special serial I/O system functions of the MS-DOS operating system.

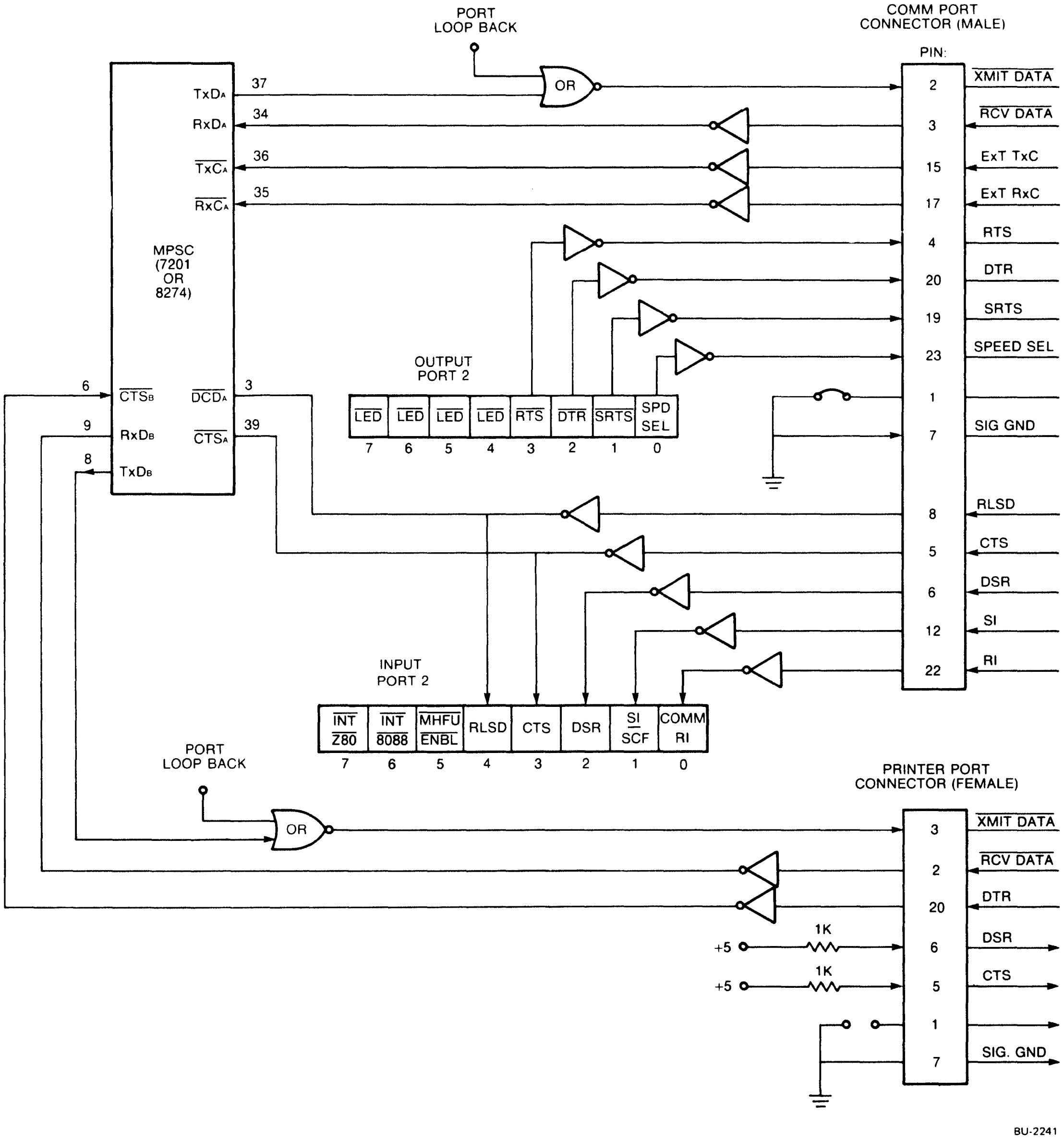


Figure 1-6: Communications and Printer Port Signal Paths

Table 1-4: Baud Rates Available

| Rate | Comm | Nibble Value | Printer | Bit 0-2 Value |
|---|---|---|---|---|
| 50 | * | 0 | | |
| 75 | * | 1 | * | 0 |
| 110 | * | 2 | | |
| 134.5 | * | 3 | | |
| 150 | * | 4 | * | 1 |
| 200 | * | 5 | | |
| 300 | * | 6 | * | 2 |
| 600 | * | 7 | * | 3 |
| 1200 | * | 8 | * | 4 |
| 1800 | * | 9 | | |
| 2000 | * | A | | |
| 2400 | * | B | * | 5 |
| 3600 | * | C | | |
| 4800 | * | D | * | 6 |
| 9600 | * | E | * | 7 |
| 19200 | * | F | | |

To set communication port baud rates, transfer a byte with two nibbles from the above table to port 06H. The high nibble sets the receive rate, and the low nibble sets the transmit rate.

To set printer baud rates, transfer a byte with bits 0-2 set to the value from the above table to port 0EH. Notice that bit 3 of this port selects the external clock for the communications port, so use care when setting printer baud rates.

## 1.14 SYSTEM RESET

You automatically clear the 8088 when you turn on the computer. You can also clear the 8088 processor at any time by pressing Ctrl/Set-Up in Set-Up mode. A special watchdog timer can also reset the system in the event it is not refreshed within a period of 100 milliseconds or the interrupts are off. The video display routines in the firmware refresh the circuit. It is important that application programs disable interrupts for no longer than 100 ms. or this circuit can reset the system.

If an application must disable interrupts for a longer period, the interrupts off circuitry can be temporarily disabled. Such a procedure can adversely affect the video display and any real-time dependent system operations.

The correct procedure for temporarily disabling the interrupts off circuitry is as follows:

1. Disable interrupts using the CLI instruction

2. Disable the interrupts off detection circuitry by transferring a 00H to 8088 I/O port 10CH.

3. Enable interrupts with the STI instruction when ready. The interrupts off detection circuit is automatically enabled.

## 1.15 AUTO-START

When you reset the Rainbow system, the system automatically tests several components. When the tests are completed, a main menu is displayed that offers a choice of disk drives from which to start the operating system. (The Rainbow 100B contains a new Set-Up feature that automatically starts the operating system without requiring you to press any keys.)

## 1.16 OPTIONAL I/O DEVICES

Three optional I/O devices are available for the Rainbow series computers:

- A color graphics option

- An extended communications option

- A hard disk option

### 1.16.1 Color Graphics Option

This option consists of a printed circuit board containing a special bit-mapped color video graphics controller chip and related logic that you mount on the motherboard.

The Rainbow computer can operate in text mode or text and graphics mode. In text mode (VT100 text mode), the graphics option video is disabled and the standard video control logic (DC011 and DC012) drives the monitor. In text and graphics mode, the option's video control logic drives the monitor. (You need a color monitor to display color graphics). Monochrome graphics can also be obtained using the VR201.

The Rainbow Computer Color/Graphics Programmers Reference Guide is included in the Rainbow 100+/100B Technical Documentation Kit, (order number QV069-GZ).

### 1.16.2 Extended Communications Option

An extended communications option provides additional communications capabilities. This option consists of a printed circuit card that mounts in two 40-pin connectors on the system module. Its communication outputs are available on D-subminiature connectors at the left rear of the system module.

The option provides the Rainbow computer with a separate high-speed communications port that can support clustering and local area networking. Asynchronous bit- and byte-synchronous capabilities are provided. This option uses a 8274 Multi-Protocol Serial Controller (MPSC) and a 8273 Direct Memory Access (DMA) controller.

The 8274 MPSC used in the extended communications option is the same as the MPSC used by the standard communications/printer ports. The BIOS of the MS-DOS Version 2.05 operating system includes a full set of routines for controlling the standard communications/printer ports and the ports of this option for asynchronous operations. The Rainbow 100 Extended Communications Option Programmer's Reference Guide, (order number AA-V172A-TV), describes the option. You cannot use this option if you have a hard disk because both options mount (and use) the same motherboard connectors.

### 1.16.3 Hard Disk Option

Two 5-1/4 inch diameter Winchester hard disks are optionally available for the Rainbow 100 and 100B computers. They differ only in the amount of storage they provide. The RD50 hard disk stores up to 5M bytes and the RD51 up to 10M bytes of data in fixed-length blocks on 5 1/4 inch (130mm) diameter rigid magnetic disks. They use what is called Winchester technology, and are contained in a sealed, non-removable enclosure. The Rainbow 100+ computer comes with a 10M byte RD51.

The controller logic is contained in a printed circuit board that occupies the extended communications option slot on the system module motherboard. It can control one ST506-interface compatible Winchester drive. (The ST506 interface is an industry standard interface.)

The MS-DOS Version 2.05 operating system contains BIOS drivers for the Winchester hard disk option, and the distribution diskette includes a backup and restore utility for use with the Winchester. Each Winchester drive comes with utilities for initializing and partitioning the disk according to your requirements. Partitioning assigns logical drive names (for example, E:, F:) to all or part of the drive, and initializes them for the operating system(s) to be used.

The ROM of the Rainbow 100+ and 100B computers contains code so you can start the system from the Winchester instead of from a diskette. An upgrade kit is available that permits a Winchester disk to be field installed in the Rainbow computer. You cannot start the operating system from the Winchester disk if you upgraded your Rainbow 100A computer to include this hard disk. This is because its ROM does not contain the necessary code. The Rainbow can configure the Winchester disk for either the CP/M-86/80 operating system, the MS-DOS operating system, or both.

The MS-DOS Version 2.05 operating system contains drivers for programming the hard disk, however, so that files and data can be written and read using the hard disk system functions. Sophisticated users can us the primitive disk operations contained in the BIOS. Chapter 3 discusses these operations.

You cannot use this option if you have an extended communications option because both options mount (and use) the same motherboard connectors.

# CHAPTER 2

# FIRMWARE

## 2.1 INTRODUCTION

This chapter describes the basic functionality of the "console" as seen by an application program. It also describes the utility functions you can use with application programs. To use this chapter, you should be familiar with:

- The Rainbow system specification for your particular Rainbow computer

- The video controller chips

## 2.2 GENERAL

The programs in ROM (firmware) include:

1. Diagnostic routines

2. Terminal and console emulation routines

3. Set-Up routines,

4. 8088 and Z80A routines for loading the operating system

The manuals that come with your Rainbow computer describe diagnostic, Set-Up and startup routines.

The Rainbow 100+/100B Terminal Emulation Manual (contained in the Rainbow 100+/100B Technical Documentation Kit, (order number QV069-GZ)), which describes terminal mode.

The Rainbow 100+ and 100B computer's firmware includes an auto-start capability. This lets you start the operating system from one of the possible disk drives without selecting the drive from the main menu.

The following list details the functions that are available in the Rainbow 100+ and 100B ROMs which differ in the Rainbow 100 ROMs:

- The BELL character does not cause a hesitation the first time it "beeps," but hesitates only for subsequent BELL characters that occur too close together.
- A new Set-Up parameter permits a choice of either Caps-Lock or Shift-Lock mode for the "Lock" key.
- Every ROM set supports all keyboard variations. At startup, the firmware asks you to select a keyboard if the stored Set-Up parameters indicate that none had previously been selected. The NVM then stores the choice.
- A new Set-Up parameter allows you to invoke a National-Replacement-Character set for country-specific use.
- The "Compose" algorithm is implemented. Details are provided in the manuals that come with your Rainbow computer.
- The printer port's DTR line is monitored when in terminal mode. If it is not asserted, printing does not occur.
- Pressing Ctrl/2 through Ctrl/8 keys generates the proper control codes.
- Ctrl/<symbol> keys are no longer shift-dependent.
- The Escape sequences for "Erase-in-line" and "Display" now accept selective parameters instead of ignoring all after the first parameter.
- All Set-Up text and all error messages are displayed in languages appropriate to the keyboard type you selected and the language-cluster supported by the installed ROM set.
- Single "Shift-2" and "Shift-3" escape sequences apply only to the next following graphic character, and are not canceled by an intervening escape sequence or other non-graphic character.
- The "Print Screen" key generates an ESC [ 1 2 ~ in console mode.
- Ctrl/xx keys auto-repeat.
- Set-Up automatically determines the amount of installed RAM and displays it.

Details of the implementation of language-clustering, national-replacement characters, and so forth, are describes in the Rainbow 100+/100B System Specification (contained in the Rainbow 100+/100B Technical Documentation Kit, (order number QV069-GZ.))

## 2.3 UTILITY FUNCTIONS

The console mode of VT102 emulation acts like a VT102 console (although without modem control or local echo). As such, it must accept characters for display and control, and it must supply characters entered at the keyboard. These, plus several other utility functions, are provided for use by programs as follows:

- Send a character to the "console" for display or control (Console-Out)
- Obtain a character from the keyboard (Level-2 Console-In)
- Determine if a keyboard key has been depressed (Console-In Status)
- Level 1 (16 bit) Console-In
- Enable and disable the cursor
- Initialize interrupt vectors
- Return the clock rate
- Set and clear LEDs on the keyboard
- Send a string of data to the screen
- Initialize the Comm/Printer 8274 chip to Set-Up parameters
- Return "Raw" keyboard data

In addition to the above, the Rainbow 100+ and 100B computers have the following functions:

- Return ROM version number
- Relocate interrupt vectors
- Ring keyboard bell
- Get/Set character set usage as per NVM

For more information, refer to the system specification for your particular Rainbow computer.

You invoke all functions by setting up the desired function number and other parameters in 8088 registers then executing a software interrupt 18H. Place the desired function code in the DI register and parameter data in other registers according to the needs of the function. The application program should save all registers it needs before issuing the INT 18H, because the firmware routines save only CS:, SS:, and DS:.

## 2.4 FUNCTION OOH CONSOLE OUT

This function sends the character in the AL register to the "console" where it is displayed. It accepts and processes ASCII and 8-bit multi-national characters as a VT102, that is, escape sequences and control characters are executed

```
ENTRY
-----
AL = Character to be displayed

EXIT
----
None
```

## 2.5 CONSOLE IN, CONSOLE IN STATUS, AND LEVEL-1 CONSOLE IN

Pressing a keyboard key generates an Interrupt 26H in the 8088 processor. The keyboard's UART passes a one-byte character code identifying the particular key to the firmware's keyboard interrupt handler. This code is analyzed to determine whether an action must be performed (such as entering Set-Up, Lock mode), or whether the key represents a character to be passed to the application.

If a character is passed to an application, the keycode is stored as the low-byte of a two byte entry in a 30-byte first-in-first-out raw key buffer. The high-byte contains three flags that are set according to whether the Shift, Ctrl, and/or Lock keys were depressed or in effect. When the character is removed from the raw key buffer, the keycode is translated to it ASCII value, or, if a function key, to a function key number. If it is a function key, a fourth function key flag is also set. Figure 2-1 illustrates this.

```
           Hi Byte           Lo Byte
Bit    7 6 5 4 3 2 1 0   7 6 5 4 3 2 1 0

    <-Not-> | | | |    <---Keycode--->
      Used  | | | |
            | | | |__Function key flag. 1=Function key
            | | |___Shift Flag.   1=Shift key in effect
            | |____Control flag.  1=Ctrl key depressed
            |_____Caps Lock Flag. 1=Caps lock in effect
```

NOTE

The function-key flag is added only when a Console-In function causes the character to be extracted from the raw key buffer.

Figure 2-1: Level-1 Character Format

Firmware functions can obtain input in either of two levels. Each 16-bit character code stored in the raw key buffer remains there until:

- A Level-1 Console-In
- A Level-1 raw key function call is made
- one of the Level-2 routines requires a character

The Level-2 Console-In routine also contains a buffer. Whenever you invoke the Level-2 Console-In function, the next character in its buffer, if any, is returned to the application. Usually, no character is present in the buffer, so the routine performs the equivalent of a Level-1 Console-In call to obtain a character from the raw key buffer.

If a character is available in the raw key buffer, it is removed. If it is an ASCII character, it is placed into the Level-2 buffer.

If the character represents a function key code, an escape sequence for the key is placed in the Level-2 buffer. That is why the Level-2 buffer is required.

DON'T MIX LEVEL-1 AND LEVEL-2 FUNCTIONS.

Notice that the Level-2 Console-In Status function also causes a character to be removed from the raw key buffer in the event the Level-2 buffer is empty and the raw key buffer is not empty. This can cause confusion, if you mix Level-1 and Level-2 Console-In and Console-In Status calls. If there were a character available in the Raw Key buffer when you performed a Level-2 Console Status function, you would be informed that a character was available. But if you then attempted to obtain that character using the Level-1 Console-In function, you would not get it, because it had been removed from the raw key buffer by the Level-2 Console Status function and transferred into the Level-2 buffer.

Programs should insure that the Level-2 buffer is empty (by repeatedly testing its status and obtaining and discarding any remaining characters) before using the Level-1 Console-In function.

The console emulation code can recognize and respond to escape sequences requesting status, cursor position, or other information. The responses are ASCII escape sequences which are placed directly into the Level-2 buffer.

## 2.6 FUNCTION 02H CONSOLE IN (LEVEL-2)

This function fetches a character from the keyboard buffer.

```
ENTRY
-----
None

EXIT
----
AL = Character from the keyboard buffer (if one was available)
CL = 00H, No character available
   = FFH, A character was returned in AL
```

NOTE

This function does NOT wait for a key to be depressed, if no character is available.

## 2.7 FUNCTION 04H CONSOLE IN STATUS (LEVEL-2)

This function determine whether a character is available in the Level-2 buffer.

```
ENTRY
-----
None

EXIT
----
CL = 00H, No character available
   = FFH, A character is available in the Level-2 buffer
```

NOTE

The character, if any, remains in the Level-2 buffer, but the function may cause the raw key buffer to become empty.

## 2.8 FUNCTION 06H CONSOLE IN (LEVEL-1)

This function fetches a 16-bit character from the Raw Key buffer.

```
ENTRY
-----
None

EXIT
----
AX = 16-bit character from Raw Key buffer
CL = 00H, if no character is available
   = 01H, if any characters remain in the Level-2 buffer from a
     previous Level-2 call, whether or not characters are in the
     Raw Key buffer
   = FFH, if a character is available (returned in AX)
```

NOTE

01H is returned to remove some of the complexity surrounding the mixing of Level-1 and Level-2 functions.

The two bytes representing the character in the raw key buffer are placed in the AX register after being analyzed and translated (see Figure 2-1 above).

For non-function keys, the function key flag is zero, and the low-byte contains the character code, either ASCII or 8-bit multinational, according to the keyboard type being used. The effect of Shift, Ctrl and Caps-Lock has already been taken into account for the character. For function keys, the Function key flag is a 1, and the low-byte contains a number that identifies which function key it is. Your Rainbow system specifications list these numbers.

## 2.9 FUNCTION 08H DISABLE CURSOR

## 2.10 FUNCTION 0AH ENABLE CURSOR

These functions make the cursor visible (Enable), and invisible (Disable). They insure that the cursor control logic is not disturbed by Function 14H. They must be used in pairs, preceding and following the use of function 14H. They are "nested" functions; it takes as many enables as disables to redisplay the cursor. If enabled when already visible, a "ghost" can appear when the cursor is moved.

## 2.11 FUNCTION 0CH INITIALIZE INTERRUPT VECTORS

Function 0CH initializes the following interrupt vectors to point to their default firmware routines:

| Type Number<br>(Hex) | Use |
|---|---|
| 02 | NMI for RAM parity error |
| 20 | Vertical frequency refresh |
| 22 | Graphics controller option |
| 23 | Ext Comm Option DMA controller |
| 25 | Ext Comm Option 8274 |
| 26 | Keyboard 8251A |
| 2C | Time Tic from Vert Refresh |

In addition, the extended communications option and the graphics option are reset to their disabled states. This function should not be used by application programs because it destroys the MS-DOS interrupt structure.

## 2.12 FUNCTION OEH RETURN CLOCK RATE

This function determines the current Set-Up clock rate.

```
ENTRY
-----
None

EXIT
----
AL = 00H, to indicate 60Hz clock
   = 01H, to indicate 50Hz clock
```

## 2.13 FUNCTION 10H SET KEYBOARD LEDS

## 2.14 FUNCTION 12H CLEAR KEYBOARD LEDS

These functions Set and Clear the keyboard light emitting diodes (LEDs). They do not cause any action that can be implied by the label of the LED being affected. Firmware normally maintains all but the "compose" LED in the proper state. (On Rainbow 100+ and 100B computers, the compose LED is also handled by the firmware.)

Register AL contains a bit pattern to set (Function 10H), or clear (Function 14H), the LEDs. A 1 in a bit indicates the LED to be set (by function 10H) or cleared (by function 12H). LEDs whose corresponding bits are zero remain unchanged. Figure 2-2 shows the keyboard LEDs:

```
     Bits       LED
7 6 5 4 3 2 1 0
|       | | | |_Wait
|       | | |___Compose
|       | |_____Lock
|       |_______Hold
|
|_______________MUST BE 1
```

Figure 2-2: Keyboard LEDs

## 2.15 FUNCTION 14H SEND DATA TO SCREEN

This function sends more than one character at a time to the screen. Entire strings, up to the length of one line, can be sent, along with their attributes, in a single function call. Your Rainbow system specification describes the meaning of characters and attributes.

```
ENTRY
-----
AX = 0000H, Characters and attributes
   = 0001H, Attributes only
   = 0002H, Characters only
BH = Column number (1 thru 80, or 1 thru 132)
BL = Line Number (1 thru 24)
```

The maximum column number is a function of the screen width (80/132) and line width (single/double). Exceeding the line length destroys the screen image.

CX = Number of characters or attributes to transfer

You are responsible for ensuring that the number of characters does not exceed the end-of-line.

DX = Offset to the start of Attributes relative to the user's DS register
SI = Offset to the start of Characters relative to the user's DS register
BP = Copy of user's DS register

Both DX and SI must be relative to the same value of DS.

## 2.16 FUNCTION 16H INITIALIZE COMMUNICATIONS/PRINTER 8274 TO SET-UP PARAMETERS

This function:

- Initializes the Communications/Printer 8274 to the Set-Up parameters

- Performs a channel reset for both channels A and B

- Sets the baud rates

- Set the number of bits per character

- Sets receive and transmit clock rates

- Enables receive and transmit for the Communication and Printer ports

All this is done according to the Set-Up parameters.

NOTE

> When data/parity is 7M or 7S, the 8274 is actually set for 8N.

## 2.17 FUNCTION 18H RAW KEYBOARD DATA

This function is provided for diagnostic purposes.

Figure 2-1 (above) indicates that the Shift, Lock and Ctrl keys can only be read in conjunction with another key by looking at the flag bits in AH. The Set-Up key is not detectable by a program, but you can look at the screen to determine if you are in Set-Up mode. The Hold Screen key is not detectable by a program, but the Hold Screen LED is lit.

An application program can determine whether or not the Rainbow is in Set-Up mode.

```
ENTRY
-----
None

EXIT
----
AX = Flag bits and keycode, as contained in the Raw Key buffer
CL = 0, no key available
   = 1, key available
```

NOTE

> The character is removed from the buffer

NOTE

> The following four functions are not available on the Rainbow 100.

## 2.18 FUNCTION 1AH RETURN ROM VERSION NUMBER

This function returns an ASCII text string, identifying the version number of the ROMs, to the user's buffer. The format is:

    MM.mm L

MM identifies the major version, mm the minor revision, both expressed in ASCII numerals. In general, minor revisions affect only one ROM, while the major version affects both ROMs. L is the language variation. mm and L are each followed by a Null (OOH) byte for delimiting purposes.

Programs or operating systems can use this function to determine the version and/or language of the ROMs. If you use this function on a Rainbow 100 computer, the buffer remains empty, identifying the unit as a Rainbow 100 computer. The Rainbow system specification contains details of version numbers and language IDs.

```
ENTRY
-----
DI = 1AH
DX = Offset Address of 8 byte buffer
BP = Segment address of 8 byte buffer

EXIT
----
None
```

## 2.19 FUNCTION 1CH RELOCATE INTERRUPT VECTORS

The Rainbow 100+ and 100B computers provide for changing interrupt codes 20H through 27H that are associated with the hardware generated interrupts to AOH through A7H under software control. When this is done it is also usually necessary to move the interrupt vectors associated with each interrupt to its new vector address. Function 1AH provides a convenient way to do this.

```
ENTRY
-----
AH = Interrupt code into whose vector the first source vector is
     relocated.  Must be either AOH, OOH (defaults to 20H), or
     20H.
AL = Interrupt code whose vector is the first of group to be
     relocated.  Must be either OOH (defaults to 20H), 20H, or
     AOH.

     (If AX = OOH, default vectors are simply initialized)
```

CX = Number of vectors to be relocated. If zero, 16 is relocated (the default).
DI = 1CH Function Number

EXIT
----
CX = 0, to indicate a successful move

NOTE

> This function is provided for use by the MS-DOS operating system and other operating systems, if required. Any use by application programs destroys the operating system's interrupt structure. The format of the calling process is general, but the only source and target addresses supported in the Version 05 ROM are 20H to/from AOH.

## 2.20 FUNCTION 1EH RING THE KEYBOARD BELL

This function rings the keyboard bell. It is a convenience for programs that use these firmware functions frequently.

## 2.21 FUNCTION 20H GET/SET DIGITAL 8/7 BIT CHARACTER CODE USAGE IN NVM

This function provides a method of reading or changing the NVM bit(s) that control character code usage. These can also be set or reset in Set-Up.

```
ENTRY
-----
AH = 0, for SET
   = 1, for GET

AL = 0, for DIGITAL-8 bit character set
   = 1, for 7-bit national replacement character codes

EXIT (GET)
----
AL = 0, for DIGITAL-8
   = 1, for 7-bit national replacement character codes
```

## 2.22 STACK CONSIDERATIONS

Both hardware and software interrupts place three words of addresses and flags onto whatever stack is currently being used, either the operating system's, your program's, or the firmware's.

In addition, the firmware's hardware interrupt handlers push their registers onto the current stack, and can be interrupted in turn by other hardware devices whose handlers push even more on the current stack. Therefore all application programs should provide a stack with at least 62 bytes more than are required by the application itself, which is the worst case possible.

Another stack-related precaution is that application handlers for hardware interrupts should NOT use either registers or the stack for passing parameters, because these might get changed by an intervening interrupt process. Nor should they use a private stack of their own. The firmware interrupt handlers test to determine whether their own stack is in place. If not, one is set up. This can cause problems. Consider the following situation. A user's hardware interrupt handler is in place for, say, the Communications port, and the handler sets up its own stack. If a character were to be received during execution of the latter part of the video refresh routine, (when interrupts are enabled), and the user's interrupt handler had set up its stack, then a keyboard interrupt was generated. The Firmware's keyboard routine would find a non-firmware stack in place, so would again set up the firmware's stack. In doing so it destroys the original video refresh routine's pushed environment, crashing the system. Software generated interrupt handlers do not have this problem since they can never interrupt a hardware handler.

## 2.23 SYSTEM PARAMETER DATA

The firmware maintains one 16-bit word of flags in Attribute RAM at location EFFFEH, which describes the system state. These flags are used primarily by the firmware, but can be used by application programs for special purposes. When this word is loaded into a register, its bits have the following meaning:

Table 2-1: System Parameter Data Flags

| Bit | Flag | Meaning when: 0 | 1 |
|---|---|---|---|
| 0 | Emulator: | Console Mode | Terminal Mode |
| 1 | On/Off Line: | On-Line | Off-Line |
| 2 | Set-Up mode: | Normal | Set-Up mode |
| 3 | Hold Screen | Normal | Hold Scrn in effect |
| 4 | Scroll I/P: | Normal | Sm Scroll in process |
| 5 | Reserved | | |
| 6 | Reserved | | |
| * 7 | Print Screen Key: | Not Pressed | Pressed |
| 8 | Comm Opt Present: | Present | Not present |
| 9 | RX50 Ctrl Bd.: | Present | Not present |
| 10 | Graphics Bd.: | Present | Not present |
| **11 | Memory Option: | Present | Not Present |
| 12 | Reserved | | |
| 13 | Reserved | | |
| 14 | Reserved | | |
| 15 | Reserved | | |

* The Print Screen Key flag bit is only in the Rainbow 100+ and 100B computers. The firmware sets this bit whenever you press the Print Screen Key, but does not reset it. If an application tests this bit more than once, it must reset the flag bit after each test.

** The memory option bit is only meaningful for Rainbow 100 computers equipped with a Rainbow 100 expansion RAM board.

## 2.24 LEVEL-1 CONSOLE-IN FUNCTION KEY CODES

The Level-1 (16-bit) Console-In function of the Rainbow computer's firmware returns a one in bit 0 of the most significant byte of the 16-bit value if the depressed key is a function key. In this case, the least significant byte is not the ASCII value of the character, but is an arbitrary code used to identify which key was depressed. The codes generated are shown in Table 2-2.

Table 2-2: Level-1 Console-In Function Key Codes

| Key | Level-1 Key Code (Hex) | Level-2 Esc Seq | Key | Level-1 Key Code (Hex) | Level-2 Esc Seq |
|---|---|---|---|---|---|
| Help | 0 | Esc [ 2 8 ~ | Down-Arrow | 29 ^ | Esc [ B |
| Do | 1 | Esc [ 2 9 ~ | Right-Arrow | 2B ^ | Esc [ C |
| Compose | 2 * | Esc [ 1 0 ~ | Left-Arrow | 2D ^ | Esc [ D |
| Print Screen | 3 + | Esc [ 1 2 ~ | Keypad 0 | 2F # | 0 |
| F4 | 5 | Esc [ 1 4 ~ | Keypad 1 | 32 # | 1 |
| F6 | 7 | Esc [ 1 7 ~ | Keypad 2 | 35 # | 2 |
| F7 | 9 | Esc [ 1 8 ~ | Keypad 3 | 38 # | 3 |
| F8 | B | Esc [ 1 9 ~ | Keypad 4 | 3B # | 4 |
| F9 | D | Esc [ 2 0 ~ | Keypad 5 | 3E # | 5 |
| F10 | F | Esc [ 2 1 ~ | Keypad 6 | 41 # | 6 |
| F14 | 11 | Esc [ 2 6 ~ | Keypad 7 | 44 # | 7 |
| F17 | 13 | Esc [ 3 1 ~ | Keypad 8 | 47 # | 8 |
| F18 | 15 | Esc [ 3 2 ~ | Keypad 9 | 4A # | 9 |
| F19 | 17 | Esc [ 3 3 ~ | Keypad Dash | 4D # | - |
| F20 | 19 | Esc [ 3 4 ~ | Keypad Comma | 50 # | , |
| Find | 1B | Esc [ 1 ~ | Keypad Period | 53 # | . |
| Insert | 1D | Esc [ 2 ~ | Keypad Enter | 56 # | (CR) |
| Remove | 1F | Esc [ 3 ~ | Keypad PF1 | 59 ~ | Esc 0 P |
| Select | 21 | Esc [ 4 ~ | Keypad PF2 | 5C ~ | Esc 0 Q |
| Prev Screen | 23 | Esc [ 5 ~ | Keypad PF3 | 5F ~ | Esc 0 R |
| Next Screen | 25 | Esc [ 6 ~ | Keypad PF4 | 62 ~ | Esc 0 S |
| Up-Arrow | 27 ^ | Esc [ A | Break | 65 | not ret. |

Notes:
* Compose key Esc sequence returned only by Rainbow 100
+ Print Screen key Esc sequence returned only by Rainbow 100+ and 100B
^ These depend on Cursor Key mode and ANSII/VT52 mode
# These depend on Keypad mode and ANSII/VT52 mode
~ These depend on ANSII/VT52 mode

Sequences shown are for ANSII mode

# CHAPTER 3

# MS-DOS

This chapter contains two parts:

- Part A describes the operation of the MS-DOS operating system (including its internal and external commands)
- Part B describes the principle parts of the MS-DOS operating system

## Part A - Operation of MS-DOS

### 3.1 OVERVIEW

MS-DOS is an operating system for micro-computers using Intel 8086 and 8088 microprocessors. An operating system is a program that controls the overall operation of a computer. It provides an environment within the computer that enables you to easily perform operations such as:

- Starting a program
- Copying files
- Displaying a directory of files
- Simplifying programming

The MS-DOS operating system provides functions for commonly-used operations and I/O operations that are hardware-independent. Thus, you can write an application program to run under the MS-DOS operating system without requiring a detailed knowledge of the computer's hardware. Such a program runs on any computers that can run the MS-DOS operating system, as long as the computer has the appropriate peripherals. The "DOS" in MS-DOS stands for Disk Operating System, which means that MS-DOS provides all the logical operations necessary for writing and reading files to and from disk storage devices.

### 3.1.1 MS-DOS Operating System Versions

The MS-DOS operating system has evolved through a number of versions. Microsoft's Version 2.0 of the MS-DOS operating system is the basis for DIGITAL's Version 2.01 and Version 2.05.

The MS-DOS Version 2.01 operating system does not contain Winchester hard disk support; the MS-DOS Version 2.05 does. Other features were also added to the BIOS of Version 2.05, and several new utilities were added to its distribution diskette. Among these are:

- MDRIVE, which lets you use excess RAM to be used as a fast, logical disk drive

- RDCPM, which reads files from Rainbow CP/M-86/80 diskettes

- MEDIACHK, which permits the checking of media to be disabled, resulting in faster disk operations

- BACKUP, a utility for making and restoring backup copies of the Winchester hard disk

Both MS-DOS operating system Version 2.01 and Version 2.05 run on the Rainbow 100, 100+, and 100B, provided it has at least 128K bytes of RAM. Only Version 2.05 includes Winchester disk drivers.

Programs written to run under the MS-DOS Version 2.01 operating system run under the MS-DOS Version 2.05 operating system.

## 3.2 LOADING THE MS-DOS OPERATING SYSTEM

DIGITAL supplies the MS-DOS operating system on a diskette. Such a diskette is called a system diskette. A system diskette contains the system files:

- IO.SYS

- MSDOS.SYS

- COMMAND.COM

These system files contain the MS-DOS operating system code. The outermost tracks of all MS-DOS-formatted diskettes store a loader program, and three data areas (for keeping track of files stored on the diskette). The number of tracks required to store the loader program and data areas varies according to the capacity of a diskette track. Rainbow computer diskettes use about three and 1/5 tracks for this.

Files are stored on the remainder of the diskette. Non-system diskettes differ from system diskettes only in that they do not contain the three system files.

The Rainbow computer contains a small ROM-resident routine that reads the first sector of the outermost track (track 0) of the diskette in the startup (BOOT) drive whenever you turn on or reset the system. This sector contains a small program that reads a loader program from the remainder of the reserved tracks into RAM and starts it executing. The loader program reads two "hidden" system files (MSDOS.SYS, IO.SYS) plus the file COMMAND.COM, from the diskette into RAM, and starts execution. This process is known as starting or BOOTing the system. IO.SYS is loaded just above the 8088 interrupt vector space, and MSDOS.SYS, which contains interrupt handlers, service routines, buffers, control areas and installed device drivers, is loaded immediately above IO.SYS.

The file COMMAND.COM contains:

- A "resident" part
- A "transient" part

The resident part is loaded just above MSDOS.SYS and contains interrupt handlers for those interrupt types used by the MS-DOS operating system plus code to reload the transient part of the file.

The transient part contains code for all of the internal commands and the batch file processor. It is loaded in the highest addressed RAM available in the system. The transient part of COMMAND.COM:

- Issues the prompt
- Reads the command from the keyboard or batch file
- Executes the command itself (internal commands) or builds a command-line
- Loads a program file (external command)
- Starts execution.

The external command is loaded into the lowest available free memory. The MS-DOS operating system maintains a table of available memory, which keeps track of where each program is loaded.

Whenever an application or external command finishes execution and returns control to the MS-DOS operating system, the resident part of COMMAND.COM determines, by a check-sum process, whether the transient part needs to be reloaded. If necessary, it reloads the transient part before issuing its prompt.

Application programs are loaded in the area of memory between the two parts of COMMAND.COM.

Figure 3-1 shows an MS-DOS memory map:

```
           Top of Available RAM
       ---------------------------
      |       COMMAND.COM          |   Displays prompt
      |     (Transient part -      |   Command Interpreter
      |     May be overlaid by     |     Gets Cmd from Kbd/.BAT
      |     External Commands)     |   Builds command line
      |                            |     Loads and Executes
      |                            |       External Commands
      |                            |     Executes Internal Cmds
       ---------------------------       Processes Batch Files
      |    256 Byte User Stack     |
      |     for .COM programs      |
       ---------------------------
      |    External Command(s)     |
      |     .COM or .EXE files     |
       ---------------------------
      |       COMMAND.COM          |   Contains Auto-Exec code
      |   (Initialization part)    |   Inits lowest Segment
      |                            |    Address
      |   (Overlaid by Ext Cmds)   |
       ---------------------------
      |                            |   Reload Transient part
      |       COMMAND.COM          |   Int. 23H (Ctrl/C exit)
      |      (Resident part)       |   Int. 24H (Fat. Err exit)
      |                            |   Error handlers
       ---------------------------
      |                            |   Buffers
      |                            |   Control Areas
      |                            |   Installed Device Drivers
        -  -  -  -  -  -  -  -  -
      |                            |   Interrupt Handlers
      |        MSDOS.SYS           |   Service Routines
      |                            |   (Int 21H Functions)
       ---------------------------
      |                            |
      |         IO.SYS             |    MS-DOS  Interface to Hardware
      |                            |
       ---------------------------
3ffH  |                            |
      |    Interrupt Vectors       |
      |                            |
000H   ---------------------------
```

Figure 3-1: MS-DOS Memory Map

## 3.3 DISTRIBUTION DISKETTE CONTENTS

Table 3-1 lists the MS-DOS operating system files stored on the system diskettes.

Table 3-1: MS-DOS Operating System Files on System Diskette

Standard MS-DOS Files

| File name | Function |
|---|---|
| COMMAND.COM | MS-DOS Command Processor |
| * MSDOS.SYS | MS-DOS Operating System |
| * IO.SYS | Hardware-Operating System Interface |
| EDLIN.COM | Line Editor |
| DEBUG.COM | Debugger |
| LINK.EXE | Linker |
| CHKDSK.COM | Checks diskettes |
| % FORMAT.COM | Formats diskettes |
| SYS.COM | Transfers System |
| DISKCOPY.COM | Copies entire diskettes |
| RECOVER.COM | Recovers diskettes |
| PRINT.COM | Print Spooler |
| MORE.COM | Reviews Text |
| SORT.EXE | Sorts Text |
| FIND.EXE | Finds a string in a file(s) or std input |
| EXE2BIN.EXE | Converts .EXE files to .COM |
| FC.EXE | Compares files |

Non-Standard MS-DOS Files

|  |  |
|---|---|
| MASM.EXE | Macro Assembler |
| CREF.EXE | Cross Reference Utility for MASM |

Files only on Rainbow computer MS-DOS V2.05 Diskette

|  |  |
|---|---|
| CONFIG.SYS | System configuration file (Req'd by MDRIVE) |
| MDRIVE.SYS | Virtual Memory diskette |
| MDRIVE.COM | " " " |
| BACKUP.EXE | Hard disk Backup & Restore Utility |
| MEDIACHK.EXE | Enables/Disables Media Checking function |
| RDCPM.EXE | Reads CP/M Formatted diskette files |
| README.HLP | Addenda and Errata to Rainbow computer documentation |

* Denotes "hidden" files

% Different levels of functionality in the two Rainbow computer MS-DOS versions

## 3.4 COMMANDS

The MS-DOS operating system uses the term "command" as both a verb and a noun. When MS-DOS issues a prompt such as "A>", it waits for you to type a command. This is the verb use. The code that performs the operation is called a command. It is the command processor part of the MS-DOS operating system that issues the prompt and invokes the program with the command name you typed.

NOTE

> Any program stored as a diskette file is called an external command. This is because such a program is not internal to the operating system. It is a command, because its name must be given in response to the command prompt.

### 3.4.1 Rainbow MS-DOS Version 2.05 Special Command Caveats

The following section describes some caveats you should keep in mind as you use the MS-DOS operating system.

FORMAT

There are two formatting processes used with diskettes. The first, and most general, is physical formatting. This is the process whereby synchronizing data is placed on a newly manufactured diskette. This synchronizing data is required in order for the diskette controller circuits to read and write data to the diskette. The MS-DOS operating system, however, uses the term "formatting" to mean an entirely different process. This process initializes a blank, but physically-formatted diskette, with an area for storing the diskette directory and a table called the File Allocation Table (FAT).

Both types of formatting must be done for MS-DOS diskettes. DIGITAL's RX50 diskettes are already physically formatted, but diskettes that have been damaged by exposure to magnetic fields may have to be physically reformatted.

The MS-DOS Version 2.01 formatting program only initializes a blank diskette and creates the file allocation table (FAT). The MS-DOS Version 2.05 formatting program physically formats the diskette and then initializes it. (Use the FORMAT command's /I switch to format both ways).

The syntax for the FORMAT command is:

```
FORMAT d:[/S][/I][/V]
```

Also, the MS-DOS Version 2.05 FORMAT command does NOT format a diskette in the default drive. See the section entitled "External Commands" later in this chapter for further information.

The MS-DOS initialized Winchester disk partitions can also be MS-DOS formatted but not physically formatted.

If you specify a volume label when the FORMAT program asks for one, you cannot later make this diskette into a system diskette using the SYS command. If you do this, the operating system displays an error message to the effect that there is no space on the diskette for system files. To make a system diskette, either format the diskette with the "/S" command option (with or without a volume label) or format the diskette without a volume label, then use SYS.

The MS-DOS Version 2.05 operating system's FORMAT command also performs MS-DOS soft formatting on Winchester partitions previously created by the partitioning formatting utility distributed with the Winchester hardware option. However, the "/I" switch for physically formatting the Winchester disk is inoperative, and does nothing if invoked. Hard formatting a Winchester disk can only be done with the above mentioned partitioning formatting utility.

## DISKCOPY

Diskettes receiving the files copied by the DISKCOPY command must have been previously "physically" formatted, not MS-DOS formatted. Diskettes that are specially formatted, such as the Lotus 1-2-3 system diskettes, should NOT be copied using the DISKCOPY command, because their special features can be corrupted.

Use a backslash character when using path names.

Pressing Ctrl/C stops a program if the running program notices the key sequence or if it uses system functions to test for the key sequence. When BREAK is on, all system functions include this test.

The command processor can read commands from the disk file as well as accepting them from the keyboard. This file must have an extension of .BAT. The file contains several MS-DOS command lines, each of which are executed by the command processor once you type the file name at the prompt.

The command processor has also been arranged to automatically search the BOOT disk for a batch file with the name "AUTOEXEC.BAT" immediately after the BOOT operation. If one is found, the file is executed as if you had entered the name AUTOEXEC at the prompt. However, the normal prompts for TIME and DATE are bypassed. If your system expects to use TIME or DATE during the session, the AUTOEXEC command file should include entries for the TIME and DATE commands.

You can use the following commands in command files:

- ECHO
- FOR
- GOTO
- IF
- PAUSE
- REM
- SHIFT

### 3.4.2 Generic Commands

This sections describes the external MS-DOS commands unique to the Rainbow computer.

BREAK

Most application programs that expect to use Ctrl/C use the system function provided for this purpose. It is not necessary, in this case, for the operator to first invoke the BREAK OFF command.

CHKDSK

Do NOT use the CHKDSK command with the /F option for an IBM diskette under Rainbow computer's MS-DOS Version 2.05 operating system. If the IBM diskette is double-sided, it can be destroyed.

COPY

You can use character I/O device names (such as PRN, AUX) as file names for the COPY command.

When copying from a serial I/O device (for example, from another computer by means of the AUX device (communications port) or the PRN device (printer port)), the operation does not work as expected. Instead, carriage return characters are removed from the data stream, characters are not echoed to the input device, and either a single Ctrl/Z or two successive carriage-return characters ends the COPY operation.

CTTY

The CTTY command changes the device used for interactive entry by the operator, but does not change any device names. For example, the command CTTY AUX causes all operator commands to be expected from the AUX port. If another terminal, such as a VT100, is connected to the communications port (AUX), it can be used for controlling the Rainbow computer. However, if, while doing this, you enter the command COPY file name CON, the file is displayed on the Rainbow computer's screen, not on your VT100. Similarly the command COPY CON file name places whatever is typed on the Rainbow computer's keyboard into the file, not the VT100's keyboard.

ECHO OFF

The ECHO OFF command is echoed.

EXE2BIN

Use this external command to convert .EXE files into .COM files. The

GOTO

A Batch-File label is defined by a leading colon (":"). The characters following the colon define a label. The characters following a GOTO must BE a label. They do not define the label.

## PRINT

The file using the /C switch, and all file entries following it in the print queue, are removed from the print queue until you type a /P switch.

The first time you use the PRINT command after starting the MS-DOS operating system, it prompts you for the name of the list device, the default being PRN (also shown in the prompt). Just press the <Return> key, unless you wish to specify some other port.

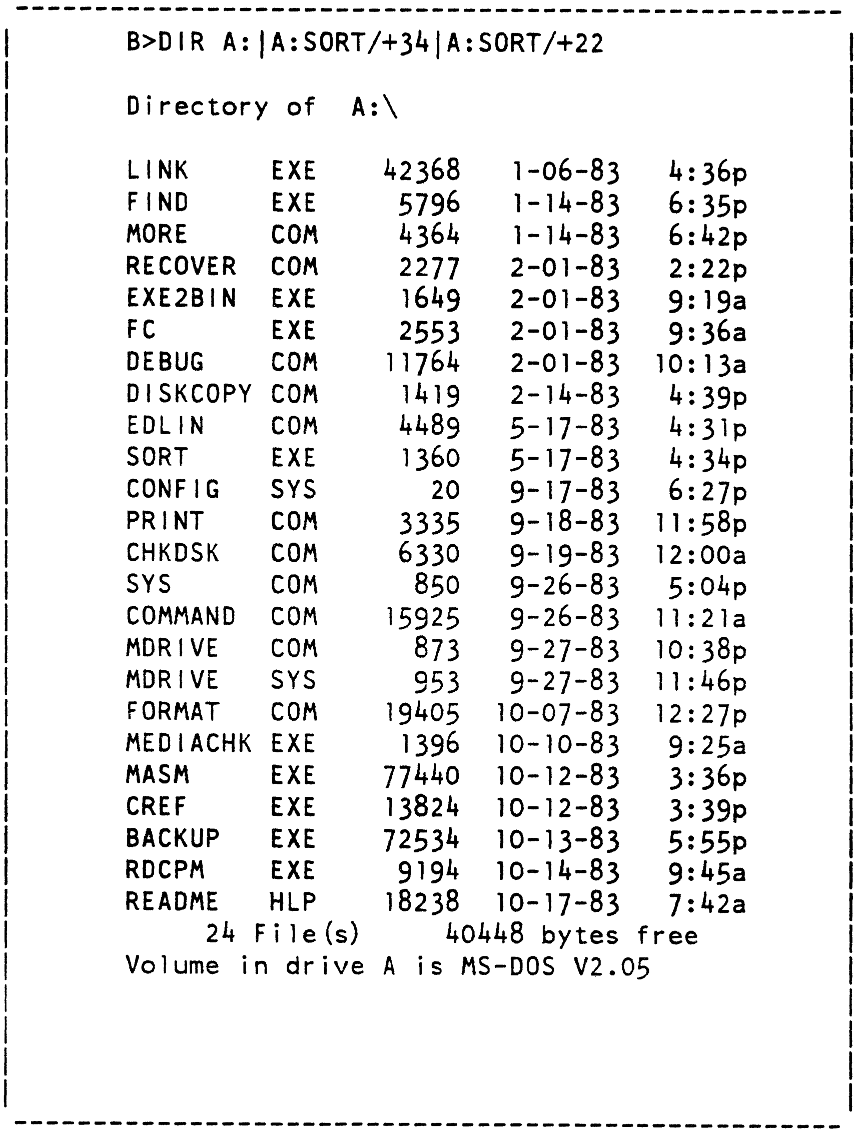
## RECOVER

RECOVER attempts to read and process data from a bad diskette. The data may be bad and can cause unpredictable, or incorrect, results. DIGITAL recommends, however, that you use this command only as a last resort to recover files from a bad diskette.

## SORT

Use the SORT command to sort any text file. Be sure that each line has the same fixed-length fields. SORT can be pipelined for multiple-field sorts, such as for DATE and TIME. When performing multiple-field sorts, the secondary, or least significant sorts, must be performed first, and the primary, or most significant sorts, last.

Figure 3-2 shows a diskette DIRectory sorted by both date and time. The two lines giving the number of files and the volume label are included in the sort process. If the last sort had been from column 24 instead of 22, the line containing 40448 would have been somewhere within the sorted list. The command was invoked from the B drive because SORT creates temporary files on the default diskette and the diskette in the A drive was write protected. The SORT command was on the diskette in the A drive.



```
B>DIR A:|A:SORT/+34|A:SORT/+22

Directory of  A:\

LINK     EXE    42368   1-06-83   4:36p
FIND     EXE     5796   1-14-83   6:35p
MORE     COM     4364   1-14-83   6:42p
RECOVER  COM     2277   2-01-83   2:22p
EXE2BIN  EXE     1649   2-01-83   9:19a
FC       EXE     2553   2-01-83   9:36a
DEBUG    COM    11764   2-01-83  10:13a
DISKCOPY COM     1419   2-14-83   4:39p
EDLIN    COM     4489   5-17-83   4:31p
SORT     EXE     1360   5-17-83   4:34p
CONFIG   SYS       20   9-17-83   6:27p
PRINT    COM     3335   9-18-83  11:58p
CHKDSK   COM     6330   9-19-83  12:00a
SYS      COM      850   9-26-83   5:04p
COMMAND  COM    15925   9-26-83  11:21a
MDRIVE   COM      873   9-27-83  10:38p
MDRIVE   SYS      953   9-27-83  11:46p
FORMAT   COM    19405  10-07-83  12:27p
MEDIACHK EXE     1396  10-10-83   9:25a
MASM     EXE    77440  10-12-83   3:36p
CREF     EXE    13824  10-12-83   3:39p
BACKUP   EXE    72534  10-13-83   5:55p
RDCPM    EXE     9194  10-14-83   9:45a
README   HLP    18238  10-17-83   7:42a
     24 File(s)     40448 bytes free
Volume in drive A is MS-DOS V2.05
```

Figure 3-2:  Sorted Diskette Directory

### 3.4.3 MS-DOS Version 2.05 Additional Commands

RDCPM

The README.HLP file, stored on the MS-DOS Version 2.05 distribution diskette, describes the RDCPM command.

```
RDCPM DIR B:[filename.typ]
```

This command displays the directory of files on the CP/M-86/80 diskette in drive B: matching the file specification. No default is allowed for the drive. The default file specification is *.*.

```
RDCPM READ B:[filename.typ] [A:][path]
```

This command copies all files on the CP/M diskette in drive B: matching the file specification to the directory described by "path" on MS-DOS drive A:. No default is allowed for B:. The default file specification is *.*. The default drive is the current MS-DOS default drive. The default path is the current directory. If the path is specified, it must exist.

```
RDCPM TYPE B:[filename.typ]
```

This command types the contents of files on the CP/M diskette in drive B: matching the file specification. No default is allowed for B:. The default file specification is *.*.

MEDIACHK

The MS-DOS Version 2.05 operating system BIOS contains code that checks the type of disk media in a disk drive on every read and write operation to that drive. This checking takes time to execute, and on large files can slow down operation. MEDIACHK is a program that disables, or enables, this media checking function. When disabled, COPY operations proceed faster than when they are enabled.

```
MEDIACHK
```

This command displays the current setting

```
MEDIACHK ON
```

This command turns MEDIACHK on

```
MEDIACHK OFF
```

This command turns MEDIACHK off

## 3.5 DEVICE NAMES

File names beginning with CON, AUX, LST, PRN, and NUL are illegal because they are used by the MS-DOS operating system to identify the following character I/O devices:

AUX - Communications Port

AUX2 - Extended Communications Port

CON - Console Device (Keyboard/Video Monitor)

PRN - Printer Port

When you use these device names, you need only type the name, although the MS-DOS operating system will also accept them followed by a colon.

## 3.6 EDITING AND FUNCTION KEYS

The MS-DOS operating system lets you easily edit command lines you have entered at the keyboard. EDLIN, the MS-DOS line editor, provides these same editing functions. EDLIN is described in the users manuals that came with your MS-DOS operating system kit. This section describes some additional information on EDLIN.

When editing an existing file, EDLIN changes the extension of the original copy to .BAK. If two files have the same name, it is difficult to determine which file was the original of the .BAK file. Therefore, avoid giving files the same file name.

To insert an "ESC" character, use the <Interrupt> key, rather than the <ESC> key.

The syntax for (C)opy includes a comma between the line number and the "C". This is optional.

When in (I)nsert mode, pressing Ctrl/C returns to the EDLIN command prompt. A Ctrl/Z sometimes also exits the (I)nsert mode. The latter, however, usually must be followed by pressing the Return key. If the Ctrl/Z is immediately followed by other characters before the Return, the Ctrl/Z code is entered in the text, and (I)nsert is not stopped.

## Part B - Principle Parts

### 3.7 MS-DOS PROGRAMS AND ROUTINES

The programs and routines that make up the MS-DOS operating system are divided into three major functional areas:

1. COMMAND.COM - Command Processor

   The file COMMAND.COM is the command processor. It is the part of the MS-DOS operating system that accepts, interprets, and acts upon the commands you enter on the console keyboard. It recognizes and executes the "internal" commands, and loads "external commands" (programs) and starts execution. It is also called an MS-DOS "shell."

2. DOS - Disk Operating System

   DOS, contained in the file MSDOS.SYS, performs logical I/O and disk operations, either in response to requests by the command processor's internal commands, or in response to your external commands. DOS also contains a number of useful "system calls," or functions. These functions provide disk and other I/O operations to programs through a standardized access method, making programs transportable from one system to another, and greatly simplifying their preparation.

3. BIOS - Basic Input - Output System

   The Basic Input - Output System (BIOS) contains routines that control the hardware of the computer system. These are custom written for each system and are contained in the file IO.SYS. Included are routines that:

   - Display a character on the logical console device

   - Send a character to the logical printing device

   - Read a record from a disk

   - Write a record to a disk

   The DOS routines and system functions access these BIOS routines to perform all I/O

Generally, the command processor and DOS portions of every MS-DOS operating system are byte-for-byte, bit-for-bit identical, as provided by Microsoft, while the BIOS is customized for the system's particular hardware.

### 3.7.1 Command Processor

After you start the MS-DOS operating system, a prompt is displayed on your screen, such as "A>". This tells you that the MS-DOS operating system is ready to receive a command, and that default disk operations are to be performed on the A: drive. The command processor displays the prompt, and either executes, or loads and starts, the command.

### 3.7.2 Command Execution

When you type a command name at the keyboard, the command processor evaluates it. First, it determines whether it is an internal command. If it is an internal command, it is immediately executed. If it is not, it attempts to find a file having the specified name on the specified drive, or if no drive was specified, on the default drive. The command processor expects that the file type associated with the file name is to be either .COM, .EXE, or .BAT, and searches for them in that order. If a command file, with the name specified in the command line, cannot be found, the operating system displays the message "Invalid Command," and issues another prompt.

When the command processor finds the specified .COM or .EXE file, it reads the file into RAM. Files of both types are usually loaded into the lowest available memory.

### 3.7.3 Program Segment Prefix

After loading an external command file, the command processor initializes the first 256 bytes of the program segment with several values for use by the loaded program. These first 256 bytes are called the Program Segment Prefix (PSP). Therefore, all application programs must begin at offset 100H. One of the DOS System functions (EXEC) can also be invoked by an application program to load another program file, and it, too, has a PSP.

After initializing the Program Segment Prefix, the command processor starts program execution. Once the external command is started, the command processor is not used again until the program terminates.

The Microsoft Operating System Programmer's Reference Manual provides a detailed description of the PSP and its contents. (This manual is contained in this kit.)

## 3.8 DOS

### 3.8.1 General

The DOS (Disk Operating System) constitutes the heart of the MS-DOS operating system. It performs all logical disk I/O operations, calling on the BIOS routines to read and write. It also provides a large number of useful system functions, which application programs can call to do disk and other I/O operations.

### 3.8.2 Disk Drives

The MS-DOS operating system can access several disk drives. Each is identified by a single letter, "A," "B," "C,". Disk drives can be either actual physical drives, or "virtual" or "logical" drives, such as portions of one large capacity hard disk.

Hard disks can often store more data than can be accommodated by one MS-DOS drive name, so they are subdivided into several partitions.

Each partition can store a portion of the total disk space. Each partition is then given a "logical" drive name.

A logical drive can even be a part of RAM. The MS-DOS Version 2.05 operating system includes a utility called MDRIVE that sets up an area of RAM as a fast access memory disk. Files can be COPYed in and out of it, as with any other logical drive. MDRIVE is useful when running programs such as editors. Of course, it is necessary to COPY files into it before editing and to COPY any that have been modified back to a physical disk before turning off the power.

### 3.8.3 Disk Files

Central to the MS-DOS operating system is its method for storing and accessing data files on disks. The MS-DOS operating system was originally written to work with diskettes, and it still does. However, it also works with hard disks.

### 3.8.4 Disk Organization

The MS-DOS operating system reserves one or more of the first tracks of a disk for storing a loader program. These first track are called the "system tracks." You us all remaining tracks for storing files.

System diskettes contain the MS-DOS operating system in three files:

- IO.SYS,
- MS-DOS.SYS
- COMMAND.COM

The two .SYS files are hidden. They are not displayed by the DIR command.

The MS-DOS operating system writes data to disk files in "logical" records, which can be any length, but are usually 128 bytes each. Data files are written wherever there are empty locations, and there can be many locations containing portions of a file. Entries are made in a Directory and a File Allocation Table for each file on a disk that tell the MS-DOS operating system where the file is stored on the disk.

### 3.8.5 Clusters

Because even a small diskette can store a great many 128 byte records, it is impossible to have the directory and File Allocation Table keep track of each one. Instead, MS-DOS divides a disk's capacity into larger groups, called "clusters," that contain several logical records each.

The data storage area of each disk is addressed in terms of these clusters. They are numbered with zero at the beginning of the data portion of the disk and "spiral" inward with increasing cluster numbers through the remainder of the disk. It is the number of clusters containing a file's data that are stored in the disk's directory and File Allocation Table.

The first few clusters on each disk are reserved to store its directory, the number depending upon the capacity of the disk and cluster size. Disks can have capacities ranging from less than 100K bytes to many megabytes. Cluster sizes can be 512, 1K, 2K, 4K, or 8K bytes. The Rainbow computer's MS-DOS RX50 diskettes use a cluster size of 512 bytes, while the Winchester drives use a cluster size of either 2K or 4K bytes, depending on the size of the partition. The first two clusters on the Rainbow computer's RX50 diskettes are reserved for the Directory.

Each MS-DOS disk contains an ID byte to identify the disk type. MS-DOS Version 2.05 determines, through this byte, the characteristics of the disks to be read.

### 3.8.6 Directory

Figure 3-3 shows the contents of the first eight directory entries for MS-DOS Version 2.05 system diskette. These are stored in the first logical record of the first cluster. The left side of the figure shows the contents in hex notation, while the right side displays the same data in ASCII (non-displayable bytes are shown as dots).

```
Byte        Contents (Hex)               Contents (ASCII)
(Hex)

0000 494F202020202020 5359532700000000  IO        SYS'....
0010 00000000000091BA 4807020000300000  ........ H....0..

0020 4D53444F53202020 5359532700000000  MSDOS     SYS'....
0030 000000000000864C 3C071A00E6420000  .......L <....B..

0040 434F4D4D414E4420 434F4D2000000000  COMMAND   COM ....
0050 000000000000AF5A 3A073C00353E0000  .......Z :.<.5>..

0060 4D532D444F532056 3230350800000000  MS-DOS V  205.....
0070 0000000000006074 4E07000000000000  ......`t N.......

0080 434F4E4649472020 5359532000000000  CONFIG    SYS ....
0090 0000000000006C93 31075C0014000000  ......l. 1.\.....

00A0 4D44524956452020 5359532000000000  MDRIVE    SYS ....
00B0 000000000000D9BD 3B075D00B9030000  ........ ;.].....

00C0 4445425547202020 434F4D2000000000  DEBUG     COM ....
00D0 000000000000A551 41065F00F42D0000  .......Q A._..-..

00E0 43484B44534B2020 434F4D2000000000  CHKDSK    COM ....
00F0 0000000000001300 33077600BA180000  ........ 3.v.....
```

Figure 3-3: MS-DOS Directory Entries

### 3.8.7 Directory Fields

Every file on a disk, whether a hard disk or diskette, has an entry in either the root directory or a subordinate directory. Subordinate directories are actually files that have a directory entry in their "parent's" directory. Each directory entry contains thirty-two bytes.

As illustrated in Figure 3-4, the first byte tells MS-DOS if it is active or not, that is, whether it contains information about a file. If not, this byte is either 00H or E5H.

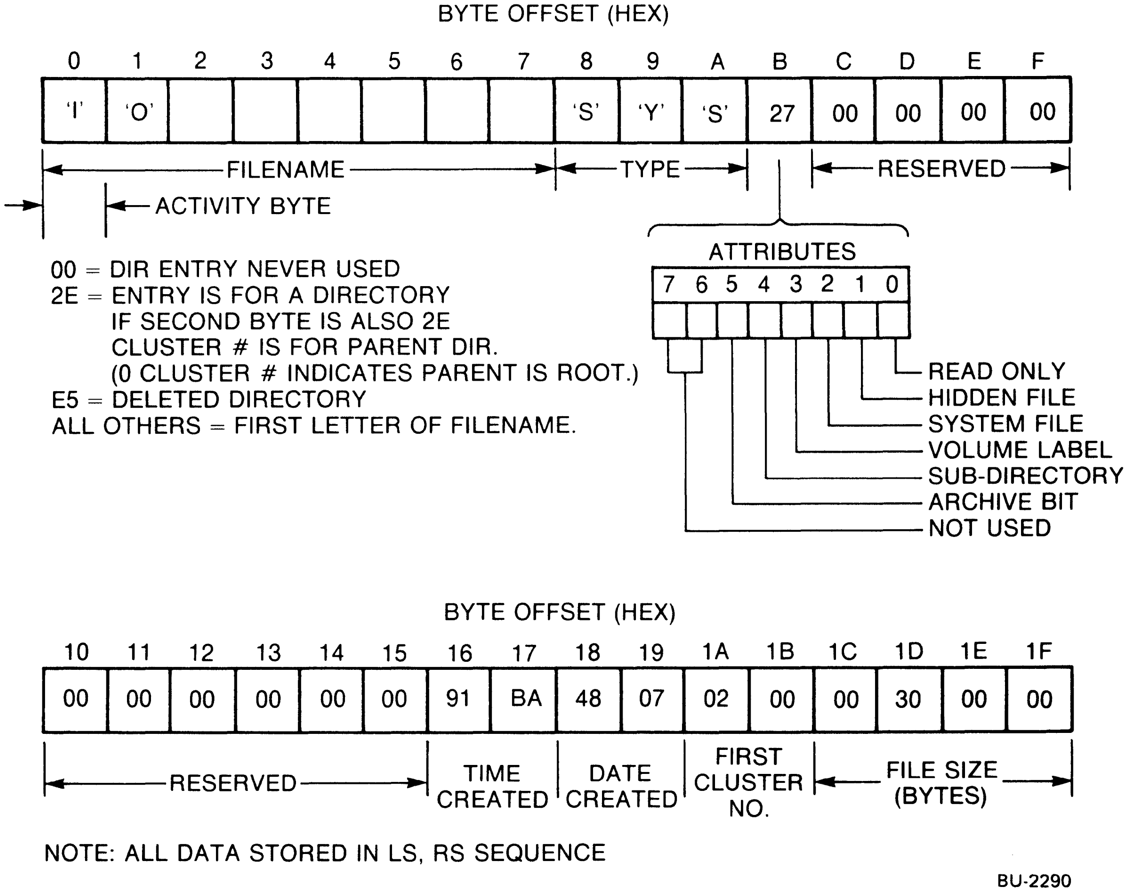

BYTE OFFSET (HEX)
0 1 2 3 4 5 6 7 8 9 A B C D E F
'I' 'O' 'S' 'Y' 'S' 27 00 00 00 00
FILENAME
TYPE
RESERVED
ACTIVITY BYTE
ATTRIBUTES
7 6 5 4 3 2 1 0
00 = DIR ENTRY NEVER USED
2E = ENTRY IS FOR A DIRECTORY
IF SECOND BYTE IS ALSO 2E
CLUSTER # IS FOR PARENT DIR.
(0 CLUSTER # INDICATES PARENT IS ROOT.)
E5 = DELETED DIRECTORY
ALL OTHERS = FIRST LETTER OF FILENAME.
READ ONLY
HIDDEN FILE
SYSTEM FILE
VOLUME LABEL
SUB-DIRECTORY
ARCHIVE BIT
NOT USED
BYTE OFFSET (HEX)
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
00 00 00 00 00 00 91 BA 48 07 02 00 00 30 00 00
RESERVED
TIME CREATED
DATE CREATED
FIRST CLUSTER NO.
FILE SIZE (BYTES)
NOTE: ALL DATA STORED IN LS, RS SEQUENCE
BU-2290

Figure 3-4: Format of a Directory Entry

When a disk or diskette is newly formatted, all directory entries have their first byte set to zero to indicate that they have never been used. When the operating system writes files to the disk or diskette, the first empty directory entries are always assigned first.

When the operating system deletes a file, the first byte of its directory entry is changed to an E5H. This arrangement speeds up directory search operations.

If a directory entry is encountered whose first byte is zero, the remainder of the disk or diskette must be empty. This first, or activity byte, is 2EH if the file is a sub-directory. Further, if its first two bytes are 2EH, the first cluster field of the directory entry contains the cluster number where the "Parent's" Directory is stored. This value is zero if the parent is the root directory.

If a directory entry is active, the first byte contains the first letter of the file name.

For active directory entries:

| Byte(s) | Contents |
|---|---|
| 00-0A | File name and file type |
| 0B | Attribute flags |
| 0C-15 | Reserved by MS-DOS |
| 16-17 | Time that the file last written |
| 18-19 | Date that the file last written |
| 1A-1B | First cluster where file is stored<br>(contained in least three significant nibbles) |
| 1C-1F | Number of bytes in the file |

When a file is DELeted, all but the first byte of its directory entry remains on the disk. All file allocation table entries containing cluster numbers allocated to the deleted file are reset to zero. This releases the disk space for use by other files. This also generally makes it impossible to reclaim a deleted file, unless the file was small enough to be contained in the single cluster whose address is in the directory entry. Small files can sometimes be reclaimed because the number of the first cluster assigned to the file is still in the deleted directory entry. This is only possible if the directory entry was not reused for a new file by an intervening COPY or other file-creating activity.

### 3.8.8 File Attributes

You can assign MS-DOS files one or more of six possible attributes, such as "Read-Only" or "Hidden." Assign these attributes by setting the appropriate bits of the attribute byte in offset OBH of the directory entry, using System Function 43H. The meaning for each bit of OBH follows:

| Bit | Meaning When Set |
|---|---|
| 7 | Not used. |
| 6 | Not used. |
| 5 | Archive bit. Set "On" each time file is written or closed. |
| 4 | File is a Sub-Directory. |
| 3 | Directory Entry is Volume Label, not file. If this bit is set, all other bits are ignored. |
| 2 | File is "System," does not display in Directory |
| 1 | File is "Hidden," does not display in Directory |
| 0 | File is Read-Only. |

The archive bit is useful for programs that make backup copies of files. All file write and close operations set this bit to On (1). A backup, or "archiving" program, should reset it to Off (0) after successfully copying the file. In this way, the backup program can determine which of the files have been changed since the last backup copy was made, and limit its copying to these.

The BACKUP utility on the MS-DOS system diskette has an option that copies only files with the archive bit set.

### 3.8.9 File Allocation Table (FAT)

The MS-DOS operating system maintains a special File Allocation Table (FAT) on each disk. The FAT stores the cluster-numbers allocated to files on the disk. Generally, disks are arranged with at least two copies of this table as insurance against accidental loss. If this table cannot be read, the files stored on the disk cannot be located, even though their data may be intact.

The FAT stores each cluster number in three half-byte "nibbles." This allows a maximum of 4096 clusters per disk file, because twelve bits can uniquely identify only 4096 locations. The first byte of the FAT usually contains a disk identifier.

Rainbow Disk Identifiers

| | V2.01 | V2.05 |
|---|---|---|
| Diskette | FC | FAH |
| Winchester Hard Disk | --- | F8H |

Rainbow computer diskettes have been assigned FAH by DIGITAL. Microsoft has assigned bytes FC,FDH,FEH, and FFH for systems disk configurations.

The second and third bytes of the FAT are always FFH. They assist in identifying the FAT, and also insure that the first FAT cluster entry begins on a byte boundary. Remember that it takes three bytes to store two FAT entries.

The first two clusters of a Rainbow computer's RX50 diskette are always assigned to the directory itself, so the first cluster number assigned to a file is number 02, because cluster numbers begin with zero.

Figure 3-5 shows the first 128 bytes of the FAT from the same MS-DOS Version 2.05 system diskette whose directory entries are shown in Figure 3-3. Notice that the three-nibble cluster entries appear mixed up. This is the way they appear on the diskette when the FAT is examined using DEBUG. When the FAT is read by the MS-DOS operating system, however, they are interpreted as illustrated in Figure 3-6. When a diskette is initialized by the FORMAT program, all FAT entries are set to zero to indicate that the cluster location is not assigned to a file, and that it is suitable for storing data. Any clusters spanning diskette defects have their FAT entries set to FF7H. The MS-DOS operating system skips over such clusters automatically, allowing the remainder of the diskette to be used.

```
Byte          Contents (Hex)
(Hex)

0000 FAFFFF0340000560 0007800009A0000B
0010 C0000DE0000F0001 1120011340011560
0020 0117800119F0FF1B C0011DE0011F0002
0030 2120022340022560 0227800229A0022B
0040 C0022DE0022F0003 3120033340033560
0050 0337800339A0033B F0FF3DE0033F0004
0060 4120044340044560 0447800449A0044B
0070 C0044DE0044F0005 5120055340055560
```

Figure 3-5: Beginning of File Allocation Table (FAT)

```
Byte          Contents (Hex)
(Hex)

0000 FAFFFF0340000560 0007800009A0000B  As stored on diskette
0000 FAFFFF0030040050 0600700800900A00  As interpreted

0010 C0000DE0000F0001 1120011340011560  As stored on diskette
0010 B00C00D00E00F01001101201301401501  As interpreted
```

Figure 3-6: Interpreted File Allocation Table Entries

The first cluster number assigned to a file is contained in the file's directory entry. When this cluster becomes filled with data, the MS-DOS operating system scans the FAT table from the top until it finds a table entry of zero, indicating an empty cluster. It then places the number of this newly assigned cluster into the FAT location of the cluster pointed to by the directory entry. Similarly, whenever a new cluster is assigned to a file, its number is placed into the FAT location for the previous cluster. In this way a file is linked from cluster to cluster. The FAT entry for the last cluster of a file is set to any value from FF8H through FFFH.

The <u>Microsoft Operating System Programmer's Reference Manual</u> describes these file and directory operations in detail. (This manual is contained in this kit.)

### 3.8.10 Sector Translation

Some computer systems store data on a disk using an interlace, that is, sectors are written in a non-contiguous sequence around a track. This is because time is required for computing and getting ready for the next sector's read or write operation. If this next sector's data were to be written to the disk's next physical sector, the latter might have already passed under the read head, thereby requiring a full revolution of the disk to access it. To avoid this, the record is written at a physical sector, separated by one or more intervening sectors. This interlacing can be accomplished in either of two ways:

- One method is to format the disk with interlaced physical sector numbers, such as in the sequence 1,3,5,7,9,2,4,6,8,10.

- The other way is to format the physical sectors sequentially and do a logical translation of the desired sector number before performing the physical read or write operation.

- When this method is used, the logical sector number specified by the MS-DOS operating system to the disk-driver is translated to obtain the correct physical sector. Files are written on Rainbow computer RX50 diskettes using this latter scheme, that is, in the sequence of physical sectors 1,3,5,7,9,2,4,6,8,10.
- For the hard disk drive, the sequence of physical sectors is 1,8,15,6,13,4,11,2,9,0,7,14,5,12,3,10.

### 3.8.11 File Control Blocks

When a program needs to read or write data to a file, certain information about the file must be passed to the MS-DOS operating system so that it can "open" the file, (locate its directory, and access its data records.) Such information can be passed to the MS-DOS operating system using a File Control Block (FCB).

A FCB consists of a 36 or 43-byte area in memory. It can be located anywhere the program finds convenient. The command processor sets up one or two default "unopened" FCBs in the Program Segment Prefix when a file is loaded, for the convenience of the programmer.

FCBs are either "unopened" or "opened." An "unopened" FCB must be set up before an "open" system function is executed. It need only contain file ID information (drive, file name, and file type). All other bytes are usually set to zero. The offset relative to DS of a file's FCB is placed by the program into the DX register prior to invoking the desired MS-DOS disk function. The "open" system function locates the file's directory and copies the remaining FCB fields from it for use by subsequent file operations. The FCB thereby becomes an "opened" FCB.

After all file operations are complete, the application program should "CLOSE" the file. The CLOSE operation copies the newly updated data from the FCB back onto the disk's directory and FAT. Closing is not necessary if only Read operations were performed, because the directory and FAT were not changed. If the close operation were omitted after data had been written to the file, however, the file-size field of the directory is not updated and would not agree with the FAT.

### 3.8.12 Extended File Control Blocks

An extended FCB is required when creating or searching for directory entries of files having special attributes. The extension consists of a seven byte prefix appended to the FCB.

| Byte(s) | Contents |
|---|---|
| 7 | FFH |
| 2-6 | OOH |
| 1 | Attributes (See above) |

### 3.8.13 Wildcards

When the MS-DOS operating system searches the directory of a disk for a matching file name, it accepts as a match any character in those positions of the FCB's file name and file type that contain question marks ("?"). Asterisks are expanded to question marks throughout the remainder of the file name and/or file type fields by the command processor and by system function 29H. This "Wild Card" feature makes multiple file operations possible.

## 3.9 BIOS

The BIOS is that part of the MS-DOS operating system that directly controls the hardware of the system. Each I/O device is controlled by a "driver." The standard drivers are contained in the file IO.SYS, one of the two system files found in all MS-DOS implementations. Certain features and functions are required by the MS-DOS operating system, since the command processor and the MS-DOS operating system use the I/O drivers provided in IO.SYS for all I/O, and they must be able to properly interface to it.

### 3.9.1 Serial I/O

3.9.1.1 Logical and Physical Device Assignments - The generic MS-DOS operating system communicates with three logical, serial I/O devices:

- Console (CON:)
- Auxiliary (AUX:)
- Print (PRN:)

The names in parentheses are the names the MS-DOS operating system recognizes as logical devices.

The Rainbow computer's MS-DOS implementations assign the keyboard/video monitor as the CON device, the communications port as the AUX device, and the printer port as the PRN device. In addition, they provide an additional device driver called AUX2 to the optional extended communications port.

The generic MS-DOS operating system also includes a CLOCK as one of its standard devices, for which there is a driver in IO.SYS.

The MS-DOS operating system lets you install additional, or alternative, device drivers. All that is necessary is for the desired driver to be on the system diskette as a file, and its name contained in a batch text file named CONFIG.SYS. When you start the operating system, it looks to see if the CONFIG.SYS file exists on the diskette, and if so, loads all drivers named. If these have the same name as a default driver, they preempt the latter. In this way special drivers can be easily loaded for new I/O operations.

The Microsoft Operating System Programmer's Reference Manual describes the rules for writing new device drivers and the operations available through the CONFIG.SYS file. (This manual is contained in this kit.)

3.9.1.2 Cross CPU Communications - The Z80A handles all diskette I/O in the Rainbow computer. Whenever the disk driver of MS-DOS is called, it calls the Z80A to perform the actual operation. The routines for these operations are contained in the Z80A's private RAM, having been placed there during the loading operation.

3.9.1.3 IBM Diskettes - The I/O drivers of MS-DOS Version 2.01 and Version 2.05 operating systems automatically identify and read RX50, IBM-8 and IBM-9 sectored single sided diskettes, and VT180 diskettes. MS-DOS files from any of the four types of single-sided diskette can be both read and written. However, data written to IBM diskettes is usually unreadable by the IBM PC, because the narrow head of the RX50 cannot fully erase the wide data tracks of these diskettes. Therefore, writing to IBM diskettes is not recommended. If it is absolutely necessary to write to an IBM diskette, try formatting a new diskette on the IBM drive, and COPY the Rainbow computer file onto it. Once you are able to read such a file on the IBM PC, COPY it to another IBM diskette for safekeeping.

3.9.1.4 MDRIVE - The Rainbow computer's MS-DOS Version 2.05 operating system includes a new device driver that can utilize some of available RAM as a logical disk drive. The system diskette contains a program called "MDRIVE.COM" You can use MDRIVE to set up this logical disk drive. Memory is assigned in groups of 64K bytes. MS-DOS can read and write files to this "disk" the same as it does to any real disk. The advantage is that files can be accessed quickly, because they are already in RAM.

You need at least 196K bytes of RAM to use MDRIVE, in which case you have one 64K block of RAM used as a logical drive. Files to be accessed must be COPYed into and out of the MDRIVE before and after use. MDRIVE stores the number of 64K blocks in MDRIVE.SYS, which is referenced in CONFIG.SYS through the DEVICE statement. (You must maintain CONFIG.SYS.) The drive name is the next available drive letter (E - I). For example, if you have 1 hard disk partition (drive E:), then MDRIVE is drive F:.

NOTE

You must always configure MDRIVE from the drive that you are going to start the operating system from.

## 3.10 SYSTEM FUNCTIONS

In addition to reading and writing disk files, the MS-DOS operating system provides some 80 different system functions for performing I/O and other tasks. Application programs can use and invoke these system functions. Generally, the system functions are invoked by loading parameters into registers and invoking interrupt 21H. Chapter 1 of the Microsoft MS-DOS O.S. Programmer's Reference Manual describes these system functions, and how to invoke them. (This manual is contained in this kit.)

NOTE

The MS-DOS operating system does not generally undertake to save any CPU registers when performing the system functions. Therefore, application programs must save the contents of those registers that need preserving BEFORE calling the function, and then restore them afterwards.

### 3.10.1 I/O Programming

The MS-DOS Version 2.05 operating system includes several additional system functions that:

- Provide enhanced serial I/O control
- Permit application programs to control the ports without the need for programming at the hardware level

You need to know how the MPSC is programmed as a prerequisite to understanding the serial I/O functions, as they all relate to MPSC programming. This information is described in:

- The Rainbow CP/M-86/80 V1.0 Tech Doc Kit (order number QV043-GZ). (This kit contains information about the Rainbow 100 computer.)
- The Rainbow MS-DOS V2.01 Technical Doc Kit (order number QV025-GZ). (This kit contains information about the Rainbow 100 computer.)
- The Rainbow 100+/100B Technical Doc Kit (order number QV069-GZ).
- The Rainbow Extended Communication Option Programmer's Reference Guide (order number AA-V172A-TV).

Chapter 5 provides details of these additional System Functions.

### 3.10.2 Interrupt Types

The 8088 interrupt types used by the MS-DOS operating system include several that are also used by the Rainbow computer's hardware and firmware. To resolve this conflict, hardware interrupts 20H through 27H are reassigned to other types. MS-DOS Version 2.01 includes a special routine in the interrupt handlers of those interrupts common to both the hardware and the MS-DOS operating system. This routine determines whether the interrupt was generated by the hardware or by software. If it determines that hardware generated the interrupt, it issues a software interrupt to the alternate interrupt type. (See Table 3-2.)

Table 3-2: Rainbow Computer MS-DOS Interrupt Vector Assignments

| Function | Interrupt Type (Hex) | | | |
|---|---|---|---|---|
| | V2.01 | | V2.05 | |
| | 100 | 100+<br>100B | 100 | 100+<br>100B |
| 8088 Processor | | | | |
| Divide by zero | 0 | 0 | 0 | 0 |
| Single step | 1 | 1 | 1 | 1 |
| NMI used for Memory Parity error or 8087 error | 2 | 2 | 2 | 2 |
| Break point instruction | 3 | 3 | 3 | 3 |
| Overflow | 4 | 4 | 4 | 4 |
| Hardware Interrupts | | | | |
| Video refresh | 40 | 40 | 40 | A0 |
| Graphics option | 42 | 42 | 42 | A2 |
| Ext. Comm. DMA | 43 | 43 | 43 | A3 |
| Comm/printer 7201 | 44 | 44 | 44 | A4 |
| Ext. Comm. int 0 (or Winchester) | 45 | 45 | 45 | A5 |
| Keyboard | 46 | 46 | 46 | A6 |
| Int from Z80A | 47 | 47 | 47 | A7 |
| Software Interrupts | | | | |
| 50/60 hz timer | 64 | 64 | 64 | 64 |
| Direct diskette I/O | 65 | 65 | 65 | 65 |
| Firmware functions | 18 | 18 | 18 | 18 |
| MS-DOS Interrupts | | | | |
| program terminate | 20 | 20 | 20 | 20 |
| function call | 21 | 21 | 21 | 21 |
| terminate address | 22/F0 | 22/F0 | 22/F0 | 22/F0 |
| Control-Break exit address | 23/F1 | 23/F1 | 23/F1 | 23/F1 |
| fatal error address | 24/F2 | 24/F2 | 24/F2 | 24/F2 |
| absolute disk read | 25 | 25 | 25 | 25 |
| absolute disk write | 26 | 26 | 26 | 26 |
| terminate, stay resident | 27 | 27 | 27 | 27 |
| Reserved by MS-DOS | 28<br>│<br>3F | 28<br>│<br>3F | 28<br>│<br>3F | 28<br>│<br>3F |

The system module used in the Rainbow 100+ and 100B computers includes circuitry that permits the hardware interrupt types to be set to either of two values, 20H through 27H, or A0H through A7H.

The firmware of the Rainbow 100+ and 100B computers includes a user callable function that moves several interrupt vectors from one location to another. The MS-DOS Version 2.05 operating system selects the alternative hardware interrupts and moves their vectors so that no conflict between hardware and software generated interrupts exists.

During system reset, the Rainbow computer's firmware initializes the interrupt vector for interrupt 28H (40 decimal) with the address of its built-in user callable functions. Interrupt 28H is also used by the operating system, however, so both MS-DOS Version 2.01 and Version 2.05 move the vector for the firmware functions to interrupt 18H. Programs that invoke the firmware routines must use software interrupt 18H.

### 3.10.3 Modifying Interrupt Vectors

Interrupt 21H, and MS-DOS system functions 25H and 35H, which set and get interrupt vectors for MS-DOS interrupts 20H through 27H, do not work for all of the interrupts. Applications which need to change these interrupt vectors should do so as follows:

- MS-DOS Version 2.05 operating system running on Rainbow 100+ or 100B computers

  Vectors for types 20H through 27H may be set and obtained by interrupt 21H, as well as by functions 25H and 35H.

- MS-DOS Version 2.05 operating system running on the Rainbow 100

  Vectors for types 22H, 23H, and 24H may be set or obtained by interrupt 21H, as well as by functions 25H and 35H.

Vectors for the remaining types must be changed by placing the new address into specific memory locations. These must be in the usual double-word format. The memory locations are:

| Type #<br>(Hex) | Memory Location<br>(Hex) |
|---|---|
| 20 | 40:0294 |
| 21 | 40:0298 |
| 25 | 40:02A0 |
| 26 | 40:02A4 |
| 27 | 40:02A8 |

- MS-DOS Version 2.01 operating system running on any Rainbow Series Computer

  Vectors for types 22H, 23H, and 24H may be set or obtained by interrupt 21H, as well as by functions 25H and 35H.

Vectors for the remaining types must be changed by placing the new address into specific memory locations. These must be in the usual double-word format. The memory locations are:

| Type #<br>(Hex) | Memory Location<br>(Hex) |
|---|---|
| 20 | 40:022B |
| 21 | 40:022F |
| 25 | 40:0233 |
| 26 | 40:0237 |
| 27 | 40:023B |

# CHAPTER 4

# MS-DOS VERSION 2.05 EXTENDED DOS FUNCTIONS

## 4.1 INTRODUCTION

This chapter describes special functions provided by the MS-DOS Version 2.05 BIOS.

DOS System Function 44H - IOCTL calls most of these special features to permit device control operations.

Chapter 2 describes some additional functions of the firmware.

## 4.2 DEVICE DRIVERS

The MS-DOS operating system requires a BIOS routine for each device on a system. These are called device drivers. ( The *Microsoft MS-DOS Operating System Programmer's Reference Manual* describes how these drivers routines are constructed, and the functions they provide.)

There are two types of device drivers:

- Serial I/O device drivers,
- Disk drivers.

The MS-DOS operating system calls serial devices "Character" devices and disk drivers "Block" drivers. The names reflect the amount of data handled during one call to the driver. One device driver can accommodate more than one actual device, as long as the devices are identical.

When the MS-DOS operating system reads or writes data to a disk file, or sends a character to a port, it calls on the driver to perform the actual read or write operation. The file IO.SYS contains drivers. Each driver provides the following functions:

- Initialization routines when you start the operating system
- Media checking function (disk drivers only)
- BIOS parameter block building (disk drivers only)
- Optional IOCTL I/O control routines
- Input and output routines
- Status routines
- Empty or "flush" routines

The optional IOCTL function is usually provided because some device characteristics may need to be changed by an application.

The MS-DOS operating system accesses these IOCTL functions through Function 44H. The MS-DOS Version 2.05 operating system has implemented several IOCTL functions for its serial I/O ports, diskettes, and Winchester hard disk. Some of the diskette and Winchester hard disk functions that would normally be accessed through Function 44H (IOCTL), have been implemented through software interrupt 65H.

Various functions provided for each device type in Version 2.05 are described below.

## 4.3 SERIAL I/O DEVICES

Functions Invoked by Function 44H - IOCTL

DOS Function 44H invokes all 25 additional serial I/O functions.

Most of these provide enhanced serial I/O port control and permit application programs to control these ports completely without the need for programming at the hardware level.

To understand these Serial I/O functions, you should be familiar with the operation of the Multi-Protocol Serial Controller chip (Nec 7201 or Intel 8274 MPSC), because many of the terms and functions relate to it.

The Serial I/O functions operate with three ports:

- Communication port (PORT A of the MPSC = AUX:)
- Printer port (PORT B of the MPSC = PRN:)
- Extended communications option port (PORT B of the option = AUX2)

Notice that the ports are not identical, nor are all the functions appropriate for all ports. Certain baud rates and modem signals are not available for printer PORT A of the Extended Communications option.

## 4.4 FUNCTIONS NOT PROVIDED

The following functions are not provided:

Full modem-control protocols
Synchronous protocols
Extended communication option's PORT A (RS 422)

### 4.4.1 PROTOCOLS

Asynchronous Support

Primitive routines are provided to support only asynchronous protocols.

Limited Modem Control

Limited Modem Control is provided so that application programs do not need to directly control the hardware. It also gives application programs a way to determine the state of the signals corresponding to each incoming and outgoing character. Characters received are assumed valid only when Receive Line Signal Detect (RLSD) is asserted, and the receive character is enabled. Characters are transmitted only when Clear To Send (CTS) is on.

Data Leads Only

Characters are transmitted and received regardless of the state of Data Terminal Ready (DTR) and Request To Send (RTS).

### 4.4.2 RECEIVE AUTO XON/XOFF

Auto XON/XOFF is supported at the driver level to handle receipt of XON/XOFF. If this feature is not selected, received XON/XOFF characters are stored as data in the circular buffer.

### 4.4.3 TRANSMIT AUTO XON/XOFF

This option monitors the receive character buffer, and automatically transmits XON and XOFF characters when the number of characters in the buffer exceeds certain limits. For this option to be effective, the application program must supply a receive character buffer of adequate size.

### 4.4.4 ALTERNATE CONTROL CHARACTERS FOR XON/XOFF PROTOCOL

The application program can specify control characters, other than DC1(11H) and DC3(13H), to be used for the XON/XOFF protocol.

### 4.4.5 MODEM SIGNAL CONTROL

The application program can set or clear the modem control signals without accessing the hardware.

### 4.4.6 BREAK TRANSMISSION AND DETECTION

Breaks can be sent for any duration, as determined by the application program. When a break is detected on a receiving device, a flag is set, which the application program can use as required.

### 4.4.7 PROGRAMMING OF PORTS

Ports can easily be programmed without directly accessing the hardware. The management of the interdependencies of ports is also provided, thereby relieving the application program of this difficult task. Application programs pass control and other information to the routines by a Communication Control Block (CCB), as described below.

### 4.4.8 RECEIVE CHARACTER AND TRANSMIT BUFFER EMPTY INTERRUPTS

Functions are provided that install and pass control to an application's interrupt handler upon:

- Receipt of a character

- Detecting that the transmitter buffer is empty

You can program the MPSC to generate interrupts under these two conditions. Any such interrupt handlers must also be disabled by the application program before termination. Functions are also provided to do this.

### 4.4.9 APPLICATION SUPPLIED BUFFER SIZE AND SPACE

An application can specify its own buffer size and location to be used for received characters. Each received character occupies two bytes in the buffer. One byte contains the character, the other byte contains information about the status of the port at the time the character was received (buffer overrun, parity error, break detection).

Applications should specify a buffer of at least 512 bytes if they use the auto transmit XON/XOFF feature. Such applications must be careful to restore the regular buffer before termination.

### 4.4.10 COMMUNICATION CONTROL BLOCK

MPSC control and setup information is passed between application programs and the functions through a Communication Control Block (CCB). Each cell of the CCB is a byte, except for the address and length of the optional user-specified buffer.

Notice that not all possible values are appropriate for all three ports, nor are all combinations of supported features allowed (for example, only Mark and Space are provided for 7 bits characters, and some values are included only for future support (for example, 7200 baud rate).

It is necessary to create or modify a CCB whenever a different configuration is required for a port. Each field of the CCB must be filled with the appropriate code for the desired parameter. Only those parameters that need to be changed must be filled. Those to remain unchanged must contain zero. When a passed value is not appropriate, the high order bit of the byte or word parameter is set to indicate an error.

NOTE

> Though some validity checking on passed parameters is performed, it is not guaranteed. The application program must insure that the rules and interfaces specified below are carefully followed.

Table 4-1: Communications Control Block Format

| Offset (Hex) | Offset Contents | Possible values |
|---|---|---|
| 0 | Device number | 1=communications, 2=printer, 3=extended comm |
| 1 | Modem Control | 1=data only, 2=limited modem control |
| 2 | Stop bits | 1=one, 2=one and one half, 3=two |
| 3 | Data bits ** | 5, 6, 7, 8, 7S, 7M |
| 4 | Parity ** | Even, Odd, None |
| 5 | Receive baud | 17 values ( no 7200 baud )<br>[printer cannot split baud rates ] |
| 6 | Transmit baud | 17 values (no 7200 baud) |
| 7 | XON character | If alt XON character to be used |
| 8 | XOFF character | If alt XOFF character to be used |
| 9 | RCV XON/XOFF | On or Off |
| A | XMT XON/XOFF | On or Off |
| B<br>\|<br>C | Alternate buffer size | If auto XMT XON/XOFF selected, user receive buffer is strongly suggested. An error is returned if the user buffer size is less than default. |
| D<br>\|<br>E<br>\|<br>F<br>\|<br>10 | start address of buffer in offset,segment format. | This field must have entry if user buffer size is non-zero. |

** Data bits and parity should be specified together. In all fields of the CCB, zero means no change is requested

If an alternate receive buffer is selected, a non-zero value greater than 50H must be entered. Otherwise the start address for the buffer is ignored and the default buffer is used.

Table 4-2: Communications Control Block Parameter Values

| Parameter Values: | | | Device Number: 1 (AUX) | 2 (PRN) | 3 (AUX2) |
|---|---|---|---|---|---|
| | | | Default Values: | | |
| Modem Control: | | | | | |
| 0 = No Change | | | | | |
| 1 = Data Leads | | | 1 | 1 | 1 |
| 2 = Limited Modem Control | | | | | |
| Auto XON/XOFF: | | | | | |
| 0 = No Change | | | | | |
| 1 = On | | | | | |
| 2 = Off | | RCV: | 2 | 2 | 2 |
| | | XMT: | 2 | 2 | 2 |
| Stop Bits: | | | | | |
| 0 = No Change | | | | | |
| 1 = One | | | 1 | 1 | 1 |
| 2 = one and one-half | | | | | |
| 3 = two (stop bits) | | | | | |
| Data Bits: | | | | | |
| 0 = No Change | Data bits and Parity | | | | |
| 1 = 5 | should be specified | | | | |
| 2 = 6 | together (see note). | | | | |
| 3 = 7 | | | | | |
| 4 = 8 | | | | 4 | |
| 5 = 7+Space | | | 5 | | 5 |
| 6 = 7+Mark | | | | | |
| Parity: | | | | | |
| 0 = No change | Data bits and Parity | | | | |
| 1 = even | should be specified | | | | |
| 2 = odd | together (see note). | | | | |
| 3 = none | | | 3 | 3 | 3 |

Note: If parity is specified without data bits also being specified for a new configuration, an error might be returned if the parity specified is odd or even and the data bits previously in effect were 7M or 7S.

| Parameter Values: | | 1 (AUX) | 2 (PRN) | 3 (AUX2) |
|---|---|---|---|---|
| XON character: | | 11H | 11H | 11H |
| XOFF character: | | 13H | 13H | 13H |
| Baud Rate - RCV | (See table for values) | 10H | 0EH | 10H |
| - XMT | | 10H | 0EH | 10H |
| Buffer Size: | (16-bit words) | 50H | 50H | 50H |

Table 4-3: Baud Rates

| | <---Hex Value placed in CCB---> | | |
|---|---|---|---|
| | Comm. | Ext. Comm. | Printer (*) |
| 50 | 1 | 1 | - |
| 75 | 2 | 2 | 2 |
| 110 | 3 | 3 | - |
| 134.5 | 4 | 4 | - |
| 150 | 5 | 5 | 5 |
| 200 | 6 | 6 | - |
| 300 | 7 | 7 | 7 |
| 600 | 8 | 8 | 8 |
| 1200 | 9 | 9 | 9 |
| 1800 | A | A | - |
| 2000 | B | B | - |
| 2400 | C | C | C |
| 3600 | D | D | - |
| 4800 | E | E | E D |
| 7200 | ** | ** | ** |
| 9600 | 10 D | 10 D | 10 |
| 19200 | 11 | 11 | - |

D Default Setting

* Transmit and Receive Baud rates for the printer port must always be the same.

** 7200 Baud is not available for any port.

- Not available

## 4.5 CALLING PROCESS

Application programs access functions by means of MS-DOS system function 44H (IOCTL). Entry conditions follow:

NOTE

> If the port number specified is invalid, the function might be ignored or an error condition might be returned. Application programs must determine whether a device is installed. This can be done by invoking the "Read Device Setup" function

## FUNCTION 44H - I/O CONTROL FOR DEVICES

ENTRY
-----

AH = 44H

AL = Function code
Both 02H and 03H work for either READ or WRITE

BX = Handle number
03H = AUX
04H = PRN
(AUX2 must be "Opened" via MS-DOS to get handle number)

DS:DX = IOCTL Packet Address

EXIT
----

Returned Data in IOCTL packet

The format of the IOCTL packet is:

| Offset | Label | Description |
|---|---|---|
| 0 | Function | Sub-function Number |
| 1 | Func_retC | Function return code:<br>FFH = successful, 0 = unsuccessful |
| 2 | Character | Character Input or Output |
| 3 | Char_Stat | Character Status |
| 4 to n | Buffer | Block Buffer |

The application program must supply the device number in the applicable buffer field. Set-up parameters must be passed each time a different set-up is required. The entry to each parameter either contains zero, if the parameter is to remain unchanged, or a value from 1 to the largest applicable number for that parameter. On return to the caller, the high order bit of the appropriate byte is set, if the particular option requested is not supported. However, this is not guaranteed if a value is outside of the range specified.

MODEM SIGNALS - Modem signals that are available are listed below. Notice that unused bits are undefined. See Functions 13 and 14 below for full description of their use.

Incoming signals.

```
           Bit
+---+---+---+---+---+---+---+---+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+-------+---+---+---+---+---+---+
|   |   |   |   |   |   |   | RI
|   |   |   |   |   |   |
|   |   |   |   |   |   | SRLSD (or SPDI, Bell 212A
|   |   |   |   |   |            speed indicator)
|   |   |   |   |   |DSR
|   |   |   |   |
|   |   |   |   |CTS
|   |   |   |
|   |   |   |RLSD
|   |   |
```

NOTE

SRLSD and SPDI signals are assigned to the same physical line. It is up to the application program to determine which one is connected. The remainder of the specification refers to this physical line as the signal SRLSD.

Also, control signals sometimes use connector pins other than those shown so that they can be monitored. The signals named in this document refer to the corresponding pin-outs and not the specific signals.

Outgoing signals that can be asserted.

```
           Bits
+---+---+---+---+---+---+---+---+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   | SP (speed select)
|   |   |   |   |   |   |
|   |   |   |   |   |   | SRTS
|   |   |   |   |   |
|   |   |   |   |   | DTR
|   |   |   |   |
|   |   |   |   | RTS
|   |   |   |
```

## 4.6 LOGICAL ASSIGNMENT OF DEVICES, AND DEFAULT SETTINGS

The two ports on the motherboard are assigned as follows:

PORT A to AUX

PORT B to PRN

PORT B on the extended communications option is assigned to AUX2

If the port (device) number is invalid, the function might be ignored or return an error condition.

It is up to the application program to determine if the device is installed. Do this by invoking the "Read Device Setup Values" function.

### FUNCTION 0 - PROGRAM DEVICE

This function reprograms the indicated device to the values specified in the CCB. It has no effect on the state of receiver enable.

```
ENTRY
-----
(FUNCTION)  = 0
(BUFFER)    = CCB

EXIT
----
(FUNC_RETC) = FFH, all functions programmed successfully

            = 0 not all functions programmed successfully. The
                fields that cannot be programmed in the control
                block are marked with high order bit set.
```

NOTE

RCV interrupts MUST be disabled before invoking this function, if the start address of the User's Buffer is changed by this function. Use Function 5 to do this.

## FUNCTION 1 - PROGRAM DEVICE TO DEFAULT NVM SETTINGS

This function reprograms the device to the default values specified in NVM. The optional communications port is set to the same values as the comm port. The receiver is enabled.

```
ENTRY
-----
(FUNCTION) = 1

EXIT
----
None
```

NOTE

> RCV interrupts MUST be disabled before invoking this function, if the start address of the User's Buffer is changed by this function. Use Function 5 to do this.

## FUNCTION 2 - SET DEVICE TO USE DEFAULT BUFFER

This function reprograms the device to use the default buffer. This must be done before program termination. The default buffer is reinitialized before it is used again.

```
ENTRY
-----
(FUNCTION) = 2

EXIT
----
None
```

NOTE

> Interrupts MUST be disabled before invoking this function, if the start address of the User's Buffer is changed by this function.

## FUNCTION 3 - READ DEVICE SET-UP VALUES

This function returns, in the CCB, information on the state of the device, that is, how it is programmed. The application must fill in the device number prior to the call to indicate which device is being asked for. All other fields must be set to zero.

```
ENTRY
-----
(FUNCTION) = 3
(BUFFER)   = CCB data

EXIT
----

(FUNC_RETC) = 0, illegal device number, or device not installed.
            = FFH, operation successful

(BUFFER)    = Current Device Settings
```

## FUNCTION 4 - ENABLE RECEIVER INTERRUPTS

This function enables the MPSC to receive characters. Characters received are placed in a circular buffer. In case of overruns, 1AH is stored in the buffer. Each character is stored along with its status. Receiver-interrupts are normally enabled.

```
ENTRY
-----
(FUNCTION) = 4

EXIT
----
None
```

NOTE

The function is ignored if the device number is invalid.

## FUNCTION 5 - DISABLE RECEIVER INTERRUPTS

This function ignores received characters.

```
ENTRY
-----
(FUNCTION) = 5

EXIT
----
None
```

## FUNCTION 6 - READ INPUT DEVICE STATUS

This function reads the status of the device's receive buffer.

```
ENTRY
-----
(FUNCTION) = 6

EXIT
----

(FUNC_RETC) = FF, character available
            = 0 , char  not available
```

## FUNCTION 7 - READ INPUT CHARACTER, RETURN IF NONE AVAILABLE

This function reads the next character from the circular buffer. If none are available, return to the caller.

```
ENTRY
-----
(FUNCTION) = 7

EXIT
----
(FUNC_RETC) = FF, character available
            = 0, char not available

(CHARACTER) = character, if available.

(CHAR_STAT) = character status

      Bit 7 = 1 Break character
      Bit 6 = 1 Framing Error
      Bit 5 = 1 Overrun error
      Bit 4 = 1 Parity error
```

## FUNCTION 8 - GET CHARACTER, RETURN WHEN AVAILABLE

This function reads the next character from the circular buffer. If none are available, wait till one is available.

```
ENTRY
-----
(FUNCTION) = 8

EXIT
----
(CHARACTER) = character

(CHAR_STAT) = status of character

     Bit 7 = 1 Break character
     Bit 6 = 1 Framing error
     Bit 5 = 1 Overrun error
     Bit 4 = 1 Parity error
```

NOTE

The function is ignored if the device number is invalid.

## FUNCTION 9 - READ OUTPUT DEVICE STATUS

This function reads the status of the transmitter.

```
ENTRY
-----
(FUNCTION) = 9

EXIT
----
(FUNC_RETC) = FF transmitter ready
            = 0 transmitter not ready or device number invalid
```

NOTE

If auto XON/XOFF is active, the state of the output status can change between 2 function calls.

## FUNCTION 10 - WRITE CHARACTER, RETURN IF UNABLE TO

This function puts the character in the transmitter. If it is unsuccessful, return to caller with appropriate error.

```
ENTRY
-----
(FUNCTION)  = 10

(CHARACTER) = character

EXIT
----
(FUNC_RETC) = FFH  transmit successful
            = 0 unable to transmit or device number invalid
```

NOTE

If auto XON/XOFF is enabled, the drivers monitor and handle XON and XOFF characters.

## FUNCTION 11 - PUT CHARACTER, RETURN WHEN SUCCESSFUL

This function puts a character into the transmit buffer, and does not return until the character is accepted.

```
ENTRY
-----
(FUNCTION)  = 11

(CHARACTER) = character

EXIT
----
None
```

NOTE

This function is ignored if the device number is invalid.

If auto XON/XOFF is enabled, the drivers monitor and handle XON and XOFF characters.

## FUNCTION 12 - TRANSMIT CHARACTER IMMEDIATELY

This function puts a character into the transmit buffer immediately, ahead of other characters in the queue, if any. It ignores the state of the modem signals, that is, if limited modem control is used, it is ineffective for this character, and the characters ahead of it in the buffer. USE THIS FUNCTION ONLY IN EMERGENCY SITUATIONS.

```
ENTRY
-----
(FUNCTION)  = 12

(CHARACTER) = character

EXIT
----
None
```

NOTE

This function is ignored if the device number is invalid.

If auto XON/XOFF is enabled, the drivers monitor and handle XON and XOFF characters.

## FUNCTION 13 - READ MODEM SIGNALS

This function fetches the incoming modem signals and the state of the transmit/receive flags. (Not applicable to the Printer port.)

```
ENTRY
-----
(FUNCTION) = 13

EXIT
----
(CHARACTER) =

      Bit 7 = 1, modem signals cannot be read
      Bit(s) 5-6,   Undefined
      Bit 4 = 1, RLSD  (comm port pin 8)
      Bit 3 = 1, CTS   (comm port pin 5)
      Bit 2 = 1, DSR   (comm port pin 6)
      Bit 1 = 1, SRLSD (comm port pin 12)
                  or
                 SPDI, bell 212A speed indicator
      Bit 0 = 1, RI on (comm port pin 22)
```

```
(CHAR_STAT) =

      Bit 7 = 1, XOFF transmitted
      Bit 6 = 1, XOFF received
      Bit 5 = 1, Receiver disabled
      Bit(s) 1-4, Undefined
      Bit 0 = 1, Transmitter empty
```

## FUNCTION 14 - SET/RESET MODEM OUTPUT SIGNALS

This function changes the state of the outgoing modem signals to the desired mask value. Notice that this also clears unmasked signals. The printer port's MODEM signals cannot be set. Modem signals are positive when asserted.

```
ENTRY
-----
(FUNCTION) = 14

(CHARACTER) = Modem signal mask

      Bit(s) 4-7, Ignored
      Bit 3 = 1, RTS set  (connector pin 4)
      Bit 2 = 1, DTR set  (connector pin 20)
      Bit 1 = 1, SRTS set (connector pin 19)
      Bit 0 = 1, SP set   (connector pin 23)

EXIT
----
(FUNC_RETC) = 0 modem signals cannot be set
            = FF modem signals set
```

## FUNCTION 15 - TRANSMIT BREAK

This function latches the transmit-data line to a space condition until "cease transmission of break" is requested. Invoke a CEASE TRANSMISSION OF BREAK function after the desired BREAK time has elapsed.

```
ENTRY
-----
(FUNCTION) = 15

EXIT
----
None
```

NOTE

> It is up to the application program to insure (if appropriate) the transmitter is empty.
>
> Notice also that the duration of the space condition is completely up to the application program.

## FUNCTION 16 - CEASE TRANSMISSION OF BREAK

This function returns the transmit-data line to a marking condition to terminate a previous "transmit break" command.

```
ENTRY
-----
(FUNCTION) = 16

EXIT
----
None
```

## FUNCTION 17 - SET RECEIVE CHARACTER INTERRUPT

This function permits a user's subroutine to be called upon each received-character interrupt. The call is made with a far call, and the user's routine must exit with a far return. The routine is called after the interrupt is serviced and has been reenabled. If a second interrupt occurs before this routine has been completed, it is not called the second time.

When called, the routine has the following conditions:

1. Interrupts off, and should remain off.

2. Stack is already changed to communications stack.

3. All register should be preserved by the called routine. Allow enough stack space for this.

The routine should not call any DOS functions.

```
ENTRY
-----
(FUNCTION) = 17

(BUFFER)   = 5-byte buffer:
    Byte 0    = device number
    Bytes 1-4 = offset and segment of routine to be called

EXIT
----
(FUNC_RETC) = 0 device invalid or not installed
            = FF successful
```

FUNCTION 18 - CANCEL RECEIVE CHARACTER INTERRUPT

This function cancels any previous function 17 calls.

```
ENTRY
-----
(FUNCTION) = 18

EXIT
----
None
```

FUNCTION 19 - TRANSMIT BUFFER BECOMING EMPTY INTERRUPT

This function permits a user's subroutine to be called upon each transmit-buffer-empty interrupt. The call is made with a far call, and the user's routine must exit with a far return. The routine is called after the interrupt is serviced and reenabled. If a second interrupt should occur before the routine has finished, it is not called the second time.

All registers should be preserved by the called routine, so allow enough stack space for this.

The routine should not call any DOS functions.

```
ENTRY
-----
(FUNCTION) = 19

(BUFFER)    = 5-byte buffer:
    Byte 0     = device number
    Bytes 1-4 = offset and segment routine to be called

EXIT
----
(FUNC_RETC) = 0  device invalid or not installed
            = FF successful
```

FUNCTION 20 - CANCEL TRANSMIT BUFFER BECOMING EMPTY INTERRUPT

This function cancels any previous Function 19 calls.

```
ENTRY
-----
(FUNCTION) = 20

EXIT
----
None
```

FUNCTION 21 - SET USER DEFINED INTERRUPT SERVICE ROUTINE

This function redefines the interrupt service routine for a given device. When an interrupt occurs for the device, the application service routine is called instead of the normally defined interrupt service routine. Use this function only under unusual circumstances, such as for asynchronous protocols, and then only if you are familiar with the operation and programming requirements of the MPSC.

Register AX on entry to the call contains twice the value of MPSC register 2B. The latter contains one of eight vector values according to the source of the interrupt.

All registers should be preserved by the called routine. Allow enough stack space for this.

The routine should not call any DOS functions.

Also, the installed driver sends an EOI instruction to the MPSC so the called routine need not send this instruction.

```
ENTRY
-----
BX         = (Contents of MPSC Register 2B) * 2
(FUNCTION) = 21

(BUFFER) = 5-byte buffer:
    Byte 0    = Device number
    Bytes 1-4 = Offset and segment of routine to be called

EXIT
----
None
```

FUNCTION 22 - RESET DEVICE INTERRUPT VECTORS

This function resets the interrupt service routine back to the system-defined service routine.

```
ENTRY
-----
(FUNCTION) = 22

EXIT
----
None
```

FUNCTION 23 - SET EXTERNAL STATUS CHANGE INTERRUPT

This function permits a user's subroutine to be called upon each external-status-change interrupt. The call is made with a far call, and the user's routine must exit with a far return.

```
ENTRY
-----
(FUNCTION) = 23

(BUFFER)   = 5-byte buffer:
    Byte 0    = device number
    Bytes 1-4 = Offset and segment of routine to be called


EXIT
----
(FUNC_RETC) = 0 Device Illegal or not installed
            = FF successfully canceled.
```

### FUNCTION 24 - CANCEL EXTERNAL STATUS CHANGE INTERRUPT

This function cancels any previous function 23 call.

```
ENTRY
-----
(FUNCTION) = 24

EXIT
----
None
```

### FUNCTION 25 - NON-DESTRUCTIVE CHARACTER READ, NO WAIT

This function returns the next available character in the device's ring-buffer, if any, but does not remove the character from the ring-buffer. This function allows the application to look ahead one input character.

```
ENTRY
-----
(FUNCTION)  = 25

EXIT
----
(FUNC_RETC) = FF if a char is available
            = 0 if no char is available

(CHARACTER) = character, if available
```

## 4.7 DISK CONTROL FUNCTIONS

All diskette IOCTL-type functions are invoked using INT 65H. This is instead of the usual MS-DOS IOCTL function 44H with INT 21H, because using function 44H would cause drive motor problems. The functions are called differently under MS-DOS Version 2.01 and Version 2.05. Both calling processes are described below to emphasize their differences.

The Winchester disk IOCTL functions are called by the MS-DOS IOCTL function 44H using INT 21H. They can be invoked only with Version 2.05, since Version 2.01 does not support Winchester drives. Since Winchester media are not removable, the media check function need not be enabled or disabled, so functions 6 - 8 are not defined for the Winchester.

## 4.8 VERSION 2.01 DISKETTE FUNCTIONS

ENTRY
-----

DS:BX = Pointer to control block:

Control Block Format

| Offset | | | |
|---|---|---|---|
| 0 | CTLBLK | DB | Function code:<br>0 Write<br>1 Read<br>2 Media check |
| 1 | | DB | Drive code (0=A, 1=B, 2=C, 3=D) |
| 2 | TRACK | DB | Track number (first is 0) |
| 3 | | DB | Sector number (first is 1) |
| 4 | | DB | Sector count (must be 1) |
| 5 | CURTRK | DB | Current track (0FFH initially) |
| 6 | | DB | Format (must be 0, RX50 media only) |
| 7-B | | DB | 0, 0, 0, 0, 0 |
| C-D | | DW | Offset of user buffer for one full sector (512 bytes) |
| E-F | | DW | Segment of user buffer (Buffer must all be in shared memory ( 00800H to 0FFFFH.) |

EXIT
----

Carry NOT SET:

| | | |
|---|---|---|
| AL = | 0 | Function successful |

Carry SET:

| | | |
|---|---|---|
| AL = | 0 | Write protected |
| | 2 | Not ready, or busy |
| | 4 | CRC error |
| | 6 | Seek error |
| | 7 | Robin media |
| | 10 | Write fault |
| | 11 | Lost data |

EXAMPLE
-------

```
           MOV   BX,offset CTLBLK
           INT   65H         ;saves only DS and ES
           MOV   AH,0FFH     ;in case of error,
                                 ;track is unknown
           JC    SETCURT
           MOV   AH,TRACK    ;otherwise, track is as requested
SETCURT:   MOV   CURTRK,AH   ;set correctly before next call
```

## 4.9 VERSION 2.05 DISKETTE FUNCTIONS

```
ENTRY
-----
DS:BX = Pointer to control block DIFFERENT from 2.01:

     Control Block Format

Offset

  0  FCTLBL  DB  Function code:
                   0  Read
                   1  Write
                   2  Write with verify
                   3  Format track
                   4  Media check
                   5  Verify disk
                   6  Enable media check function
                   7  Disable media check function
                   8  Status of media check function
  1     DB  Drive code (0=A, 1=B, 2=C, 3=D,
                     0FFH=physical unit)
  2     DB  Sector # (one-based, skewed if track > 1)
  3     DB  Physical unit ( only if drive code = 0FFH)
              High nibble = unit number
                (0 to 3 for diskettes A to D)
              Low nibble  = head number
                (normally 0, for first side)
 4-5   DW  Track number (first is 0)
 6-7   DW  Sector count (always 1, ignored)
 8-9   DW  Offset of user buffer (512 bytes)
A-B    DW  Segment of user buffer (Buffer may be anywhere in
           memory, except "format track" must be in absolute
           00800H-0FFFFH.)

EXIT
----
Carry NOT SET:

   For functions 0 - 5:
      AL = 0  Function successful

   For functions 6 - 8:
      AL = 0  Media check function enabled
           1  Media check function disabled
```

```
Carry SET (functions 0 - 5 only):
      AL = 0  Write protected
           2  Not ready, or busy
           4  CRC error
           6  Seek error
           7  Robin media
          10  Write fault
          11  Lost data
```

EXAMPLE
-------

```
MOV   BX,offset FCTLBL
INT   65H       ;Functions 0 - 5 save BX, DS, ES
                ;Functions 6 - 8 save all except AX
```

## 4.10 VERSION 2.05 WINCHESTER

ENTRY
-----

```
AH = 44H
AL = Function code (must be 4 or 5)
BL = Drive (0=default, 5=E, 6=F, 7=G, 8=H)
DS:DX = Pointer to control block DIFFERENT from diskette
```

Control Block Format

Offset

```
  0  DB  Function code:
          0  Read
          1  Write
          2  Write with verify
          3  Format track
          4  Media check
          5  Verify disk (simply returns AL=5)
  1  DB  Drive code
          (4=E, 5=F, 6=G, 7=H, 0FFH=physical unit)
           Notice difference from drive value in BL
  2  DB  Sector number
          (first is 1, skewed if track > 1)
  3  DB  Physical unit ( only if drive code = 0FFH)
            High nibble = unit
               (0, has only one physical unit)
            Low nibble  = head or surface
               (0 to 3, unit has 4 heads)
            Notice that track = cylinder and is not  offset,  and
            that  sector  is not skewed, when performing physical
            unit I/O.
```

```
Offset

 4-5  DW  Track number
             (first is 0, offset from partition start)
 6-7  DW  Count of sectors (functions 0-2)
             or tracks  (function 3)
 8-9  DW  Offset of user buffer for either:
             sector data (512 bytes each for read  or  write),  or
             track  data  (for  formatting,  32 bytes per track, 2
             each for 16 sectors, first is 00H for normal  or  80H
             for bad, second is logical sector number to write)
 A-B  DW  Segment of user buffer (may be anywhere in memory)
   C  DB  Returned status from driver
   D  DB  Error type bits (= AH, valid if status is 1, 3, or 0DH)
```

```
EXIT
----
AL = Returned status:
     00 Function successful
     01 Error (during read, write, or format)
     03 Write fault at drive (or 8-second timeout)
     05 Invalid drive in BL or in WCTLBL
    0DH Write verify error (during reread after write)
    0FH Soft read error (retries were done and worked)
   0FFH Driver not called (only way to tell)

AH = Error type bits (for functions 0-3 only):
     01H Data Address Mark not found (during read)
     02H Track 0 error (cannot restore to track 0)
     04H Aborted (new command issued while busy)
     10H ID not found (or track number too large)
     40H CRC error in data field (after 8 retries)
     80H Bad block detect (sector is marked as bad)
```

```
EXAMPLE
-------
MOV   AX,4404H  ;IOCTL function for disk
MOV   BL,5      ;for drive E, first Winchester partition
MOV   DX,offset WCTLBL
INT   21H       ;saves DS, ES, SI, DI, BP
```

# CHAPTER 5

# MS-DOS PROGRAMMING NOTES

## 5.1 DIRECT ROM CALLS

### 5.1.1 IBM ROM Calls (INT 10)

- Set mode
- Set cursor type
- Set cursor position
- Read cursor position
- Select active display page
- Scroll active page up
- Scroll active page down
- Read attribute and character at current cursor position
- Write attribute and character at current cursor position
- Write character at current position

### 5.1.2 Rainbow ROM Calls (INT 18H under the MS-DOS operating system)

- Write character at current position
- Disable/enable cursor o Send data to screen

### 5.1.3 Set Mode

IBM computer
------------

40x25 black _white
80x25 black _white

RAINBOW computer
----------------

ANSI Escape Sequences

```
ESC#5   - 80 x 24
ESC#6   - 40 x 24
```

### 5.1.4 Set/Read Cursor position

RAINBOW computer
----------------

ANSI Escape Sequences

```
ESC[#;#H    Set Cursor position
ESC[#;#R    Read Cursor position
```

The # refers to a numeric value.

### 5.1.5 Paging

RAINBOW computer
----------------

The Rainbow computer does not support a paging scheme for the video. If paging is required, you must write your own support routine(s).

### 5.1.6 Read Attribute and Character

RAINBOW computer
----------------

The Rainbow computer does not support reading attribute and character. You can write a routine to read the cursor position, calculate the memory location, then return the character and attribute set.

### 5.1.7 Write Character at the Current Cursor Position

| | IBM Computer | Rainbow Computer |
|---|---|---|
| Function | AH = 10 | DI = 0 |
| Character | AL | AL |
| Page | BH | N/A |
| # Repeat Char | CX | N/A |

### 5.1.8 Write Character and Attribute

RAINBOW computer
----------------

A single system call can write multiple characters/attributes (see below).

Rainbow Fast Access to Video Memory by ROM (INT 18H)

```
DI = 14
AX = Transfer type
   0 = Character _Attribute
   1 = Attribute only
   2 = Character only
BL = Line number (1-24)
BH = Column number (1-132)
CX = Number of Char/Attr to transfer
DX = Start address of Attributes
SI = Start address of Characters
BP = Segment code
```

Table 5-1: Memory Map for Video

```
IBM computer
------------

B0000 to B3FFF  16K Character/Attribute RAM

   Even bytes are Character RAM
   Odd bytes are Attribute RAM

RAINBOW computer
----------------

EE000 to EEFFF  4K Character RAM
EF000 to EFFFF  4K Attribute RAM
```

Rainbow Video Memory Map - RAM Addresses

Direct video memory access routines start by issuing escape sequence ESC[?31 with a generic INT 21H call. After receiving this escape sequence, the video memory appears as follows for an 80 column display:

(The addresses are for the first line of the display.)

| | |
|---|---|
| EE00:12 - EE00:61 | (80 characters) |
| :62 | Termination Char (FF) |
| :63-64 | Address of next line |

Subsequent line addresses are listed below:

| | | |
|---|---|---|
| LINE | 2 | 99H |
| LINE | 3 | 120H |
| LINE | 4 | 1A7H |
| LINE | 5 | 22EH |
| LINE | 6 | 2B5H |
| LINE | 7 | 33CH |
| LINE | 8 | 3C3H |
| LINE | 9 | 44AH |
| LINE | 10 | 4D1H |
| LINE | 11 | 558H |
| LINE | 12 | 5DFH |
| LINE | 13 | 666H |
| LINE | 14 | 6EDH |
| LINE | 15 | 774H |

| | | | |
|---|---|---|---|
| LINE | 16 | 7FBH | |
| LINE | 17 | 882H | |
| LINE | 18 | 909H | |
| LINE | 19 | 990H | |
| LINE | 20 | A17H | |
| LINE | 21 | A9EH | |
| LINE | 22 | B25H | |
| LINE | 23 | BACH | |
| LINE | 24 | C33H | |
| LINE | 25 | CBAH | (Used when Scrolling) |

Table 5-2: Character Attributes

| | IBM Computer | RAINBOW Computer |
|---|---|---|
| REVERSE VIDEO | YES | YES |
| BLINK | YES | YES |
| UNDERSCORE | YES | YES |
| BOLD | YES | YES |

Table 5-3: IBM Attribute Byte Bit Mask

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Blink | r | g | b | Intensity | r | g | b |
| | Foreground | | | | Background | | |

| Background | | | Foreground | | | Function |
|---|---|---|---|---|---|---|
| r | g | b | r | g | b | |
| 0 | 0 | 0 | 0 | 0 | 0 | Non display |
| 0 | 0 | 0 | 0 | 0 | 1 | Underline |
| 0 | 0 | 0 | 1 | 1 | 1 | White character |
| 1 | 1 | 1 | 0 | 0 | 0 | Reverse video |

Table 5-4: Rainbow Attribute Byte Bit Mask

```
Bit 3 = Underscore
        0 = Underscore
        1 = No underscore

Bit 2 = Blink
        0 = Blink
        1 = No blink

Bit 1 = Bold
        0 = Bold
        1 = No bold

Bit 0 = Reverse Video
        0 = Normal
        1 = Reverse video
```

## 5.2 KEYBOARD INTERFACE (IBM COMPUTER)

### 5.2.1 Generic MS-DOS calls (INT 21H)

- Function 1 (Read Keyboard and Echo)
- Function 6 (Direct Keyboard I/O)
- Function 7 (Direct Keyboard Input)
- Function 8 (Read Keyboard)
- Function 10 (Buffer Keyboard Input)
- Function 11 (Check Keyboard Input)
- Function 12 (Flush Buffer, Read Keyboard)

NOTE

These generic calls work the same in both IBM and Rainbow systems

## 5.3 DIRECT ROM CALLS

### 5.3.1 IBM ROM Calls (INT 16)

- Keyboard Input (AH=0, character returned in AL)
- Keyboard Status (AH=1, Z-flag set = no character, character in AX) (Notice that this allows non-destructive read.)
- Return the current shift status

### 5.3.2 Rainbow ROM Calls (INT 40)

- Keyboard Input (DI=2, character returned in AL)
- Keyboard Status (DI=4, CL=0 no char / CL=FFH, AL=char)
- Level 1 Keyboard Input ("raw" key-code)

## 5.4 IBM SHIFT KEY STATUS

The Shift status is the second charter of a keyboard. Function keys and other keys generate two-byte inputs.

The Rainbow computer generates an escape sequence when you type special keys (for example, keypad) and function keys. These escape sequences must then be parsed by the application.

## 5.5 DISK INTERFACE

### 5.5.1 Generic MS-DOS Calls

All generic disk calls are equivalent in the two systems.

## 5.6 DIRECT HARDWARE/FIRMWARE/BIOS CALLS

### 5.6.1 IBM ROM Calls (INT 13)

- Reset disk system
- Read status
- Read disk sector
- Write disk sector
- Verify disk sector
- Format disk

### 5.6.2 Rainbow Floppy Diskette Calls (INT 65) and Winchester Calls (IOCTL 44)

See Chapter Four for details.

- Read sector
- Write sector
- Write and verify
- Format
- Media check
- Verify media (bad block check)

## 5.7 SAMPLE PROGRAM

The following sample program shows the use of MS-DOS system function 4BH (EXEC). "Shelling" (a term used by the UNIX and XENIX operating systems) means to invoke the operating system's command processor to load and start a program. The MS-DOS system function 4B is called "EXEC". The sample program is written for assembly by Microsoft's MASM.

```
TITLE: Exec call test
PAGE:  60,132
```

Notice that no stack segment is defined. This is because the program is linked as a .COM file that, when loaded, has all segment registers set to the same value. Notice that the program is ORGed at 100H. This is necessary for

```
code    SEGMENT 'codesg'
        ASSUME  CS:code,DS:code,ES:code

        ORG     0100H                   ; Program entry point

exectest:                               ;

        MOV     SP,OFFSET stack         ; Set up local stack

        MOV     AH,09H                  ; Print "Before shell"
        MOV     DX,OFFSET mess1         ;
        INT     21H                     ;

        MOV     BX,OFFSET lastloc+15    ; BX := program size in
        MOV     CX,4                    ;  paragraphs
        SHR     BX,CL                   ;

        MOV     AX,4A00H                ; Deallocate unused memory
        INT     21H                     ;

        MOV     SI,2CH                  ; Get environment address
        MOV     AX,CS:[SI]              ; from PSP+2CH
        MOV     WORD PTR parmblk,AX     ;

        MOV     AX,CS                   ; Set segment registers
        MOV     WORD PTR parm4,AX       ; in parameter block
        MOV     WORD PTR parm8,AX       ;
        MOV     WORD PTR parmC,AX       ;

        MOV     DX,OFFSET filenam       ; Set up exec call
        MOV     BX,OFFSET parmblk       ;
        MOV     AX,4B00H                ;

        PUSH    DS                      ; Save machine state
        PUSH    ES                      ;
        MOV     CS:savess,SS            ;
        MOV     CS:savesp,SP            ;

        INT     21H                     ; Shell to DOS

        MOV     SP,CS:savesp            ; Restore machine state
        MOV     SS,CS:savess            ;
        POP     ES                      ;
        POP     DS                      ;
        MOV     AH,09H                  ; Print "After shell"
        MOV     DX,OFFSET mess2         ;
        INT     21H                     ;

        INT     20H                     ; Terminate program

savess  DW      ?                       ; Holders for SS:SP
savesp  DW      ?                       ;

mess1   DB      'Before shell',0DH,0AH,'$'
mess2   DB      'After shell',0DH,0AH,'$'

filenam DB      'A:\COMMAND.COM',0      ; Assume COMMAND.COM on A:

parmblk DW      00                      ; Parameter block
        DW      OFFSET comline          ;
        parm4   DW      00              ;
                DW      5CH             ;
        parm8   DW      00              ;
                DW      6CH             ;
        parmC   DW      00              ;
        comline DB      09H,'/C dir A:',0DH   ; Command line

        PAGE

                DB      128  DUP (?)    ; Stack
        stack   LABEL   BYTE            ;

        lastloc LABEL   BYTE            ; End of program

        code    ENDS
                END     exectest
```

## 5.8 PROGRAMMING HINTS

Keep the following programming hints in mind when using the MS-DOS Version 2.05 operating system:

1. Serial I/O Function 14 "Set/Clear Modem Signals" does not work.

2. Serial I/O Function 21 "Program Device Interrupt Vector" does not work.

3. Data received from the communications port and printer port when the character format has been set to 7M (7 bits, 8th bit always Mark) always have the eighth bit set. The MPSC does not strip off this bit, so the application must strip it.

4. The BIOS reads IBM single-sided, 8- or 9-sectored diskettes. However, it does NOT check to determine if a diskette in the specified drive is double sided. As a result, the top side (which contains the directory and FAT) is read if you try to read a double-sided IBM diskette. This falsely indicates that all of the data can be accessed. Copy operations do not work, however, because files are stored on both sides of the double-sided diskettes, and the data on the lower side cannot be read.

5. The MS-DOS Version 2.05 system diskette contains Z80A code used for starting (BOOTing) the operating system. The computer stops if you try to hard format a write-protected diskette using the /I switch.

6. The FORMAT program on the MS-DOS Version 2.05 distribution diskette contains an error that fails to place an error code in the FAT if the last sector on the last track of a diskette contained a hard error. This is of minimal significance, since diskettes having errors in sector 10 of track 79 are probably extremely rare.

# APPENDIX A

# RAINBOW ON DISK STRUCTURES

This appendix defines the layout of all software related disk structures. It also defines the contents of all fixed data blocks necessary for supporting various system configurations and applications.

## A.1 INTRODUCTION

The on-disk structure accommodates disks having larger capacitties, and removable media.

## A.2 GOALS

1. Provide a disk structure suitable for supporting multiple operating systems, as well as multiple logical disks.

2. Be able to support removable (mountable) media.

3. Support media interchange between Rainbow and non-Rainbow computers.

## A.3 CONVENTIONS

### A.3.1 Disk Address Space

Internal pointers to disk addresses are in the form:

```
     Disk Addresses
    +----+----+----+            +----+----+
    !  track  !sect! (3-bytes)  ! length  !  (opt. 2-bytes)
    +----+----+----+            +----+----+
```

The intersection of a cylinder and surface is a track. Each track has a unique number assigned to it. The numbers are assigned sequentially starting with track 0 located at cylinder 0 - surface 0 and proceeding downward within the cylinder until the last surface is reached. The next track is the top surface of the next cylinder. The formula for computing track number from cylinder and surface numbers is:

track = surface + cylinder x (# of surfaces on disk)

For example, the track located on surface 2 of cylinder 3 on a 4-surface disk would be track 14.

Disk addresses can be viewed as monotonically increasing block numbers starting at 1 (Cylinder 0, Surface 0, Sector 1) and continuing until the end of the last usable cylinder-surface-sector. All sectors on a track are used sequentially until the last sector of that track. The next block is then the first sector of the next track. For example (for RD51):

```
     Block  1 -- Track 0 (Cylinder 0, Surface 0), Sector 1
     Block 16 -- Track 0 (Cylinder 0, Surface 0), Sector 16
     Block 17 -- Track 1 (Cylinder 0, Surface 1), Sector 1
     Block 64 -- Track 3 (Cylinder 0, Surface 3), Sector 16
     Block 65 -- Track 4 (Cylinder 1, Surface 0), Sector 1
```

### A.3.2 Checksums

Each reserved area of the on-disk structure contains a 16-bit checksum value in bytes 4 and 5. This value is computed by zeroing the checksum bytes, then performing a cumulative modulo-16 addition of the reserved area data. The data is treated as 16-bit quantities starting with bytes 0 and 1. The resulting sum is complemented and a value of 1 is added to it. This number becomes the checksum. When a reserved area is read, the checksum is verified by performing the same modulo-16 addition, including the checksum quantity. The resulting sum should be zero.

## A.4 RESERVED AREAS

Tracks 0 and 1 contain information used to configure and maintain system areas on the hard disk. This data is accessed by various utilities that need to know the extent of disk partitions and bad regions. This data is duplicated on tracks 3 and 4. This area is not usable by any operating system for their respective file structures. The information stored on these tracks is:

- PRE-BOOT - This block is to be read by new firmware. It contains a small program that reads and starts the primary boot program.
- HOM - This block contains the volume ID of the disk, a description of the physical disk layout, and pointers to the other disk system areas.
- BAT (Bad Address Table) - This area contains a bit map identifying all bad sectors on the disk.
- AST (Alternate Sector Table) - This area contains the addresses of bad sectors on the disk along with an alternate sector address for each.
- DPD (Disk Partition Data) - This area contains a description of each disk partition, its logical assignment, and operating system code.
- OSN (Operating System Name Table) - This area contains the name strings of each known operation system type code found in the DPD.
- BOOT - This area is reserved for the Winchester boot program.

Other reserved areas are:

1. PAS (Partition Alternate Sector Table) - This area contains the addresses of all bad sectors in the partition along with an alternate sector address for each. It occupies the first sector in the first track of each partition (CP/M and Concurrent CP/M only).
2. Alternate Sector Area - This area is reserved for use as alternate sectors in place of known bad sectors.
3. Maintenance Cylinder - This is the next to the last cylinder and is reserved for use by the hard disk diagnostic.
4. Manufacturing Cylinder - This is the last cylinder and is reserved for bad spot information written during the manufacturing process.

### A.4.1 Pre-Boot

This block contains the code necessary to read in the HOM block and locate the Disk Boot Program. It then reads in this program from consecutive sectors as specified by the HOM block and starts it at a fixed address (to be determined at a later time). When a new HOM block is written to the disk, this block should be initialized with a small program that displays the message: "There is no bootable system on this disk."

### A.4.2 Home Block (HOM)

This block contains the data necessary to locate all other reserved areas on the disk. It contains the Volume ID (for removable media) as well as the physical disk parameters. The home block data must fit into 128 bytes in order to be compatible with disks that have a sector size of 128 bytes. If the disk sector size is greater than 128 bytes, then the remainder of the home block must be filled with zeros to maintain a proper checksum.

| Byte | Contents |
|---|---|
| 0-2 | HOM Block Identification (ASCII characters). |
| 3 | Flag: $00 if partitioned; else $FF |
| 4-5 | Checksum [Block Checksum := 0 (mod 16-bits)] |
| 6-13 | 8-character VOL ID (Default := "RAINBOW") |
| 14-17 | System ID or serial number (Default := 0) |
| 18-22 | Disk Addresses/length of BAT or 0 if none. |
| 23-27 | Disk Addresses/length of DPD or 0 if none. |
| 28-32 | Disk Addresses/length of OSN or 0 if none. |
| 33-37 | Disk Addresses/length of BOOT or 0 if none. |
| 38-42 | Disk Addresses/length of AST or 0 if none. |
| 43-44 | Starting track number for alternate sector area. |
| 45 | Number of tracks reserved for alternate sector area. |

| Byte | Contents |
|---|---|
| 46-63 | Auto-boot parameters: |
| 46 | Auto-boot Flag: $FF if no auto-boot<br>$nn if auto-boot: nn is index into DPD entries section of DPD block |
| 47-48 | Partition Boot Track |
| 64-126 | Physical Disk Parameters: |
| 64-65 | # of Cylinders |
| 66 | Sectors/Track |
| 67-68 | Sector size (bytes) |
| 69 | # of Surfaces |
| 70-71 | Maint. Cylinder # |
| 72-73 | Mfg. Data Cylinder # |
| 74-75 | Write Pre-comp value |
| 76 | Step Rate |
| 77 | Disk Type Code, 10 = RD51; 12 = RD50 |
| 127 | Physical block # of this block (for example, 01 for HOM) |
| 128-511 | Must be zero (MBZ) |

### A.4.3 Bad Address Table (BAT)

This table contains a bit map identifying known bad sectors on the disk. The map is created by the hard disk utility program using the factory bad spot data and the diagnostic mapping data. The table is treated as an array of 16-bit words. The number of words required depends on the size of the disk. (For the RD51, 1224 words are needed.) A bit is accessed first by locating the word containing it and then by locating the bit position within the word. The bit locations are a function of the sector disk address. The word offset and bit position are computed as follows:

table value = (trk number X sectors/track) + (sector number - 1)

word offset = bits 4-23 of table value

bit position = bits 0-3 of table value

Because the entire table cannot fit in a single sector, multiple BAT blocks are required. The header of each block identifies the range of sector addresses whose bits are represented in that block's portion of the table.

| Byte | Contents |
|---|---|
| 0-2 | BAT Block Identification (ASCII Characters) |
| 3 | Logical Block Number (LBN) - Relative block number within BAT table. (That is, the first BAT block is 0, the second is 1, and so forth.) |
| 4-5 | Checksum [Block Checksum := 0 (mod 16-bits)] |
| 6-8 | Disk Address of the sector corresponding to the first bit entry in the block |
| 9-11 | Disk Address of the sector corresponding to the last bit entry in the block |
| 12-511 | BAT entries, one bit each: |

| Bit value | Meaning |
|---|---|
| 0 | Corresponding sector is good |
| 1 | Corresponding sector is bad |

### A.4.4 Alternate Sector Table (AST)

This block contains an ordered list of known bad sectors on the disk along with an alternate good sector for each. The disk utility programs assign alternate good sectors to each bad sector specified in the BAT blocks. The bad sector/alternate sector information relevant to each partition is duplicated in the PAS block located at the beginning of the partition.

| Byte | Contents |
|---|---|
| 0-2 | "AST" Block Identification (ASCII Characters) |
| 3 | Logical Block Number (LBN) - Relative block number within AST table. (That is, the first AST block is 0, the second is 1, and so forth.) |
| 4-5 | Checksum [Block Checksum := 0 (mod 16-bits)] |
| 6-7 | Maximum_Entry_Count - The maximum number of AST entries allowed in this block. (Max := 100 for 512 byte sectors.) |

Byte Contents (Cont.)

8-9 Entry_Count - The number of entries in this block, counting bad sectors with or without alternate sectors and unassigned alternate sectors.

10-11 Reserved

12-511 AST entries, 5-bytes each:

> Offset Contents
>
> 0-2 Disk Address of bad sector: 0 means associated alternate sector is unassigned.
>
> 3 Track offset of alternate good sector (from first alternate track)
>
> 4 Sector number of alternate good sector

### A.4.5 Disk Partition Data (DPD)

This area contains the data for partitioning the disk into logical, assignable areas. Areas are categorized by name and OS type. This data is constructed at sub-system installation time or whenever the user wishes to repartition the disk.

Byte Contents

0-2 "DPD" Block Identification (ASCII Characters)

3 Logical Block Number (LBN) - Relative block number within DPD table. (That is, the first DPD block is 0, the second is 1, and so forth.)

4-5 Checksum [Block Checksum := 0 (mod 16-bits)]

6-7 Maximum_Entry_Count - The maximum number of DPD entries allowed in this block. (Max := 15 for 512 byte sectors.)

8-9 Entry_Count - The number of DPD entries in this block.

10-31 Reserved

Byte Contents (Cont.)

32-511 DPD entries, 32-bytes each:

Offset Contents

0 Flag: $FF := Non-existent; $0F := not initialized;
$F0 := initialized.

1 Logical unit: $00 := Unassigned; 1-63 := Assignment within OS (For CP/M: 1 = A, 2 = B, 3 = C, and so forth.)

2-9 Partition name (ASCII Alphanumeric characters)
[Default := "DISKnn"; nn := 01, 02, etc.]

10 Partition occurrence number (i.e. 0,1,2,... for partitions of the same name).

11 OS type code: Index into OS Name table (OSN)

12-13 First track number

14-15 Last track number

16 Number of PAS blocks at the beginning of the partition (CP/M and Concurrent CP/M partitions only)

17 if Concurrent CP/M partition, directory size code:
0 = default (256 entries for <5MB partition,
512 for >5MB partition)
1 = 256 entries
2 = 512 entries
3 = 1024 entries
4 = 2048 entries

18 if MS-DOS partition, cluster size code:
0 = 2KB cluster
1 = 4KB cluster
2 = 8KB cluster
3 = 16KB cluster

19 if MS-DOS partition, number of FAT sectors (only if Version 3.0 or later of Hard Disk Utility is used)

20-31 Reserved for Backup/Restore system.

### A.4.6 Operating System Name Table (OSN)

This area contains the operating system name string table representing the operating system type codes found in the DPD. This table may be sparsely populated, therefore, it is not necessary to allocate consecutive entries. Each entry is a zero filled 16-byte text string.

| Byte | Contents |
|---|---|
| 0-2 | "OSN" Block Identification (ASCII Characters) |
| 3 | Logical Block Number (LBN) - Relative block number within OSN table. (That is, the first OSN block is 0, the second is 1, and so forth.) |
| 4-5 | Checksum [Block Checksum := 0 (mod 16-bits)] |
| 6-7 | Maximum_Entry_Count - The maximum number of OSN entries allowed in this block. (Max := 31 for 512 byte sectors.) |
| 8-15 | Reserved |
| 16-511 | OSN entries (16-bytes). Operating system name strings created by the partition utility. Name strings may consist of 1 to 16 ASCII characters excluding CONTROL, SPACE and DEL characters. |

### A.4.7 System Boot Program Area

This area contains a secondary bootstrap routine. This routine asks you what operating system and partition should be used for booting. It optionally uses the "Auto-boot" data if present in the HOM block. Whenever this area is created, the PRE-BOOT block (0) should also be written.

### A.4.8 Track Layout

For a 10M byte RD51 disk with 16 sectors of 512 bytes each per track.

Track 0:

```
        1      2      3      4      5      6      7      8
  +------+------+------+------+------+------+------+------+
  !PBOOT ! HOM  !DPD(0)!DPD(1)!OSN(0)!BAT(0)!BAT(1)!BAT(2)!
  +------+------+------+------+------+------+------+------+

        9     10     11     12     13     14     15     16
  +------+------+------+------+------+------+------+------+
  |BAT(3)|BAT(4)|AST(0)|AST(1)|AST(2)|  *   |  *   |  *   |
  +------+------+------+------+------+------+------+------+
               * = reserved
```

Track 1:

```
        1      2      3      4      5      6      7      8
  +------+------+------+------+------+------+------+------+
  !            Secondary Bootstrap Program                !
  +------+------+------+------+------+------+------+------+

        9     10     11     12     13     14     15     16
  +------+------+------+------+------+------+------+------+
  !            Secondary Bootstrap Program                !
  +------+------+------+------+------+------+------+------+
```

| | |
|---|---|
| Track 2: | same as track 0 |
| Track 3: | same as track 1 |
| Tracks 4-19: | Alternate sectors |
| Tracks 20-1215: | Partitions |
| Tracks 1216-1219: | Maintenance tracks |
| Tracks 1220-1223: | Manufacturing tracks |

For a 5M byte RD50 disk with 16 sectors of 512 bytes each per track.

Track 0:

```
       1      2      3      4      5      6      7      8
+------+------+------+------+------+------+------+------+
!PBOOT ! HOM  !DPD(0)!DPD(1)!OSN(0)!BAT(0)!BAT(1)!BAT(2)!
+------+------+------+------+------+------+------+------+

       9     10     11     12     13     14     15     16
+------+------+------+------+------+------+------+------+
|AST(0)|AST(1)|  *   |  *   |  *   |  *   |  *   |  *   |
+------+------+------+------+------+------+------+------+
             * = reserved
```

Track 1:

```
       1      2      3      4      5      6      7      8
+------+------+------+------+------+------+------+------+
!            Secondary Bootstrap Program                !
+------+------+------+------+------+------+------+------+

       9     10     11     12     13     14     15     16
+------+------+------+------+------+------+------+------+
!            Secondary Bootstrap Program                !
+------+------+------+------+------+------+------+------+
```

Track 2:          same as track 0

Track 3:          same as track 1

Tracks 4-12:      Alternate sectors

Tracks 20-603:    Partitions

Tracks 604-607:   Maintenance tracks

Tracks 608-611:   Manufacturing tracks

### A.4.9 Partition Alternate Sector Table (PAS)

This block contains an ordered list of known bad sectors within the partition along with an alternate good sector for each. This information is a subset of that which is contained in the AST block. It is provided for use by the operating system to prevent use of known bad sectors. It is located in the first sector on the first track of a CP/M or Concurrent CP/M partition. There is no PAS block for MS-DOS partitions.

Byte Contents

0-2 "PAS" Block Identification (ASCII Characters)

3 Logical Block Number (LBN) - Relative block number within a possible sequence of PAS blocks (That is, the first PAS block is 0, the second is 1, and so forth.)

4-5 Checksum [Block Checksum := 0 (mod 16-bits)]

6-7 Maximum_Entry_Count - The maximum number of PAS entries allowed in this block. (Max := 100 for 512 byte sectors.)

8-9 Entry_Count - The number of PAS entries in this block.

10-11 Reserved

12-511 PAS entries, 5-bytes each:

> Offset Contents
>
> 0-2 Disk Addresses of bad sector
>
> 3 Track offset of alternate good sector (from first alternate track)
>
> 4 Sector number of alternate good sector

### A.4.10 Operating System Bootstraps

Individual operating system bootstraps reside on reserved areas within their respective partitions, starting at the first sector after the PAS block in the partition.

# APPENDIX B

# MICROSOFT ARTICLES FOR PROGRAMMERS AND OEMS

This appendix includes articles written by Microsoft Corporation. Generally, they cover the same information covered in the MS-DOS "Programmers Reference Manual." They are included since they provide additional, useful information.

## B.1 CONFIGURATION FILES IN MS-DOS VERSION 2.0

The following are a list of commands for the configuration file CONFIG.SYS:

BUFFERS = <number>

> This is the number of additional sector buffers to add to the system list. The effect of several BUFFERS commands is to allocate a series of buffers.

FILES = <number>

> This is the number of open files that the XENIX system calls can access.

DEVICE = <filename>

> This installs the device driver in <filename> into the system list.

BREAK = <ON or OFF>

> If ON is specified (the default is OFF), a check for ^C at the console input is made every time the system is called. ON improves the ability to abort programs over previous versions of the DOS.

SWITCHAR = <char>

Causes the DOS to return <char> as the current switch designator character when the DOS call to return the switch character is made. Default is '/'.

AVAILDEV = <true or false>

The default is TRUE, which means both \dev\<dev> and <dev> reference the device <dev>. If FALSE is selected, only \dev\<dev> refers to device <dev>, <dev> by itself means a file in the current directory with the same name as one of the devices.

SHELL = <filename>

This begins execution of the shell (top-level command processor) from <filename>. Used when COMMAND.COM is not in the current directory.

A typical configuration file might look like this:

```
BUFFERS = 10
FILES = 10
DEVICE = \bin\network.sys
BREAK = ON
SWITCHAR = -
SHELL = a:\bin\COMMAND.COM a:\bin /p
```

The default value for BUFFERS is OEM specific in that the OEM can specify the number in the BIOS. A typical value is 2, the minimal value is one. (The Rainbow's default is 4). The default value for FILES is usually 8 (as above, it may be set by OEM BIOS) , so "FILES = 10" actually allocates only 2 new file channels. If a number less than, or equal to, five is specified, the command is ignored. (The Rainbow's default is 8). BREAK defaults to OFF, SWITCHAR to /, and AVAILDEV to TRUE. Notice that the setting of SWITCHAR may effect characters used on the SHELL line (this is true of COMMAND.COM).

## B.2 VERSION INCOMPATIBILITIES

Areas where Version 2.0 is not compatible with previous versions of the DOS.

- Direct calls to the BIOS

  Any program which jumped directly to the BIOS by way of the jump table at 40:0 no longer works.

- FAT pointer calls

  Programs that used system calls 27 and 28 to get a pointer to the FAT no longer work. Because the FAT is now cached with other disk resources, there is no fixed location in memory to pass the address to. The calls still exist, however, and have the same format. THEY CAN ONLY BE USED TO GET THE FAT ID BYTE. On return ES:BX points to a FAT ID BYTE for the Drive. Doing anything except READing this ONE byte will probably crash the system. In order to get at the FAT, programs first call DSKRESET (call 13) to flush out any dirty buffers, and then make a GETDPB call (call 31 or 50) to find out which sector on the disk the FAT starts at, how big it is, and how many copies of it there are. Then INT 25H and INT 26H can be used to transfer the FAT in and out of the programs memory space.

- INT 25H and INT 26H

  In order for the above to work, and in order to maintain some order in the world of multi-surface disks, it is required that INT 25H and 26H use the MS-DOS sector mapping rather than some rather arbitrary head-cylinder-sector mapping.

The following subroutine reads the FAT into the area of memory specified by DS:BX. DL contains the drive number, DL=0 means read the FAT from the default drive, DL=1 indicates read from drive A:, and so on.

```
getfat:
        push    bx              ; save pointer to FAT area
        push    ds
        mov     ah,50           ; request the dpb
        int     21h
        mov     cx,[bx+15]      ; get FAT sector count
        mov     dx,[bx+6]       ; first sector of FAT
        pop     ds              ; restore FAT area pointer
        pop     bx
        mov     al,dl           ; is it the default drive?
        or      al,al
        jnz     driveok         ; if not, load FAT

        mov     ah,19h          ; ask for default drive
        int     21h             ; get the default drive
        inc     al              ; map a=0 to a=1
driveok:
        dec     al              ; map a=1 to a=0
        int     25h             ; read the FAT into DS:BX
        pop     ax              ; clean up the stack
        ret
```

## B.3 DIFFERENCES AND ADDITIONS TO INT 24H HARD ERROR HANDLER(S)

For MS-DOS 2.0

Additional Constraints:

Under previous versions it was not explicitly stated that an INT 24H handler must preserve the ES register. It is now required that INT 24H handlers preserve ES.

When it is desired to ignore an error, the same registers must be preserved as when it is desired to retry the operation (SS,SP,DS,BX,CX,DX).

It was not clearly stated in the past, but it was true that only system calls 1-12 can be made by an INT 24H handler. Making any other calls destroys the DOS stack and its ability to retry or ignore an error.

INT 24H handlers should always return to the DOS on a retry, ignore, or abort. Failure to return to the DOS leaves the DOS in an unstable state until a non 1-12 function call is made.

Additional features:

Character device errors are now handled by the INT 24H mechanism. Previously only disk I/O errors were handled by the INT 24H handler. Additional information is now passed to the INT 24H handler in the BP and SI registers (which need not be preserved).

BP:SI is a DWORD pointer to the Device Header of the device causing the error. Information can be gotten from this header to determine whether or not the device is a block or character device; if the device is a character device, the name of the device can also be obtained. The DEVICE-DRIVERS document for 2.0 contains the definition of this header format.

NOTE

> AL (drive number for Disk errors) is indeterminate on character device errors. Bit 7 of AH is always 1 for character device errors. Previously bit 7 was 1 only in the case of a bad memory image of the FAT.

LIST OF INT 24H ERROR CODES PASSED IN DI

| | | |
|---|---|---|
| | 0 | Write protect violation |
| * | 1 | Unknown unit |
| | 2 | Drive not ready |
| * | 3 | Unknown command |
| | 4 | CRC error |
| * | 5 | Bad drive request structure length |
| * | 6 | Seek error |
| * | 7 | Unknown media |
| | 8 | Sector not found |
| * | 9 | Printer out of paper |
| | A | Write fault |
| * | B | Read fault |
| | C | General failure |

* Denotes New Function

As mentioned above BP:SI points to the device header:

```
    BP:SI->
           +--------------------------------------+
           | DWORD Pointer to next device         |
           |  (-1 if last device)                 |
           +--------------------------------------+
           | WORD Attributes                      |
           |  Bit 15 = 1 if char device, 0 if blk |
           |  if bit 15 is 1                      |
           |      Bit 0 = 1 if Current sti device |
           |      Bit 1 = 1 if Current sto output |
           |      Bit 2 = 1 if Current NUL device |
           |      Bit 3 = 1 if Current CLOCK dev  |
           |  Bit 14 is the IOCTL bit (see below) |
           |  Bit 13 is the NON IBM FORMAT bit    |
           +--------------------------------------+
           | WORD Pointer to Device strategy      |
           |      entry point                     |
           +--------------------------------------+
           | WORD Pointer to Device interrupt     |
           |      entry point                     |
           +--------------------------------------+
           | 8-BYTE character device name field   |
           | Character devices set a device name  |
           | For block devices the first byte is  |
           | The number of units                  |
           +--------------------------------------+
```

To tell if the error occurred on a block or character device you must look at bit 15 in the attribute field (WORD at BP:SI+4).

If the name of the character device is desired, look at the eight bytes starting at BP:SI+10.

## B.4 MS-DOS 2.0 UTILITY EXTENSIONS

The following notation is used below:

|  |  |
|---|---|
| [item] | item is optional |
| item* | item is repeated 0 or more times |
| item+ | item is repeated 1 or more times |
| {item1 \| item2} | item1 is present or item 2 is present, but not both |
| <object> | indicates a syntactic variable |

BATCH COMMAND invocation

COMMAND [[<drive>:]<path>] [<CTTYDEV>] [/D] [/P] [/C <string>]

/P If present, COMMAND is permanent, otherwise this is a transient command.

/D If present, COMMAND does not prompt for DATE and TIME when it comes up.

drive: Specifies device where command looks for COMMAND.COM current default drive if absent.

<path> Specifies a directory on device drive: root directory if absent.

<CTTYDEV> Name of the CTTY device. /DEV/CON if absent and command is permanent. The /DEV/ may be left off if AVAILDEV is TRUE (see AVAILDEV above).

/C <string> If present, /C must be the last switch. This causes COMMAND to try to execute the string as if the user had typed it at the standard input. COMMAND executes this single command string then exits. If the /P switch is present it is ignored (can't have a single command, permanent COMMAND).

NOTE

ALL of the text on the command line after the /C is just passed on. It is not processed for more arguments, this is why /C must be last.

Redirection of standard input/standard output.

Programs that read from the keyboard and write to the screen are said to be doing I/O to the standard input and standard output. Using any of the following results in I/O to these standard devices:

Writing to default handles 1 / read from default handle 0.

Doing byte I/O using system calls 1, 2, 6-12.

These standard devices may be redirected to/from files by the following in command line arguments:

> <filename>
  causes <filename> to be created (or truncated to zero length) and then assigns standard output to that file. All output from the command is placed in the file.

< <filename>
  causes standard input to be assigned to <filename>. All input to the command comes from this file. If end-of-file is reached, then system calls 1, 2, 6-12 returns ^Z, while reading from handle 0 returns zero characters.

>> <filename>
  causes <filename> to be opened (created if necessary) and positions the write pointer at the end of the file so that all output is appended to the file.

Notice that the above does not appear in the command line that the program being invoked sees.

Examples:

```
DIR *.ASM >FOO.LST
```

Sends the output of the dir command to the file FOO.LST.

```
FOR %O IN (*.ASM) DO MASM %O; >>ERRS.LST
```

Sends all error output from assembling every .ASM file into the file ERRS.LST.

Piping of standard I/O

It is often useful for the output of one program to be sent as input to another program. A typical case is a program that produces columnar output that must later be sorted.

The pipe feature allows this to occur naturally as the programs do all of their I/O to the standard devices.

For example, if we had a program SORT that read all of it's standard input, sorted it and then wrote it to the standard output, we could get a sorted directory listing as follows:

```
DIR | SORT
```

The | would cause all standard output generated by the left-hand command to be sent to the standard input of the right-hand command.

If we wanted the sorted directory to be sent to a file, we type:

```
DIR | SORT >FILE
```

and away it goes.

The piping feature is implemented as sequential execution of the procedures with redirection to and from temporary files. In the example above, the following would be an exact equivalent:

```
DIR >\tmp\std1

SORT >\tmp\std1 >FILE
```

The pipe is not a real pipe but rather a quasi-pipe that uses temporary files to hold the input and output as it sequentially executes the elements of the pipe. These files are created in the current directory, of the current drive and have the form %PIPEx%.$$$, where x is 1 or 2. This means that any program that runs in the pipe must be sure to restore the current directory and drive, if it has changed them, or the pipe files are lost.

COMMAND extensions

BREAK [{ON | OFF}]

"BREAK ON" turns on the Control C check in the DOS function dispatcher. "BREAK OFF" turns it off. If no argument is given the setting of BREAK is printed to the standard output in the form:

```
BREAK is xxx
```

Where xxx is "on" or "off".

CHDIR [{<drive>: | <path>}]

Change directory, or print current directory. If no argument is given, the current directory on the default drive is printed. If drive: alone is given, the current directory of drive is printed. Otherwise the current directory is set to path.

"CD" is accepted as an abbreviation.

CLS

Clear screen causes the ANSI escape sequence ESC[2J to be sent to standard output.

COMMAND internal commands take path arguments.

DIR <path>

COPY <path> <path>

DEL(ERASE) <path>

If the path is a dir, all files in that dir are deleted.

NOTE

The "Are you sure (Y/N)" prompt for DEL and ERASE now uses buffered standard input, so users must type a return after their answer. This gives them the chance to correct if they type 'y' by mistake.

TYPE <path> (must specify a file)

CTTY \DEV\dev - Change console TTY. For instance:

CTTY \DEV\AUX

Would move all command I/O to the AUX port.

CTTY \DEV\CON

Would move it back to the normal device. The \dev\ prefix may be left off if AVAILDEV is TRUE (see configuration-file doc).

ECHO [{ON | OFF | <message>}]

Normally, commands in a BATCH file are echoed onto the standard output as they are seen by COMMAND. ECHO OFF turns off this feature. ECHO ON turns echoing back on. If ON or OFF is not specified and there is text following the command, that text (a message) is echoed to standard output. If there are no arguments at all, the current setting of echo (on or off) is echoed to the standard output in the form:

ECHO is xxx

Where xxx is "on" or "off".

EXIT

For COMMANDs run without the P switch, this causes COMMAND to return. For a normal COMMAND it causes a return to itself.

GOTO <label>

Causes commands to be taken from the batch file beginning with the line after the <label> definition. If no label has been defined, the current batch file terminates.

Example:

```
:foo
REM looping...
GOTO foo
```

produces an infinite sequence of messages:

```
REM looping...'
```

NOTE

Labels are case insensitive, :FOO == :foo == :Foo

IF <condition> <command>

where <condition> is one of the following:

ERRORLEVEL <number>

true if and only if the previous program EXECed by COMMAND had an exit code of <number> or higher.

<string1> == <string2>

true if and only if <string1> and <string2> are identical after parameter substitution. Strings may not have embedded delimiters.

EXIST <filename>

true if and only if <filename> exists.

NOT <condition>

true if and only if <condition> is false.

The IF statement allows conditional execution of commands. When the <condition> is true, then the <command> is executed otherwise, the <command> is skipped.

Examples:

```
IF not exist \tmp\foo ECHO Can't find file \tmp\foo

IF $1x == x ECHO Need at least one parameter

IF NOT ERRORLEVEL 3 LINK $1,,;
```

```
FOR %%<c> IN <set> DO <command>
```

<c> can be any character but 0,1,2,3,..,9 (so there is no confusion with the %0 - %9 batch parameters).

<set> is ( <item> * )

The %%<c> variable is sequentially set to each member of <set> and then <command> is evaluated. If a member of <set> is an expression involving * and/or ?, then the variable is set to each matching pattern from disk. In this case only one such <item> may be in the set, any <item>s after the first are ignored.

Example:

```
FOR %%f IN ( *.ASM ) DO MASM %%f;

for %%f in (FOO BAR BLECH) do REM %%f to you
```

NOTE

> The '%%' is needed so that after Batch parameter (%0 - %9) processing is done, there is one '%' left. If only '%f' were there, the batch parameter processor would see the '%' then look at 'f', decide that '%f' was an error (bad parameter reference) and throw out the '%f' so that FOR would never see it. If the FOR is NOT in a batch file, then only ONE '%' should be used.

:<label>

This is essentially a no-op. It defines a label in the batch file for a subsequent GOTO. It may also be used to put comment lines in batch files since all lines that start with ':' are ignored.

MKDIR <path> - Make a directory.

"MD" is accepted as an abbreviation.

PATH [<path>{;<path>} *]

Set command search paths. This allows users to set directories that should be searched for external commands after a search of the current directory is made. The default value is NO PATH. In addition there are two special cases: PATH with no arguments prints the current path. Path with the single argument ';' (i.e., "PATH ;") sets the NUL path (no directories other than the current one searched). If no argument is given, the current value of PATH is printed to the standard output in the form:

```
PATH=text of path
  or
No path
```

PROMPT [<prompt-text>]

Set the system prompt. MS-DOS prompts are now user settable, all of the text on the command line is taken to be the new prompt. If no text is present the prompt is set to the default prompt. There are meta strings for various special prompts. These are of the form '$c' where c is one of the following:

$ - The '$' character.

t - The time.

d - The date.

p - The current directory of the default drive.

v - The version number.

n - The default drive.

g - The '>' character.

l - The '<' character.

b - The '|' character.

s - The ' ' character.

e - The ESC character.

_ - A CR LF sequence.

EXAMPLE:

```
PROMPT $n>
```

Would set the normal MS-DOS prompt.

```
PROMPT Time = $t$_Date = $d
```

Would set a two line prompt which printed

```
Time = (current time)

Date = (current date)
```

For '$c' sequences, lower case = upper case, and any character not on the above list is mapped to nothing.

RMDIR <path> - Remove a directory.

"RD" is accepted as an abbreviation.

The directory must be empty except for '.' and '..'.

<path> - A standard XENIX style path with the optional addition of a drive spec:

```
A:\FOO\BAR Full path

\FOO\BAR Full path, current drive

FOO\BAR Current dir relative

A:FOO\BAR  "     "     "
```

SET (ENVNAME)=(ENVTEXT)

Set environment strings.

This command inserts strings in COMMAND's environment. For instance:

```
SET PROMPT=$n>
```

Duplicates the function of the PROMPT command.

```
SET PATH=p1;p2
```

Duplicates the function of the PATH command.

SET foo=bar

Puts the string FOO=bar into the environment (notice the case mapping of (ENVNAME)).

NOTE

Environments are very flexible, and almost anything can be put into the environment with the SET command; the only requirement is that a single '=' be present in the string.

SHIFT

Currently, command files are limited to handling 10 parameters: %0 through %9. To allow access to more than these, the command SHIFT performs a 'pop' of the command line parameters:

```
if  %0 = "foo
    %1 = "bar"
    %2 = "blech"
    %3...%9  are empty
```

then a SHIFT results in the following:

```
    %0 = "bar"
    %1 = "blech"
    %2...%9 are empty
```

If there are more than 10 parameters given on a command line, those that appear after the 10th (%9) are shifted one at a time into %9 by successive shifts.

VER

Prints DOS version number.

VERIFY [{ON | OFF}]

Select/Deselect verify after write mode. This supplements the V switch to the COPY command. Once turned ON, it stays on until some program changes it (via the set verify system call) or the VERIFY OFF command is given. If no argument is given, the current setting of VERIFY is printed to the standard output in the form:

```
VERIFY is xxx
```

Where xxx is "on" or "off".

VOL [<drive>:]

Prints the volume ID of the disk in drive:. No drive: it does the default drive.

# INDEX

Rainbow™
MS™–DOS V2.05
Programmer's Guide
QV068–GZ

## READER'S COMMENTS

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

______________________________

______________________________

______________________________

______________________________

______________________________

Did you find errors in this manual? If so, specify the error and the page number.

______________________________

______________________________

______________________________

______________________________

______________________________

Please indicate the type of reader that you most nearly represent.

□ First-time computer user
□ Experienced computer user
□ Application package user
□ Programmer
□ Other (please specify)______________________________

Name______________________________

Date______________________________

Organization______________________________

Street______________________________

City______________________________

State______________________________

Zip Code
or Country______________________________

------Do Not Tear - Fold Here and Tape------

digital

No Postage
Necessary
if Mailed in the
United States

**BUSINESS REPLY MAIL**
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

**SOFTWARE PUBLICATIONS**
200 FOREST STREET MRO1-2/L12
MARLBOROUGH, MA 01752

----- Do Not Tear - Fold Here and Tape------

Cut Along Dotted Line

# Rainbow™

# MS™-DOS V2.05
# BIOS Listings

digital equipment corporation

**First Printing, November 1984**

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

Printed in U.S.A.

```
Warning: No STACK segment

 Start  Stop   Length  Name                   Class
 00000H 0255CH 255DH  CODE                   CODE
 02560H 02D53H 07F4H  SYSINITSEG             SYSTEM_INIT

 Origin   Group
 0000:0   CGROUP
```

```
  Address         Publics by Name

 0000:10F9       AUX
 0000:10E9       AUX2
 0000:0081       AUX2DEV
 0256:07E0       BADCOM
 0256:07CE       BADLD
 0256:0786       BADOPM
 0256:07AD       BADSIZ
 0000:2236       BIOSINIT
 0000:1E8E       BRKOFF
 0000:1E7B       BRKON
 0000:2036       BUFFER
 0256:0012       BUFFERS
 0000:1105       CLK
 0000:1348       CLKISR
 0000:1323       CLK_16_20
 0000:1322       CLK_60_50
 0000:1325       CLK_ADJ
 0000:0CFE       CLTPN
 0000:0594       CMDSTR
 0000:110B       CON
 0256:07AA       CRLFM
 0256:0005       CURRENT_DOS_LOCATION
 0000:1603       DATBCOM
 0000:1603       DATBPRT
 0000:0093       DECDISK
 0256:0011       DEFAULT_DRIVE
 0256:000B       DEVICE_LIST
 0000:0015       DEVLST
 0000:1C62       DFLT7201
 0000:1CA4       DFLTBUF
 0000:159D       DFLTCOM
 0000:15AE       DFLTPRT
 0000:15BF       DFLTXCOM
 0000:2433       DMAADR
 0000:0294       DOS20VEC
 0000:0298       DOS21VEC
 0000:029C       DOS24VEC
 0000:02A0       DOS25VEC
 0000:02A4       DOS26VEC
 0000:02A8       DOS27VEC
 0000:0004       DOS_POINTER
 0000:242F       DRVNO
 0000:05BE       DSK
 0000:005D       DSKDEV
 0000:0418       DSKTBL
 0000:05BD       DSKTMR
 0000:0DC8       DSK_INIT
 0000:0932       DSK_INT
 0000:2236       ENDBIOS
 0000:2435       EXSTAT
 0000:11C3       FASTCON
 0256:0013       FILES
 0256:0009       FINAL_DOS_LOCATION
 0000:242D       FNCCOD
 0000:1D4B       GETCHAR
 0000:0A85       GETFTN
 0000:0A83       GETFTNO
 0000:0A93       GETPEA
 0000:1F1B       HACK7201
```

```
0000:0079       HDRIVES
0000:08DE       HDSKO
0000:008F       HDSKDEV
0000:0D9B       HD_INIT
0000:0BEE       HFORMAT
0000:0C5F       HINIT
0000:0A9D       HIOCTL
0000:0C34       HMCHK
0000:0DDD       HNFMT
0000:0B41       HREAD
0000:0C38       HVDISK
0000:13EF       HVECTOR
0000:0B8E       HWRITE
0000:0B87       HWRITEV
0000:242B       I88PKT
0000:1D11       INCHAR
0000:1FBA       INITCOM
0000:040F       INITTBL
0000:092A       INI_TAB
0000:1CFD       INSTAT
0000:02AD       INT20
0000:02D4       INT21
0000:02FB       INT24
0000:0322       INT25
0000:0349       INT26
0000:0370       INT27
0000:0D13       INTHDL
0000:13B8       INTRET
0000:2446       IOINIT
0000:0B36       MAXTRK
0000:08CD       MC_FLG
0256:000F       MEMORY_SIZE
0000:0004  Abs  MNPARTS
0000:0E7D       NDRV
0000:2432       NSECT
0000:15D0       NVMCOM
0000:15E1       NVMPRT
0000:15F2       NVMXCOM
0000:1DB0       OUTCHAR
0000:13B9       OUTHEX
0000:1DEC       OUTNOW
0000:1D92       OUTSTAT
0000:242D       PACKET
0000:2429       PACKET_ADR
0000:241D       PACKET_BASE
0000:0A1D       PARTITION
0000:0006  Abs  PART_SIZE
0000:13FF       PCCOM
0000:1489       PCPRT
0000:1513       PCXCOM
0000:0100       PHYSDISK
0000:0B39       PRECOMP
0000:19C1       PRG7201
0000:10F1       PRN
0000:1613       PRTBAUD
0000:160B       PRTYCOM
0000:160B       PRTYPRT
0000:13E2       PSTR
0000:0011       PTRSAVE
0000:1DE2       PUTCHAR
0000:040D       R100
0000:1ECA       RCVCANCEL
```

```
0000:1CE6       RCVDIS
0000:1CD5       RCVENA
0000:1EA7       RCVINT
0000:1E33       RDMODEM
0000:163D       RDNVMCOM
0000:1CB5       RDSETUP
0000:0A35       RDXLT
0000:0396       RE_INIT
0000:0E7E       RHOME
0000:16EB       RUPT7201
0000:2430       SECNO
0000:1E60       SETMODEM
0000:1F8D       STATCHG
0000:242E       STATUS
0000:1FB0       STCANCEL
0000:0B38       STEPRATE
0000:0006       STRATEGY
0256:0000       SYSINIT
0256:07F4       SYSSIZE
0000:108F       TEMP_SEC
0000:108D       TEMP_TRK
0000:2421       TFORMAT
0000:1321       TICKER
0000:2431       TRACKN
0000:241D       TTRACK
0000:0FF5       UP_BAT
0000:1F80       VECT7201
0000:161B       XBAUD1
0000:162C       XBAUD2
0000:2429       XFRPKT
0000:0B40       XLT_F
0000:1F04       XMTCANCEL
0000:1ED4       XMTMTY
0000:040E       XOPTION
0000:16F5       XRUPT7201
0000:021A       Z80ISR
```

```
 Address         Publics by Value

0000:0004  Abs  MNPARTS
0000:0004       DOS_POINTER
0000:0008  Abs  PART_SIZE
0000:0008       STRATEGY
0000:0011       PTRSAVE
0000:0015       DEVLST
0000:005D       DSKDEV
0000:006F       HDSKDEV
0000:0079       HDRIVES
0000:0081       AUX2DEV
0000:0083       DECDISK
0000:0100       PHYSDISK
0000:021A       Z80ISR
0000:0294       DOS20VEC
0000:0298       DOS21VEC
0000:029C       DOS24VEC
0000:02A0       DOS25VEC
0000:02A4       DOS26VEC
0000:02A8       DOS27VEC
0000:02AD       INT20
0000:02D4       INT21
0000:02FB       INT24
0000:0322       INT25
0000:0349       INT26
0000:0370       INT27
0000:0398       RE_INIT
0000:040D       R100
0000:040E       XOPTION
0000:040F       INITTBL
0000:0418       DSKTBL
0000:0594       CMDSTR
0000:05BD       DSKTMR
0000:05BE       DSK
0000:08CD       MC_FLG
0000:08DE       HDSK0
0000:092A       INI_TAB
0000:0932       DSK_INT
0000:0A1D       PARTITION
0000:0A35       RDXLT
0000:0A83       GETFTNO
0000:0A85       GETFTN
0000:0A93       GETPEA
0000:0A9D       HIOCTL
0000:0B36       MAXTRK
0000:0B38       STEPRATE
0000:0B39       PRECOMP
0000:0B40       XLT_F
0000:0B41       HREAD
0000:0B87       HWRITEV
0000:0B8E       HWRITE
0000:0BEE       HFORMAT
0000:0C34       HMCHK
0000:0C38       HVDISK
0000:0C5F       HINIT
0000:0CFE       CLTPN
0000:0D13       INTHDL
0000:0D9B       HD_INIT
0000:0DC8       DSK_INIT
0000:0DDD       HNFMT
```

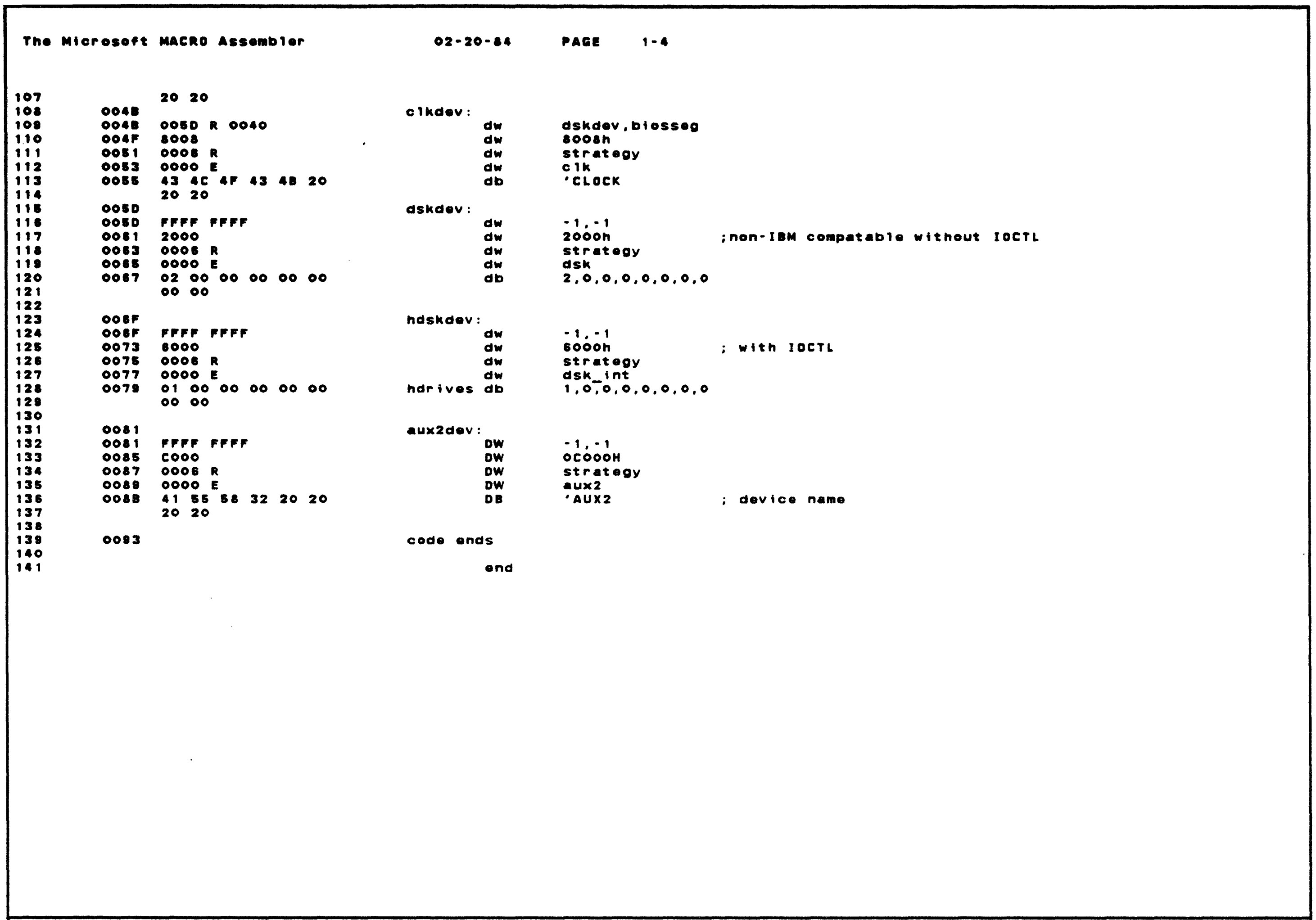
```
0000:0E7D       NDRV
0000:0E7E       RHOME
0000:0FF5       UP_BAT
0000:108D       TEMP_TRK
0000:108F       TEMP_SEC
0000:10E9       AUX2
0000:10F1       PRN
0000:10F9       AUX
0000:1105       CLK
0000:110B       CON
0000:11C3       FASTCON
0000:1321       TICKER
0000:1322       CLK_60_50
0000:1323       CLK_16_20
0000:1325       CLK_ADJ
0000:1348       CLKISR
0000:13B8       INTRET
0000:13B9       OUTHEX
0000:13E2       PSTR
0000:13EF       HVECTOR
0000:13FF       PCCOM
0000:1489       PCPRT
0000:1513       PCXCOM
0000:159D       DFLTCOM
0000:15AE       DFLTPRT
0000:15BF       DFLTXCOM
0000:15D0       NVMCOM
0000:15E1       NVMPRT
0000:15F2       NVMXCOM
0000:1603       DATBCOM
0000:1603       DATBPRT
0000:160B       PRTYPRT
0000:160B       PRTYCOM
0000:1613       PRTBAUD
0000:161B       XBAUD1
0000:162C       XBAUD2
0000:163D       RDNVMCOM
0000:16EB       RUPT7201
0000:16F5       XRUPT7201
0000:19C1       PRG7201
0000:1C62       DFLT7201
0000:1CA4       DFLTBUF
0000:1CB5       RDSETUP
0000:1CD5       RCVENA
0000:1CE6       RCVDIS
0000:1CFD       INSTAT
0000:1D11       INCHAR
0000:1D4B       GETCHAR
0000:1D92       OUTSTAT
0000:1DB0       OUTCHAR
0000:1DE2       PUTCHAR
0000:1DEC       OUTNOW
0000:1E33       RDMODEM
0000:1E60       SETMODEM
0000:1E7B       BRKON
0000:1E8E       BRKOFF
0000:1EA7       RCVINT
0000:1ECA       RCVCANCEL
0000:1ED4       XMTMTY
0000:1F04       XMTCANCEL
0000:1F1B       HACK7201
0000:1F60       VECT7201
```

```
0000:1F8D         STATCHG
0000:1FB0         STCANCEL
0000:1FBA         INITCOM
0000:2036         BUFFER
0000:2236         BIOSINIT
0000:2236         ENDBIOS
0000:241D         PACKET_BASE
0000:241D         TTRACK
0000:2421         TFORMAT
0000:2429         XFRPKT
0000:2429         PACKET_ADR
0000:242B         I88PKT
0000:242D         FNCCOD
0000:242D         PACKET
0000:242E         STATUS
0000:242F         DRVNO
0000:2430         SECNO
0000:2431         TRACKN
0000:2432         NSECT
0000:2433         DMAADR
0000:2435         EXSTAT
0000:2446         IOINIT
0256:0000         SYSINIT
0256:0005         CURRENT_DOS_LOCATION
0256:0009         FINAL_DOS_LOCATION
0256:000B         DEVICE_LIST
0256:000F         MEMORY_SIZE
0256:0011         DEFAULT_DRIVE
0256:0012         BUFFERS
0256:0013         FILES
0256:0786         BADOPM
0256:07AA         CRLFM
0256:07AD         BADSIZ
0256:07CE         BADLD
0256:07E0         BADCOM
0256:07F4         SYSSIZE
```

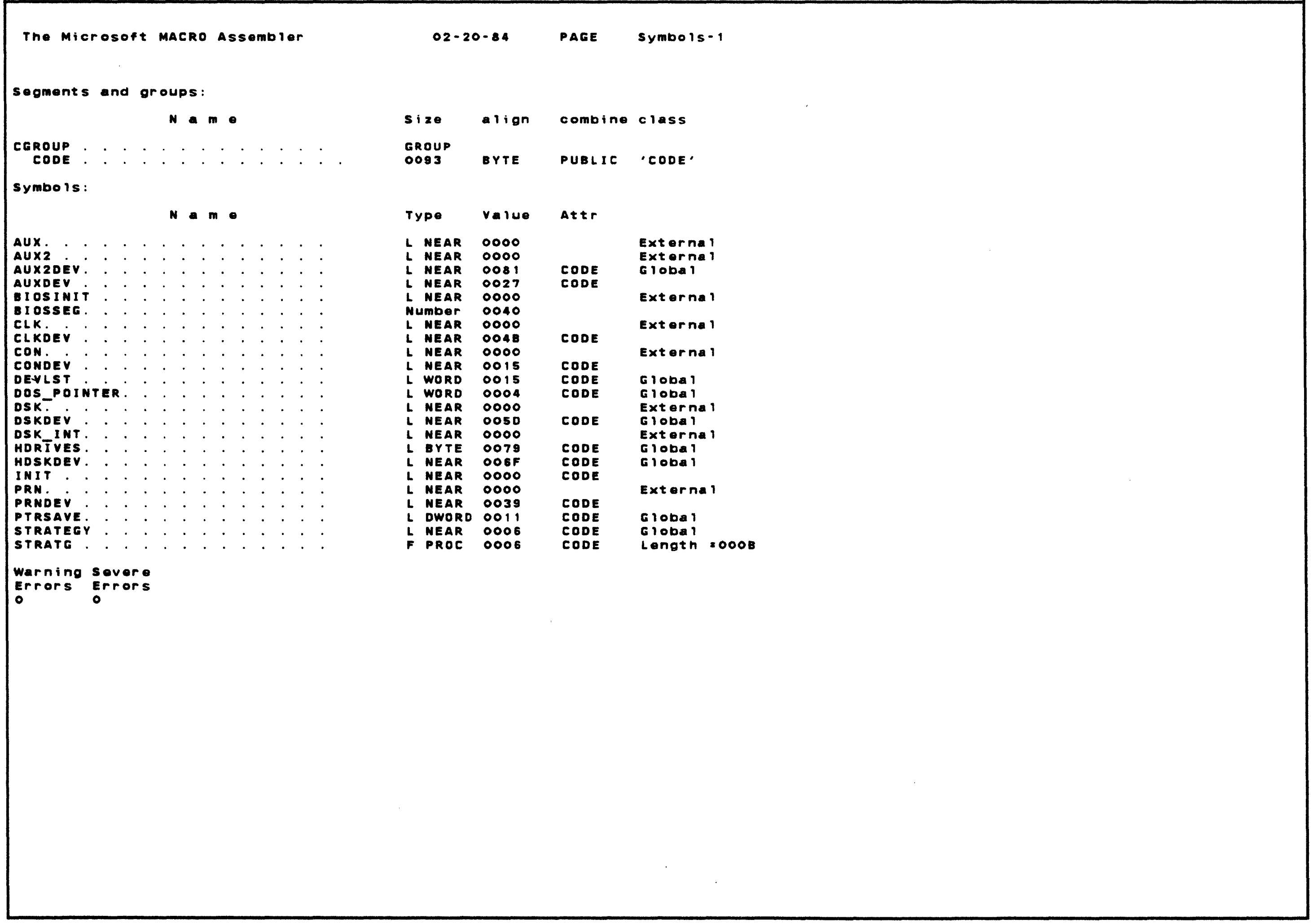
```
                                        PAGE    60,132

                                ;
                                ;       COMPANY CONFIDENTIAL
                                ;       Copyright (C) 1983 Digital Equipment Corporation
                                ;       All rights reserved.
                                ;
                                ;       10/05/83


                                cgroup group code

        = 0040                  biosseg equ 40h

                                ;
                                ;Top level I/O system module: contains the
                                ;device tables for the minimum system.
                                ;
                                public  strategy,ptrsave,devlst,hdskdev,dos_pointer,hdrives
                                public  dskdev,aux2dev

                                extrn   biosinit:near
                                extrn   CON:NEAR,AUX:NEAR,PRN:NEAR, AUX2:NEAR
                                extrn   clk:near,dsk:near
                                extrn   dsk_int:near
```

```
                                 page
0000                             code segment byte public 'code'
                                 assume cs:cgroup,ds:cgroup
                                 ;
                                 ;Initialize ourselves and what's necessary for
                                 ;SYSINIT, then pass control there.
                                 ;
0000  FA                         init:    cli
0001  E9 0000 E                           jmp      biosinit

0004  0000                       dos_pointer        dw    0        ;Filled in by BIOSINIT used by FORMAT
                                 ;
                                 ;Simplistic strategy routine for 2.00.
                                 ;
0006                             stratg proc far

0006                             strategy:
0006  2E: 89 1E 0011 R                    mov word ptr cs:ptrsave,bx
000B  2E: 8C 06 0013 R                    mov word ptr cs:ptrsave+2,es
0010  CB                                  ret

0011                             stratg endp
                                 ;
                                 ;Public pointer save.
                                 ;
0011  ????????                   ptrsave dd       (?)
```

```
                                 page
                                 ;
                                 ;I/O device table. This contains the five
                                 ;minimum devices: CON, AUX, PRN, CLOCK and
                                 ;disks. SYSINIT may load others later, but
                                 ;that's no concern to us here.
                                 ;
                                 ;-----------------------------------------------+
                                 ;      DWORD pointer to next device            !
                                 ;          (-1,-1 if last device)              |
                                 ;-----------------------------------------------+
                                 ;      Device attribute WORD                   ;
                                 ;        Bit 15 = 1 for chacter devices.       ;
                                 ;                 0 for Block devices.         ;
                                 ;                                              ;
                                 ;        Charcter devices. (Bit 15=1)          ;
                                 ;          Bit 0 = 1  current sti device.      ;
                                 ;          Bit 1 = 1  current sto device.      ;
                                 ;          Bit 2 = 1  current NUL device.      ;
                                 ;          Bit 3 = 1  current Clock device.    ;
                                 ;                                              ;
                                 ;          Bit 14 = 1 IOCTL control bit.       ;
                                 ;-----------------------------------------------+
                                 ;      Device strategy pointer.                ;
                                 ;-----------------------------------------------+
                                 ;      Device interrupt pointer.               ;
                                 ;-----------------------------------------------+
                                 ;      Device name field.                      ;
                                 ;        Character devices are any valid name;
                                 ;          left justified, in a space filled ;
                                 ;          field.                              ;
                                 ;        Block devices contain # of units in ;
                                 ;          the first byte.                     ;
                                 ;-----------------------------------------------+

0015                             devlst label word
0015  0027 R 0040                condev:  dw      auxdev,biosseg  ;ptr to next,
0019  8003                                dw      8003h           ;char,sti,sto,
001B  0006 R                              dw      strategy
001D  0000 E                              dw      con
001F  43 4F 4E 20 20 20                   db      'CON     '
      20 20
0027                             auxdev:
0027  0039 R 0040                         dw      prndev,biosseg
002B  C000                                dw      0C000h
002D  0006 R                              dw      strategy
002F  0000 E                              dw      aux
0031  41 55 58 20 20 20                   db      'AUX     '
      20 20
0039                             prndev:
0039  004B R 0040                         dw      clkdev,biosseg
003D  C000                                dw      0C000h
003F  0006 R                              dw      strategy
0041  0000 E                              dw      prn
0043  50 52 4E 20 20 20                   db      'PRN     '
```



```
 The Microsoft MACRO Assembler              02-20-84      PAGE     1-4


               20 20
        004B                               clkdev:
        004B   005D R 0040                          dw       dskdev,biosseg
        004F   8008                                 dw       8008h
        0051   0006 R                               dw       strategy
        0053   0000 E                               dw       clk
        0055   43 4C 4F 43 4B 20                    db       'CLOCK
               20 20
        005D                               dskdev:
        005D   FFFF FFFF                            dw       -1,-1
        0061   2000                                 dw       2000h          ;non-IBM compatable without IOCTL
        0063   0006 R                               dw       strategy
        0065   0000 E                               dw       dsk
        0067   02 00 00 00 00 00                    db       2,0,0,0,0,0,0,0
               00 00

        006F                               hdskdev:
        006F   FFFF FFFF                            dw       -1,-1
        0073   6000                                 dw       6000h          ; with IOCTL
        0075   0006 R                               dw       strategy
        0077   0000 E                               dw       dsk_int
        0079   01 00 00 00 00 00           hdrives db       1,0,0,0,0,0,0,0
               00 00

        0081                               aux2dev:
        0081   FFFF FFFF                            DW       -1,-1
        0085   C000                                 DW       0C000H
        0087   0006 R                               DW       strategy
        0089   0000 E                               DW       aux2
        008B   41 55 58 32 20 20                    DB       'AUX2          ; device name
               20 20

        0093                               code ends

                                                    end
```



```
 The Microsoft MACRO Assembler              02-20-84      PAGE     Symbols-1


Segments and groups:

                N a m e                    Size    align   combine class

CGROUP . . . . . . . . . . . . . .         GROUP
  CODE . . . . . . . . . . . . . . .       0093    BYTE    PUBLIC  'CODE'

Symbols:

                N a m e                    Type    Value   Attr

AUX. . . . . . . . . . . . . . . .         L NEAR  0000            External
AUX2 . . . . . . . . . . . . . . .         L NEAR  0000            External
AUX2DEV. . . . . . . . . . . . . .         L NEAR  0081    CODE    Global
AUXDEV . . . . . . . . . . . . . .         L NEAR  0027    CODE
BIOSINIT . . . . . . . . . . . . .         L NEAR  0000            External
BIOSSEG. . . . . . . . . . . . . .         Number  0040
CLK. . . . . . . . . . . . . . . .         L NEAR  0000            External
CLKDEV . . . . . . . . . . . . . .         L NEAR  004B    CODE
CON. . . . . . . . . . . . . . . .         L NEAR  0000            External
CONDEV . . . . . . . . . . . . . .         L NEAR  0015    CODE
DEVLST . . . . . . . . . . . . . .         L WORD  0015    CODE    Global
DOS_POINTER. . . . . . . . . . . .         L WORD  0004    CODE    Global
DSK. . . . . . . . . . . . . . . .         L NEAR  0000            External
DSKDEV . . . . . . . . . . . . . .         L NEAR  005D    CODE    Global
DSK_INT. . . . . . . . . . . . . .         L NEAR  0000            External
HDRIVES. . . . . . . . . . . . . .         L BYTE  0079    CODE    Global
HDSKDEV. . . . . . . . . . . . . .         L NEAR  006F    CODE    Global
INIT . . . . . . . . . . . . . . .         L NEAR  0000    CODE
PRN. . . . . . . . . . . . . . . .         L NEAR  0000            External
PRNDEV . . . . . . . . . . . . . .         L NEAR  0039    CODE
PTRSAVE. . . . . . . . . . . . . .         L DWORD 0011    CODE    Global
STRATEGY . . . . . . . . . . . . .         L NEAR  0006    CODE    Global
STRATG . . . . . . . . . . . . . .         F PROC  0006    CODE    Length =000B

Warning Severe
Errors  Errors
0       0
```

```
 Symbol Cross Reference                    (# is definition)       Cref-1

AUX. . . . . . . . . . . . . . . .        22#    98
AUX2 . . . . . . . . . . . . . . .        22#   135
AUX2DEV. . . . . . . . . . . . . .        19    131#
AUXDEV . . . . . . . . . . . . . .        88     94#

BIOSINIT . . . . . . . . . . . . .        21#    34
BIOSSEG. . . . . . . . . . . . . .        13#    88      95     102     109

CGROUP . . . . . . . . . . . . . .        11     28      28
CLK. . . . . . . . . . . . . . . .        23#   112
CLKDEV . . . . . . . . . . . . . .       102    108#
CODE . . . . . . . . . . . . . . .        11     27#     27     139
CON. . . . . . . . . . . . . . . .        22#    91
CONDEV . . . . . . . . . . . . . .        88#

DEVLST . . . . . . . . . . . . . .        18     87#
DOS_POINTER. . . . . . . . . . . .        18     36#
DSK. . . . . . . . . . . . . . . .        23#   119
DSKDEV . . . . . . . . . . . . . .        19    109     115#
DSK_INT. . . . . . . . . . . . . .        24#   127

HDRIVES. . . . . . . . . . . . . .        18    128#
HDSKDEV. . . . . . . . . . . . . .        18    123#

INIT . . . . . . . . . . . . . . .        33#

PRN. . . . . . . . . . . . . . . .        22#   105
PRNDEV . . . . . . . . . . . . . .        95    101#
PTRSAVE. . . . . . . . . . . . . .        18     43      44      51#

STRATEGY . . . . . . . . . . . . .        18     42#     90      97     104     111     118     126     134
STRATG . . . . . . . . . . . . . .        40#    47
```

```
 The Microsoft MACRO Assembler                    02-20-84      PAGE     1-1
DEC Rainbow Rx-50/Z80 disk driver


1                                                      PAGE     60,132
2                                                      TITLE    DEC Rainbow Rx-50/Z80 disk driver
3                                                      NAME     DECDISK
4                                              ;
5                                              ;       COMPANY CONFIDENTIAL
6                                              ;       Copyright (C) 1983 Digital Equipment Corporation
7                                              ;       All rights reserved.
8                                              ;
9                                              ;       10/07/83
10
11                                             cgroup group code
12                                             ;
13                                             ;Converts the standard BIOS data packet to
14                                             ;the Rainbow Z80 packet and executes it.
15                                             ;Returns normal MSDOS error codes.
16                                             ;
17                                             public  decdisk,z80isr,physdisk
18                                             extrn   buffer:word, MC_FLG:BYTE
19                                             ;
20                                             ;Z80 function codes:
21                                             ;
22       = 0013                                QKRDCOM equu     13h      ;disk read,
23       = 0014                                QKWTCOM equ      14h      ;disk write,
24       = 0015                                QKCMCOM equ      15h      ;media check,
25       = 0016                                QKWVFCOM equ     16h      ;write with verify,
26       = 0023                                QKVFYCOM equ     23h      ;verify disk command
27       = 0024                                QKFMTCOM equ     24h      ;format track/disk command
28
29       = 0000                                intz80   equ     0        ;z80 interrupt port,
30       = 0002                                gscr     equ     2        ;interrupt status port,
31
32       = 0000                                XCOMMAND EQU     0        ;IOCTL PACKET COMMAND OFFSET
33       = 0001                                XDRIVE   EQU     1        ;DRIVE/SIDE OFFSET
34       = 0002                                XSECTOR  EQU     2        ;SECTOR OFFSET
35       = 0003                                XPHDRV   EQU     3        ;PHYSICAL DRIVE CODE
36       = 0004                                XTRACK   EQU     4        ;TRACK # TO USE
37       = 0006                                XCOUNT   EQU     6        ;# OF SECTORS TO XFER
38       = 0008                                XDMAOFF  EQU     8        ;OFFSET OF DMA XFER
39       = 000A                                XDMASEG  EQU     0AH      ;SEGMENT OF DMA XFER
40
41
42       = 0000                                rx50     EQU     00h      ;return value for RX50 disks.
43       = 0008                                ibm8     EQU     08h      ;media check return value for IBM 8 sector
44       = 0002                                ibm9     EQU     02h      ;mediacheck return value for IBM 9 sector
45
46       = 0000                                rx50val equ      00h      ;MSDOS value for rx50 disks
47       = 0001                                ibm8val equ      01h      ;IBM 8 sector disks
48       = 0002                                ibm9val equ      02h      ;IBM 9 sector disks
49
50
51                                          C  include iodef.ash
52                                          C  ;Rainbow Interrupt numbers.
53                                          C  ;1-Apr-83 sgs added dskio int vector
54                                          C  ;19-Mar-83 sgs added profile int vector [user clock]
55                                          C  ;the int profile is called by the RTC interrupt service for each tick.
```

```
                                C  ;The ax and ds register don't need to be saved [done in clkisr].
                                C  ;
        = 0064                  C  profile equ     64h     ;100. user interface to clock interrupt
        = 002C                  C  clk_int equ     2ch     ;60Hz int,
        = 0065                  C  dskio   equ     65h     ;direct disk io for format
                                C  ;
                                C  ;17-Mar-83 sgs changed to include int 20-26
                                C  ;interrupt 22-24h are duplicated at 42-44h for consistency.
                                C  ;These are the relocated Rainbow interrupts.
                                C  ;interrupts, 20h-26h moved to 40h-46h
                                C  ;ATTENTION Modules IOINIT and REINIT may have to be
                                C  ;changed if int vectors 40h to 46h are changed
                                C  ;
        = 0040                  C  vert    equ     40h     ;new vert. freq.
        = 0041                  C  spare41 equ     41h
        = 0042                  C  spare42 equ     42h
        = 0043                  C  comdma  equ     43h     ;DMA ctrl optional comm. board
        = 0044                  C  aux_prn equ     44h     ;7201 comm./printer
        = 0045                  C  excom   equ     45h     ;extented comm. option
        = 0046                  C  uart    equ     46h     ;new UART vector,
        = 0047                  C  z80     equ     47h     ;Z80 interrupt,
        = 0018                  C  rom     equ     18h     ;new ROM access,
                                C  ;
                                C  ;DEC Rainbow IO port stuff.
                                C  ;
        = 0010                  C  kdp     equ     10h     ;8251 data port,
        = 0011                  C  ksp     equ     11h     ;8251 status,
        = 0040                  C  auxdp   equ     40h     ;7201 data,
        = 0041                  C  prndp   equ     41h
        = 0042                  C  auxp    equ     42h     ;7201 command,
        = 0043                  C  prnp    equ     43h
                                C  ;
                                C  ; Values in XOPTION
                                C  ;
        = 0001                  C  wprsnt  equ     01      ;Winnie preset
        = 0002                  C  xcprsnt equ     02      ;XCOMM present
                                C  include disk.ash
                                C  ;
                                C  ;Physical disk IO data packet.
                                C  ;
        = 0000                  C  command equ     0       ;read or write,
        = 0001                  C  drive   equ     1       ;phs. drive,
        = 0002                  C  track   equ     2       ;track,
        = 0003                  C  sector  equ     3       ;starting sector,
        = 0004                  C  count   equ     4       ;sector count,
        = 0005                  C  curtrk  equ     5       ;current track,
        = 0006                  C  density equ     6       ;N.A.
        = 0007                  C  gaplen  equ     7       ;N.A.
        = 0008                  C  enn     equ     8       ;N.A.
        = 0009                  C  dtl     equ     9       ;N.A.
        = 000A                  C  secsiz  equ     10      ;sector size,
        = 000C                  C  dmaoff  equ     12      ;DMA offset,
        = 000E                  C  dmaseg  equ     14      ;DMA segment
        = 0010                  C  spt     equ     16      ;sectors/track
        = 0012                  C  head    equ     18      ;head,
```

```
                                   page

                                   ;This is what the Z80 thinks memory looks like.

        0000                       z80seg  segment at 0
        0000                       z80seg  ends

                                   extrn   packet_adr:near
                                   extrn   tformat:byte    ;Table for current formats, one byte per drive.
                                   extrn   ttrack:byte     ;Current tracks, one per drive. As in TFORMAT,

                                   ;Pointers to the Z80 and 8088 packets.

                                   extrn   xfrpkt:word             ;pointer to "packet"
                                   extrn   i88pkt:word             ;returned pointer from Z80

                                   ;The packet for the z80.

                                   extrn   packet:byte             ;Z80 packet pointers
                                   extrn   fnccod:byte             ;Function code
                                   extrn   status:byte             ;retnd status,
                                   extrn   drvno:byte              ;drive and side,
                                   extrn   secno:byte              ;sector #
                                   extrn   trackn:byte             ;track #
                                   extrn   nsect:byte              ;# of sectors to process
                                   extrn   dmaadr:word             ;DMA address (abs)
                                   extrn   exstat:byte             ;extended status
```

```
                              page

0000                          code segment byte public 'code'
                              assume cs:cgroup,ds:cgroup,es:z80seg
                              ;
                              ;Convert the BIOS packet for the Z80, make the
                              ;request. The absolute address range must be
                              ;1000 - ffff hex, as below that is Z80 private,
                              ;and the Z80 cannot reach above it. A special
                              ;'disk' module is requred.
                              ;
0000  06                      decdisk:push    es
0001  B8  ---- R                      mov     ax,seg z80seg
0004  8E C0                           mov     es,ax

0006  32 D2                           xor     dl,dl             ;clear drive byte & assume single side
0008  8A 57 01                        mov     dl,drive[bx]      ;make drive
000B  D0 E2                           shl     dl,1              ;select bits,
000D  D0 E2                           shl     dl,1              ;SHL DL,1
000F  D0 E2                           shl     dl,1              ;4 times is
0011  D0 E2                           shl     dl,1              ;much faster
0013  26: 88 16 0000 E                mov     es:drvno,dl
                              ;
                              ;Only if the media is RX-50, and the track is
                              ;above 1, skew the sector. RX-50 media tracks
                              ;0 and 1 are the boot, and non-RX-50 media
                              ;is assumed to be IBM.
                              ;
0018  8A 47 03                        mov     al,sector[bx]
001B  80 7F 06 00                     cmp     byte ptr density[bx],0
001F  75 0E                           jne     zw0               ;if RX-50,
0021  80 7F 02 02                     cmp     byte ptr track[bx],2
0025  72 08                           jb      zw0               ;and trk 2-,

0027  B4 00                           mov     ah,0              ;skew it, SI=
0029  8B F0                           mov     si,ax             ;index to tbl
002B  8A 84 01EE R                    mov     al,skewtbl_2[si]

002F  26: A2 0000 E           zw0:    mov     es:secno,al       ;set it
0033  8A 47 02                        mov     al,track[bx]
0036  26: A2 0000 E                   mov     es:trackn,al
003A  8A 47 04                        mov     al,count[bx]
003D  26: A2 0000 E                   mov     es:nsect,al
                              ;
                              ;Make the absolute address in AX. Ignore out of
                              ;bounds addresses; DISK is supposed to process
                              ;them for us.
                              ;
0041  8B 47 0E                        mov     ax,dmaseg[bx]     ;segment
0044  D1 E0                           shl     ax,1              ;times 2,
0046  D1 E0                           shl     ax,1              ;times 4,
0048  D1 E0                           shl     ax,1              ;times 8,
004A  D1 E0                           shl     ax,1              ;times 16,
004C  03 47 0C                        add     ax,dmaoff[bx]     ;plus offset,
004F  26: A3 0000 E                   mov     es:dmaadr,ax
```

```

0053  8A 07                           mov     al,command[bx]    ;what to do?
0055  98                              cbw                       ;expand to word
0056  8B F0                           mov     si,ax             ;copy command
0058  8A 84 0065 R                    mov     al,zdskfcn[si]    ;get actual command
005C  26: A2 0000 E                   mov     es:fnccod,al      ;store Z80 command
                              ;
                              ;Execute the packet.
                              ;
0060  E8 019C R                       call    execz80           ;Invoke Z80 and report error if any
0063  07                              pop     es
0064  C3                              ret

                              ;translate table for internal command to Z80 command

0065  14 13 15 16             zdskfcn db      qkwtcom, qkrdcom, qkcmcom, qkwvfcom

```

```
                                page
                                ;
                                ;for use by format, direct disk i/o
                                ;same as decdisk.
                                ;
0069     04 [                           db      4 dup (0) ;this space maker needed for exe2bin 1-Apr-83 sgs no joke
               00
                    ]



                                ;       check media function switch
                                ;       COMMAND = 6 then turn the function ON
                                ;       COMMAND = 7 then turn the function OFF
                                ;       COMMAND = 8 then read the function status in AL

006D                            physdisk label near
006D                            physdisk1 proc far

006D  8A 07                             MOV     AL,XCOMMAND[BX]            ; get command
006F  3C 06                             CMP     AL,6                       ; check media function?
0071  72 1A                             JB      PHYS1
0073  1E                                PUSH    DS                         ; save user's DS
0074  0E                                PUSH    CS
0075  1F                                POP     DS                         ; DS=CS

0076  B4 00                             MOV     AH,0                       ; assume turn function on
0078  74 06                             JZ      PHYS2
007A  FE C4                             INC     AH                         ; assume turn function off
007C  3C 07                             CMP     AL,7                       ; if > 8 then do as 8
007E  75 04                             JNZ     PHYS3
0080  88 26 0000 E              PHYS2:  MOV     BYTE PTR MC_FLG,AH
0084  33 C0                     PHYS3:  XOR     AX,AX
0086  A0 0000 E                         MOV     AL,BYTE PTR MC_FLG
0089  1F                                POP     DS
008A  CA 0002                           RET     2
008D                            PHYS1:
008D  53                                push    bx
008E  06                                push    es
                                ;
                                ;ds set to calling segment for [bx] param.
                                ;
008F  B8  ---- R                        mov     ax,seg z80seg
0092  8E C0                             mov     es,ax

0094  32 D2                             xor     dl,dl
0096  8A 57 01                          mov     dl,xdrive[bx]    ;make drive
0099  80 FA FF                          cmp     dl,0ffh          ;is this a physical disk oper?
009C  75 06                             jnz     logdrv           ;no,its logical
009E  8A 57 03                          mov     dl,xphdrv[bx]    ;get user's pre defined drive code
00A1  EB 09 90                          jmp     stdrv            ;else set the drive
00A4  D0 E2                     logdrv: shl     dl,1             ;move drive # to high nybble
00A6  D0 E2                             shl     dl,1             ;SHL DL,1
00A8  D0 E2                             shl     dl,1             ;4 times is
00AA  D0 E2                             shl     dl,1             ;much faster
```

```
00AC  26: 88 16 0000 E          stdrv:  mov     es:drvno,dl      ;put drive/side in packet
00B1  8B 4F 04                          mov     cx,xtrack[bx]    ;get track #
00B4  33 C0                             xor     ax,ax            ;clear high byte
00B6  8A 47 02                          mov     al,xsector[bx]   ;get sector #
00B9  83 F9 02                          cmp     cx,02            ;is track # > 2
00BC  72 07                             jb      skwed            ;if not, dont skew
00BE  8B F0                             mov     si,ax            ;else index into skew table
00C0  2E: 8A 84 01EE R                  mov     al,cs:skewtbl_2[si] ;and get skewed value
00C5  26: 88 0E 0000 E          skwed:  mov     es:trackn,cl     ;save track #
00CA  26: A2 0000 E                     mov     es:secno,al      ;save processed sector
00CE  26: C6 06 0000 E 01               mov     byte ptr es:nsect,1

00D4  32 E4                             xor     ah,ah            ;clear high byte
00D6  8A 07                             mov     al,xcommand[bx]  ;what to do?
00D8  8B F0                             mov     si,ax            ;index into function table
00DA  3C 00                             cmp     al,0             ;if read, use intermediate buffer and move later
00DC  74 22                             jz      redfndl          ;
00DE  8B 4F 0A                          mov     cx,xdmaseg[bx]   ;get users data buffer ptr
00E1  8B 47 08                          mov     ax,xdmaoff[bx]   ;it is, after all, a dword type
00E4  83 FE 03                          cmp     si,03            ;are we doing a format?
00E7  74 1C                             jz      fmtfndl          ;then use the users buffer
00E9  56                        movlst: push    si               ;save the registers we are going to cream
00EA  57                                push    di               ;in the block move
00EB  1E                                push    ds               ;
00EC  06                                push    es
00ED  FC                                cld                      ;make sure we are going in the right direction
00EE  0E                                push    cs               ;destination is our segment
00EF  07                                pop     es
00F0  BF 0000 E                         mov     di,offset buffer          ;and the default buffer
00F3  8E D9                             mov     ds,cx            ;and the source is the users buffer
00F5  8B F0                             mov     si,ax            ;
00F7  B9 0100                           mov     cx,100h          ;move full sector
00FA  F3/ A5                            rep     movsw            ;
00FC  07                                pop     es
00FD  1F                                pop     ds
00FE  5F                                pop     di
00FF  5E                                pop     si               ;restore registers

0100                            redfndl:
0100  8C C9                             mov     cx,cs            ;use our buffer
0102  B8 0000 E                         mov     ax,offset buffer;for read, and move after
0105                            fmtfndl:
0105  D1 E1                             shl     cx,1
0107  D1 E1                             shl     cx,1
0109  D1 E1                             shl     cx,1
010B  D1 E1                             shl     cx,1
010D  03 C8                             add     cx,ax            ;make absolute address
010F  26: 89 0E 0000 E                  mov     es:dmaadr,cx     ;and put it in the packet

0114  2E: 8A 84 017F R                  mov     al,byte ptr cs:xdskfcn[si]      ;get function code
0119  26: A2 0000 E                     mov     es:fnccod,al     ;put code in packet
                                ;
                                ;now set ds to cs for local code
                                ;
011D  1E                        pzw1:   push    ds
```

```
0120  8E D8                          mov     ds,ax            ;set ds
0122  52                             push    dx               ;save drive #
0123  56                             push    si               ;get function code
0124  E8 019C R                      call    execz80          ;Invoke Z80 and wait for status
0127  5A                             pop     dx               ;recover function code
0128  5E                             pop     si               ;recover drive #
0129  9C                             pushf                    ;save flags around grand munging we do
012A  83 FA 04                       cmp     dx,04            ;was this a media check?
012D  75 2A                          jnz     eggsit           ;if not, just bail out
012F  B1 04                          mov     cl,4             ;restore drive value
0131  D3 CE                          ror     si,cl            ;to its virgin state
0133  81 E6 000F                     and     si,0fh           ;JIC any garbage
0137  26: 8A A4 0000 E               mov     ah,es:byte ptr tformat[si] ;get media type
013C  80 FC 00                       cmp     ah,0             ;is this an RX50?
013F  74 0C                          jz      fmtset           ;just put 0 in buffer
0141  80 FC 08                       cmp     ah,ibm8          ;Is this an IBM 8 sector
0144  75 05                          jnz     oterfmt          ;no,must be an IBM 9
0146  B4 01                          mov     ah,ibm8val       ;save MSDOS code
0148  EB 03 90                       jmp     fmtset           ;for users
014B  B4 02                 oterfmt:mov      ah,ibm9val       ;assume IBM 9 sector for now
014D  26: 8B 1E 0000 E      fmtset: mov      bx,es:dmaadr     ;get pointer to xfer buffer
0152  26: 88 27                      mov     es:[bx],ah       ;Save media type in buffer
0155  33 D2                          xor     dx,dx            ;do move the buffer, but
0157  33 C0                          xor     ax,ax            ;dont return a funny status
0159                        eggsit:
0159  9D                             popf
015A  1F                             pop     ds
015B  07                             pop     es
015C  5B                             pop     bx       ;restore users request buffer
015D  9C                             pushf            ;save flags again
015E  80 3F 00                       cmp     byte ptr xcommand[bx],0 ;was this a read?
0161  75 18                          jnz     outhere

0163  1E                             push    ds
0164  06                             push    es
0165  56                             push    si
0166  57                             push    di
0167  8E 47 0A                       mov     es,xdmaseg[bx]  ;put data where the user expects to see it
016A  8B 7F 08                       mov     di,xdmaoff[bx]  ;
016D  0E                             push    cs
016E  1F                             pop     ds              ;
016F  BE 0000 E                      mov     si,offset buffer
0172  B9 0100                        mov     cx,100h
0175  F3/ A5                         rep     movsw
0177  5F                             pop     di
0178  5E                             pop     si
0179  07                             pop     es
017A  1F                             pop     ds

017B                        outhere:
017B  9D                             popf            ;return flags
017C  CA 0002                        ret     2       ;called with INT (clean stack)

017F  13                    XDSKFCN: db      QKRDCOM
```

```
0180  14                             db      QKWTCOM
0181  16                             db      QKWVFCOM
0182  24                             db      QKFMTCOM
0183  15                             db      QKCMCOM
0184  23                             db      QKVFYCOM
0185                        physdisk1 endp
```

```
                             page
                             ;
                             ;Interrupt service from the Z80. The Z80
                             ;indicates it is done by setting Z80PKT ::
                             ;I88PKT, and interrupting us. Once we clear
                             ;our interrupt, it then zeros the packet
                             ;pointer. The reason for all this interlocking
                             ;pointer fiddling is a mystery.
                             ;
0185 0000                    zpkt    dw      0

0187 50                      z80isr: push    ax
0188 06                              push    es
0189 B8 ---- R                       mov     ax,seg z80seg   ;if the ptr is
018C 8E C0                           mov     es,ax           ;0, do nothing,
018E 26: 87 06 0000 E                xchg    ax,es:xfrpkt    ;see if packet back yet
0193 2E: A3 0185 R                   mov     cs:zpkt,ax       ;if zppkt still 0, upper level not back
0197 E4 00                   zi1:    in      al,intz80       ;clear the int.
0199 07                              pop     es
019A 58                              pop     ax
019B CF                              iret
                             ;
                             ;As in the ISR, we must play strange games with
                             ;packet pointers. Set the i88 packet pointer,
                             ;interrupt the Z80, when it clears it's int-
                             ;errupt, zero the packet pointer, then wait
                             ;for an interrupt from the Z80.
                             ;
019C                         execz80:
019C 26: C7 06 0000 E 0000 E         mov     es:xfrpkt,offset z80seg:packet   ;set xfer packet ready
01A3 2E: C7 06 0185 R 0000           mov     cs:zpkt,0       ;set status to gone
01AA 26: C6 06 0000 E 00             mov     es:byte ptr status,0
01B0 E6 00                           out     intz80,al       ;start z80,
                             ;
                             ;Wait until the Z80 accepts the interrupt,
                             ;then zero the packet pointer.
                             ;
01B2 26: 83 3E 0000 E 00     xz0:    cmp     word ptr es:xfrpkt,0
01B8 75 F8                           jnz     xz0

                             ;Wait for the Z80 to interrupt us.
                             ;
01BA FA                      xz1:    cli                     ;lock out Z80,
01BB 2E: 83 3E 0185 R 00             cmp     cs:zpkt,0       ;see if gone
01C1 75 04                           jne     xz2             ;done.
01C3 FB                              sti
01C4 F4                              hlt                     ;wait for an
01C5 EB F3                           jmp     xz1             ;interrupt.

01C7 FB                      xz2:    sti
01C8 26: A0 0000 E                   mov     al,es:status    ;OK?
01CC 3C 00                           cmp     al,0
01CE 75 02                           jnz     xz80err         ;if no error,
01D0 F8                              clc                     ;make sure carry bit clear
01D1 C3                              ret                     ; and return
```

```

01D2                         xz80err:
01D2 BE 0000                         mov     si,0            ;ptr to code,
01D5 26: A0 0000 E                   mov     al,es:exstat    ;get raw machine status
01D9 3C 00                           cmp     al,0            ;this may be zero if no hardware error
01DB 75 04                           jnz     xz4
01DD B0 0C                           mov     al,0Ch          ;return "general failure"
01DF EB 08                           jmp short xz5

01E1 D0 C0                   xz4:    rol     al,1            ;shift to Cy,
01E3 72 03                           jc      xz3
01E5 46                              inc     si              ;next bit ...
01E6 EB F9                           jmp short xz4

01E8 8A 84 01F9 R            xz3:    mov     al,errcode[si]  ,get return
01EC F9                      xz5:    stc                     ;code,
01ED C3                              ret
```

```
454                                      page
455                                      ;
456                                      ;DEC RX-50 sector skew table. . Only
457                                      ;used on tracks 2 to 79. Tracks 0 and 1 are
458                                      ;unskewed.
459                                      ;
460                                      ;
461                                      ;      skew :: 2
462      01EE                            skewtbl_2 label byte
463
464      01EE  00                               db     0                 ;no sector 0,
465      01EF  01                               db     1
466      01F0  03                               db     3
467      01F1  05                               db     5
468      01F2  07                               db     7
469      01F3  09                               db     9
470      01F4  02                               db     2
471      01F5  04                               db     4
472      01F6  06                               db     6
473      01F7  08                               db     8
474      01F8  0A                               db     10
475                                      ;
476                                      ;179x -> MSDOS disk error code translate table.
477                                      ;
478      01F9                            errcode label byte
479
480      01F9  02                               db     2        ;bit 7: not ready,
481      01FA  00                               db     c        ;bit 6: write protect,
482      01FB  0A                               db     10       ;bit 5: write fault,
483      01FC  06                               db     6        ;bit 4: seek error,
484      01FD  04                               db     4        ;bit 3: CRC error,
485      01FE  0B                               db     11       ;bit 2: lost data,
486      01FF  07                               db     7        ;bit 1: **Robin media**
487      0200  02                               db     2        ;bit 0: busy
488
489      0201                            code ends
490
491                                             end
```

```
Segments and groups:

                N a m e                     Size     align    combine class

CGROUP . . . . . . . . . . . . . .          GROUP
  CODE . . . . . . . . . . . . . . . .      0201     BYTE     PUBLIC   'CODE'
Z80SEG . . . . . . . . . . . . . .          0000     AT       0000

Symbols:

                N a m e                     Type     Value    Attr

AUXDP. . . . . . . . . . . . . . .          Number   0040
AUXP . . . . . . . . . . . . . . .          Number   0042
AUX_PRN. . . . . . . . . . . . . .          Number   0044
BUFFER . . . . . . . . . . . . . .          V WORD   0000              External
CLK_INT. . . . . . . . . . . . . .          Number   002C
COMDMA . . . . . . . . . . . . . .          Number   0043
COMMAND. . . . . . . . . . . . . .          Number   0000
COUNT. . . . . . . . . . . . . . .          Number   0004
CURTRK . . . . . . . . . . . . . .          Number   0005
DECDISK. . . . . . . . . . . . . .          L NEAR   0000     CODE     Global
DENSITY. . . . . . . . . . . . . .          Number   0006
DMAADR . . . . . . . . . . . . . .          V WORD   0000              External
DMAOFF . . . . . . . . . . . . . .          Number   000C
DMASEG . . . . . . . . . . . . . .          Number   000E
DRIVE. . . . . . . . . . . . . . .          Number   0001
DRVNO. . . . . . . . . . . . . . .          V BYTE   0000              External
DSKIO. . . . . . . . . . . . . . .          Number   0065
DTL. . . . . . . . . . . . . . . .          Number   0009
EGGSIT . . . . . . . . . . . . . .          L NEAR   0159     CODE
ENN. . . . . . . . . . . . . . . .          Number   0008
ERRCODE. . . . . . . . . . . . . .          L BYTE   01F9     CODE
EXCOM. . . . . . . . . . . . . . .          Number   0045
EXECZ80. . . . . . . . . . . . . .          L NEAR   019C     CODE
EXSTAT . . . . . . . . . . . . . .          V BYTE   0000              External
FMTFNDL. . . . . . . . . . . . . .          L NEAR   0105     CODE
FMTSET . . . . . . . . . . . . . .          L NEAR   014D     CODE
FNCCOD . . . . . . . . . . . . . .          V BYTE   0000              External
GAPLEN . . . . . . . . . . . . . .          Number   0007
GSCR . . . . . . . . . . . . . . .          Number   0002
HEAD . . . . . . . . . . . . . . .          Number   0012
I88PKT . . . . . . . . . . . . . .          V WORD   0000              External
IBM8 . . . . . . . . . . . . . . .          Number   0008
IBM8VAL. . . . . . . . . . . . . .          Number   0001
IBM9 . . . . . . . . . . . . . . .          Number   0002
IBM9VAL. . . . . . . . . . . . . .          Number   0002
INTZ80 . . . . . . . . . . . . . .          Number   0000
KDP. . . . . . . . . . . . . . . .          Number   0010
KSP. . . . . . . . . . . . . . . .          Number   0011
LOGDRV . . . . . . . . . . . . . .          L NEAR   00A4     CODE
MC_FLG . . . . . . . . . . . . . .          V BYTE   0000              External
MOV1ST . . . . . . . . . . . . . .          L NEAR   00E9     CODE
NSECT. . . . . . . . . . . . . . .          V BYTE   0000              External
OTERFMT. . . . . . . . . . . . . .          L NEAR   014B     CODE
```

```
OUTHERE. . . . . . . . . . . . . .      L NEAR  017B    CODE
PACKET . . . . . . . . . . . . . .      V BYTE  0000            External
PACKET_ADR . . . . . . . . . . . .      L NEAR  0000            External
PHYS1. . . . . . . . . . . . . . .      L NEAR  008D    CODE
PHYS2. . . . . . . . . . . . . . .      L NEAR  0080    CODE
PHYS3. . . . . . . . . . . . . . .      L NEAR  0084    CODE
PHYSDISK . . . . . . . . . . . . .      L NEAR  006D    CODE    Global
PHYSDISK1. . . . . . . . . . . . .      F PROC  006D    CODE    Length =0118
PRNDP. . . . . . . . . . . . . . .      Number  0041
PRNP . . . . . . . . . . . . . . .      Number  0043
PROFILE. . . . . . . . . . . . . .      Number  0064
PZW1 . . . . . . . . . . . . . . .      L NEAR  011D    CODE
QKCMCOM. . . . . . . . . . . . . .      Number  0015
QKFMTCOM . . . . . . . . . . . . .      Number  0024
QKRDCOM. . . . . . . . . . . . . .      Number  0013
QKVFYCOM . . . . . . . . . . . . .      Number  0023
QKWTCOM. . . . . . . . . . . . . .      Number  0014
QKWVFCOM . . . . . . . . . . . . .      Number  0016
REDFNDL. . . . . . . . . . . . . .      L NEAR  0100    CODE
ROM. . . . . . . . . . . . . . . .      Number  0018
RX50 . . . . . . . . . . . . . . .      Number  0000
RX50VAL. . . . . . . . . . . . . .      Number  0000
SECNO. . . . . . . . . . . . . . .      V BYTE  0000            External
SECSIZ . . . . . . . . . . . . . .      Number  000A
SECTOR . . . . . . . . . . . . . .      Number  0003
SKEWTBL_2. . . . . . . . . . . . .      L BYTE  01EE    CODE
SKWED. . . . . . . . . . . . . . .      L NEAR  00C5    CODE
SPARE41. . . . . . . . . . . . . .      Number  0041
SPARE42. . . . . . . . . . . . . .      Number  0042
SPT. . . . . . . . . . . . . . . .      Number  0010
STATUS . . . . . . . . . . . . . .      V BYTE  0000            External
STDRV. . . . . . . . . . . . . . .      L NEAR  00AC    CODE
TFORMAT. . . . . . . . . . . . . .      V BYTE  0000            External
TRACK. . . . . . . . . . . . . . .      Number  0002
TRACKN . . . . . . . . . . . . . .      V BYTE  0000            External
TTRACK . . . . . . . . . . . . . .      V BYTE  0000            External
UART . . . . . . . . . . . . . . .      Number  0046
VERT . . . . . . . . . . . . . . .      Number  0040
WPRSNT . . . . . . . . . . . . . .      Number  0001
XCOMMAND . . . . . . . . . . . . .      Number  0000
XCOUNT . . . . . . . . . . . . . .      Number  0006
XCPRSNT. . . . . . . . . . . . . .      Number  0002
XDMAOFF. . . . . . . . . . . . . .      Number  0008
XDMASEG. . . . . . . . . . . . . .      Number  000A
XDRIVE . . . . . . . . . . . . . .      Number  0001
XDSKFCN. . . . . . . . . . . . . .      L NEAR  017F    CODE
XFRPKT . . . . . . . . . . . . . .      V WORD  0000            External
XPHDRV . . . . . . . . . . . . . .      Number  0003
XSECTOR. . . . . . . . . . . . . .      Number  0002
XTRACK . . . . . . . . . . . . . .      Number  0004
XZ0. . . . . . . . . . . . . . . .      L NEAR  01B2    CODE
XZ1. . . . . . . . . . . . . . . .      L NEAR  01BA    CODE
XZ2. . . . . . . . . . . . . . . .      L NEAR  01C7    CODE
XZ3. . . . . . . . . . . . . . . .      L NEAR  01E8    CODE
XZ4. . . . . . . . . . . . . . . .      L NEAR  01E1    CODE
```

```
XZ5. . . . . . . . . . . . . . . .      L NEAR  01EC    CODE
XZ80ERR. . . . . . . . . . . . . .      L NEAR  01D2    CODE
Z80. . . . . . . . . . . . . . . .      Number  0047
Z80ISR . . . . . . . . . . . . . .      L NEAR  0187    CODE    Global
ZDSKFCN. . . . . . . . . . . . . .      L BYTE  0065    CODE
ZI1. . . . . . . . . . . . . . . .      L NEAR  0197    CODE
ZPKT . . . . . . . . . . . . . . .      L WORD  0185    CODE
ZWO. . . . . . . . . . . . . . . .      L NEAR  002F    CODE

Warning Severe
Errors  Errors
0       0
```

```
 DEC Rainbow Rx-50/Z80 disk driver

 Symbol Cross Reference                    (# is definition)       Cref-1

AUXDP. . . . . . . . . . . . . . .   83#
AUXP . . . . . . . . . . . . . . .   85#
AUX_PRN. . . . . . . . . . . . . .   73#

BUFFER . . . . . . . . . . . . . .   18#    284     306     383

CGROUP . . . . . . . . . . . . . .   11     142     142
CLK_INT. . . . . . . . . . . . . .   59#
CODE . . . . . . . . . . . . . . .   11     141#    141     489
COMDMA . . . . . . . . . . . . . .   72#
COMMAND. . . . . . . . . . . . . .   98#    195
COUNT. . . . . . . . . . . . . . .  100#    180
CURTRK . . . . . . . . . . . . . .  101#

DECDISK. . . . . . . . . . . . . .   17     150#
DENSITY. . . . . . . . . . . . . .  102#    188
DMAADR . . . . . . . . . . . . . .  138#    193     313     342
DMAOFF . . . . . . . . . . . . . .  107#    192
DMASEG . . . . . . . . . . . . . .  108#    187
DRIVE. . . . . . . . . . . . . . .   97#    155
DRVNO. . . . . . . . . . . . . . .  132#    160     286
DSKIO. . . . . . . . . . . . . . .   80#
DTL. . . . . . . . . . . . . . . .  105#

EGGSIT . . . . . . . . . . . . . .  330     348#
ENN. . . . . . . . . . . . . . . .  104#
ERRCODE. . . . . . . . . . . . . .  451     478#
EXCOM. . . . . . . . . . . . . . .   74#
EXECZ80. . . . . . . . . . . . . .  203     325     410#
EXSTAT . . . . . . . . . . . . . .  137#    440

FMTFNDL. . . . . . . . . . . . . .  286     307#
FMTSET . . . . . . . . . . . . . .  336     340     342#
FNCCOD . . . . . . . . . . . . . .  130#    199     316

GAPLEN . . . . . . . . . . . . . .  103#
GSCR . . . . . . . . . . . . . . .   30#

HEAD . . . . . . . . . . . . . . .  110#

I88PKT . . . . . . . . . . . . . .  125#
IBM8 . . . . . . . . . . . . . . .   43#    337
IBM8VAL. . . . . . . . . . . . . .   47#    339
IBM9 . . . . . . . . . . . . . . .   44#
IBM9VAL. . . . . . . . . . . . . .   48#    341
INTZ80 . . . . . . . . . . . . . .   29#    399     414

KDP. . . . . . . . . . . . . . . .   81#
KSP. . . . . . . . . . . . . . . .   82#

LOGDRV . . . . . . . . . . . . . .  259     262#

MC_FLG . . . . . . . . . . . . . .   18#    242     244
MOV1ST . . . . . . . . . . . . . .  287#

NSECT. . . . . . . . . . . . . . .  135#    181     276
```

```
 DEC Rainbow Rx-50/Z80 disk driver

 Symbol Cross Reference                    (# is definition)       Cref-2

OTERFMT. . . . . . . . . . . . . .  338     341#
OUTHERE. . . . . . . . . . . . . .  353     371#

PACKET . . . . . . . . . . . . . .  129#    411
PACKET_ADR . . . . . . . . . . . .  118#
PHYS1. . . . . . . . . . . . . . .  232     247#
PHYS2. . . . . . . . . . . . . . .  238     242#
PHYS3. . . . . . . . . . . . . . .  241     243#
PHYSDISK . . . . . . . . . . . . .   17     227#
PHYSDISK1. . . . . . . . . . . . .  228#    381
PRNDP. . . . . . . . . . . . . . .   84#
PRNP . . . . . . . . . . . . . . .   86#
PROFILE. . . . . . . . . . . . . .   58#
PZW1 . . . . . . . . . . . . . . .  320#

QKCMCOM. . . . . . . . . . . . . .   24#    209     379
QKFMTCOM . . . . . . . . . . . . .   27#    378
QKRDCOM. . . . . . . . . . . . . .   22#    209     375
QKVFYCOM . . . . . . . . . . . . .   26#    380
QKWTCOM. . . . . . . . . . . . . .   23#    209     376
QKWVFCOM . . . . . . . . . . . . .   25#    209     377

REDFNDL. . . . . . . . . . . . . .  282     304#
ROM. . . . . . . . . . . . . . . .   77#
RX50 . . . . . . . . . . . . . . .   42#
RX50VAL. . . . . . . . . . . . . .   46#

SECNO. . . . . . . . . . . . . . .  133#    177     275
SECSIZ . . . . . . . . . . . . . .  106#
SECTOR . . . . . . . . . . . . . .   99#    167
SKEWTBL_2. . . . . . . . . . . . .  175     273     462#
SKWED. . . . . . . . . . . . . . .  271     274#
SPARE41. . . . . . . . . . . . . .   70#
SPARE42. . . . . . . . . . . . . .   71#
SPT. . . . . . . . . . . . . . . .  109#
STATUS . . . . . . . . . . . . . .  131#    413     432
STDRV. . . . . . . . . . . . . . .  261     266#

TFORMAT. . . . . . . . . . . . . .  119#    334
TRACK. . . . . . . . . . . . . . .   98#    170     178
TRACKN . . . . . . . . . . . . . .  134#    179     274
TTRACK . . . . . . . . . . . . . .  120#

UART . . . . . . . . . . . . . . .   75#

VERT . . . . . . . . . . . . . . .   69#

WPRSNT . . . . . . . . . . . . . .   90#

XCOMMAND . . . . . . . . . . . . .   32#    230     279     352
XCOUNT . . . . . . . . . . . . . .   37#
XCPRSNT. . . . . . . . . . . . . .   91#
XDMAOFF. . . . . . . . . . . . . .   38#    284     360
XDMASEG. . . . . . . . . . . . . .   39#    283     359
XDRIVE . . . . . . . . . . . . . .   33#    257
XDSKFCN. . . . . . . . . . . . . .  315     375#
XFRPKT . . . . . . . . . . . . . .  124#    397     411     419
```

```
 DEC Rainbow Rx-50/Z80 disk driver

 Symbol Cross Reference                    (# is definition)       Cref-3

XPHDRV . . . . . . . . . . . . . .      35#   260
XSECTOR. . . . . . . . . . . . . .      34#   269
XTRACK . . . . . . . . . . . . . .      36#   267
XZ0. . . . . . . . . . . . . . . .     419#   420
XZ1. . . . . . . . . . . . . . . .     424#   429
XZ2. . . . . . . . . . . . . . . .     426    431#
XZ3. . . . . . . . . . . . . . . .     447    451#
XZ4. . . . . . . . . . . . . . . .     442    446#   449
XZ5. . . . . . . . . . . . . . . .     444    452#
XZ80ERR. . . . . . . . . . . . . .     434    438#

Z80. . . . . . . . . . . . . . . .      76#
Z80ISR . . . . . . . . . . . . . .      17    393#
Z80SEG . . . . . . . . . . . . . .     115#   116    142    151    253    395    411
ZDSKFCN. . . . . . . . . . . . . .     198    209#
ZI1. . . . . . . . . . . . . . . .     399#
ZPKT . . . . . . . . . . . . . . .     391#   398    412    425
ZWO. . . . . . . . . . . . . . . .     169    171    177#
```

```
 The Microsoft MACRO Assembler                    02-20-84      PAGE     1-1
Rainbow Interrupt fix routines

                                                 PAGE    60,132
                                                 TITLE   Rainbow Interrupt fix routines
                                                 NAME    INTFIX

                                          ;
                                          ;      COMPANY CONFIDENTIAL
                                          ;      Copyright (C) 1983 Digital Equipment Corporation
                                          ;      All rights reserved.
                                          ;

                                          cgroup group code
                                          ;
                                          ;Placed at the 2X interrupts, dispatch to
                                          ;either the proper MSDOS interrupt or
                                          ;the hardware vector, determined by looking
                                          ;at the last executed instruction.
                                          ;
                                          public  int20,int21,int24,int25,int26,int27
                                          public  dos20vec,dos21vec,dos24vec
                                          public  dos25vec,dos26vec,dos27vec

        0000                              code segment byte public 'code'
                                          assume cs:cgroup,ds:nothing
                                      C   include iodef.ash
                                      C   ;Rainbow Interrupt numbers.
                                      C   ;1-Apr-83 sgs added dskio int vector
                                      C   ;19-Mar-83 sgs added profile int vector [user clock]
                                      C   ;the int profile is called by the RTC interrupt service for each tick.
                                      C   ;The ax and ds register don't need to be saved [done in clkisr].
                                      C   ;
        = 0064                        C   profile equu     64h      ;100. user interface to clock interrupt
        = 002C                        C   clk_int equ      2ch      ;60Hz int,
        = 0065                        C   dskio   equ      65h      ;direct disk io for format
                                      C   ;
                                      C   ;17-Mar-83 sgs changed to include int 20-26
                                      C   ;interrupt 22-24h are duplicated at 42-44h for consistency.
                                      C   ;These are the relocated Rainbow interrupts.
                                      C   ;interrupts, 20h-26h moved to 40h-46h
                                      C   ;ATTENTION Modules IDINIT and REINIT may have to be
                                      C   ;changed if int vectors 40h to 46h are changed
                                      C   ;
        = 0040                        C   vert    equ      40h      ;new vert. freq.
        = 0041                        C   spare41 equ      41h
        = 0042                        C   spare42 equ      42h
        = 0043                        C   comdma  equ      43h      ;DMA ctrl optional comm. board
        = 0044                        C   aux_prn equ      44h      ;7201 comm./printer
        = 0045                        C   excom   equ      45h      ;extented comm. option
        = 0046                        C   uart    equ      46h      ;new UART vector,
        = 0047                        C   z80     equ      47h      ;Z80 interrupt,
        = 0018                        C   rom     equ      18h      ;new ROM access,
                                      C   ;
                                      C   ;DEC Rainbow IO port stuff.
                                      C   ;
        = 0010                        C   kdp     equ      10h      ;8251 data port,
        = 0011                        C   ksp     equ      11h      ;8251 status,
 = 0040                    C  auxdp    equ     40h     ;7201 data,
 = 0041                    C  prndp    equ     41h
 = 0042                    C  auxp     equ     42h     ;7201 command,
 = 0043                    C  prnp     equ     43h
                           C  ;
                           C  ; Values in XOPTION
                           C  ;
 = 0001                    C  wprsnt   equ     01      ;Winnie preset
 = 0002                    C  xcprsnt  equ     02      ;XCOMM present
```

```
                             page
                             ;
                             ;Dword pointers into the DOS.
                             ;ATTENTION must be one block for block move
                             ;
 0000  ????????              dos20vec dd (?)
 0004  ????????              dos21vec dd (?)
 0008  ????????              dos24vec dd (?)
 000C  ????????              dos25vec dd (?)
 0010  ????????              dos26vec dd (?)
 0014  ????????              dos27vec dd (?)
                             ;
                             ;INT 20 code. We must tell a software interrupt
                             ;from a hardware one.
                             ;
                             ;
                             ;int_20_code:
                             ;
                             ;       if flag set,
                             ;               hardware_interrupt,
                             ;               clear flag,
                             ;               iret.
                             ;
                             ;       set flag,
                             ;       enable interrupts,
                             ;       nop,
                             ;       disable interrupts,
                             ;       if CD 20 on stack,
                             ;               if flag20 set or IP == 2,
                             ;                       clear flag,
                             ;                       software_interrupt,
                             ;       iret.
                             ;
```

```
page
intmac  macro   nn,hwi
;
;Attempt to figure out where to go. The "soft
;parts" (our ambiguity) are marked with <*>.
;
flag&nn db      0

int&nn proc far
        dec     cs:flag&nn      ;<*> if flag was
        jnz     i&nn&a          ;<*> set, must
        int     hwi             ;be hardware
        iret                    ;service it.
;
;Dont know what type of interrupt yet. NOTE:
;The STI is done first: any pending interrupt
;will NOT be serviced until the MOV FLAG,1 is
;executed. Absolute minimum ambiguity.
;
i&nn&a: sti                     ;<*> enable ints,
        mov     cs:flag&nn,1    ;<*> set flag,
        cli                     ;off again,
        cmp     cs:flag&nn,0    ;if flag clear,
        jnz     i&nn&b          ;it's hardware
;
;Hmm. Flag cleared. Either a hardware interrupt
;or a software interrupt that got interrupted
;in our soft parts. All we can do is assume
;it was a hardware interrupt. Goodbye.
;
        iret
;
;We reenabled interrupts, and didn't get inter-
;rupted. Must be software.
;
i&nn&b: mov     cs:flag&nn,0
        jmp     dword ptr dos&nn&vec

int&nn endp
        endm
```

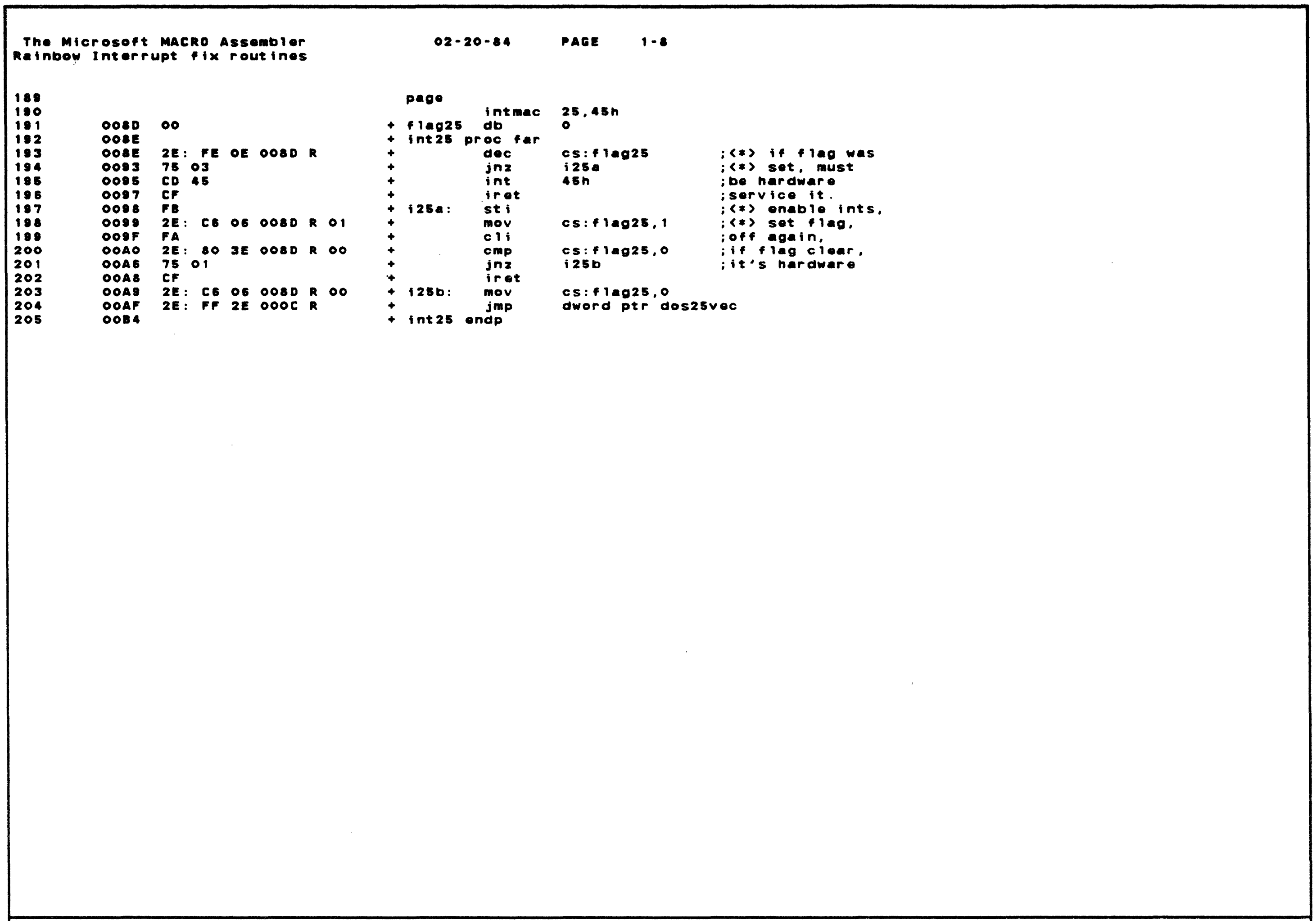
```
                                    page
                                            intmac  20,vert
0018  00                          + flag20  db      0
0019                              + int20 proc far
0019  2E: FE 0E 0018 R            +         dec     cs:flag20       ;<*> if flag was
001E  75 03                       +         jnz     i20a            ;<*> set, must
0020  CD 40                       +         int     vert            ;be hardware
0022  CF                          +         iret                    ;service it.
0023  FB                          + i20a:   sti                     ;<*> enable ints,
0024  2E: C6 06 0018 R 01         +         mov     cs:flag20,1     ;<*> set flag,
002A  FA                          +         cli                     ;off again,
002B  2E: 80 3E 0018 R 00         +         cmp     cs:flag20,0     ;if flag clear,
0031  75 01                       +         jnz     i20b            ;it's hardware
0033  CF                          +         iret
0034  2E: C6 06 0018 R 00         + i20b:   mov     cs:flag20,0
003A  2E: FF 2E 0000 R            +         jmp     dword ptr dos20vec
003F                              + int20 endp
```

Rainbow Interrupt fix routines

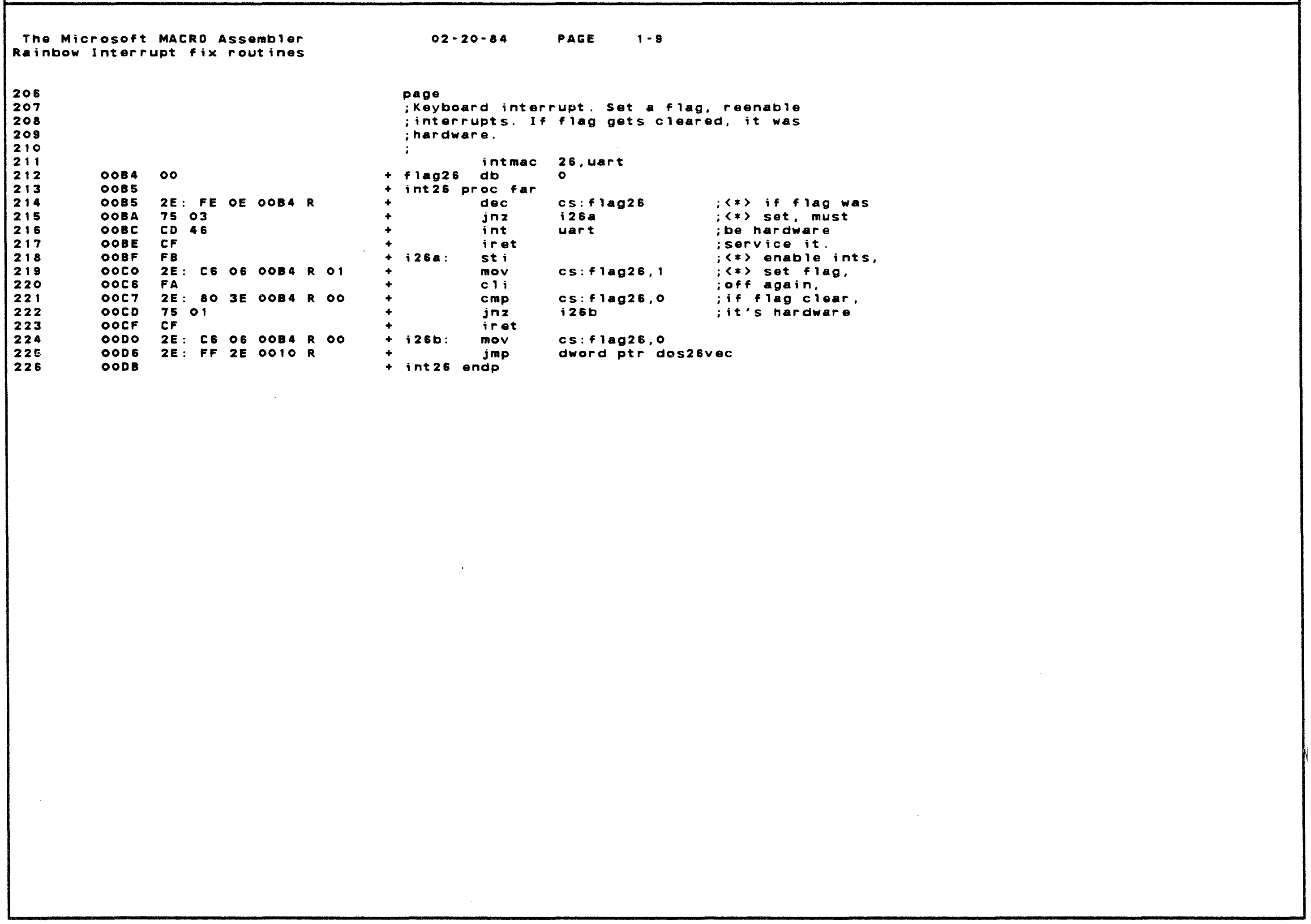
```
                                         page
                                                intmac  21,41h
        003F  00                       + flag21  db      0
        0040                           + int21 proc far
        0040  2E: FE 0E 003F R         +         dec     cs:flag21        ;(*) if flag was
        0045  75 03                    +         jnz     i21a             ;(*) set, must
        0047  CD 41                    +         int     41h              ;be hardware
        0049  CF                       +         iret                     ;service it.
        004A  FB                       + i21a:   sti                      ;(*) enable ints,
        004B  2E: C6 06 003F R 01      +         mov     cs:flag21,1      ;(*) set flag,
        0051  FA                       +         cli                      ;off again,
        0052  2E: 80 3E 003F R 00      +         cmp     cs:flag21,0      ;if flag clear,
        0058  75 01                    +         jnz     i21b             ;it's hardware
        005A  CF                       +         iret
        005B  2E: C6 06 003F R 00      + i21b:   mov     cs:flag21,0
        0061  2E: FF 2E 0004 R         +         jmp     dword ptr dos21vec
        0066                           + int21 endp
```

Rainbow Interrupt fix routines

```
                                         page
                                                intmac  24,44h
        0066  00                       + flag24  db      0
        0067                           + int24 proc far
        0067  2E: FE 0E 0066 R         +         dec     cs:flag24        ;(*) if flag was
        006C  75 03                    +         jnz     i24a             ;(*) set, must
        006E  CD 44                    +         int     44h              ;be hardware
        0070  CF                       +         iret                     ;service it.
        0071  FB                       + i24a:   sti                      ;(*) enable ints,
        0072  2E: C6 06 0066 R 01      +         mov     cs:flag24,1      ;(*) set flag,
        0078  FA                       +         cli                      ;off again,
        0079  2E: 80 3E 0066 R 00      +         cmp     cs:flag24,0      ;if flag clear,
        007F  75 01                    +         jnz     i24b             ;it's hardware
        0081  CF                       +         iret
        0082  2E: C6 06 0066 R 00      + i24b:   mov     cs:flag24,0
        0088  2E: FF 2E 0008 R         +         jmp     dword ptr dos24vec
        008D                           + int24 endp
```



```
 The Microsoft MACRO Assembler             02-20-84      PAGE    1-8
Rainbow Interrupt fix routines


189                                        page
190                                                intmac  25,45h
191      008D  00                        + flag25  db      0
192      008E                            + int25 proc far
193      008E  2E: FE 0E 008D R          +         dec     cs:flag25        ;<*> if flag was
194      0093  75 03                     +         jnz     i25a             ;<*> set, must
195      0095  CD 45                     +         int     45h              ;be hardware
196      0097  CF                        +         iret                     ;service it.
197      0098  FB                        + i25a:   sti                      ;<*> enable ints,
198      0099  2E: C6 06 008D R 01       +         mov     cs:flag25,1      ;<*> set flag,
199      009F  FA                        +         cli                      ;off again,
200      00A0  2E: 80 3E 008D R 00       +         cmp     cs:flag25,0      ;if flag clear,
201      00A6  75 01                     +         jnz     i25b             ;it's hardware
202      00A8  CF                        +         iret
203      00A9  2E: C6 06 008D R 00       + i25b:   mov     cs:flag25,0
204      00AF  2E: FF 2E 000C R          +         jmp     dword ptr dos25vec
205      00B4                            + int25 endp
```



```
 The Microsoft MACRO Assembler             02-20-84      PAGE    1-9
Rainbow Interrupt fix routines


206                                        page
207                                        ;Keyboard interrupt. Set a flag, reenable
208                                        ;interrupts. If flag gets cleared, it was
209                                        ;hardware.
210                                        ;
211                                                intmac  26,uart
212      00B4  00                        + flag26  db      0
213      00B5                            + int26 proc far
214      00B5  2E: FE 0E 00B4 R          +         dec     cs:flag26        ;<*> if flag was
215      00BA  75 03                     +         jnz     i26a             ;<*> set, must
216      00BC  CD 46                     +         int     uart             ;be hardware
217      00BE  CF                        +         iret                     ;service it.
218      00BF  FB                        + i26a:   sti                      ;<*> enable ints,
219      00C0  2E: C6 06 00B4 R 01       +         mov     cs:flag26,1      ;<*> set flag,
220      00C6  FA                        +         cli                      ;off again,
221      00C7  2E: 80 3E 00B4 R 00       +         cmp     cs:flag26,0      ;if flag clear,
222      00CD  75 01                     +         jnz     i26b             ;it's hardware
223      00CF  CF                        +         iret
224      00D0  2E: C6 06 00B4 R 00       + i26b:   mov     cs:flag26,0
225      00D6  2E: FF 2E 0010 R          +         jmp     dword ptr dos26vec
226      00DB                            + int26 endp
```

```
                                           page
                                           ;Z80 interrupt. Set the flag, if we get
                                           ;reinterrupted, it was the Z80.
                                           ;
                                                   intmac  27,z80
     00DB  00                            + flag27  db      0
     00DC                                + int27 proc far
     00DC  2E: FE 0E 00DB R              +         dec     cs:flag27        ;(*) if flag was
     00E1  75 03                         +         jnz     i27a             ;(*) set, must
     00E3  CD 47                         +         int     z80              ;be hardware
     00E5  CF                            +         iret                     ;service it.
     00E6  FB                            + i27a:   sti                      ;(*) enable ints,
     00E7  2E: C6 06 00DB R 01           +         mov     cs:flag27,1      ;(*) set flag,
     00ED  FA                            +         cli                      ;off again,
     00EE  2E: 80 3E 00DB R 00           +         cmp     cs:flag27,0      ;if flag clear,
     00F4  75 01                         +         jnz     i27b             ;it's hardware
     00F6  CF                            +         iret
     00F7  2E: C6 06 00DB R 00           + i27b:   mov     cs:flag27,0
     00FD  2E: FF 2E 0014 R              +         jmp     dword ptr dos27vec
     0102                                + int27 endp

     0102                                  code ends

                                                   end
```

```
Macros:

                Name                    Length

INTMAC . . . . . . . . . . . . . .      001C

Segments and groups:

                Name                    Size    align   combine class

CGROUP . . . . . . . . . . . . . .      GROUP
  CODE . . . . . . . . . . . . . . .    0102    BYTE    PUBLIC  'CODE'

Symbols:

                Name                    Type    Value   Attr

AUXDP. . . . . . . . . . . . . . .      Number  0040
AUXP . . . . . . . . . . . . . . .      Number  0042
AUX_PRN. . . . . . . . . . . . . .      Number  0044
CLK_INT. . . . . . . . . . . . . .      Number  002C
COMDMA . . . . . . . . . . . . . .      Number  0043
DOS20VEC . . . . . . . . . . . . .      L DWORD 0000    CODE    Global
DOS21VEC . . . . . . . . . . . . .      L DWORD 0004    CODE    Global
DOS24VEC . . . . . . . . . . . . .      L DWORD 0008    CODE    Global
DOS25VEC . . . . . . . . . . . . .      L DWORD 000C    CODE    Global
DOS26VEC . . . . . . . . . . . . .      L DWORD 0010    CODE    Global
DOS27VEC . . . . . . . . . . . . .      L DWORD 0014    CODE    Global
DSKIO. . . . . . . . . . . . . . .      Number  0065
EXCOM. . . . . . . . . . . . . . .      Number  0045
FLAG20 . . . . . . . . . . . . . .      L BYTE  0018    CODE
FLAG21 . . . . . . . . . . . . . .      L BYTE  003F    CODE
FLAG24 . . . . . . . . . . . . . .      L BYTE  0066    CODE
FLAG25 . . . . . . . . . . . . . .      L BYTE  008D    CODE
FLAG26 . . . . . . . . . . . . . .      L BYTE  00B4    CODE
FLAG27 . . . . . . . . . . . . . .      L BYTE  00DB    CODE
I20A . . . . . . . . . . . . . . .      L NEAR  0023    CODE
I20B . . . . . . . . . . . . . . .      L NEAR  0034    CODE
I21A . . . . . . . . . . . . . . .      L NEAR  004A    CODE
I21B . . . . . . . . . . . . . . .      L NEAR  005B    CODE
I24A . . . . . . . . . . . . . . .      L NEAR  0071    CODE
I24B . . . . . . . . . . . . . . .      L NEAR  0082    CODE
I25A . . . . . . . . . . . . . . .      L NEAR  0098    CODE
I25B . . . . . . . . . . . . . . .      L NEAR  00A9    CODE
I26A . . . . . . . . . . . . . . .      L NEAR  00BF    CODE
I26B . . . . . . . . . . . . . . .      L NEAR  00D0    CODE
I27A . . . . . . . . . . . . . . .      L NEAR  00E6    CODE
I27B . . . . . . . . . . . . . . .      L NEAR  00F7    CODE
INT20. . . . . . . . . . . . . . .      F PROC  0019    CODE    Global  Length =0026
INT21. . . . . . . . . . . . . . .      F PROC  0040    CODE    Global  Length =0026
INT24. . . . . . . . . . . . . . .      F PROC  0067    CODE    Global  Length =0026
INT25. . . . . . . . . . . . . . .      F PROC  008E    CODE    Global  Length =0026
INT26. . . . . . . . . . . . . . .      F PROC  00B5    CODE    Global  Length =0026
INT27. . . . . . . . . . . . . . .      F PROC  00DC    CODE    Global  Length =0026
KDP. . . . . . . . . . . . . . . .      Number  0010
```

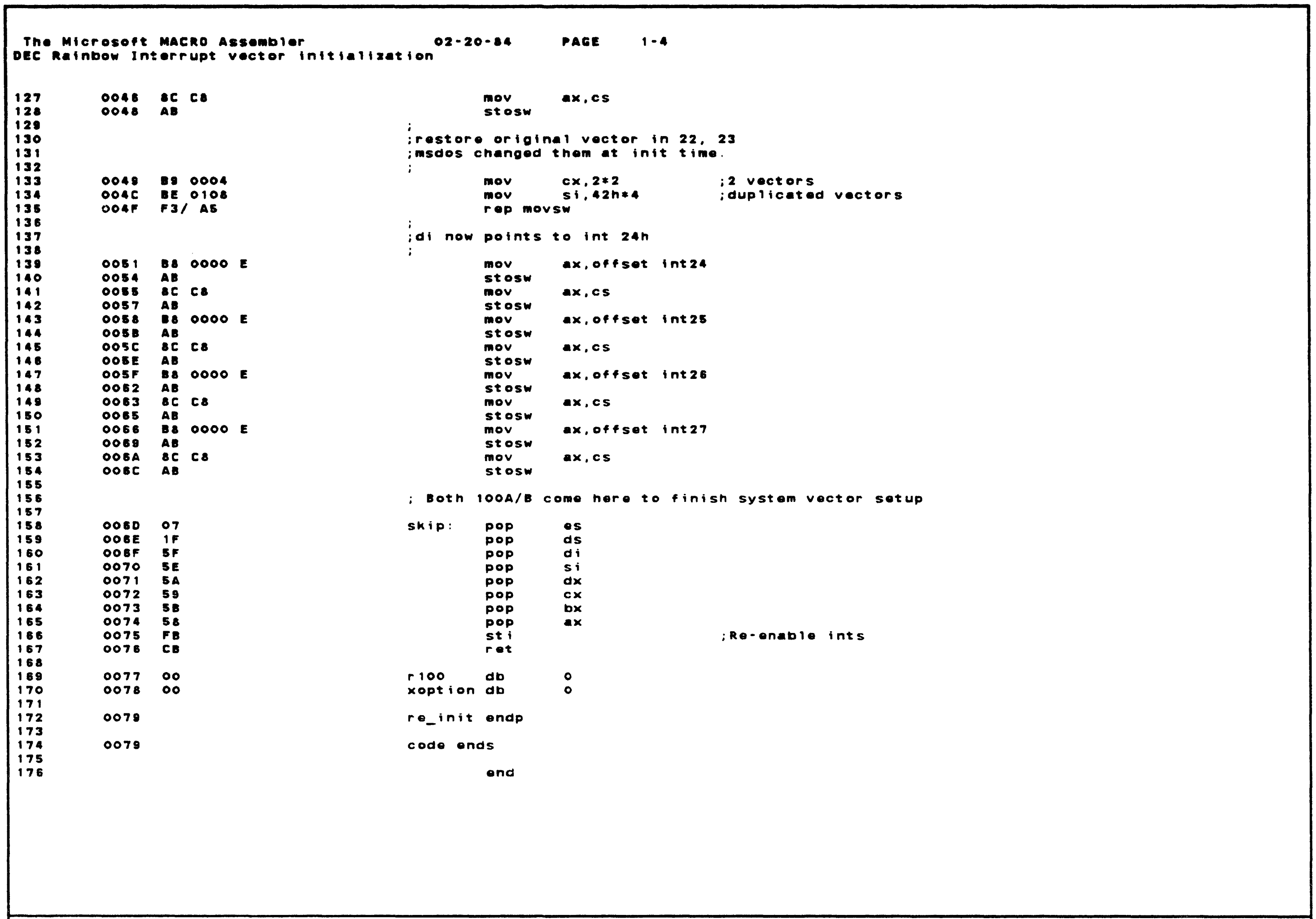
```
 The Microsoft MACRO Assembler              02-20-84      PAGE    Symbols-2
Rainbow Interrupt fix routines


KSP. . . . . . . . . . . . . . .                Number   0011
PRNDP. . . . . . . . . . . . . .                Number   0041
PRNP . . . . . . . . . . . . . .                Number   0043
PROFILE. . . . . . . . . . . . .                Number   0064
ROM. . . . . . . . . . . . . . .                Number   0018
SPARE41. . . . . . . . . . . . .                Number   0041
SPARE42. . . . . . . . . . . . .                Number   0042
UART . . . . . . . . . . . . . .                Number   0046
VERT . . . . . . . . . . . . . .                Number   0040
WPRSNT . . . . . . . . . . . . .                Number   0001
XCPRSNT. . . . . . . . . . . . .                Number   0002
Z80. . . . . . . . . . . . . . .                Number   0047

Warning Severe
Errors  Errors
0       0
```

```
 Rainbow Interrupt fix routines

 Symbol Cross Reference                    (# is definition)        Cref-1

AUXDP. . . . . . . . . . . . . .          56#
AUXP . . . . . . . . . . . . . .          58#
AUX_PRN. . . . . . . . . . . . .          46#

CGROUP . . . . . . . . . . . . .           1      23
CLK_INT. . . . . . . . . . . . .          32#
CODE . . . . . . . . . . . . . .          11      22#     22      248
COMDMA . . . . . . . . . . . . .          45#

DOS20VEC . . . . . . . . . . . .          19      70#    153
DOS21VEC . . . . . . . . . . . .          19      71#    170
DOS24VEC . . . . . . . . . . . .          19      72#    187
DOS25VEC . . . . . . . . . . . .          20      73#    204
DOS26VEC . . . . . . . . . . . .          20      74#    225
DOS27VEC . . . . . . . . . . . .          20      75#    245
DSKIO. . . . . . . . . . . . . .          33#

EXCOM. . . . . . . . . . . . . .          47#

FLAG20 . . . . . . . . . . . . .         140#    142     147     149     152
FLAG21 . . . . . . . . . . . . .         157#    159     164     166     169
FLAG24 . . . . . . . . . . . . .         174#    176     181     183     186
FLAG25 . . . . . . . . . . . . .         191#    193     198     200     203
FLAG26 . . . . . . . . . . . . .         212#    214     219     221     224
FLAG27 . . . . . . . . . . . . .         232#    234     239     241     244

I20A . . . . . . . . . . . . . .         143     146#
I20B . . . . . . . . . . . . . .         150     152#
I21A . . . . . . . . . . . . . .         160     163#
I21B . . . . . . . . . . . . . .         167     169#
I24A . . . . . . . . . . . . . .         177     180#
I24B . . . . . . . . . . . . . .         184     186#
I25A . . . . . . . . . . . . . .         194     197#
I25B . . . . . . . . . . . . . .         201     203#
I26A . . . . . . . . . . . . . .         215     218#
I26B . . . . . . . . . . . . . .         222     224#
I27A . . . . . . . . . . . . . .         235     238#
I27B . . . . . . . . . . . . . .         242     244#
INT20. . . . . . . . . . . . . .          18     141#    154
INT21. . . . . . . . . . . . . .          18     158#    171
INT24. . . . . . . . . . . . . .          18     175#    188
INT25. . . . . . . . . . . . . .          18     192#    205
INT26. . . . . . . . . . . . . .          18     213#    226
INT27. . . . . . . . . . . . . .          18     233#    246
INTMAC . . . . . . . . . . . . .         139     156     173     190     211     231

KDP. . . . . . . . . . . . . . .          54#
KSP. . . . . . . . . . . . . . .          55#

PRNDP. . . . . . . . . . . . . .          57#
PRNP . . . . . . . . . . . . . .          59#
PROFILE. . . . . . . . . . . . .          31#

ROM. . . . . . . . . . . . . . .          50#

SPARE41. . . . . . . . . . . . .          43#
SPARE42. . . . . . . . . . . . .          44#
```

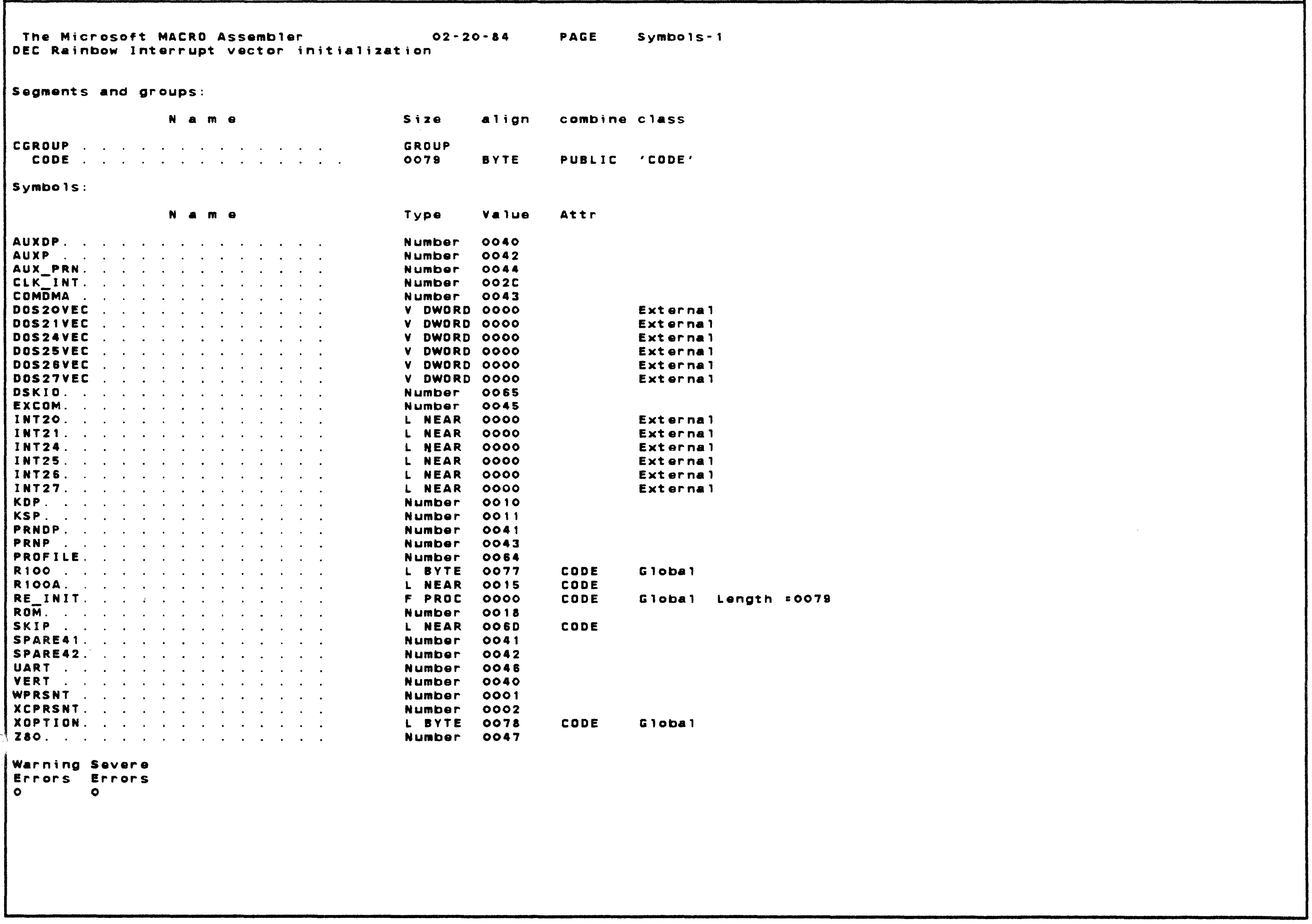
```
 Rainbow Interrupt fix routines

 Symbol Cross Reference                    (# is definition)       Cref-2

UART . . . . . . . . . . . . . . . .   48#   216

VERT . . . . . . . . . . . . . . . .   42#   144

WPRSNT . . . . . . . . . . . . . . .   63#

XCPRSNT. . . . . . . . . . . . . . .   64#

Z80. . . . . . . . . . . . . . . . .   49#   236
```

```
 The Microsoft MACRO Assembler              02-20-84       PAGE     1-1
DEC Rainbow Interrupt vector initialization


1                                                      PAGE     60,132
2                                                      TITLE    DEC Rainbow Interrupt vector initialization
3                                                      NAME     REINIT
4
5                                              ;
6                                              ;       COMPANY CONFIDENTIAL
7                                              ;       Copyright (C) 1983 Digital Equipment Corporation
8                                              ;       All rights reserved.
9                                              ;
10
11                                             cgroup group code
12                                             ;
13                                             ;       Copies the MSDOS interrupt vectors
14                                             ;from the 2x block to the Fx block, installs
15                                             ;the magic routine at 2x, which will determine
16                                             ;the interrupt source, and dispatch to the right
17                                             ;vector.
18                                             ;
19
20                                             public re_init,r100,xoption
21
22                                             extrn   int20:near,int21:near
23                                             extrn   int25:near,int24:near
24                                             extrn   int26:near,int27:near
25                                             extrn   dos20vec:dword,dos21vec:dword
26                                             extrn   dos25vec:dword,dos24vec:dword
27                                             extrn   dos26vec:dword,dos27vec:dword
28
29         0000                                code segment byte public 'code'
30                                             assume cs:cgroup,ds:cgroup
31                                         C   include iodef.ash
32                                         C   ;Rainbow Interrupt numbers.
33                                         C   ;1-Apr-83 sgs added dskio int vector
34                                         C   ;19-Mar-83 sgs added profile int vector [user clock]
35                                         C   ;the int profile is called by the RTC interrupt service for each tick.
36                                         C   ;The ax and ds register don't need to be saved [done in clkisr].
37                                         C   ;
38         = 0064                          C   profile equu     64h      ;100. user interface to clock interrupt
39         = 002C                          C   clk_int equ      2ch      ;60Hz int,
40         = 0065                          C   dskio   equ      65h      ;direct disk io for format
41                                         C   ;
42                                         C   ;17-Mar-83 sgs changed to include int 20-26
43                                         C   ;interrupt 22-24h are duplicated at 42-44h for consistency.
44                                         C   ;These are the relocated Rainbow interrupts.
45                                         C   ;interrupts, 20h-26h moved to 40h-46h
46                                         C   ;ATTENTION Modules IOINIT and REINIT may have to be
47                                         C   ;changed if int vectors 40h to 46h are changed
48                                         C   ;
49         = 0040                          C   vert    equ      40h      ;new vert. freq.
50         = 0041                          C   spare41 equ      41h
51         = 0042                          C   spare42 equ      42h
52         = 0043                          C   comdma  equ      43h      ;DMA ctrl optional comm. board
53         = 0044                          C   aux_prn equ      44h      ;7201 comm./printer
54         = 0045                          C   excom   equ      45h      ;extented comm. option
55         = 0046                          C   uart    equ      46h      ;new UART vector,
56        = 0047                        C  z80      equ      47h       ;Z80 interrupt,
57        = 0018                        C  rom      equ      18h       ;new ROM access,
58                                      C  ;
59                                      C  ;DEC Rainbow IO port stuff.
60                                      C  ;
61        = 0010                        C  kdp      equ      10h       ;8251 data port,
62        = 0011                        C  ksp      equ      11h       ;8251 status,
63        = 0040                        C  auxdp    equ      40h       ;7201 data,
64        = 0041                        C  prndp    equ      41h
65        = 0042                        C  auxp     equ      42h       ;7201 command,
66        = 0043                        C  prnp     equ      43h
67                                      C  ;
68                                      C  ; Values in XOPTION
69                                      C  ;
70        = 0001                        C  wprsnt   equ      01        ;Winnie preset
71        = 0002                        C  xcprsnt  equ      02        ;XCOMM present
```

```
72                                         page
73
74        0000                             re_init proc far
75                                         ;
76                                         ;Copy the DOS interrupts to a safe place, then
77                                         ;install our magic routines there.
78                                         ;
79        0000  FA                                 cli
80        0001  50                                 push     ax
81        0002  53                                 push     bx
82        0003  51                                 push     cx
83        0004  52                                 push     dx
84        0005  56                                 push     si
85        0006  57                                 push     di
86        0007  1E                                 push     ds
87        0008  06                                 push     es
88
89        0009  0E                                 push     cs
90        000A  1F                                 pop      ds
91        000B  F6 06 0077 R FF                    test     byte ptr r100,0ffh        ; check 100A or 100B
92        0010  74 03                              jz       r100a
93        0012  EB 59 90                           jmp      skip
94
95                                         ;Copy INT 20, 21, 24, 25, 26 and 27 vectors to the BIOS
96                                         ;dword places. dosxxvec must be squential [intfix.asm]
97
98        0015  B8 0000                    r100a:  mov      ax,0            ;DS:= low mem,
99        0018  8E D8                              mov      ds,ax
100       001A  0E                                 push     cs
101       001B  07                                 pop      es              ;ES:= BIOS,
102
103       001C  FC                                 cld
104       001D  B9 0004                            mov      cx,2*2          ;2 vectors
105       0020  BE 0080                            mov      si,20h*4        ;save INT 20,21
106       0023  BF 0000 E                          mov      di,offset dos20vec ;in our CS:,
107       0026  F3/ A5                             rep movsw
108
109       0028  B9 0008                            mov      cx,4*2          ;4 vectors
110       002B  BE 0090                            mov      si,24h*4        ;save INT 24, 25, 26, 27
111       002E  BF 0000 E                          mov      di,offset dos24vec ;in our CS:,
112       0031  F3/ A5                             rep movsw
113                                        ;
114                                        ;Install our drivers in the interrupt vectors.
115                                        ;
116       0033  B8 0000                            mov      ax,0            ;ES:= low mem,
117       0036  8E C0                              mov      es,ax
118
119       0038  BF 0080                            mov      di,20h*4        ;install ours,
120       003B  B8 0000 E                          mov      ax,offset int20
121       003E  AB                                 stosw
122       003F  8C C8                              mov      ax,cs
123       0041  AB                                 stosw
124
125       0042  B8 0000 E                          mov      ax,offset int21
126       0045  AB                                 stosw
```



```
 The Microsoft MACRO Assembler              02-20-84      PAGE     1-4
DEC Rainbow Interrupt vector initialization


127       0046  8C C8                            mov      ax,cs
128       0048  AB                               stosw
129                                      ;
130                                      ;restore original vector in 22, 23
131                                      ;msdos changed them at init time.
132                                      ;
133       0049  B9 0004                          mov      cx,2*2          ;2 vectors
134       004C  BE 0108                          mov      si,42h*4        ;duplicated vectors
135       004F  F3/ A5                           rep movsw
136                                      ;
137                                      ;di now points to int 24h
138                                      ;
139       0051  B8 0000 E                        mov      ax,offset int24
140       0054  AB                               stosw
141       0055  8C C8                            mov      ax,cs
142       0057  AB                               stosw
143       0058  B8 0000 E                        mov      ax,offset int25
144       005B  AB                               stosw
145       005C  8C C8                            mov      ax,cs
146       005E  AB                               stosw
147       005F  B8 0000 E                        mov      ax,offset int26
148       0062  AB                               stosw
149       0063  8C C8                            mov      ax,cs
150       0065  AB                               stosw
151       0066  B8 0000 E                        mov      ax,offset int27
152       0069  AB                               stosw
153       006A  8C C8                            mov      ax,cs
154       006C  AB                               stosw
155
156                                      ; Both 100A/B come here to finish system vector setup
157
158       006D  07                       skip:   pop      es
159       006E  1F                               pop      ds
160       006F  5F                               pop      di
161       0070  5E                               pop      si
162       0071  5A                               pop      dx
163       0072  59                               pop      cx
164       0073  5B                               pop      bx
165       0074  58                               pop      ax
166       0075  FB                               sti                      ;Re-enable ints
167       0076  CB                               ret
168
169       0077  00                       r100    db       0
170       0078  00                       xoption db       0
171
172       0079                           re_init endp
173
174       0079                           code ends
175
176                                              end
```



```
 The Microsoft MACRO Assembler              02-20-84      PAGE     Symbols-1
DEC Rainbow Interrupt vector initialization


Segments and groups:

                N a m e                  Size    align    combine class

CGROUP . . . . . . . . . . . . . .       GROUP
  CODE . . . . . . . . . . . . . . .     0079    BYTE     PUBLIC  'CODE'

Symbols:

                N a m e                  Type    Value    Attr

AUXDP. . . . . . . . . . . . . . .       Number  0040
AUXP . . . . . . . . . . . . . . .       Number  0042
AUX_PRN. . . . . . . . . . . . . .       Number  0044
CLK_INT. . . . . . . . . . . . . .       Number  002C
COMDMA . . . . . . . . . . . . . .       Number  0043
DOS20VEC . . . . . . . . . . . . .       V DWORD 0000             External
DOS21VEC . . . . . . . . . . . . .       V DWORD 0000             External
DOS24VEC . . . . . . . . . . . . .       V DWORD 0000             External
DOS25VEC . . . . . . . . . . . . .       V DWORD 0000             External
DOS26VEC . . . . . . . . . . . . .       V DWORD 0000             External
DOS27VEC . . . . . . . . . . . . .       V DWORD 0000             External
DSKIO. . . . . . . . . . . . . . .       Number  0065
EXCOM. . . . . . . . . . . . . . .       Number  0045
INT20. . . . . . . . . . . . . . .       L NEAR  0000             External
INT21. . . . . . . . . . . . . . .       L NEAR  0000             External
INT24. . . . . . . . . . . . . . .       L NEAR  0000             External
INT25. . . . . . . . . . . . . . .       L NEAR  0000             External
INT26. . . . . . . . . . . . . . .       L NEAR  0000             External
INT27. . . . . . . . . . . . . . .       L NEAR  0000             External
KDP. . . . . . . . . . . . . . . .       Number  0010
KSP. . . . . . . . . . . . . . . .       Number  0011
PRNDP. . . . . . . . . . . . . . .       Number  0041
PRNP . . . . . . . . . . . . . . .       Number  0043
PROFILE. . . . . . . . . . . . . .       Number  0064
R100 . . . . . . . . . . . . . . .       L BYTE  0077     CODE    Global
R100A. . . . . . . . . . . . . . .       L NEAR  0015     CODE
RE_INIT. . . . . . . . . . . . . .       F PROC  0000     CODE    Global  Length =0079
ROM. . . . . . . . . . . . . . . .       Number  0018
SKIP . . . . . . . . . . . . . . .       L NEAR  006D     CODE
SPARE41. . . . . . . . . . . . . .       Number  0041
SPARE42. . . . . . . . . . . . . .       Number  0042
UART . . . . . . . . . . . . . . .       Number  0046
VERT . . . . . . . . . . . . . . .       Number  0040
WPRSNT . . . . . . . . . . . . . .       Number  0001
XCPRSNT. . . . . . . . . . . . . .       Number  0002
XOPTION. . . . . . . . . . . . . .       L BYTE  0078     CODE    Global
Z80. . . . . . . . . . . . . . . .       Number  0047

Warning Severe
Errors  Errors
0       0
```

```
 DEC Rainbow Interrupt vector initialization

 Symbol Cross Reference                       (# is definition)       Cref-1

AUXDP. . . . . . . . . . . . . . . .          63#
AUXP . . . . . . . . . . . . . . . .          65#
AUX_PRN. . . . . . . . . . . . . . .          53#

CGROUP . . . . . . . . . . . . . . .          11       30       30
CLK_INT. . . . . . . . . . . . . . .          39#
CODE . . . . . . . . . . . . . . . .          11       29#      29      174
COMDMA . . . . . . . . . . . . . . .          52#

DOS20VEC . . . . . . . . . . . . . .          25#     106
DOS21VEC . . . . . . . . . . . . . .          25#
DOS24VEC . . . . . . . . . . . . . .          26#     111
DOS25VEC . . . . . . . . . . . . . .          26#
DOS26VEC . . . . . . . . . . . . . .          27#
DOS27VEC . . . . . . . . . . . . . .          27#
DSKIO. . . . . . . . . . . . . . . .          40#

EXCOM. . . . . . . . . . . . . . . .          54#

INT20. . . . . . . . . . . . . . . .          22#     120
INT21. . . . . . . . . . . . . . . .          22#     125
INT24. . . . . . . . . . . . . . . .          23#     139
INT25. . . . . . . . . . . . . . . .          23#     143
INT26. . . . . . . . . . . . . . . .          24#     147
INT27. . . . . . . . . . . . . . . .          24#     151

KDP. . . . . . . . . . . . . . . . .          61#
KSP. . . . . . . . . . . . . . . . .          62#

PRNDP. . . . . . . . . . . . . . . .          64#
PRNP . . . . . . . . . . . . . . . .          66#
PROFILE. . . . . . . . . . . . . . .          38#

R100 . . . . . . . . . . . . . . . .          20       91      169#
R100A. . . . . . . . . . . . . . . .          92       98#
RE_INIT. . . . . . . . . . . . . . .          20       74#     172
ROM. . . . . . . . . . . . . . . . .          57#

SKIP . . . . . . . . . . . . . . . .          93      158#
SPARE41. . . . . . . . . . . . . . .          50#
SPARE42. . . . . . . . . . . . . . .          51#

UART . . . . . . . . . . . . . . . .          55#

VERT . . . . . . . . . . . . . . . .          49#

WPRSNT . . . . . . . . . . . . . . .          70#

XCPRSNT. . . . . . . . . . . . . . .          71#
XOPTION. . . . . . . . . . . . . . .          20      170#

Z80. . . . . . . . . . . . . . . . .          56#
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE     1-1
Disk definition tables for MSDOS 2.00

                                                  PAGE    60,132
                                                  TITLE   Disk definition tables for MSDOS 2.00
                                                  NAME    DSKTBL

                                          ;
                                          ;       COMPANY CONFIDENTIAL
                                          ;       Copyright (C) 1983 Digital Equipment Corporation
                                          ;       All rights reserved.
                                          ;

                                          cgroup group code
        0000                              code segment byte public 'code'
                                          assume cs:cgroup,ds:cgroup

                                          public  dsktbl,inittbl
                                          extrn   decdisk:near      ;;,serdsk:near
                                          ;
                                          ;Define the disk data block used by the BIOS.
                                          ;
                                          dskblk macro bpbp,dchk,drvr,ctrk,secsiz,spt,dens,drv,dtrk,gpl,enn,dtl
                                                  dw      bpbp            ;BPB ptr,
                                                  dw      dchk            ;1= fixed dsk,
                                                  dw      drvr            ;driver addr,
                                                  dw      ctrk            ;curr trk ptr,
                                                  dw      secsiz          ;sector size,
                                                  dw      spt             ;sectors/trk,
                                                  dw      dens    ;density: 0=RX 1=IBM8 2=IBM9
                                                  dw      drv             ;drive #,
                                                  dw      dtrk            ;dens. chk trk
                                                  dw      gpl             ;765 gap len,
                                                  dw      enn             ;765 N,
                                                  dw      dtl             ;765 data len,

                                                  endm
                                          ;
                                          ;Create a BPB block for the system.
                                          ;
                                          bpb macro secsiz,clssiz,ressec,fats,dirs,dsksiz,media,spf
                                                  dw      secsiz          ;sector size,
                                                  db      clssiz          ;cluster size,
                                                  dw      ressec          ;resv'd sectors
                                                  db      fats            ;# FATs,
                                                  dw      dirs            ;# dirs,
                                                  dw      dsksiz          ;total sectors,
                                                  db      media           ;media byte,
                                                  dw      spf             ;sectors/FAT,

                                                  endm
```

```
                                    page
                                    ;
                                    ;Initialization table. Merely a set of
                                    ;pointers, one per unit, to the largest
                                    ;BPB for that unit.
                                    ;
      0000  04                      inittbl db      4
      0001  004C R                          dw      bpbdec
      0003  004C R                          dw      bpbdec
      0005  004C R                          dw      bpbdec
      0007  004C R                          dw      bpbdec
                                    ;;      dw      bpbibm  ;DEBUG
                                    ;
                                    ;BIOS internal data tables. These define all
                                    ;physical characteristics and current format
                                    ;for each drive. The first byte is the number
                                    ;of drives.
                                    ;
      0009  04                      dsktbl  db      4       ;# disks,
      000A  0012 R                          dw      diska
      000C  001A R                          dw      diskb
      000E  0022 R                          dw      diskc
      0010  002A R                          dw      diskd
                                    ;;      dw      diskx   ;DEBUG
                                    ;
                                    ;For each unit there are three alternative
                                    ;formats. This table defines the three formats
                                    ;for each drive, and pointer to the byte value
                                    ;that tells which to use.
                                    ;
                                    ;;diskx dw      dmibma
                                    ;;      dw      dmibma
                                    ;;      dw      currx

      0012  0059 R                  diska   dw      deca
      0014  00B9 R                          dw      ibma
      0016  0119 R                          dw      ibm9a
      0018  0181 R                          dw      curra

      001A  0071 R                  diskb   dw      decb
      001C  00D1 R                          dw      ibmb
      001E  0131 R                          dw      ibm9b
      0020  0182 R                          dw      currb

      0022  0089 R                  diskc   dw      decc
      0024  00E9 R                          dw      ibmc
      0026  0149 R                          dw      ibm9c
      0028  0183 R                          dw      currc

      002A  00A1 R                  diskd   dw      decd
      002C  0101 R                          dw      ibmd
      002E  0161 R                          dw      ibm9d
      0030  0184 R                          dw      currd
```

```
                                    page
                                    ;
                                    ;These tables describe each logical drive's
                                    ;physical characteristics. Each logical disk
                                    ;has a table entry.
                                    ;ATTENTION the fat id byte for each table
                                    ; must be different.
                                    ;
      0032                          bpbibm: bpb     512,1,1,2,64,8*40,0feh,1
      0032  0200               +            dw      512             ;sector size,
      0034  01                 +            db      1               ;cluster size,
      0035  0001               +            dw      1               ;resv'd sectors
      0037  02                 +            db      2               ;# FATs,
      0038  0040               +            dw      64              ;# dirs,
      003A  0140               +            dw      8*40            ;total sectors,
      003C  FE                 +            db      0feh            ;media byte,
      003D  0001               +            dw      1               ;sectors/FAT,
      003F                          bpbibm9: bpb    512,1,1,2,64,9*40,0fch,2
      003F  0200               +            dw      512             ;sector size,
      0041  01                 +            db      1               ;cluster size,
      0042  0001               +            dw      1               ;resv'd sectors
      0044  02                 +            db      2               ;# FATs,
      0045  0040               +            dw      64              ;# dirs,
      0047  0168               +            dw      9*40            ;total sectors,
      0049  FC                 +            db      0fch            ;media byte,
      004A  0002               +            dw      2               ;sectors/FAT,
      004C                          bpbdec: bpb     512,1,2*10,2,96,80*10,0fah,3
      004C  0200               +            dw      512             ;sector size,
      004E  01                 +            db      1               ;cluster size,
      004F  0014               +            dw      2*10            ;resv'd sectors
      0051  02                 +            db      2               ;# FATs,
      0052  0060               +            dw      96              ;# dirs,
      0054  0320               +            dw      80*10           ;total sectors,
      0056  FA                 +            db      0fah            ;media byte,
      0057  0003               +            dw      3               ;sectors/FAT,
                                    ;
                                    ;Density here is:
                                    ;
                                    ;       0 :: RX-50
                                    ;       1 :: IBM 8 sectors
                                    ;       2 :: IBM 9 sectors or (Robin)
                                    ;This is used by the Z80 for 80/40 track switch.
                                    ;If 0 then 80 tracks, else 40 tracks.
                                    ;
      0059                          deca:   dskblk  bpbdec,0,decdisk,tracka,512,10,0,0,2,0,0,0
      0059  004C R             +            dw      bpbdec          ;BPB ptr,
      005B  0000               +            dw      0               ;1= fixed dsk,
      005D  0000 E             +            dw      decdisk         ;driver addr,
      005F  0179 R             +            dw      tracka          ;curr trk ptr,
      0061  0200               +            dw      512             ;sector size,
      0063  000A               +            dw      10              ;sectors/trk,
      0065  0000               +            dw      0       ;density: 0=RX 1=IBM8 2=IBM9
      0067  0000               +            dw      0               ;drive #,
      0069  0002               +            dw      2               ;dens. chk trk
      006B  0000               +            dw      0               ;765 gap len,
157       006D  0000                 +          dw      0                      ;765 N,
158       006F  0000                 +          dw      0                      ;765 data len,
159       0071                         decb:    dskblk  bpbdec,0,decdisk,trackb,512,10,0,1,2,0,0,0
160       0071  004C R               +          dw      bpbdec                 ;BPB ptr,
161       0073  0000                 +          dw      0                      ;1= fixed dsk,
162       0075  0000 E               +          dw      decdisk                ;driver addr,
163       0077  017A R               +          dw      trackb                 ;curr trk ptr,
164       0079  0200                 +          dw      512                    ;sector size,
165       007B  000A                 +          dw      10                     ;sectors/trk,
166       007D  0000                 +          dw      0        ;density: 0=RX 1=IBM8 2=IBM9
167       007F  0001                 +          dw      1                      ;drive #,
168       0081  0002                 +          dw      2                      ;dens. chk trk
169       0083  0000                 +          dw      0                      ;765 gap len,
170       0085  0000                 +          dw      0                      ;765 N,
171       0087  0000                 +          dw      0                      ;765 data len,
172       0089                         decc:    dskblk  bpbdec,0,decdisk,trackc,512,10,0,2,2,0,0,0
173       0089  004C R               +          dw      bpbdec                 ;BPB ptr,
174       008B  0000                 +          dw      0                      ;1= fixed dsk,
175       008D  0000 E               +          dw      decdisk                ;driver addr,
176       008F  017B R               +          dw      trackc                 ;curr trk ptr,
177       0091  0200                 +          dw      512                    ;sector size,
178       0093  000A                 +          dw      10                     ;sectors/trk,
179       0095  0000                 +          dw      0        ;density: 0=RX 1=IBM8 2=IBM9
180       0097  0002                 +          dw      2                      ;drive #,
181       0099  0002                 +          dw      2                      ;dens. chk trk
182       009B  0000                 +          dw      0                      ;765 gap len,
183       009D  0000                 +          dw      0                      ;765 N,
184       009F  0000                 +          dw      0                      ;765 data len,
185       00A1                         decd:    dskblk  bpbdec,0,decdisk,trackd,512,10,0,3,2,0,0,0
186       00A1  004C R               +          dw      bpbdec                 ;BPB ptr,
187       00A3  0000                 +          dw      0                      ;1= fixed dsk,
188       00A5  0000 E               +          dw      decdisk                ;driver addr,
189       00A7  017C R               +          dw      trackd                 ;curr trk ptr,
190       00A9  0200                 +          dw      512                    ;sector size,
191       00AB  000A                 +          dw      10                     ;sectors/trk,
192       00AD  0000                 +          dw      0        ;density: 0=RX 1=IBM8 2=IBM9
193       00AF  0003                 +          dw      3                      ;drive #,
194       00B1  0002                 +          dw      2                      ;dens. chk trk
195       00B3  0000                 +          dw      0                      ;765 gap len,
196       00B5  0000                 +          dw      0                      ;765 N,
197       00B7  0000                 +          dw      0                      ;765 data len,
198
199       00B9                         ibma:    dskblk  bpbibm,0,decdisk,tracke,512, 8,2,0,0,0,0,0
200       00B9  0032 R               +          dw      bpbibm                 ;BPB ptr,
201       00BB  0000                 +          dw      0                      ;1= fixed dsk,
202       00BD  0000 E               +          dw      decdisk                ;driver addr,
203       00BF  017D R               +          dw      tracke                 ;curr trk ptr,
204       00C1  0200                 +          dw      512                    ;sector size,
205       00C3  0008                 +          dw      8                      ;sectors/trk,
206       00C5  0002                 +          dw      2        ;density: 0=RX 1=IBM8 2=IBM9
207       00C7  0000                 +          dw      0                      ;drive #,
208       00C9  0000                 +          dw      0                      ;dens. chk trk
209       00CB  0000                 +          dw      0                      ;765 gap len,
210       00CD  0000                 +          dw      0                      ;765 N,
211       00CF  0000                 +          dw      0                      ;765 data len,
```

```
212       00D1                         ibmb:    dskblk  bpbibm,0,decdisk,trackf,512, 8,2,1,0,0,0,0
213       00D1  0032 R               +          dw      bpbibm                 ;BPB ptr,
214       00D3  0000                 +          dw      0                      ;1= fixed dsk,
215       00D5  0000 E               +          dw      decdisk                ;driver addr,
216       00D7  017E R               +          dw      trackf                 ;curr trk ptr,
217       00D9  0200                 +          dw      512                    ;sector size,
218       00DB  0008                 +          dw      8                      ;sectors/trk,
219       00DD  0002                 +          dw      2        ;density: 0=RX 1=IBM8 2=IBM9
220       00DF  0001                 +          dw      1                      ;drive #,
221       00E1  0000                 +          dw      0                      ;dens. chk trk
222       00E3  0000                 +          dw      0                      ;765 gap len,
223       00E5  0000                 +          dw      0                      ;765 N,
224       00E7  0000                 +          dw      0                      ;765 data len,
225       00E9                         ibmc:    dskblk  bpbibm,0,decdisk,trackg,512, 8,2,2,0,0,0,0
226       00E9  0032 R               +          dw      bpbibm                 ;BPB ptr,
227       00EB  0000                 +          dw      0                      ;1= fixed dsk,
228       00ED  0000 E               +          dw      decdisk                ;driver addr,
229       00EF  017F R               +          dw      trackg                 ;curr trk ptr,
230       00F1  0200                 +          dw      512                    ;sector size,
231       00F3  0008                 +          dw      8                      ;sectors/trk,
232       00F5  0002                 +          dw      2        ;density: 0=RX 1=IBM8 2=IBM9
233       00F7  0002                 +          dw      2                      ;drive #,
234       00F9  0000                 +          dw      0                      ;dens. chk trk
235       00FB  0000                 +          dw      0                      ;765 gap len,
236       00FD  0000                 +          dw      0                      ;765 N,
237       00FF  0000                 +          dw      0                      ;765 data len,
238       0101                         ibmd:    dskblk  bpbibm,0,decdisk,trackh,512, 8,2,3,0,0,0,0
239       0101  0032 R               +          dw      bpbibm                 ;BPB ptr,
240       0103  0000                 +          dw      0                      ;1= fixed dsk,
241       0105  0000 E               +          dw      decdisk                ;driver addr,
242       0107  0180 R               +          dw      trackh                 ;curr trk ptr,
243       0109  0200                 +          dw      512                    ;sector size,
244       010B  0008                 +          dw      8                      ;sectors/trk,
245       010D  0002                 +          dw      2        ;density: 0=RX 1=IBM8 2=IBM9
246       010F  0003                 +          dw      3                      ;drive #,
247       0111  0000                 +          dw      0                      ;dens. chk trk
248       0113  0000                 +          dw      0                      ;765 gap len,
249       0115  0000                 +          dw      0                      ;765 N,
250       0117  0000                 +          dw      0                      ;765 data len,
251
252       0119                         ibm9a:   dskblk  bpbibm9,0,decdisk,tracke,512,9,2,0,0,0,0,0
253       0119  003F R               +          dw      bpbibm9                ;BPB ptr,
254       011B  0000                 +          dw      0                      ;1= fixed dsk,
255       011D  0000 E               +          dw      decdisk                ;driver addr,
256       011F  017D R               +          dw      tracke                 ;curr trk ptr,
257       0121  0200                 +          dw      512                    ;sector size,
258       0123  0009                 +          dw      9                      ;sectors/trk,
259       0125  0002                 +          dw      2        ;density: 0=RX 1=IBM8 2=IBM9
260       0127  0000                 +          dw      0                      ;drive #,
261       0129  0000                 +          dw      0                      ;dens. chk trk
262       012B  0000                 +          dw      0                      ;765 gap len,
263       012D  0000                 +          dw      0                      ;765 N,
264       012F  0000                 +          dw      0                      ;765 data len,
265       0131                         ibm9b:   dskblk  bpbibm9,0,decdisk,trackf,512,9,2,1,0,0,0,0
266       0131  003F R               +          dw      bpbibm9                ;BPB ptr,
0133  0000                  +          dw      0                  ;1= fixed dsk,
0135  0000 E                +          dw      decdisk            ;driver addr,
0137  017E R                +          dw      trackf             ;curr trk ptr,
0139  0200                  +          dw      512                ;sector size,
013B  0009                  +          dw      9                  ;sectors/trk,
013D  0002                  +          dw      2         ;density: 0=RX 1=IBM8 2=IBM9
013F  0001                  +          dw      1                  ;drive #,
0141  0000                  +          dw      0                  ;dens. chk trk
0143  0000                  +          dw      0                  ;765 gap len,
0145  0000                  +          dw      0                  ;765 N,
0147  0000                  +          dw      0                  ;765 data len,
0149                        ibm9c:     dskblk  bpbibm9,0,decdisk,trackg,512,9,2,2,0,0,0,0
0149  003F R                +          dw      bpbibm9            ;BPB ptr,
014B  0000                  +          dw      0                  ;1= fixed dsk,
014D  0000 E                +          dw      decdisk            ;driver addr,
014F  017F R                +          dw      trackg             ;curr trk ptr,
0151  0200                  +          dw      512                ;sector size,
0153  0009                  +          dw      9                  ;sectors/trk,
0155  0002                  +          dw      2         ;density: 0=RX 1=IBM8 2=IBM9
0157  0002                  +          dw      2                  ;drive #,
0159  0000                  +          dw      0                  ;dens. chk trk
015B  0000                  +          dw      0                  ;765 gap len,
015D  0000                  +          dw      0                  ;765 N,
015F  0000                  +          dw      0                  ;765 data len,
0161                        ibm9d:     dskblk  bpbibm9,0,decdisk,trackh,512,9,2,3,0,0,0,0
0161  003F R                +          dw      bpbibm9            ;BPB ptr,
0163  0000                  +          dw      0                  ;1= fixed dsk,
0165  0000 E                +          dw      decdisk            ;driver addr,
0167  0180 R                +          dw      trackh             ;curr trk ptr,
0169  0200                  +          dw      512                ;sector size,
016B  0009                  +          dw      9                  ;sectors/trk,
016D  0002                  +          dw      2         ;density: 0=RX 1=IBM8 2=IBM9
016F  0003                  +          dw      3                  ;drive #,
0171  0000                  +          dw      0                  ;dens. chk trk
0173  0000                  +          dw      0                  ;765 gap len,
0175  0000                  +          dw      0                  ;765 N,
0177  0000                  +          dw      0                  ;765 data len,

                            ;;dmibma:          dskblk  bpbibm,-1,serdsk,trackx,512,8,3,1,0,0,0,0
```

```
                            page
                            ;
                            ;Current tracks. The current track is kept
                            ;for each PHYSICAL drive; note that the tables
                            ;above that use the same physical drive point
                            ;to the same current track. No need to fiddle
                            ;with these, except maybe to add one if more
                            ;drives are added.
                            ;
0179  FF                    tracka  db      255     ;same drives as
017A  FF                    trackb  db      255     ;below, but different
017B  FF                    trackc  db      255     ;number of tracks.
017C  FF                    trackd  db      255

017D  FF                    tracke  db      255
017E  FF                    trackf  db      255
017F  FF                    trackg  db      255
0180  FF                    trackh  db      255

                            ;;trackx        db      255
                            ;
                            ;Current format selections. One per logical
                            ;drive. These get updated by the media check
                            ;disk call, to either 0,1 or 2.
                            ;
0181  00                    curra   db      0
0182  00                    currb   db      0
0183  00                    currc   db      0
0184  00                    currd   db      0

                            ;;currx db      0

0185                        code ends

                                    end
```

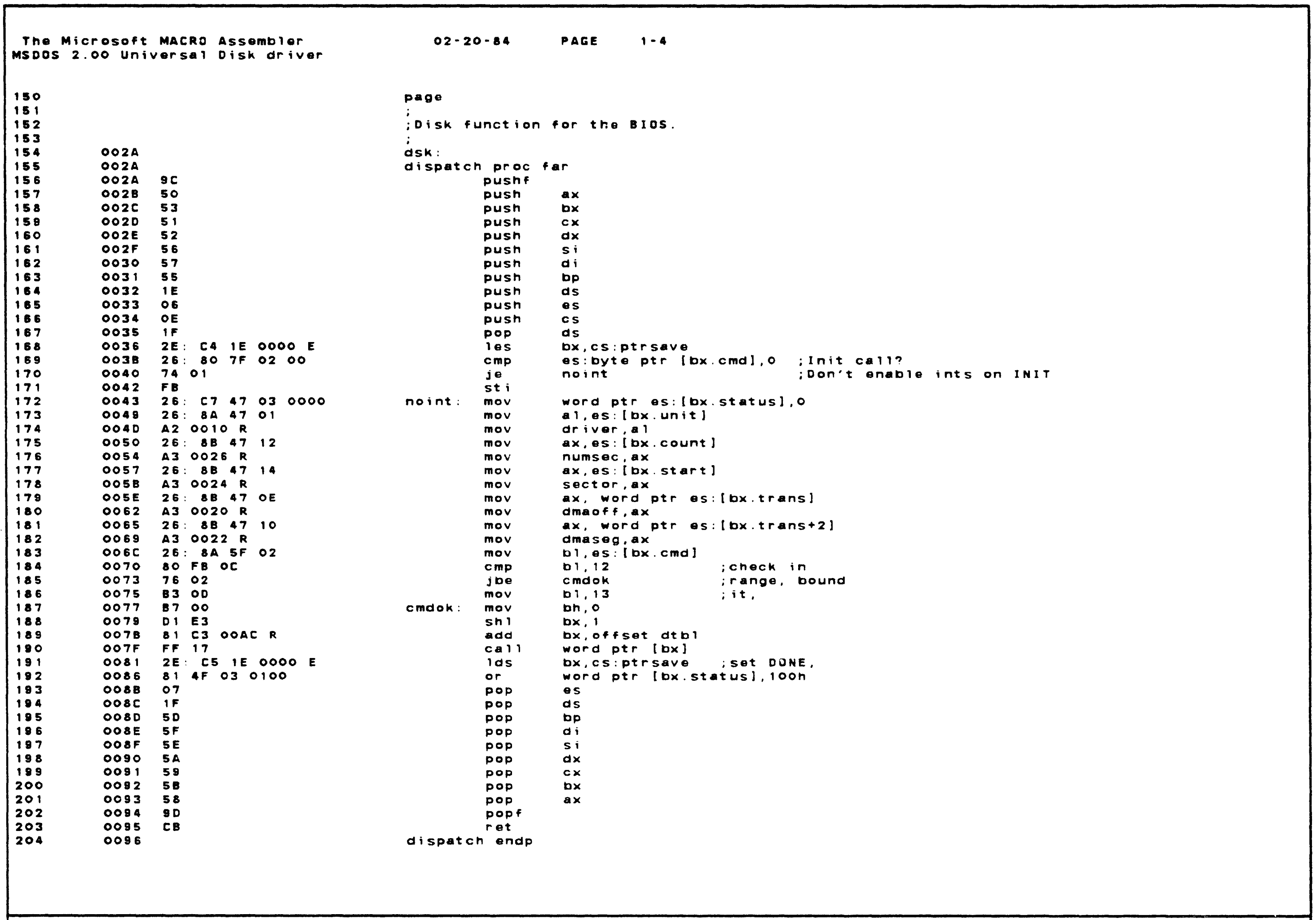
```
 The Microsoft MACRO Assembler           02-20-84      PAGE     Symbols-1
Disk definition tables for MSDOS 2.00


Macros:

                N a m e                 Length

BPB. . . . . . . . . . . . . . .        0006
DSKBLK . . . . . . . . . . . . .        0009

Segments and groups:

                N a m e                 Size    align   combine class

CGROUP . . . . . . . . . . . . .        GROUP
  CODE . . . . . . . . . . . . . .      0185    BYTE    PUBLIC  'CODE'

Symbols:

                N a m e                 Type    Value   Attr

BPBDEC . . . . . . . . . . . . .        L NEAR  004C    CODE
BPBIBM . . . . . . . . . . . . .        L NEAR  0032    CODE
BPBIBM9. . . . . . . . . . . . .        L NEAR  003F    CODE
CURRA. . . . . . . . . . . . . .        L BYTE  0181    CODE
CURRB. . . . . . . . . . . . . .        L BYTE  0182    CODE
CURRC. . . . . . . . . . . . . .        L BYTE  0183    CODE
CURRD. . . . . . . . . . . . . .        L BYTE  0184    CODE
DECA . . . . . . . . . . . . . .        L NEAR  0059    CODE
DECB . . . . . . . . . . . . . .        L NEAR  0071    CODE
DECC . . . . . . . . . . . . . .        L NEAR  0089    CODE
DECD . . . . . . . . . . . . . .        L NEAR  00A1    CODE
DECDISK. . . . . . . . . . . . .        L NEAR  0000    CODE    External
DISKA. . . . . . . . . . . . . .        L WORD  0012    CODE
DISKB. . . . . . . . . . . . . .        L WORD  001A    CODE
DISKC. . . . . . . . . . . . . .        L WORD  0022    CODE
DISKD. . . . . . . . . . . . . .        L WORD  002A    CODE
DSKTBL . . . . . . . . . . . . .        L BYTE  0009    CODE    Global
IBM9A. . . . . . . . . . . . . .        L NEAR  0119    CODE
IBM9B. . . . . . . . . . . . . .        L NEAR  0131    CODE
IBM9C. . . . . . . . . . . . . .        L NEAR  0149    CODE
IBM9D. . . . . . . . . . . . . .        L NEAR  0161    CODE
IBMA . . . . . . . . . . . . . .        L NEAR  00B9    CODE
IBMB . . . . . . . . . . . . . .        L NEAR  00D1    CODE
IBMC . . . . . . . . . . . . . .        L NEAR  00E9    CODE
IBMD . . . . . . . . . . . . . .        L NEAR  0101    CODE
INITTBL. . . . . . . . . . . . .        L BYTE  0000    CODE    Global
TRACKA . . . . . . . . . . . . .        L BYTE  0179    CODE
TRACKB . . . . . . . . . . . . .        L BYTE  017A    CODE
TRACKC . . . . . . . . . . . . .        L BYTE  017B    CODE
TRACKD . . . . . . . . . . . . .        L BYTE  017C    CODE
TRACKE . . . . . . . . . . . . .        L BYTE  017D    CODE
TRACKF . . . . . . . . . . . . .        L BYTE  017E    CODE
TRACKG . . . . . . . . . . . . .        L BYTE  017F    CODE
TRACKH . . . . . . . . . . . . .        L BYTE  0180    CODE

Warning Severe
Errors  Errors
```

```
 The Microsoft MACRO Assembler           02-20-84      PAGE     Symbols-2
Disk definition tables for MSDOS 2.00


0       0
```

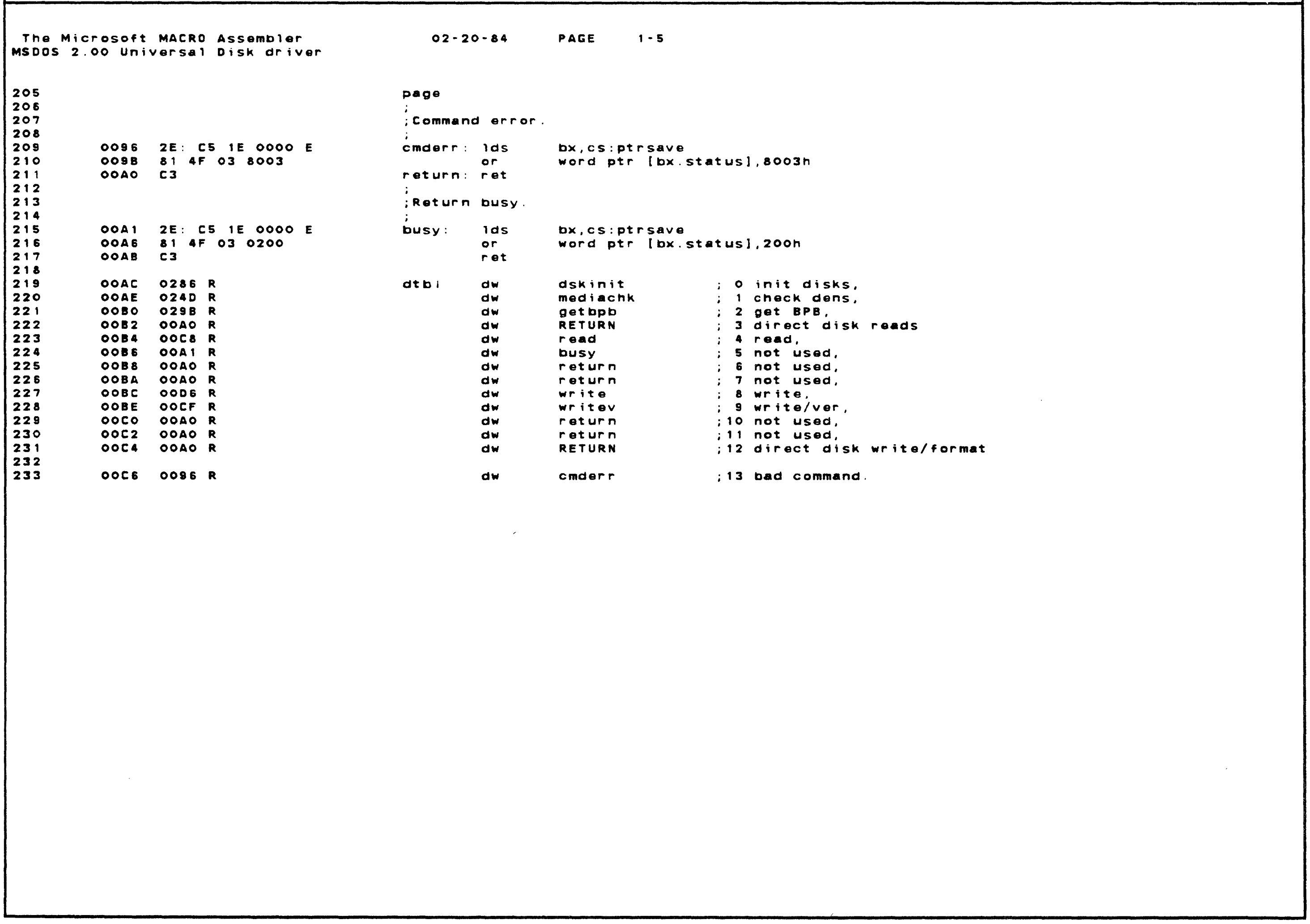
```
 Disk definition tables for MSDOS 2.00

 Symbol Cross Reference                  (# is definition)      Cref-1

BPB. . . . . . . . . . . . . . . .    110     119     128
BPBDEC . . . . . . . . . . . . . .     56      57      58      59     128#    147     160     173     186
BPBIBM . . . . . . . . . . . . . .    110#    200     213     226     239
BPBIBM9. . . . . . . . . . . . . .    119#    253     266     279     292

CGROUP . . . . . . . . . . . . . .     11      13      13
CODE . . . . . . . . . . . . . . .     11      12#     12     338
CURRA. . . . . . . . . . . . . . .     86     331#
CURRB. . . . . . . . . . . . . . .     91     332#
CURRC. . . . . . . . . . . . . . .     96     333#
CURRD. . . . . . . . . . . . . . .    101     334#

DECA . . . . . . . . . . . . . . .     83     146#
DECB . . . . . . . . . . . . . . .     88     159#
DECC . . . . . . . . . . . . . . .     93     172#
DECD . . . . . . . . . . . . . . .     98     185#
DECDISK. . . . . . . . . . . . . .     16#    149     162     175     188     202     215     228     241     255     268     281     294
DISKA. . . . . . . . . . . . . . .     68      83#
DISKB. . . . . . . . . . . . . . .     69      88#
DISKC. . . . . . . . . . . . . . .     70      93#
DISKD. . . . . . . . . . . . . . .     71      98#
DSKBLK . . . . . . . . . . . . . .    146     159     172     185     199     212     225     238     252     265     278     291
DSKTBL . . . . . . . . . . . . . .     15      67#

IBM9A. . . . . . . . . . . . . . .     85     252#
IBM9E. . . . . . . . . . . . . . .     90     265#
IBM9C. . . . . . . . . . . . . . .     95     278#
IBM9D. . . . . . . . . . . . . . .    100     291#
IBMA . . . . . . . . . . . . . . .     84     199#
IBMB . . . . . . . . . . . . . . .     89     212#
IBMC . . . . . . . . . . . . . . .     94     225#
IBMD . . . . . . . . . . . . . . .     99     238#
INITTBL. . . . . . . . . . . . . .     15      55#

TRACKA . . . . . . . . . . . . . .    150     315#
TRACKB . . . . . . . . . . . . . .    163     316#
TRACKC . . . . . . . . . . . . . .    176     317#
TRACKD . . . . . . . . . . . . . .    189     318#
TRACKE . . . . . . . . . . . . . .    203     256     320#
TRACKF . . . . . . . . . . . . . .    216     269     321#
TRACKG . . . . . . . . . . . . . .    229     282     322#
TRACKH . . . . . . . . . . . . . .    242     295     323#
```

```
 The Microsoft MACRO Assembler                02-20-84      PAGE     1-1
MSDOS 2.00 Universal Disk driver

1                                                      PAGE    60,132
2                                                      TITLE   MSDOS 2.00 Universal Disk driver
3                                                      NAME    DISK
4                                              ;
5                                              ;       COMPANY CONFIDENTIAL
6                                              ;       Copyright (C) 1983 Digital Equipment Corporation
7                                              ;       All rights reserved.
8                                              ;
9                                              ;       10/05/83
10
11                                             cgroup group code
12
13                                             public  dsk, MC_FLG
14                                             public  cmdstr, dsktmr
15
16                                             extrn   ptrsave:dword
17                                             extrn   dsktbl:word,inittbl:word
18
19                                             ;
20                                             ;This is a special version for the Rainbow.
21                                             ;It prevents disk access outside the range
22                                             ;of 01000 - offffh. Much optimization has
23                                             ;been done to speed things up.
24
25                                          C  include iodef.ash
26                                          C  ;Rainbow Interrupt numbers.
27                                          C  ;1-Apr-83 sgs added dskio int vector
28                                          C  ;19-Mar-83 sgs added profile int vector [user clock]
29                                          C  ;the int profile is called by the RTC interrupt service for each tick.
30                                          C  ;The ax and ds register don't need to be saved [done in clkisr].
31                                          C  ;
32        = 0064                            C  profile equ     64h     ;100. user interface to clock interrupt
33        = 002C                            C  clk_int equ     2ch     ;60Hz int,
34        = 0065                            C  dskio   equ     65h     ;direct disk io for format
35                                          C  ;
36                                          C  ;17-Mar-83 sgs changed to include int 20-26
37                                          C  ;interrupt 22-24h are duplicated at 42-44h for consistency.
38                                          C  ;These are the relocated Rainbow interrupts.
39                                          C  ;interrupts, 20h-26h moved to 40h-46h
40                                          C  ;ATTENTION Modules IDINIT and REINIT may have to be
41                                          C  ;changed if int vectors 40h to 46h are changed
42                                          C  ;
43        = 0040                            C  vert    equ     40h     ;new vert. freq.
44        = 0041                            C  spare41 equ     41h
45        = 0042                            C  spare42 equ     42h
46        = 0043                            C  comdma  equ     43h     ;DMA ctrl optional comm. board
47        = 0044                            C  aux_prn equ     44h     ;7201 comm./printer
48        = 0045                            C  excom   equ     45h     ;extented comm. option
49        = 0046                            C  uart    equ     46h     ;new UART vector,
50        = 0047                            C  z80     equ     47h     ;Z80 interrupt,
51        = 0018                            C  rom     equ     18h     ;new ROM access,
52                                          C  ;
53                                          C  ;DEC Rainbow IO port stuff.
54                                          C  ;
55        = 0010                            C  kdp     equ     10h     ;8251 data port,
       = 0011                    C  ksp     equ     11h      ;8251 status,
       = 0040                    C  auxdp   equ     40h      ;7201 data,
       = 0041                    C  prndp   equ     41h
       = 0042                    C  auxp    equ     42h      ;7201 command,
       = 0043                    C  prnp    equ     43h
                                 C  ;
                                 C  ; Values in XOPTION
                                 C  ;
       = 0001                    C  wprsnt  equ     01       ;Winnie preset
       = 0002                    C  xcprsnt equ     02       ;XCOMM present

                                    extrn   buffer:near      ;defined in END.ASM

       0000                         code segment byte public 'code'
                                    assume cs:cgroup,ds:cgroup
                                    ;
                                    ;       High level disk interface for MSDOS
                                    ;2.00. The MSDOS request for N logical sectors
                                    ;is broken down to a bunch of calls to the
                                    ;phsical driver to do sectors within a given
                                    ;physical track. Address wrap around due to
                                    ;hardware limitations (16 bit address, 4 bit
                                    ;page register) is taken care of here, as well
                                    ;as I/O requests larger than 64K bytes that
                                    ;would wrap around a segment.
                                    ;
                                 C  include ms200.ash
                                 C  ;
                                 C  ;MSDOS 2.00 IO device data packet layout.
                                 C  ;
                                 C  iodata struc
       0000  ??                  C  cmdlen  db      (?)      ;packet length,
       0001  ??                  C  unit    db      (?)      ;unit number,
       0002  ??                  C  cmd     db      (?)      ;command,
       0003  ????                C  status  dw      (?)      ;returned status,
       0005      08 [            C          db 8 dup (?)
                         ??      C
                              ]  C
                                 C
       000D  ??                  C  media   db      (?)      ;descriptor byte
       000E  ????????            C  trans   dd      (?)      ;transfer address
       0012  ????                C  count   dw      (?)      ;data count
       0014  ????                C  start   dw      (?)      ;starting record
       0016                      C  iodata ends
                                 C
```

```
                                    page
       = 0000                       fwrite  equ     0        ;write
       = 0001                       fread   equ     1        ;read
       = 0002                       fdens   equ     2        ;check density
       = 0003                       fwritev equu    3        ;write with verify
       = 0065                       dskdrct equ     065h     ;direct disk access interrupt
       = 0002                       tmrcnt  equ     2        ;2 sec. timeout


                                    ;
                                    ;Physical disk driver parameter block. The
                                    ;read or write command is built here, and
                                    ;a pointer to it is passed to the actual
                                    ;driver.
                                    ;
       0000                         cmdstr:
       0000  ??                     command db      (?)      ;floppy command,
       0001  ??                     dskdrv  db      (?)      ;disk drive,
       0002  ??                     track   db      (?)      ;seek track,
       0003  ??                     physec  db      (?)      ;seek sector,
       0004  ??                     physcnt db      (?)      ;sector count,
       0005  ??                     curtrk  db      (?)      ;current track,
       0006  ??                     density db      (?)      ;density/size flag,
       0007  ??                     gaplen  db      (?)      ;766 gap length,
       0008  ??                     enn     db      (?)      ;765 sec size,
       0009  ??                     dtl     db      (?)      ;765 data length,
       000A  ????                   secsiz  dw      (?)      ;sector size,
       000C  ????                   dskoff  dw      (?)      ;DMA offset,
       000E  ????                   dskseg  dw      (?)      ;DMA segment,
                                    ;
                                    ;End of the command string, start of
                                    ;internal stuff.
                                    ;
       0010  ??                     driver  db      (?)      ;driver number,
       0011  ????                   trkptr  dw      (?)      ;ptr to current track,
       0013  ????                   bpbptr  dw      (?)      ;ptr to BPB,
       0015  ????                   typptr  dw      (?)      ;ptr to current format,
       0017  ??                     chkbyte db      (?)      ;1 if non removable,
       0018  ??                     dentrk  db      (?)      ;trk for density test,
       0019  ????                   spt     dw      (?)      ;sectors per track,
       001B  ????                   scount  dw      (?)      ;physical sector count,
       001D  ??                     errmsk  db      (?)      ;status mask,
       001E  ????                   calladdr dw     (?)      ;driver address,
       0020  ????                   dmaoff  dw      (?)      ;DMA offset,(don't change these DMA order)
       0022  ????                   dmaseg  dw      (?)      ;DMA segment,
       0024  ????                   sector  dw      (?)      ;MSDOS sector number,
       0026  ????                   numsec  dw      (?)      ;MSDOS sector count,

       0028  FF                     tmrdrv  db      0ffh     ;Last drive used successfully
       0029  02                     dsktmr  db      tmrcnt   ;timer loc.
```



```
 The Microsoft MACRO Assembler              02-20-84      PAGE     1-4
MSDOS 2.00 Universal Disk driver


                                      page
                                      ;
                                      ;Disk function for the BIOS.
                                      ;
0 02A                                 dsk:
002A                                  dispatch proc far
002A  9C                                        pushf
002B  50                                        push    ax
002C  53                                        push    bx
002D  51                                        push    cx
002E  52                                        push    dx
002F  56                                        push    si
0030  57                                        push    di
0031  55                                        push    bp
0032  1E                                        push    ds
0033  06                                        push    es
0034  0E                                        push    cs
0035  1F                                        pop     ds
0036  2E: C4 1E 0000 E                          les     bx,cs:ptrsave
003B  26: 80 7F 02 00                           cmp     es:byte ptr [bx.cmd],0  ;Init call?
0040  74 01                                     je      noint                   ;Don't enable ints on INIT
0042  FB                                        sti
0043  26: C7 47 03 0000               noint:    mov     word ptr es:[bx.status],0
0048  26: 8A 47 01                              mov     al,es:[bx.unit]
004D  A2 0010 R                                 mov     driver,al
0050  26: 8B 47 12                              mov     ax,es:[bx.count]
0054  A3 0026 R                                 mov     numsec,ax
0057  26: 8B 47 14                              mov     ax,es:[bx.start]
005B  A3 0024 R                                 mov     sector,ax
005E  26: 8B 47 0E                              mov     ax, word ptr es:[bx.trans]
0062  A3 0020 R                                 mov     dmaoff,ax
0065  26: 8B 47 10                              mov     ax, word ptr es:[bx.trans+2]
0069  A3 0022 R                                 mov     dmaseg,ax
006C  26: 8A 5F 02                              mov     bl,es:[bx.cmd]
0070  80 FB 0C                                  cmp     bl,12           ;check in
0073  76 02                                     jbe     cmdok           ;range, bound
0075  B3 0D                                     mov     bl,13           ;it,
0077  B7 00                           cmdok:    mov     bh,0
0079  D1 E3                                     shl     bx,1
007B  81 C3 00AC R                              add     bx,offset dtbl
007F  FF 17                                     call    word ptr [bx]
0081  2E: C5 1E 0000 E                          lds     bx,cs:ptrsave   ;set DONE,
0086  81 4F 03 0100                             or      word ptr [bx.status],100h
008B  07                                        pop     es
008C  1F                                        pop     ds
008D  5D                                        pop     bp
008E  5F                                        pop     di
008F  5E                                        pop     si
0090  5A                                        pop     dx
0091  59                                        pop     cx
0092  5B                                        pop     bx
0093  58                                        pop     ax
0094  9D                                        popf
0095  CB                                        ret
0096                                  dispatch endp
```



```
 The Microsoft MACRO Assembler              02-20-84      PAGE     1-5
MSDOS 2.00 Universal Disk driver


                                      page
                                      ;
                                      ;Command error.
                                      ;
0096  2E: C5 1E 0000 E                cmderr:   lds     bx,cs:ptrsave
009B  81 4F 03 8003                             or      word ptr [bx.status],8003h
00A0  C3                              return:   ret
                                      ;
                                      ;Return busy.
                                      ;
00A1  2E: C5 1E 0000 E                busy:     lds     bx,cs:ptrsave
00A6  81 4F 03 0200                             or      word ptr [bx.status],200h
00AB  C3                                        ret

00AC  0286 R                          dtbl      dw      dskinit         ; 0 init disks,
00AE  024D R                                    dw      mediachk        ; 1 check dens,
00B0  029B R                                    dw      getbpb          ; 2 get BPB,
00B2  00A0 R                                    dw      RETURN          ; 3 direct disk reads
00B4  00C8 R                                    dw      read            ; 4 read,
00B6  00A1 R                                    dw      busy            ; 5 not used,
00B8  00A0 R                                    dw      return          ; 6 not used,
00BA  00A0 R                                    dw      return          ; 7 not used,
00BC  00D6 R                                    dw      write           ; 8 write,
00BE  00CF R                                    dw      writev          ; 9 write/ver,
00C0  00A0 R                                    dw      return          ;10 not used,
00C2  00A0 R                                    dw      return          ;11 not used,
00C4  00A0 R                                    dw      RETURN          ;12 direct disk write/format

00C6  0096 R                                    dw      cmderr          ;13 bad command.
```

```
                               page
                               ;
                               ;Disk read/ write entry point. PTRSAV points
                               ;to the data packet. Perform the function,
                               ;return:
                               ;       # sectors actually read,
                               ;       status byte (See below)
                               ;       error byte (see below)
                               ;
                               ;       0       write protect,
                               ;       1       unknown unit
                               ;       2       not ready,
                               ;       3       command error
                               ;       4       data error,
                               ;       5       bad structure length,
                               ;       6       seek error,
                               ;       7       unkown media,
                               ;       8       sector not found,
                               ;       9       out of paper,
                               ;       10      write fault,
                               ;       11      read fault,
                               ;       12      other failure.
                               ;

00C8  C6 06 0000 R 01          read:   mov     command,fread
00CD  EB 0C                            jmp short comn

00CF  C6 06 0000 R 03          writev: mov     command,fwritev
00D4  EB 05                            jmp short comn

00D6  C6 06 0000 R 00          write:  mov     command,fwrite
00DB  E8 01EC R                comn:   call    set_drv         ;set parms,
00DE  72 5C                            jc      rdwret          ;ret bad unit
```

```
                               page
                               ;
                               ;Calculate the track, given the logical sector
                               ;and the number of sectors/track. Loop over
                               ;this point til all sectors read.
                               ;
                               ; What it does:
                               ;
                               ;1/     Find phys. track, sector,
                               ;2/     If (scount < sectors left on track) {
                               ;                 read sectors on track
                               ;       } else {
                               ;               read (scount) sectors
                               ;       }
                               ;3/     scount =scount-sectors read,
                               ;       log. sec = log. sec + sectors read,
                               ;4/     if scount .gt. 0, goto 1/.
                               ;
                               ;All necessary physical parameters are calcu-
                               ;lated here, and passed to lower level guys
                               ;that actually do the read or write.
                               ;
00E0  A1 0026 R                rdwsec: mov     ax,numsec       ;if all done,
00E3  3D 0000                          cmp     ax,0            ;return.
00E6  74 54                            jz      rdwret

                               ;
                               ;offset is as small as possible (will be less
                               ;reading or writing more than 64K per BIOS
                               ;call. Assumes less than 64K per disk track.
                               ;
00E8  A1 0020 R                        mov     ax,dmaoff
00EB  D1 C8                            ror     ax,1            ;divide offset
00ED  D1 C8                            ror     ax,1            ;by 16, leave
00EF  D1 C8                            ror     ax,1            ;remainder in
00F1  D1 C8                            ror     ax,1            ;bits 15-12,
00F3  8B D0                            mov     dx,ax
00F5  25 0FFF                          and     ax,0fffh
00F8  01 06 0022 R                     add     dmaseg,ax       ;adjust segment
00FC  81 E2 F000                       and     dx,0f000h
0100  D1 C2                            rol     dx,1            ;put remainder
0102  D1 C2                            rol     dx,1            ;in LSB's,
0104  D1 C2                            rol     dx,1
0106  D1 C2                            rol     dx,1
0108  89 16 0020 R                     mov     dmaoff,dx       ;adjusted off,
                                page
                                ;
                                ;Figure out how many sectors to read/write and
                                ;the track to do it on.
                                ;
 010C  A1 0024 R                        mov     ax,sector          ;logical sector
 010F  8B 0E 0019 R                     mov     cx,spt             ;sectors/track,
 0113  F6 F1                            div     cl                 ;now AL= track,
 0115  FE C4                            inc     ah                 ;AH = sector,
 0117  A2 0002 R                        mov     track,al           ;save track,
 011A  88 26 0003 R                     mov     physec,ah          ;save sector
                                ;
                                ;Start the read or write process. The number of
                                ;sectors to do may bring us past the end of the
                                ;cylinder; if so, do (sectors left on track)
                                ;sectors, then repeat for each track until done
                                ;
 011E  8A 0E 0003 R             dosec:  mov     cl,physec          ;find # sectors
 0122  B5 00                            mov     ch,0               ;left on
 0124  A1 0019 R                        mov     ax,spt             ;this track,
 0127  40                               inc     ax                 ;(start at 1,)
 0128  2B C1                            sub     ax,cx              ;leave in AX,
 012A  8B 0E 0026 R                     mov     cx,numsec          ;comp to total
 012E  3B C1                            cmp     ax,cx              ;if .lt. total,
 0130  72 02                            jb      doleft             ;do left track,
 0132  8B C1                                    mov     ax,cx      ;else requested
 0134  A3 001B R                doleft: mov     scount,ax          ;amt. set count
 0137  E8 015A R                        call    readwrite          ;do the oper
 013A  73 A4                            jnc     rdwsec             ;rpt no error,
                                ;
                                ;Return to MSDOS. If no error, just return: the
                                ;number of sectors read ('count') is correct.
                                ;If error, set the error code, and adjust the
                                ;count to reflect the number actually done.
                                ;
 013C  73 12                    rdwret: jnc     rwr1               ;if error,
 013E  2E: C4 1E 0000 E                 les     bx,cs:ptrsave
 0143  B4 80                            mov     ah,80h             ;err flag,
 0145  26: 09 47 03                     or      es:[bx.status],ax ;set error,
 0149  A1 0026 R                        mov     ax,numsec
 014C  26: 29 47 12                     sub     es:[bx.count],ax
 0150  A0 0005 R                rwr1:   mov     al,curtrk          ;save current
 0153  8B 1E 0011 R                     mov     bx,trkptr
 0157  88 07                            mov     [bx],al            ;track,
 0159  C3                               ret
```

```
                                page
                                ;
                                ;Read or write physical sectors. The physical
                                ;parameters are all set; the sector count is
                                ;equal to or less than the number of sectors
                                ;on the track. Returns carry set and the MSDOS
                                ;error code in AL if error.
                                ;
                                ;Do all Rainbow sectors one at a time for 2
                                ;reasons: a bug in the Z80 code prevents
                                ;multiple sectors from working, and the
                                ;Z80 can only access addresses from 01000 -
                                ;0ffffh.
                                ;
 015A                           readwrite:
 015A  E8 0184 R                        call    physbuf            ;do buffered,
 015D  72 24                            jc      rdwrret            ;exit if err,
 015F  A0 0001 R                        mov     al,dskdrv          ;no error - reset timer/drive
 0162  A2 0028 R                        mov     tmrdrv,al
 0165  C6 06 0029 R 02                  mov     dsktmr,tmrcnt      ;reset counter
                                ;
                                ;Update the number of sectors, physical sector,
                                ;and all that stuff. If no more to do, return.
                                ;NOTE: This assumes that only one sector is
                                ;done per pass.
                                ;
 016A  A1 000A R                rwchk:  mov     ax,secsiz
 016D  01 06 0020 R                     add     dmaoff,ax          ; adjust offset
 0171  FE 06 0003 R                     inc     physec             ; next phys sec
 0175  FF 06 0024 R                     inc     sector             ; next log  sec
 0179  FF 0E 0026 R                     dec     numsec             ; total to do,
 017D  FF 0E 001B R                     dec     scount             ; new count,
 0181  75 D7                            jnz     readwrite
 0183  C3                       rdwrret:ret                        ;done this trk
```

```
                                   page
                                   ;
                                   ;Read or write data to our local buffer. Copy
                                   ;the data in or out of the buffer as necessary.
                                   ;Return carry set and AL error code if bad.
                                   ;
0184                               physbuf:
0184  C7 06 000C R 0000 E                  mov     dskoff,offset buffer
018A  8C 1E 000E R                         mov     dskseg,ds         ;set local buf
018E  C6 06 0004 R 01                      mov     physcnt,1         ;one sector,
0193  80 3E 0000 R 03                      cmp     command,fwritev ;write verify
0198  74 07                                je      wrtbuf
019A  80 3E 0000 R 00                      cmp     command,fwrite
019F  75 1D                                jne     dord              ;if write,
01A1                               wrtbuf:
01A1  8B 0E 000A R                         mov     cx,secsiz
01A5  8B 36 0020 R                         mov     si,dmaoff         ;from DMA
01A9  BF 0000 E                            mov     di,offset buffer          ;to buffer,
01AC  8C DA                                mov     dx,ds             ;save DS,
01AE  8E C2                                mov     es,dx             ;ES= buffer,
01B0  2E: 8E 1E 0022 R                     mov     ds,cs:dmaseg      ;DS= dest.,
01B5  FC                                   cld
01B6  F3/ A4                               rep movsb                 ;copy to buf
01B8  8E DA                                mov     ds,dx             ;restore DS,
01BA  E8 01D7 R                            call    dophys            ;write to disk,
01BD  C3                                   ret                       ;return status,

01BE  E8 01D7 R                    dord:   call    dophys            ;read, do it,
01C1  72 13                                jc      pret              ;stop if error,
01C3  8B 0E 000A R                         mov     cx,secsiz
01C7  BE 0000 E                            mov     si,offset buffer          ;copy data out
01CA  8B 3E 0020 R                         mov     di,dmaoff
01CE  2E: 8E 06 0022 R                     mov     es,cs:dmaseg      ;to seg:off,
01D3  FC                                   cld
01D4  F3/ A4                               rep movsb                 ;copy from buf
01D6  C3                           pret:   ret                       ;return status.
```

```
                                   page
                                   ;
                                   ;Do the physical read or write as specified by
                                   ;the disk command block. The driver to use was
                                   ;found in the 'dsktbl'. The physical drivers
                                   ;return carry set and the MSDOS error code in
                                   ;AL if error.
                                   ;
                                   ;After the read or write, update the current
                                   ;track for this disk. If error, set the current
                                   ;track to -1, to indicate an error. (The driver
                                   ;will recal the disk before seeking if the
                                   ;current track is -1.)
                                   ;
01D7                               dophys:
01D7  BB 0000 R                            mov     bx,offset cmdstr ;setup command
01DA  FF 16 001E R                         call    word ptr calladdr ;do it,
01DE  C6 06 0005 R FF                      mov     curtrk,-1         ;assume bad,
01E3  72 06                                jc      dp1
01E5  A0 0002 R                            mov     al,track          ;if good, set
01E8  A2 0005 R                            mov     curtrk,al         ;current track,
01EB  C3                           dp1:    ret
```

```
                            page
                            ;
                            ;Given the drive number, pull out the physical
                            ;parameters necessary to read and write.
                            ;The parameters are:
                            ;        BPB pointer,
                            ;        fixed/removable disk,
                            ;        read/write driver address,
                            ;        Physical drive number,
                            ;        Density flag,
                            ;        Sector size,
                            ;        Current track,
                            ;        Sectors per track,
                            ;        Density check track,
                            ;        NEC765 parameters.
                            ;
01EC                        set_drv:
01EC  8A 1E 0010 R                   mov     bl,driver
01F0  3A 1E 0000 E                   cmp     bl,byte ptr dsktbl;legal unit?
01F4  72 04                          jb      sdrv1
01F6  F9                                     stc               ;no,
01F7  B0 01                                  mov      al,1     ;unknown unit,
01F9  C3                                     ret
01FA  B7 00                 sdrv1:   mov     bh,0              ;1st time, get
01FC  D1 E3                          shl     bx,1              ;dsk data ptr,
01FE  8B 9F 0001 E                   mov     bx,dsktbl[bx+1] ;BX= disk ptr to ptr,
0202  8B 77 06                       mov     si,[bx+6]         ;SI= type ptr (ptr to currxx)
0205  89 36 0015 R                   mov     typptr,si         ;save type ptr,
0209  8A 04                          mov     al,[si]           ;get curr formt
020B  25 0007                        and     ax,7              ;three choices,
020E  03 D8                          add     bx,ax             ;add word offset
0210  03 D8                          add     bx,ax             ;get dsk data,
0212  8B 37                          mov     si,[bx]

0214  FC                             cld
0215  AD                             lodsw                     ;BPB pointer,
0216  A3 0013 R                      mov     bpbptr,ax
0219  AD                             lodsw                     ;removable flag
021A  A2 0017 R                      mov     chkbyte,al
021D  AD                             lodsw                     ;driver address
021E  A3 001E R                      mov     calladdr,ax
0221  AD                             lodsw                     ;cur. trk ptr,
0222  A3 0011 R                      mov     trkptr,ax
0225  8B D8                          mov     bx,ax             ;current track,
0227  8A 07                          mov     al,[bx]
0229  A2 0005 R                      mov     curtrk,al

022C  AD                             lodsw                     ;sector size,
022D  A3 000A R                      mov     secsiz,ax
0230  AD                             lodsw                     ;sectors/track,
0231  A3 0019 R                      mov     spt,ax
0234  AD                             lodsw                     ;density byte,
0235  A2 0006 R                      mov     density,al
0238  AD                             lodsw                     ;physical drive
0239  A2 0001 R                      mov     dskdrv,al
```

```
023C  AD                             lodsw
023D  A2 0018 R                      mov     dentrk,al         ;dens. chk. trk
0240  AD                             lodsw
0241  A2 0007 R                      mov     gaplen,al         ;gap length,
0244  AD                             lodsw
0245  A2 0008 R                      mov     enn,al            ;sector size,
0248  AD                             lodsw
0249  A2 0009 R                      mov     dtl,al            ;data length,
024C  C3                             ret
```

```
                              page
                              ;
                              ;Disk change detect routine.
                              ;always returns dont know
                              ;Returns 1 if fixed disk (didn't change).
                              ;
024D                          mediachk:
024D  E8 01EC R                           call    set_drv           ;select drive,
0250  B0 01                               mov     al,1              ;unknown unit,
0252  B4 FF                               mov     ah,-1             ;unknown change
0254  72 16                               jc      samdsk            ;return error,

0256  33 C0                               xor     ax,ax             ;assume don't know
0258  8A 1E 0001 R                        mov     bl,dskdrv         ;current disk
025C  3A 1E 0028 R                        cmp     bl,tmrdrv         ;match last one used?
0260  75 09                               jnz     maybec
0262  80 3E 0029 R 00                     cmp     dsktmr,0          ;timer expired?
0267  74 02                               je      maybec            ;jump if too late
0269  B4 01                               mov     ah,1              ;flag not changed
026B                          maybec:
026B  F8                                  clc                       ;no error,
026C                          samdsk:
026C  8B 36 0011 R                        mov     si,trkptr
0270  8A 0E 0005 R                        mov     cl,curtrk
0274  88 0C                               mov     [si],cl  ;phys track=cur.track

0276  2E: C5 1E 0000 E                    lds     bx,cs:ptrsave     ;set for ret,
027B  88 67 0E                            mov     [bx.media+1],ah ;set ret code,
027E  73 05                               jnc     dcr1              ;if error,

0280  B4 80                               mov     ah,80h            ;set err bit,
0282  09 47 03                            or      [bx.status],ax    ;return error,
0285                          dcr1:
0285  C3                                  ret
```

```
                              page
                              ;
                              ;Disk initialization. Return a pointer to the
                              ;initialization table, and the number of units.
                              ;
0286                          dskinit:
0286  BE 0000 E                           mov     si,offset inittbl;table ptr,
0289  8A 04                               mov     al,[si]           ;# disks,
028B  46                                  inc     si                ;ptr to ptrs,
028C  2E: C5 1E 0000 E                    lds     bx,cs:ptrsave
0291  88 47 0D                            mov     [bx.media],al     ;# units,
0294  89 77 12                            mov     [bx.count],si
0297  8C 4F 14                            mov     [bx.start],cs
029A  C3                                  ret

                              ;
                              ;Build BPB pointer. All it does is 'select'
                              ;the disk internally, and return a pointer to
                              ;the BPB table.
                              ;
029B                          getbpb:
029B  E8 02B1 R                           call    chkmed            ;check media type for drive
029E  E8 01EC R                           call    set_drv
02A1  A1 0013 R                           mov     ax,bpbptr
02A4  C5 1E 0000 E                        lds     bx,ptrsave
02A8  89 47 12                            mov     [bx.count],ax     ;return DWORD
02AB  8C C8                               mov     ax,cs             ;ptr to table,
02AD  89 47 14                            mov     [bx.start],ax
02B0  C3                                  ret

= 0000                        iofun       equ     0                 ;ioctl packet function offset
= 0001                        iodrv       equ     1                 ;ioctl packet drive offset
= 0002                        iosec       equ     2                 ;ioctl packet sector offset
= 0004                        iotrk       equ     4                 ;ioctl track offset
= 0006                        iocnt       equ     6                 ;ioctl sector count
= 0008                        iooff       equ     8                 ;ioctl buffer offset
= 000A                        ioseg       equ     0ah               ;ioctl buffer segment

02B1  A0 0010 R               chkmed:     mov     al,driver         ;get current drive to check
02B4  3A 06 0000 E                        cmp     al,byte ptr dsktbl ;see if it is a valid drive
02B8  72 06                               jb      ckmda1
02BA  B0 01                               mov     al,1
02BC  F9                                  stc
02BD  EB 45 90                            jmp     bailout
02C0                          ckmda1:
02C0  80 3E 0339 R 01                     cmp     byte ptr mc_flg,1         ; check media check flag
02C5  75 04                               jnz     normal                    ; do check
02C7  33 C0                               xor     ax,ax                     ; set to RAINBOW media
02C9  EB 25                               jmp     short okmdia              ;  and skip media check codes
02CB                          normal:
02CB  BB 033A R                           mov     bx,offset iopkt
02CE  8C 5F 0A                            mov     ioseg[bx],ds              ;io xfer buffer segment
02D1  C7 47 08 0000 E                     mov     word ptr iooff[bx],offset buffer ;io xfer buffer
02D6  88 47 01                            mov     iodrv[bx],al              ;drive to check
02D9  C6 07 04                            mov     byte ptr iofun[bx],04     ;check media function
```

```

      02DC  CD 65                       int     dskdrct           ;direct disk driver interrupt

      02DE  B0 01                       mov     al,01             ;in case of error
      02E0  72 22                       jc      bailout           ;blast out of here
      02E2  BB 0000 E                   mov     bx,offset buffer         ;media type is in first byte of buffer
      02E5  33 C0                       xor     ax,ax            ;clear field for weird
      02E7  8A 07                       mov     al,[bx]          ;get media type
      02E9  3C 01                       cmp     al,01            ;if this is a rainbow or ibm8, Ok
      02EB  76 03                       jbe     okmdia           ;else fin out what it really is
      02ED  E8 0305 R                   call    softchk          ;do a soft media check (read disk ID)
      02F0                      okmdia:
      02F0  33 DB                       xor bx,bx                ;clear pointer
      02F2  8A 1E 0010 R                mov     bl,driver         ;get current disk again
      02F6  D1 E3                       shl     bx,1              ;make worb index
      02F8  8B 9F 0001 E                mov     bx,dsktbl[bx+1]   ;and get media type table for drive
      02FC  8B 77 06                    mov     si,[bx+6]         ;get pointer to media type holder
      02FF  88 04                       mov     [si],al           ;and save media type
      0301  32 C0                       xor     al,al             ;clear status
      0303  F8                          clc                       ;accross the board
      0304                      bailout:
      0304  C3                          ret

      0305                      softchk:             ;coft media check to see if real 9 sector
      0305  BB 033A R                   mov     bx,offset iopkt ;read the disk id byte to check illogica
      0308  A0 0010 R                   mov     al,driver        ;get drive type
      030B  88 47 01                    mov     iodrv[bx],al     ;drive to read
      030E  C6 47 02 02                 mov     byte ptr iosec[bx],2    ;type byte is on sector 2
      0312  C7 47 04 0000               mov     word ptr iotrk[bx],0  ;on track 0
      0317  C7 47 06 0001               mov     word ptr iocnt[bx],1 ;only reat 1 sector, of course
      031C  C7 47 08 0000 E             mov     word ptr iooff[bx],offset buffer;default sector buffer
      0321  8C 5F 0A                    mov     ioseg[bx],ds     ;default data segment
      0324  C6 07 00                    mov     byte ptr iofun[bx],0    ;read data command

      0327  CD 65                       int     dskdrct           ;direct disk interrupt
      0329  72 0D                       jc      exito             ;exito: greek for throwing garbage out

      032B  BB 0000 E                   mov     bx,offset buffer        ;get data byte
      032E  8A 07                       mov     al,[bx]          ;get id byte
      0330  3C FE                       cmp     al,0feh          ;is it an ibm8
      0332  B0 01                       mov     al,01            ;if so, say so
      0334  74 02                       jz      exito            ;and leave
      0336  B0 02                       mov     al,02            ;else it is an ibm9
      0338  C3                  exito:  ret                      ;return with media type in al
```

```
                                        PAGE

      0339  00                  MC_FLG  DB      0                ; media check flag, 0 = check default
                                                                 ; 1 = don't check assume RAINBOW disk


      033A     10 [             iopkt:  db 10h  dup (?)          ;space for packet
                        ??
                             ]


      034A                      code    ends
                                        end
```

```
Structures and records:

                N a m e                        Width    # fields
                                               Shift    Width    Mask     Initial

IODATA . . . . . . . . . . . . . . . .         0016     0009
  CMDLEN . . . . . . . . . . . . . . . . .     0000
  UNIT . . . . . . . . . . . . . . . . . .     0001
  CMD. . . . . . . . . . . . . . . . . . .     0002
  STATUS . . . . . . . . . . . . . . . . .     0003
  MEDIA. . . . . . . . . . . . . . . . . .     000D
  TRANS. . . . . . . . . . . . . . . . . .     000E
  COUNT. . . . . . . . . . . . . . . . . .     0012
  START. . . . . . . . . . . . . . . . . .     0014

Segments and groups:

                N a m e                        Size     align    combine class

CGROUP . . . . . . . . . . . . . . . .         GROUP
  CODE . . . . . . . . . . . . . . . . . .     034A     BYTE     PUBLIC  'CODE'

Symbols:

                N a m e                        Type     Value    Attr

AUXDP. . . . . . . . . . . . . . . . .         Number   0040
AUXP . . . . . . . . . . . . . . . . .         Number   0042
AUX_PRN. . . . . . . . . . . . . . . .         Number   0044
BAILOUT. . . . . . . . . . . . . . . .         L NEAR   0304     CODE
BPBPTR . . . . . . . . . . . . . . . .         L WORD   0013     CODE
BUFFER . . . . . . . . . . . . . . . .         L NEAR   0000              External
BUSY . . . . . . . . . . . . . . . . .         L NEAR   00A1     CODE
CALLADDR . . . . . . . . . . . . . . .         L WORD   001E     CODE
CHKBYTE. . . . . . . . . . . . . . . .         L BYTE   0017     CODE
CHKMED . . . . . . . . . . . . . . . .         L NEAR   02B1     CODE
CKMDA1 . . . . . . . . . . . . . . . .         L NEAR   02C0     CODE
CLK_INT. . . . . . . . . . . . . . . .         Number   002C
CMDERR . . . . . . . . . . . . . . . .         L NEAR   0096     CODE
CMDOK. . . . . . . . . . . . . . . . .         L NEAR   0077     CODE
CMDSTR . . . . . . . . . . . . . . . .         L NEAR   0000     CODE     Global
COMDMA . . . . . . . . . . . . . . . .         Number   0043
COMMAND. . . . . . . . . . . . . . . .         L BYTE   0000     CODE
COMN . . . . . . . . . . . . . . . . .         L NEAR   00DB     CODE
CURTRK . . . . . . . . . . . . . . . .         L BYTE   0005     CODE
DCR1 . . . . . . . . . . . . . . . . .         L NEAR   0285     CODE
DENSITY. . . . . . . . . . . . . . . .         L BYTE   0006     CODE
DENTRK . . . . . . . . . . . . . . . .         L BYTE   0018     CODE
DISPATCH . . . . . . . . . . . . . . .         F PROC   002A     CODE     Length =006C
DMAOFF . . . . . . . . . . . . . . . .         L WORD   0020     CODE
DMASEG . . . . . . . . . . . . . . . .         L WORD   0022     CODE
DOLEFT . . . . . . . . . . . . . . . .         L NEAR   0134     CODE
DOPHYS . . . . . . . . . . . . . . . .         L NEAR   01D7     CODE
DORD . . . . . . . . . . . . . . . . .         L NEAR   01BE     CODE
DOSEC. . . . . . . . . . . . . . . . .         L NEAR   011E     CODE
DP1. . . . . . . . . . . . . . . . . .         L NEAR   01EB     CODE
```

```
DRIVER . . . . . . . . . . . . . . . .         L BYTE   0010     CODE
DSK. . . . . . . . . . . . . . . . . .         L NEAR   002A     CODE     Global
DSKDRCT. . . . . . . . . . . . . . . .         Number   0065
DSKDRV . . . . . . . . . . . . . . . .         L BYTE   0001     CODE
DSKINIT. . . . . . . . . . . . . . . .         L NEAR   0286     CODE
DSKIO. . . . . . . . . . . . . . . . .         Number   0065
DSKOFF . . . . . . . . . . . . . . . .         L WORD   000C     CODE
DSKSEG . . . . . . . . . . . . . . . .         L WORD   000E     CODE
DSKTBL . . . . . . . . . . . . . . . .         V WORD   0000              External
DSKTMR . . . . . . . . . . . . . . . .         L BYTE   0029     CODE     Global
DTBL . . . . . . . . . . . . . . . . .         L WORD   00AC     CODE
DTL. . . . . . . . . . . . . . . . . .         L BYTE   0009     CODE
ENN. . . . . . . . . . . . . . . . . .         L BYTE   0008     CODE
ERRMSK . . . . . . . . . . . . . . . .         L BYTE   001D     CODE
EXCOM. . . . . . . . . . . . . . . . .         Number   0045
EXITO. . . . . . . . . . . . . . . . .         L NEAR   0338     CODE
FDENS. . . . . . . . . . . . . . . . .         Number   0002
FREAD. . . . . . . . . . . . . . . . .         Number   0001
FWRITE . . . . . . . . . . . . . . . .         Number   0000
FWRITEV. . . . . . . . . . . . . . . .         Number   0003
GAPLEN . . . . . . . . . . . . . . . .         L BYTE   0007     CODE
GETBPB . . . . . . . . . . . . . . . .         L NEAR   029B     CODE
INITTBL. . . . . . . . . . . . . . . .         V WORD   0000              External
IOCNT. . . . . . . . . . . . . . . . .         Number   0006
IODRV. . . . . . . . . . . . . . . . .         Number   0001
IOFUN. . . . . . . . . . . . . . . . .         Number   0000
IOOFF. . . . . . . . . . . . . . . . .         Number   0008
IOPKT. . . . . . . . . . . . . . . . .         L NEAR   033A     CODE
IOSEC. . . . . . . . . . . . . . . . .         Number   0002
IOSEG. . . . . . . . . . . . . . . . .         Number   000A
IOTRK. . . . . . . . . . . . . . . . .         Number   0004
KDP. . . . . . . . . . . . . . . . . .         Number   0010
KSP. . . . . . . . . . . . . . . . . .         Number   0011
MAYBEC . . . . . . . . . . . . . . . .         L NEAR   026B     CODE
MC_FLG . . . . . . . . . . . . . . . .         L BYTE   0339     CODE     Global
MEDIACHK . . . . . . . . . . . . . . .         L NEAR   024D     CODE
NOINT. . . . . . . . . . . . . . . . .         L NEAR   0043     CODE
NORMAL . . . . . . . . . . . . . . . .         L NEAR   02CB     CODE
NUMSEC . . . . . . . . . . . . . . . .         L WORD   0026     CODE
OKMDIA . . . . . . . . . . . . . . . .         L NEAR   02F0     CODE
PHYSBUF. . . . . . . . . . . . . . . .         L NEAR   0184     CODE
PHYSCNT. . . . . . . . . . . . . . . .         L BYTE   0004     CODE
PHYSEC . . . . . . . . . . . . . . . .         L BYTE   0003     CODE
PRET . . . . . . . . . . . . . . . . .         L NEAR   01D6     CODE
PRNDP. . . . . . . . . . . . . . . . .         Number   0041
PRNP . . . . . . . . . . . . . . . . .         Number   0043
PROFILE. . . . . . . . . . . . . . . .         Number   0064
PTRSAVE. . . . . . . . . . . . . . . .         V DWORD  0000              External
RDWRET . . . . . . . . . . . . . . . .         L NEAR   013C     CODE
RDWRRET. . . . . . . . . . . . . . . .         L NEAR   0183     CODE
RDWSEC . . . . . . . . . . . . . . . .         L NEAR   00E0     CODE
READ . . . . . . . . . . . . . . . . .         L NEAR   00C8     CODE
READWRITE. . . . . . . . . . . . . . .         L NEAR   015A     CODE
RETURN . . . . . . . . . . . . . . . .         L NEAR   00A0     CODE
ROM. . . . . . . . . . . . . . . . . .         Number   0018
```

```
 The Microsoft MACRO Assembler                02-20-84      PAGE     Symbols-3
MSDOS 2.00 Universal Disk driver


RWCHK. . . . . . . . . . . . . . . .          L NEAR  016A      CODE
RWR1 . . . . . . . . . . . . . . . .          L NEAR  0150      CODE
SAMDSK . . . . . . . . . . . . . . .          L NEAR  026C      CODE
SCOUNT . . . . . . . . . . . . . . .          L WORD  001B      CODE
SDRV1. . . . . . . . . . . . . . . .          L NEAR  01FA      CODE
SECSIZ . . . . . . . . . . . . . . .          L WORD  000A      CODE
SECTOR . . . . . . . . . . . . . . .          L WORD  0024      CODE
SET_DRV. . . . . . . . . . . . . . .          L NEAR  01EC      CODE
SOFTCHK. . . . . . . . . . . . . . .          L NEAR  0305      CODE
SPARE41. . . . . . . . . . . . . . .          Number  0041
SPARE42. . . . . . . . . . . . . . .          Number  0042
SPT. . . . . . . . . . . . . . . . .          L WORD  0019      CODE
TMRCNT . . . . . . . . . . . . . . .          Number  0002
TMRDRV . . . . . . . . . . . . . . .          L BYTE  0028      CODE
TRACK. . . . . . . . . . . . . . . .          L BYTE  0002      CODE
TRKPTR . . . . . . . . . . . . . . .          L WORD  0011      CODE
TYPPTR . . . . . . . . . . . . . . .          L WORD  0015      CODE
UART . . . . . . . . . . . . . . . .          Number  0046
VERT . . . . . . . . . . . . . . . .          Number  0040
WPRSNT . . . . . . . . . . . . . . .          Number  0001
WRITE. . . . . . . . . . . . . . . .          L NEAR  00D6      CODE
WRITEV . . . . . . . . . . . . . . .          L NEAR  00CF      CODE
WRTBUF . . . . . . . . . . . . . . .          L NEAR  01A1      CODE
XCPRSNT. . . . . . . . . . . . . . .          Number  0002
Z80. . . . . . . . . . . . . . . . .          Number  0047

Warning Severe
Errors  Errors
0       0
```

```
 MSDOS 2.00 Universal Disk driver

 Symbol Cross Reference                    (# is definition)       Cref-1

AUXDP. . . . . . . . . . . . . . . .     57#
AUXP . . . . . . . . . . . . . . . .     59#
AUX_PRN. . . . . . . . . . . . . . .     47#

BAILOUT. . . . . . . . . . . . . . .    590     606     622#
BPBPTR . . . . . . . . . . . . . . .    135#    485     570
BUFFER . . . . . . . . . . . . . . .     67#    398     408     421     599     607     632     639
BUSY . . . . . . . . . . . . . . . .    215#    224

CALLADDR . . . . . . . . . . . . . .    142#    443     489
CGROUP . . . . . . . . . . . . . . .      1:     70      70
CHKBYTE. . . . . . . . . . . . . . .    137#    487
CHKMED . . . . . . . . . . . . . . .    568     585#
CKMDA1 . . . . . . . . . . . . . . .    587     591#
CLK_INT. . . . . . . . . . . . . . .     33#
CMD. . . . . . . . . . . . . . . . .     89#    169     183
CMDERR . . . . . . . . . . . . . . .    209#    233
CMDLEN . . . . . . . . . . . . . . .     87#
CMDOK. . . . . . . . . . . . . . . .    185     187#
CMDSTR . . . . . . . . . . . . . . .     14     115#    442
CODE . . . . . . . . . . . . . . . .     11      69#     69     657
COMDMA . . . . . . . . . . . . . . .     46#
COMMAND. . . . . . . . . . . . . . .    116#    258     261     264     401     403
COMN . . . . . . . . . . . . . . . .    259     262     265#
COUNT. . . . . . . . . . . . . . . .     97#    175     352     558     572
CURTRK . . . . . . . . . . . . . . .    121#    353     444     447     494     536

DCR1 . . . . . . . . . . . . . . . .    541     545#
DENSITY. . . . . . . . . . . . . . .    122#    501
DENTRK . . . . . . . . . . . . . . .    138#    505
DISPATCH . . . . . . . . . . . . . .    155#    204
DMAOFF . . . . . . . . . . . . . . .    143#    180     298     311     384     407     422
DMASEG . . . . . . . . . . . . . . .    144#    182     305     411     423
DOLEFT . . . . . . . . . . . . . . .    336     338#
DOPHYS . . . . . . . . . . . . . . .    415     418     441#
DORD . . . . . . . . . . . . . . . .    404     418#
DOSEC. . . . . . . . . . . . . . . .    329#
DP1. . . . . . . . . . . . . . . . .    445     448#
DRIVER . . . . . . . . . . . . . . .    133#    174     466     585     615     627
DSK. . . . . . . . . . . . . . . . .     13     154#
DSKDRCT. . . . . . . . . . . . . . .    106#    603     636
DSKDRV . . . . . . . . . . . . . . .    117#    374     503     526
DSKINIT. . . . . . . . . . . . . . .    219     552#
DSKIO. . . . . . . . . . . . . . . .     34#
DSKOFF . . . . . . . . . . . . . . .    127#    398
DSKSEG . . . . . . . . . . . . . . .    128#    399
DSKTBL . . . . . . . . . . . . . . .     17#    467     474     586     617
DSKTMR . . . . . . . . . . . . . . .     14     149#    376     529
DTBL . . . . . . . . . . . . . . . .    189     219#
DTL. . . . . . . . . . . . . . . . .    125#    511

ENN. . . . . . . . . . . . . . . . .    124#    509
ERRMSK . . . . . . . . . . . . . . .    141#
EXCOM. . . . . . . . . . . . . . . .     48#
EXITO. . . . . . . . . . . . . . . .    637     643     645#

FDENS. . . . . . . . . . . . . . . .    104#
```

```
 MSDOS 2.00 Universal Disk driver

 Symbol Cross Reference                    (# is definition)       Cref-2

FREAD. . . . . . . . . . . . . .     103#    258
FWRITE . . . . . . . . . . . . .     102#    264     403
FWRITEV. . . . . . . . . . . . .     105#    261     401

GAPLEN . . . . . . . . . . . . .     123#    507
GETBPB . . . . . . . . . . . . .     221     567#

INITTBL. . . . . . . . . . . . .      17#    553
IOCNT. . . . . . . . . . . . . .     581#    631
IODATA . . . . . . . . . . . . .      86#     99
IODRV. . . . . . . . . . . . . .     578#    600     628
IOFUN. . . . . . . . . . . . . .     577#    601     634
IOOFF. . . . . . . . . . . . . .     582#    599     632
IOPKT. . . . . . . . . . . . . .     597     626     652#
IOSEC. . . . . . . . . . . . . .     579#    629
IOSEG. . . . . . . . . . . . . .     583#    598     633
IOTRK. . . . . . . . . . . . . .     580#    630

KDP. . . . . . . . . . . . . . .      55#
KSP. . . . . . . . . . . . . . .      56#

MAYBEC . . . . . . . . . . . . .     528     530     532#
MC_FLG . . . . . . . . . . . . .      13     592     648#
MEDIA. . . . . . . . . . . . . .      95#    540     557
MEDIACHK . . . . . . . . . . . .     220     519#

NOINT. . . . . . . . . . . . . .     170     172#
NORMAL . . . . . . . . . . . . .     593     596#
NUMSEC . . . . . . . . . . . . .     146#    176     289     334     351     387

OKMDIA . . . . . . . . . . . . .     595     611     613#

PHYSBUF. . . . . . . . . . . . .     372     397#
PHYSCNT. . . . . . . . . . . . .     120#    400
PHYSEC . . . . . . . . . . . . .     119#    322     329     385
PRET . . . . . . . . . . . . . .     419     426#
PRNDP. . . . . . . . . . . . . .      58#
PRNP . . . . . . . . . . . . . .      60#
PROFILE. . . . . . . . . . . . .      32#
PTRSAVE. . . . . . . . . . . . .      16#    168     191     209     215     348     539     556     571

RDWRET . . . . . . . . . . . . .     266     291     347#
RDWRRET. . . . . . . . . . . . .     373     390#
RDWSEC . . . . . . . . . . . . .     289#    340
READ . . . . . . . . . . . . . .     223     258#
READWRITE. . . . . . . . . . . .     339     371#    389
RETURN . . . . . . . . . . . . .     211#    222     225     226     229     230     231
ROM. . . . . . . . . . . . . . .      51#
RWCHK. . . . . . . . . . . . . .     383#
RWR1 . . . . . . . . . . . . . .     347     353#

SAMDSK . . . . . . . . . . . . .     523     534#
SCOUNT . . . . . . . . . . . . .     140#    338     388
SDRV1. . . . . . . . . . . . . .     468     472#
SECSIZ . . . . . . . . . . . . .     126#    383     406     420     497
SECTOR . . . . . . . . . . . . .     145#    178     317     386
SET_DRV. . . . . . . . . . . . .     265     465#    520     569
```

```
 MSDOS 2.00 Universal Disk driver

 Symbol Cross Reference                    (# is definition)       Cref-3

SOFTCHK. . . . . . . . . . . . .     612     625#
SPARE41. . . . . . . . . . . . .      44#
SPARE42. . . . . . . . . . . . .      45#
SPT. . . . . . . . . . . . . . .     139#    318     331     499
START. . . . . . . . . . . . . .      98#    177     559     574
STATUS . . . . . . . . . . . . .      90#    172     192     210     216     350     544

TMRCNT . . . . . . . . . . . . .     107#    149     376
TMRDRV . . . . . . . . . . . . .     148#    375     527
TRACK. . . . . . . . . . . . . .     118#    321     446
TRANS. . . . . . . . . . . . . .      96#    179     181
TRKPTR . . . . . . . . . . . . .     134#    354     491     535
TYPPTR . . . . . . . . . . . . .     136#    476

UART . . . . . . . . . . . . . .      49#
UNIT . . . . . . . . . . . . . .      88#    173

VERT . . . . . . . . . . . . . .      43#

WPRSNT . . . . . . . . . . . . .      64#
WRITE. . . . . . . . . . . . . .     227     264#
WRITEV . . . . . . . . . . . . .     228     261#
WRTBUF . . . . . . . . . . . . .     402     405#

XCPRSNT. . . . . . . . . . . . .      65#

Z80. . . . . . . . . . . . . . .      50#
```

```
                                        PAGE 60,132
                                        TITLE   Hard Disk driver
                                        NAME    HDISK

                                ;
                                ;       COMPANY CONFIDENTIAL
                                ;       Copyright (C) 1983 Digital Equipment Corporation
                                ;       All rights reserved.
                                ;
                                ;       Hard Disk Driver for Version 2.05 of MS-DOS.
                                ;       09/22/83

                                CGROUP  GROUP   CODE
        0000                    CODE    SEGMENT BYTE PUBLIC 'CODE'
                                ASSUME  CS:CGROUP, DS:CGROUP, ES:CGROUP, SS:CGROUP

                                        EXTRN   HREAD:NEAR, HWRITE:NEAR, HWRITEV:NEAR
                                        EXTRN   HFORMAT:NEAR, HINIT:NEAR, HMCHK:NEAR
                                        EXTRN   HVDISK:NEAR, INTHDL:NEAR, DSK_INIT:NEAR
                                        EXTRN   XLT_F:BYTE, PTRSAVE:DWORD, NDRV:BYTE
                                        EXTRN   HIOCTL:NEAR, UP_BAT:NEAR
                                        EXTRN   TEMP_TRK:WORD, TEMP_SEC:BYTE

                                        PUBLIC  INI_TAB, RDXLT
                                        PUBLIC  GETFTN, GETFTNO, GETPEA
                                        PUBLIC  PARTITION, MNPARTS, PART_SIZE
                                        PUBLIC  HDSKO,DSK_INT

                                        .LIST

                                        SUBTTL  Common Drive parameter block definitions
```

```
                                        PAGE
                                DBP     STRUC

                                ;-------  Start of Drive Parameter Block.

        0000  ????              SECSZ   DW      ?               ;Sector size in bytes.                  (dpb)
        0002  ??                ALLOC   DB      ?               ;Number of sectors per alloc. block.    (dpb)
        0003  ????              RESSEC  DW      ?               ;Reserved sectors.                      (dpb)
        0005  ??                FATS    DB      ?               ;Number of FAT's.                       (dpb)
        0006  ????              MAXDIR  DW      ?               ;Number of root directory entries.      (dpb)
        0008  ????              SECTORS DW      ?               ;Number of sectors per diskette.        (dpb)
        000A  ??                MEDIAID DB      ?               ;Media byte ID.                         (dpb)
        000B  ????              FATSEC  DW      ?               ;Number of FAT Sectors.                 (dpb)

                                ;-------  End of Drive Parameter Block. (Extensions follow)

        000D  ????              SECTRK  DW      ?               ;Number of Sectors per track.
        000F  ????              HEADS   DW      ?               ;Number of heads per cylinder.
        0011  ????              HIDDEN  DW      ?               ;Number of hidden sectors.

        0013                    DBP     ENDS

        0000  0200              HDSKO   DBP     <512,4,2*16,2,256,0,0F8H,0,16,4,0>
        0002  04
        0003  0020
        0005  02
        0006  0100
        0008  0000
        000A  F8
        000B  0000
        000D  0010
        000F  0004
        0011  0000

        0013  0200              HDSK1   DBP     <512,4,2*16,2,256,0,0F8H,0,16,4,0>
        0015  04
        0016  0020
        0018  02
        0019  0100
        001B  0000
        001D  F8
        001E  0000
        0020  0010
        0022  0004
        0024  0000

        0026  0200              HDSK2   DBP     <512,4,2*16,2,256,0,0F8H,0,16,4,0>
        0028  04
        0029  0020
        002B  02
        002C  0100
        002E  0000
        0030  F8
        0031  0000
        0033  0010
```

```
0035  0004
0037  0000

0039  0200                       HDSK3   DBP     <512,4,2*16,2,256,0,0F8H,0,16,4,0>
003B  04
003C  0020
003E  02
003F  0100
0041  0000
0043  F8
0044  0000
0046  0010
0048  0004
004A  0000


004C  0000 R                     INI_TAB DW      OFFSET HDSK0
004E  0013 R                             DW      OFFSET HDSK1
0050  0026 R                             DW      OFFSET HDSK2
0052  0039 R                             DW      OFFSET HDSK3

                                         SUBTTL  Strategy and Software Interrupt routines.
```

```
                                         PAGE
                                 ;Define offsets for io data packet


                                 IODAT   STRUC
0000  ??                         CMDLEN  DB      ?               ;LENGTH OF THIS COMMAND
0001  ??                         UNIT    DB      ?               ;SUB UNIT SPECIFIER
0002  ??                         CMD     DB      ?               ;COMMAND CODE
0003  ????                       STATUS  DW      ?               ;STATUS
0005     08 [                            DB      8 DUP (?)
               ??
                   ]

000D  ??                         MEDIA   DB      ?               ;MEDIA DESCRIPTOR
000E  ????????                   TRANS   DD      ?               ;TRANSFER ADDRESS
0012  ????                       COUNT   DW      ?               ;COUNT OF BLOCKS OR CHARACTERS
0014  ????                       START   DW      ?               ;FIRST BLOCK TO TRANSFER
0016                             IODAT   ENDS

                                 ;***********************************************************
                                 ;
                                 ;       DSK_INT - Disk driver interrupt routine
                                 ;
                                 ;       When perform disk function command, register contains:
                                 ;       AL : Unit code.
                                 ;       AH : Media descriptor.
                                 ;       CX : Contains byte/sector count.
                                 ;       DX : Starting Logical sector.
                                 ;       ES:DI : Transfer address
                                 ;
                                 ;***********************************************************

0054                             dispatch proc far

0054                             DSK_INT:
0054  50                                 PUSH    AX
0055  53                                 PUSH    BX
0056  51                                 PUSH    CX
0057  52                                 PUSH    DX
0058  56                                 PUSH    SI
0059  57                                 PUSH    DI
005A  55                                 PUSH    BP
005B  1E                                 PUSH    DS
005C  06                                 PUSH    ES
005D  2E: C5 1E 0000 E                   LDS     BX,CS:[PTRSAVE]  ;Retrieve pointer to I/O Packet.
0062  80 7F 02 00                        CMP     BYTE PTR [BX.CMD],0     ;INIT?
0066  74 01                              JE      NOINT                   ;DON'T RENABLE INTS ON INIT
0068  FB                                 STI
0069  C7 47 03 0000              NOINT:  MOV     WORD PTR [BX.STATUS],0  ;Clear status initially

006E  8A 47 01                           MOV     AL,[BX.UNIT]    ;AL = Unit number
0071  8A 67 0D                           MOV     AH,[BX.MEDIA]   ;AH = Media descriptor.
0074  8B 4F 12                           MOV     CX,[BX.COUNT]   ;CX = Contains byte/sector count.
0077  8B 57 14                           MOV     DX,[BX.START]   ;DX = Starting Logical sector.
007A  97                                 XCHG    AX,DI           ;Save Unit and Media Temporarily.
007B  8A 47 02                           MOV     AL,[BX.CMD]     ;Retrieve Command type. (1 => 11)
```

```
164
165     007E  3C 0C                       CMP     AL,12           ;CHECK IN
166     0080  76 02                       JBE     CMDOK           ;RANGE, BOUND
167     0082  B0 0D                       MOV     AL,13           ;LIT,
168     0084                      CMDOK:
169     0084  32 E4                       XOR     AH,AH
170     0086  D1 E0                       SHL     AX,1
171     0088  05 00A9 R                   ADD     AX,OFFSET DSK_TBL
172     008B  8B F0                       MOV     SI,AX
173     008D  97                          XCHG    AX,DI           ; get unit and media back
174     008E  C4 7F 0E                    LES     DI,[BX.TRANS]   ;DI = transfer addr
175                                                               ;ES=segment
176     0091  0E                          PUSH    CS
177     0092  1F                          POP     DS
178     0093  FF 14                       CALL    WORD PTR [SI]
179     0095  2E: C5 1E 0000 E            LDS     BX,CS:[PTRSAVE]
180     009A  81 4F 03 0100               OR      WORD PTR [BX.STATUS],100H        ; set done flag
181
182     009F  07                          POP     ES
183     00A0  1F                          POP     DS
184     00A1  5D                          POP     BP
185     00A2  5F                          POP     DI
186     00A3  5E                          POP     SI
187     00A4  5A                          POP     DX
188     00A5  59                          POP     CX
189     00A6  5B                          POP     BX
190     00A7  58                          POP     AX
191     00A8  CB                          RET
192
193     00A9                      DISPATCH ENDP
```

```
194                                       PAGE
195     00A9                      DSK_TBL:
196     00A9  0000 E                      DW      DSK_INIT        ;0  - Initialize Driver.
197     00AB  00DB R                      DW      MEDIA_CHK       ;1  - Return current media code.
198     00AD  00E5 R                      DW      GET_BPB         ;2  - Get Bios Parameter Block.
199     00AF  0000 E                      DW      HIOCTL          ;3  - Hard disk IOCTL
200     00B1  00FB R                      DW      DSK_READ        ;4  - Block read.
201     00B3  00D0 R                      DW      BUSY            ;5  - (Not used, return busy flag)
202     00B5  00CF R                      DW      RETURN          ;6  - Return status. (Not used)
203     00B7  00CF R                      DW      RETURN          ;7  - Flush input buffer. (Not used.)
204     00B9  010B R                      DW      DSK_WRT         ;8  - Block write.
205     00BB  0103 R                      DW      DSK_WRTV        ;9  - Block write with verify.
206     00BD  00CF R                      DW      RETURN          ;10 - Return output status.
207     00BF  00CF R                      DW      RETURN          ;11 - Flush output buffer. (Not used.)
208     00C1  0000 E                      DW      HIOCTL          ;12 - Hard disk IOCTL
209     00C3  00C5 R                      DW      CMDERR          ;13 - Error
210
211                                       SUBTTL  Common error and exit points.
```

```
                                        PAGE
                                ;  Common error processing routine.
                                ;    AL contains actual error code.
                                ;
                                ;    Error # 0 = Write Protect violation.
                                ;            1 = Unkown unit.
                                ;            2 = Drive not ready.
                                ;            3 = Unknown command in I/O packet.
                                ;            4 = CRC error.
                                ;            5 = Bad drive request structure length.
                                ;            6 = Seek error.
                                ;            7 = Unknown media discovered.
                                ;            8 = Sector not found.
                                ;            9 = Printer out of paper.
                                ;           10 = Write fault.
                                ;           11 = Read fault.
                                ;           12 = General failure.
                                ;       command error

 00C5                           CMDERR:
 00C5  2E: C5 1E 0000 E                 LDS     BX,CS:[PTRSAVE]
 00CA  81 4F 03 8003                    OR      WORD PTR [BX.STATUS],8003H
 00CF  C3                       RETURN: RET


                                ;       return busy

 00D0                           BUSY:
 00D0  2E: C5 1E 0000 E                 LDS     BX,CS:[PTRSAVE]
 00D5  81 4F 03 0200                    OR      WORD PTR [BX.STATUS],200H
 00DA  C3                               RET

                                        SUBTTL  Media check routine
```

```
                                        PAGE
                                ;
                                ; Media check routine.
                                ; On entry:
                                ;       AL = memory driver unit number.
                                ;       AH = media byte
                                ; On exit:
                                ;
                                ;       [MEDIA FLAG] = -1 (FF hex) if disk is changed.
                                ;       [MEDIA FLAG] = 0 if don't know.
                                ;       [MEDIA FLAG] = 1 if not changed.
                                ;

 00DB                           MEDIA_CHK:
 00DB  2E: C5 1E 0000 E                 LDS     BX,CS:[PTRSAVE]
 00E0  C6 47 0E 01                      MOV     BYTE PTR [BX.TRANS],1
 00E4  C3                               RET

                                        SUBTTL  Build and return Bios Parameter Block for a diskette.
```

```
                                            PAGE
                                ;
                                ; Build Bios Parameter Blocks.
                                ;
                                ;           On entry:  ES:BX contains the address of a scratch sector buffer.
                                ;                      AL = Unit number.
                                ;                      AH = Current media byte.
                                ;
                                ;           On exit:   Return a DWORD pointer to tne associated BPB
                                ;                      in the Request packet.
                                ;

00E5                            GET_BPB:
00E5  32 E4                                 XOR     AH,AH                   ; clear high byte
00E7  D1 E0                                 SHL     AX,1                    ; shift to word offset
00E9  8B F0                                 MOV     SI,AX
00EB  8B 84 004C R                          MOV     AX,WORD PTR INI_TAB[SI] ; get BPB

00EF  2E: C5 1E 0000 E                      LDS     BX,CS:[PTRSAVE]
00F4  89 47 12                              MOV     WORD PTR [BX.COUNT],AX
00F7  8C 4F 14                              MOV     WORD PTR [BX.COUNT+2],CS
00FA  C3                                    RET

                                            SUBTTL  MSDOS 2.x Disk I/O drivers.
```

```
                                            PAGE
                                ;***********************************************************
                                ;
                                ;           Disk READ/WRITE functions.
                                ;
                                ;           ENTRY:
                                ;                   AL = Disk I/O driver number
                                ;                   AH = Media byte.
                                ;                   CX = Number of sectors to transfer
                                ;                   DX = Logical starting sector.
                                ;                   ES:DI = Disk transfer address
                                ;
                                ;           EXIT:
                                ;
                                ;***********************************************************

00FB                            DSK_READ:
00FB  E8 0167 R                             CALL    SETPAR
00FE  E8 0000 E                             CALL    HREAD
0101  EB 0E                                 JMP     SHORT COMM

0103                            DSK_WRTV:
0103  E8 0167 R                             CALL    SETPAR
0106  E8 0000 E                             CALL    HWRITEV
0109  EB 06                                 JMP     SHORT COMM

010B                            DSK_WRT:
010B  E8 0167 R                             CALL    SETPAR
010E  E8 0000 E                             CALL    HWRITE

0111                            COMM:
0111  73 2B                                 JNC     COMM1           ; exit if no error
0113  3C 0F                                 CMP     AL,SERR         ; soft error
0115  74 27                                 JZ      COMM1           ; return as good
0117  2E: C5 1E 0000 E                      LDS     BX,CS:[PTRSAVE]
011C  C7 47 12 0000                         MOV     WORD PTR [BX.COUNT],0   ; any error set count =0
0121  B9 0102                               MOV     CX,0102H        ; assume drive not ready error
0124  3C 03                                 CMP     AL,DRIVEDF
0126  74 10                                 JZ      COMM2
0128  B1 04                                 MOV     CL,4            ; assume CRC error
012A  F6 C4 40                              TEST    AH,CRCER
012D  75 09                                 JNZ     COMM2
012F  B1 08                                 MOV     CL,8            ; assume sector not found
0131  F6 C4 10                              TEST    AH,IDNF
0134  75 02                                 JNZ     COMM2
0136  B1 0C                                 MOV     CL,0CH          ; set as general error
0138                            COMM2:
0138  09 4F 03                              OR      WORD PTR [BX.STATUS],CX

013B  E8 0000 E                             CALL    UP_BAT          ; update BAT
013E                            COMM1:
013E  C3                                    RET

                                            SUBTTL  Partition Table
```

```
                                        PAGE
                                ;************************************************************
                                ;
                                ;       PARTITION
                                ;
                                ;       This is the logical drive partitions for this OS.
                                ;       If any one of the drive is not present, the first track #
                                ;       should contain a 0. (since track 0 will not be used for any OS)
                                ;       Note: this table will have more information about each partition.
                                ;       So, PART_SIZE will change!!!
                                ;
                                ;************************************************************


      = 0004                    MNPARTS         EQU     4       ; # of max. partitions in this system
      = 0006                    PART_SIZE       EQU     6       ; partition info size

      013F                      PARTITION       LABEL   WORD

      013F  0000                        DW      0               ; first track #
      0141  0000                        DW      0               ; last track #
      0143  0000                        DW      0               ; reserved

      0145  0000 0000 0000              DW      0,0,0
      014B  0000 0000 0000              DW      0,0,0
      0151  0000 0000 0000              DW      0,0,0


      0157  01 08 0F 06 0D 04   RDXLT   DB      1,8,15,6,13,4,11,2
            0B 02
      015F  09 10 07 0E 05 0C           DB      9,16,7,14,5,12,3,10
            03 0A
```

```
                                        PAGE
                                ;************************************************************
                                ;
                                ;       SETPAR - set up disk interface parameters
                                ;
                                ;       ENTRY:
                                ;               AL = unit code
                                ;               DX = logical sector #
                                ;       EXIT:   BL = sector #
                                ;               BH = surface #
                                ;               DX = cylinder #
                                ;               (XLT_F) = logical/physical flag
                                ;
                                ;************************************************************

      0167                      SETPAR:
      0167  C6 06 0000 E 01             MOV     BYTE PTR XLT_F,1        ; assume using translate table
      016C  8B F2                       MOV     SI,DX           ; save sector #
      016E  E8 01A7 R                   CALL    GETFTN          ; get partition first track #

      0171  8B C6                       MOV     AX,SI           ; get sector #
      0173  81 E6 000F                  AND     SI,0FH          ; form logical sector #
      0177  8A 9C 0157 R                MOV     BL,BYTE PTR RDXLT[SI]   ; get sector # from translate table

      017B  D1 E8                       SHR     AX,1            ; shift to track offset
      017D  D1 E8                       SHR     AX,1
      017F  D1 E8                       SHR     AX,1
      0181  D1 E8                       SHR     AX,1
      0183  3D 0002                     CMP     AX,2            ; check within first 2 track
      0186  7D 09                       JGE     SETPAR1
      0188  8B DE                       MOV     BX,SI           ; no skew on first 2 track
      018A  FE C3                       INC     BL              ; sector # is base 1
      018C  C6 06 0000 E 00             MOV     BYTE PTR XLT_F,0        ; don't use translate table
      0191                      SETPAR1:
      0191  03 D0                       ADD     DX,AX           ; add partition offset
      0193  89 16 0000 E                MOV     TEMP_TRK,DX     ; save track # in case error
      0197  88 1E 0000 E                MOV     TEMP_SEC,BL     ; save sector #
                                ;       JMP     CLTPN


                                ;************************************************************
                                ;
                                ;       CLTPN - Convert Logical Track # to Physical surface/cylinder #
                                ;
                                ;       ENTRY:  DX = logical track #
                                ;       EXIT:   BH = surface #
                                ;               DX = cylinder #
                                ;       USE:    NONE
                                ;
                                ;************************************************************

      019B                      CLTPN:
      019B  8A FA                       MOV     BH,DL           ; get low track byte
      019D  80 E7 03                    AND     BH,03H          ; keep surface only
      01A0  D1 EA                       SHR     DX,1            ; get rid of surface bits
```

```
 01A2  D1 EA                        SHR     DX,1
 01A4  C3                           RET

```

```
                                    PAGE
                            ;**************************************************************
                            ;
                            ;       GETFTN - GET partition First Track Number
                            ;
                            ;       ENTRY:
                            ;               GETFTN: AL = hard disk partition # (from 0 to MNPARTS-1)
                            ;               GETFTNO: AL = MS-DOS disk drive offset
                            ;       EXIT:   DX = start track # for this partition
                            ;               Z flag is not set if the partition is not exist
                            ;                 (start track # = 0), or wrong drive #
                            ;               BX = partition entry start addr
                            ;       USES:   AX, BX
                            ;
                            ;**************************************************************

 01A5                       GETFTNO:
 01A5  2C 04                        SUB     AL,HDISK            ; set to Wini offset
 01A7                       GETFTN:
 01A7  33 D2                        XOR     DX,DX               ; assume wrong drive #
 01A9  3C 04                        CMP     AL,MNPARTS          ; within limit
 01AB  73 05                        JNB     GETFTN1
 01AD  E8 01B5 R                    CALL    GETPEA              ; get partition start addr
 01B0  8B 17                        MOV     DX,[BX]             ; get start track #
 01B2                       GETFTN1:
 01B2  23 D2                        AND     DX,DX               ; set Z flag
 01B4  C3                           RET



                            ;**************************************************************
                            ;
                            ;       GETPEA - GET Partition Entry Addr
                            ;
                            ;       ENTRY:  AL = hard disk partition # (from 0 to MNPARTS-1)
                            ;       EXIT:   AX = BX = partition entry start addr
                            ;       USES:   AX, BX
                            ;
                            ;**************************************************************

 01B5                       GETPEA:
 01B5  B3 06                        MOV     BL,PART_SIZE                ; get partition size
 01B7  F6 E3                        MUL     BL                          ; set to partition entry offset
 01B9  05 013F R                    ADD     AX,OFFSET PARTITION         ; add start addr
 01BC  8B D8                        MOV     BX,AX                       ; be nice to BX
 01BE  C3                           RET


 01BF                       CODE    ENDS
                                    END
```

```
Structures and records:

                N a m e                  Width   # fields
                                         Shift   Width   Mask    Initial

DBP. . . . . . . . . . . . . . . .       0013    000B
  SECSZ. . . . . . . . . . . . . . .     0000
  ALLOC. . . . . . . . . . . . . . .     0002
  RESSEC . . . . . . . . . . . . . .     0003
  FATS . . . . . . . . . . . . . . .     0005
  MAXDIR . . . . . . . . . . . . . .     0006
  SECTORS. . . . . . . . . . . . . .     0008
  MEDIAID. . . . . . . . . . . . . .     000A
  FATSEC . . . . . . . . . . . . . .     000B
  SECTRK . . . . . . . . . . . . . .     000D
  HEADS. . . . . . . . . . . . . . .     000F
  HIDDEN . . . . . . . . . . . . . .     0011
IODAT. . . . . . . . . . . . . . .       0016    0009
  CMDLEN . . . . . . . . . . . . . .     0000
  UNIT . . . . . . . . . . . . . . .     0001
  CMD. . . . . . . . . . . . . . . .     0002
  STATUS . . . . . . . . . . . . . .     0003
  MEDIA. . . . . . . . . . . . . . .     000D
  TRANS. . . . . . . . . . . . . . .     000E
  COUNT. . . . . . . . . . . . . . .     0012
  START. . . . . . . . . . . . . . .     0014

Segments and groups:

                N a m e                  Size    align   combine class

CGROUP . . . . . . . . . . . . . .       GROUP
  CODE . . . . . . . . . . . . . . .     01BF    BYTE    PUBLIC  'CODE'

Symbols:

                N a m e                  Type    Value   Attr

ABRTC. . . . . . . . . . . . . . .       Number  0004
AST_OFF. . . . . . . . . . . . . .       Number  0026
ATRETRY. . . . . . . . . . . . . .       Number  0000
BADBD. . . . . . . . . . . . . . .       Number  0080
BAT_OFF. . . . . . . . . . . . . .       Number  0012
BID_AST. . . . . . . . . . . . . .       Number  0005
BID_BAT. . . . . . . . . . . . . .       Number  0002
BID_BOT. . . . . . . . . . . . . .       Number  0008
BID_DPD. . . . . . . . . . . . . .       Number  0003
BID_HOM. . . . . . . . . . . . . .       Number  0001
BID_OSN. . . . . . . . . . . . . .       Number  0004
BID_PAS. . . . . . . . . . . . . .       Number  0006
BID_PBT. . . . . . . . . . . . . .       Number  0007
BID_UKN. . . . . . . . . . . . . .       Number  0000
BIT0 . . . . . . . . . . . . . . .       Number  0001
BIT1 . . . . . . . . . . . . . . .       Number  0002
BIT15. . . . . . . . . . . . . . .       Number  8000
BIT2 . . . . . . . . . . . . . . .       Number  0004
```

```
BIT3 . . . . . . . . . . . . . . .       Number  0008
BIT4 . . . . . . . . . . . . . . .       Number  0010
BIT5 . . . . . . . . . . . . . . .       Number  0020
BIT6 . . . . . . . . . . . . . . .       Number  0040
BIT7 . . . . . . . . . . . . . . .       Number  0080
BK_EC_OFF. . . . . . . . . . . . .       Number  0008
BK_LBN_OFF . . . . . . . . . . . .       Number  0003
BK_MEC_OFF . . . . . . . . . . . .       Number  0006
BOOT_OFF . . . . . . . . . . . . .       Number  0021
BUSY . . . . . . . . . . . . . . .       L NEAR  00D0    CODE
CBSY . . . . . . . . . . . . . . .       Number  0080
CBSY2. . . . . . . . . . . . . . .       Number  0001
CLRIL. . . . . . . . . . . . . . .       Number  0004
CLRSPL . . . . . . . . . . . . . .       Number  0008
CLTPN. . . . . . . . . . . . . . .       L NEAR  019B    CODE
CMDERR . . . . . . . . . . . . . .       L NEAR  00C5    CODE
CMDIP. . . . . . . . . . . . . . .       Number  0002
CMDOK. . . . . . . . . . . . . . .       L NEAR  0084    CODE
COMM . . . . . . . . . . . . . . .       L NEAR  0111    CODE
COMM1. . . . . . . . . . . . . . .       L NEAR  013E    CODE
COMM2. . . . . . . . . . . . . . .       L NEAR  0138    CODE
CPM8680. . . . . . . . . . . . . .       Number  0001
CR . . . . . . . . . . . . . . . .       Number  000D
CRCER. . . . . . . . . . . . . . .       Number  0040
CTRL_Q . . . . . . . . . . . . . .       Number  0011
CTRL_S . . . . . . . . . . . . . .       Number  0013
CTRL_Z . . . . . . . . . . . . . .       Number  001A
CYLDH. . . . . . . . . . . . . . .       Number  0065
CYLDL. . . . . . . . . . . . . . .       Number  0064
DAMNF. . . . . . . . . . . . . . .       Number  0001
DATA . . . . . . . . . . . . . . .       Number  0060
DATARQ . . . . . . . . . . . . . .       Number  0008
DISKDF . . . . . . . . . . . . . .       Number  0005
DISPATCH . . . . . . . . . . . . .       F PROC  0054    CODE    Length =0055
DPDE_FLAG. . . . . . . . . . . . .       Number  0000
DPDE_FTN . . . . . . . . . . . . .       Number  000C
DPDE_INIT. . . . . . . . . . . . .       Number  00F0
DPDE_LTN . . . . . . . . . . . . .       Number  000E
DPDE_LU. . . . . . . . . . . . . .       Number  0001
DPDE_OFF . . . . . . . . . . . . .       Number  0020
DPDE_OST . . . . . . . . . . . . .       Number  000B
DPDE_PN. . . . . . . . . . . . . .       Number  0002
DPDE_PON . . . . . . . . . . . . .       Number  000A
DPD_OFF. . . . . . . . . . . . . .       Number  0017
DPD_START_OFF. . . . . . . . . . .       Number  0020
DRDY . . . . . . . . . . . . . . .       Number  0040
DRDY2. . . . . . . . . . . . . . .       Number  0040
DRIVE0 . . . . . . . . . . . . . .       Number  0000
DRIVE1 . . . . . . . . . . . . . .       Number  0008
DRIVE2 . . . . . . . . . . . . . .       Number  0010
DRIVE3 . . . . . . . . . . . . . .       Number  0018
DRIVEDF. . . . . . . . . . . . . .       Number  0003
DRIVEM . . . . . . . . . . . . . .       Number  0018
DRVSEL . . . . . . . . . . . . . .       Number  0001
DSK_INIT . . . . . . . . . . . . .       L NEAR  0000    CODE    External
```

```
DSK_INT. . . . . . . . . . . . .            L NEAR  0054    CODE    Global
DSK_READ . . . . . . . . . . . .            L NEAR  00FB    CODE
DSK_TBL. . . . . . . . . . . . .            L NEAR  00A9    CODE
DSK_WRT. . . . . . . . . . . . .            L NEAR  010B    CODE
DSK_WRTV . . . . . . . . . . . .            L NEAR  0103    CODE
DSTAT0 . . . . . . . . . . . . .            Number  0068
DSTAT1 . . . . . . . . . . . . .            Number  0069
DWRTF. . . . . . . . . . . . . .            Number  0020
DWRTF2 . . . . . . . . . . . . .            Number  0020
ERRF . . . . . . . . . . . . . .            Number  0001
ERROR. . . . . . . . . . . . . .            Number  0061
FALSE. . . . . . . . . . . . . .            Number  0000
FMTTRK . . . . . . . . . . . . .            Number  0050
GETFTN . . . . . . . . . . . . .            L NEAR  01A7    CODE    Global
GETFTN0. . . . . . . . . . . . .            L NEAR  01A5    CODE    Global
GETFTN1. . . . . . . . . . . . .            L NEAR  01B2    CODE
GETPEA . . . . . . . . . . . . .            L NEAR  01B5    CODE    Global
GET_BPB. . . . . . . . . . . . .            L NEAR  00E5    CODE
HDIR . . . . . . . . . . . . . .            Number  0008
HDISK. . . . . . . . . . . . . .            Number  0004
HDSK0. . . . . . . . . . . . . .            L 0013  0000    CODE    Global
HDSK1. . . . . . . . . . . . . .            L 0013  0013    CODE
HDSK2. . . . . . . . . . . . . .            L 0013  0026    CODE
HDSK3. . . . . . . . . . . . . .            L 0013  0039    CODE
HEAD0. . . . . . . . . . . . . .            Number  0000
HEAD02 . . . . . . . . . . . . .            Number  0000
HEAD1. . . . . . . . . . . . . .            Number  0001
HEAD12 . . . . . . . . . . . . .            Number  0002
HEAD2. . . . . . . . . . . . . .            Number  0002
HEAD22 . . . . . . . . . . . . .            Number  0004
HEAD3. . . . . . . . . . . . . .            Number  0003
HEAD32 . . . . . . . . . . . . .            Number  0006
HEADM. . . . . . . . . . . . . .            Number  0007
HEADM2 . . . . . . . . . . . . .            Number  000E
HFORMAT. . . . . . . . . . . . .            L NEAR  0000    CODE    External
HINIT. . . . . . . . . . . . . .            L NEAR  0000    CODE    External
HIOCTL . . . . . . . . . . . . .            L NEAR  0000    CODE    External
HIVEC_A. . . . . . . . . . . . .            Number  0114
HIVEC_B. . . . . . . . . . . . .            Number  0294
HMCHK. . . . . . . . . . . . . .            L NEAR  0000    CODE    External
HREAD. . . . . . . . . . . . . .            L NEAR  0000    CODE    External
HVDISK . . . . . . . . . . . . .            L NEAR  0000    CODE    External
HWRITE . . . . . . . . . . . . .            L NEAR  0000    CODE    External
HWRITEV. . . . . . . . . . . . .            L NEAR  0000    CODE    External
IDBIT. . . . . . . . . . . . . .            Number  00E0
IDBITV . . . . . . . . . . . . .            Number  00A0
IDNF . . . . . . . . . . . . . .            Number  0010
INI_TAB. . . . . . . . . . . . .            L WORD  004C    CODE    Global
INTHDL . . . . . . . . . . . . .            L NEAR  0000    CODE    External
LF . . . . . . . . . . . . . . .            Number  000A
LINDEX . . . . . . . . . . . . .            Number  0002
LSTEPP . . . . . . . . . . . . .            Number  0004
MEDIA_CHK. . . . . . . . . . . .            L NEAR  00DB    CODE
MNPARTS. . . . . . . . . . . . .            Number  0004            Global
MSCTF. . . . . . . . . . . . . .            Number  0004
```

```
MSDOS. . . . . . . . . . . . . .            Number  0002
NCYLD_OFF. . . . . . . . . . . .            Number  0040
NDRV . . . . . . . . . . . . . .            V BYTE  0000    CODE    External
NOINT. . . . . . . . . . . . . .            L NEAR  0069    CODE
NT_AST_OFF . . . . . . . . . . .            Number  002D
NUMHEAD. . . . . . . . . . . . .            Number  0004
OSN_OFF. . . . . . . . . . . . .            Number  001C
PARTITION. . . . . . . . . . . .            L WORD  013F    CODE    Global
PART_FTN . . . . . . . . . . . .            Number  0000
PART_LTN . . . . . . . . . . . .            Number  0002
PART_PAS . . . . . . . . . . . .            Number  0004
PART_SIZE. . . . . . . . . . . .            Number  0006            Global
PASE_OFF . . . . . . . . . . . .            Number  0005
PAS_ASN. . . . . . . . . . . . .            Number  0004
PAS_ATN. . . . . . . . . . . . .            Number  0003
PAS_BSN. . . . . . . . . . . . .            Number  0002
PAS_BTN. . . . . . . . . . . . .            Number  0000
PRE_READ . . . . . . . . . . . .            Alias   FALSE
PTRSAVE. . . . . . . . . . . . .            V DWORD 0000    CODE    External
RDBASE . . . . . . . . . . . . .            Number  0060
RDCMD. . . . . . . . . . . . . .            Number  0067
RDCMD2 . . . . . . . . . . . . .            Number  0068
RDINTF . . . . . . . . . . . . .            Number  0008
RDREAD . . . . . . . . . . . . .            Number  0020
RDSTAT . . . . . . . . . . . . .            Number  0067
RDWRITE. . . . . . . . . . . . .            Number  0030
RDXLT. . . . . . . . . . . . . .            L BYTE  0157    CODE    Global
RESTOR . . . . . . . . . . . . .            Number  0010
RETURN . . . . . . . . . . . . .            L NEAR  00CF    CODE
SBUFR. . . . . . . . . . . . . .            Number  0001
SCANID . . . . . . . . . . . . .            Number  0040
SCTEXT . . . . . . . . . . . . .            Number  0080
SDH. . . . . . . . . . . . . . .            Number  0086
SECPTRK. . . . . . . . . . . . .            Number  0010
SECSIZE. . . . . . . . . . . . .            Number  0200
SECTC. . . . . . . . . . . . . .            Number  0062
SECTN. . . . . . . . . . . . . .            Number  0063
SECTSZ . . . . . . . . . . . . .            Number  0060
SEEK . . . . . . . . . . . . . .            Number  0070
SEEKC. . . . . . . . . . . . . .            Number  0010
SEEKC2 . . . . . . . . . . . . .            Number  0010
SERR . . . . . . . . . . . . . .            Number  000F
SETPAR . . . . . . . . . . . . .            L NEAR  0167    CODE
SETPAR1. . . . . . . . . . . . .            L NEAR  0191    CODE
SINIT. . . . . . . . . . . . . .            Number  0002
SSCTF. . . . . . . . . . . . . .            Number  0000
SSZ128 . . . . . . . . . . . . .            Number  0060
SSZ1K. . . . . . . . . . . . . .            Number  0040
SSZ256 . . . . . . . . . . . . .            Number  0000
SSZ512 . . . . . . . . . . . . .            Number  0020
STEPR0 . . . . . . . . . . . . .            Number  0000
STEPR1 . . . . . . . . . . . . .            Number  0001
STEPR2 . . . . . . . . . . . . .            Number  0002
STEPR3 . . . . . . . . . . . . .            Number  0003
STEPR4 . . . . . . . . . . . . .            Number  0004
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE     Symbols-5
Hard Disk driver


STEPR5 . . . . . . . . . . . . . . . .        Number   0005
STEPR6 . . . . . . . . . . . . . . . .        Number   0006
STEPR7 . . . . . . . . . . . . . . . .        Number   0007
STEPR8 . . . . . . . . . . . . . . . .        Number   0008
STEPR9 . . . . . . . . . . . . . . . .        Number   0009
STEPRA . . . . . . . . . . . . . . . .        Number   000A
STEPRB . . . . . . . . . . . . . . . .        Number   000B
STEPRC . . . . . . . . . . . . . . . .        Number   000C
STEPRD . . . . . . . . . . . . . . . .        Number   000D
STEPRE . . . . . . . . . . . . . . . .        Number   000E
STEPRF . . . . . . . . . . . . . . . .        Number   000F
STEPR_OFF. . . . . . . . . . . . . . .        Number   004C
ST_AST_OFF . . . . . . . . . . . . . .        Number   002B
TEMP_SEC . . . . . . . . . . . . . . .        V BYTE   0000     CODE     External
TEMP_TRK . . . . . . . . . . . . . . .        V WORD   0000     CODE     External
TRK02. . . . . . . . . . . . . . . . .        Number   0080
TRK0E. . . . . . . . . . . . . . . . .        Number   0002
TRUE . . . . . . . . . . . . . . . . .        Number   - 0001
UP_BAT . . . . . . . . . . . . . . . .        L NEAR   0000     CODE     External
WGATE. . . . . . . . . . . . . . . . .        Number   0010
WINI_OFFSET. . . . . . . . . . . . . .        Number   0094
WINI_SEG . . . . . . . . . . . . . . .        Number   0096
WPC_OFF. . . . . . . . . . . . . . . .        Number   004A
WPRCMP . . . . . . . . . . . . . . . .        Number   0061
WPRSNT . . . . . . . . . . . . . . . .        Number   0001
WTVERR . . . . . . . . . . . . . . . .        Number   000D
XBB_BO . . . . . . . . . . . . . . . .        Number   0008
XBB_BS . . . . . . . . . . . . . . . .        Number   000A
XBB_CN . . . . . . . . . . . . . . . .        Number   0004
XBB_DISKF. . . . . . . . . . . . . . .        Number   0000
XBB_DSN. . . . . . . . . . . . . . . .        Number   0003
XBB_ERRC . . . . . . . . . . . . . . .        Number   000C
XBB_LDN. . . . . . . . . . . . . . . .        Number   0001
XBB_SC . . . . . . . . . . . . . . . .        Number   0006
XBB_SN . . . . . . . . . . . . . . . .        Number   0002
XCPRSNT. . . . . . . . . . . . . . . .        Number   0002
XLT_F. . . . . . . . . . . . . . . . .        V BYTE   0000     CODE     External

Warning Severe
Errors  Errors
0       0
```

```
 Hard Disk driver

 Symbol Cross Reference                     (# is definition)        Cref-1

ABRTC. . . . . . . . . . . . . . . .       29#
ALLOC. . . . . . . . . . . . . . . .       38#
AST_OFF. . . . . . . . . . . . . . .       29#
ATRETRY. . . . . . . . . . . . . . .       29#

BADBD. . . . . . . . . . . . . . . .       29#
BAT_OFF. . . . . . . . . . . . . . .       29#
BID_AST. . . . . . . . . . . . . . .       29#
BID_BAT. . . . . . . . . . . . . . .       29#
BID_BOT. . . . . . . . . . . . . . .       29#
BID_DPD. . . . . . . . . . . . . . .       29#
BID_HOM. . . . . . . . . . . . . . .       29#
BID_OSN. . . . . . . . . . . . . . .       29#
BID_PAS. . . . . . . . . . . . . . .       29#
BID_PBT. . . . . . . . . . . . . . .       29#
BID_UKN. . . . . . . . . . . . . . .       29#
BIT0 . . . . . . . . . . . . . . . .       29#
BIT1 . . . . . . . . . . . . . . . .       29#
BIT15. . . . . . . . . . . . . . . .       29#
BIT2 . . . . . . . . . . . . . . . .       29#
BIT3 . . . . . . . . . . . . . . . .       29#
BIT4 . . . . . . . . . . . . . . . .       29#
BIT5 . . . . . . . . . . . . . . . .       29#
BIT6 . . . . . . . . . . . . . . . .       29#
BIT7 . . . . . . . . . . . . . . . .       29#
BK_EC_OFF. . . . . . . . . . . . . .       29#
BK_LBN_OFF . . . . . . . . . . . . .       29#
BK_MEC_OFF . . . . . . . . . . . . .       29#
BOOT_OFF . . . . . . . . . . . . . .       29#
BUSY . . . . . . . . . . . . . . . .      201      239#

CBSY . . . . . . . . . . . . . . . .       29#
CBSY2. . . . . . . . . . . . . . . .       29#
CGROUP . . . . . . . . . . . . . . .       13       15       15       15       15
CLRIL. . . . . . . . . . . . . . . .       29#
CLRSPL . . . . . . . . . . . . . . .       29#
CLTPN. . . . . . . . . . . . . . . .      426#
CMD. . . . . . . . . . . . . . . . .      115#     153      163
CMDERR . . . . . . . . . . . . . . .      209      231#
CMDIP. . . . . . . . . . . . . . . .       29#
CMDLEN . . . . . . . . . . . . . . .      113#
CMDOK. . . . . . . . . . . . . . . .      166      168#
CODE . . . . . . . . . . . . . . . .       13       14#      14      481
COMM . . . . . . . . . . . . . . . .      307      312      318#
COMM1. . . . . . . . . . . . . . . .      319      321      338#
COMM2. . . . . . . . . . . . . . . .      326      329      332      334#
COUNT. . . . . . . . . . . . . . . .      123#     160      283      284      323
CPM8680. . . . . . . . . . . . . . .       29#
CR . . . . . . . . . . . . . . . . .       29#
CRCER. . . . . . . . . . . . . . . .       29#     328
CTRL_Q . . . . . . . . . . . . . . .       29#
CTRL_S . . . . . . . . . . . . . . .       29#
CTRL_Z . . . . . . . . . . . . . . .       29#
CYLDH. . . . . . . . . . . . . . . .       29#
CYLDL. . . . . . . . . . . . . . . .       29#

DAMNF. . . . . . . . . . . . . . . .       29#
```

```
Hard Disk driver

Symbol Cross Reference                    (# is definition)       Cref-2

DATA . . . . . . . . . . . . . . . .      29#
DATARQ . . . . . . . . . . . . . . .      29#
DBP. . . . . . . . . . . . . . . . .      33#     52
DISKDF . . . . . . . . . . . . . . .      29#
DISPATCH . . . . . . . . . . . . . .     140#    193
DPDE_FLAG. . . . . . . . . . . . . .      29#
DPDE_FTN . . . . . . . . . . . . . .      29#
DPDE_INIT. . . . . . . . . . . . . .      29#
DPDE_LTN . . . . . . . . . . . . . .      29#
DPDE_LU. . . . . . . . . . . . . . .      29#
DPDE_OFF . . . . . . . . . . . . . .      29#
DPDE_OST . . . . . . . . . . . . . .      29#
DPDE_PN. . . . . . . . . . . . . . .      29#
DPDE_PON . . . . . . . . . . . . . .      29#
DPD_OFF. . . . . . . . . . . . . . .      29#
DPD_START_OFF. . . . . . . . . . . .      29#
DRDY . . . . . . . . . . . . . . . .      29#
DRDY2. . . . . . . . . . . . . . . .      29#
DRIVE0 . . . . . . . . . . . . . . .      29#
DRIVE1 . . . . . . . . . . . . . . .      29#
DRIVE2 . . . . . . . . . . . . . . .      29#
DRIVE3 . . . . . . . . . . . . . . .      29#
DRIVEDF. . . . . . . . . . . . . . .      29#    325
DRIVEM . . . . . . . . . . . . . . .      29#
DRVSEL . . . . . . . . . . . . . . .      29#
DSK_INIT . . . . . . . . . . . . . .      19#    196
DSK_INT. . . . . . . . . . . . . . .      27     142#
DSK_READ . . . . . . . . . . . . . .     200     304#
DSK_TBL. . . . . . . . . . . . . . .     171     195#
DSK_WRT. . . . . . . . . . . . . . .     204     314#
DSK_WRTV . . . . . . . . . . . . . .     205     309#
DSTAT0 . . . . . . . . . . . . . . .      29#
DSTAT1 . . . . . . . . . . . . . . .      29#
DWRTF. . . . . . . . . . . . . . . .      29#
DWRTF2 . . . . . . . . . . . . . . .      29#

ERRF . . . . . . . . . . . . . . . .      29#
ERROR. . . . . . . . . . . . . . . .      29#

FALSE. . . . . . . . . . . . . . . .      29#     29
FATS . . . . . . . . . . . . . . . .      40#
FATSEC . . . . . . . . . . . . . . .      44#
FMTTRK . . . . . . . . . . . . . . .      29#

GETFTN . . . . . . . . . . . . . . .      25     393     451#
GETFTNO. . . . . . . . . . . . . . .      25     449#
GETFTN1. . . . . . . . . . . . . . .     454     457#
GETPEA . . . . . . . . . . . . . . .      25     455     473#
GET_BPB. . . . . . . . . . . . . . .     198     276#

HDIR . . . . . . . . . . . . . . . .      29#
HDISK. . . . . . . . . . . . . . . .      29#    450
HDSK0. . . . . . . . . . . . . . . .      27      54#    103
HDSK1. . . . . . . . . . . . . . . .      66#    104
HDSK2. . . . . . . . . . . . . . . .      78#    105
HDSK3. . . . . . . . . . . . . . . .      90#    106
HEAD0. . . . . . . . . . . . . . . .      29#
```

```
Hard Disk driver

Symbol Cross Reference                    (# is definition)       Cref-3

HEAD02 . . . . . . . . . . . . . . .      29#
HEAD1. . . . . . . . . . . . . . . .      29#
HEAD12 . . . . . . . . . . . . . . .      29#
HEAD2. . . . . . . . . . . . . . . .      29#
HEAD22 . . . . . . . . . . . . . . .      29#
HEAD3. . . . . . . . . . . . . . . .      29#
HEAD32 . . . . . . . . . . . . . . .      29#
HEADM. . . . . . . . . . . . . . . .      29#
HEADM2 . . . . . . . . . . . . . . .      29#
HEADS. . . . . . . . . . . . . . . .      49#
HFORMAT. . . . . . . . . . . . . . .      18#
HIDDEN . . . . . . . . . . . . . . .      50#
HINIT. . . . . . . . . . . . . . . .      18#
HIOCTL . . . . . . . . . . . . . . .      21#    199     208
HIVEC_A. . . . . . . . . . . . . . .      29#
HIVEC_B. . . . . . . . . . . . . . .      29#
HMCHK. . . . . . . . . . . . . . . .      18#
HREAD. . . . . . . . . . . . . . . .      17#    306
HVDISK . . . . . . . . . . . . . . .      19#
HWRITE . . . . . . . . . . . . . . .      17#    316
HWRITEV. . . . . . . . . . . . . . .      17#    311

IDBIT. . . . . . . . . . . . . . . .      29#
IDBITV . . . . . . . . . . . . . . .      29#
IDNF . . . . . . . . . . . . . . . .      29#    331
INI_TAB. . . . . . . . . . . . . . .      24     103#    280
INTHDL . . . . . . . . . . . . . . .      19#
IODAT. . . . . . . . . . . . . . . .     112#    125

LF . . . . . . . . . . . . . . . . .      29#
LINDEX . . . . . . . . . . . . . . .      29#
LSTEPP . . . . . . . . . . . . . . .      29#

MAXDIR . . . . . . . . . . . . . . .      41#
MEDIA. . . . . . . . . . . . . . . .     121#    159
MEDIAID. . . . . . . . . . . . . . .      43#
MEDIA_CHK. . . . . . . . . . . . . .     197     258#
MNPARTS. . . . . . . . . . . . . . .      26     357#    453
MSCTF. . . . . . . . . . . . . . . .      29#
MSDOS. . . . . . . . . . . . . . . .      29#

NCYLD_OFF. . . . . . . . . . . . . .      29#
NDRV . . . . . . . . . . . . . . . .      20#
NOINT. . . . . . . . . . . . . . . .     154     156#
NT_AST_OFF . . . . . . . . . . . . .      29#
NUMHEAD. . . . . . . . . . . . . . .      29#

OSN_OFF. . . . . . . . . . . . . . .      29#

PARTITION. . . . . . . . . . . . . .      26     360#    476
PART_FTN . . . . . . . . . . . . . .      29#
PART_LTN . . . . . . . . . . . . . .      29#
PART_PAS . . . . . . . . . . . . . .      29#
PART_SIZE. . . . . . . . . . . . . .      26     358#    474
PASE_OFF . . . . . . . . . . . . . .      29#
PAS_ASN. . . . . . . . . . . . . . .      29#
PAS_ATN. . . . . . . . . . . . . . .      29#
```

```
 Hard Disk driver

 Symbol Cross Reference                   (# is definition)      Cref-4

PAS_BSN. . . . . . . . . . . . .    29#
PAS_BTN. . . . . . . . . . . . .    29#
PRE_READ . . . . . . . . . . . .    29#
PTRSAVE. . . . . . . . . . . . .    20#    152    179    232    240    259    282    322

RDBASE . . . . . . . . . . . . .    29#     29     29     29     29     29     29     29     29     29     29     29     29     29
RDCMD. . . . . . . . . . . . . .    29#
RDCMD2 . . . . . . . . . . . . .    29#
RDINTF . . . . . . . . . . . . .    29#
RDREAD . . . . . . . . . . . . .    29#
RDSTAT . . . . . . . . . . . . .    29#
RDWRITE. . . . . . . . . . . . .    29#
RDXLT. . . . . . . . . . . . . .    24     371#   397
RESSEC . . . . . . . . . . . . .    39#
RESTOR . . . . . . . . . . . . .    29#
RETURN . . . . . . . . . . . . .   202     203    206    207    234#

SBUFR. . . . . . . . . . . . . .    29#
SCANID . . . . . . . . . . . . .    29#
SCTEXT . . . . . . . . . . . . .    29#
SDH. . . . . . . . . . . . . . .    29#
SECPTRK. . . . . . . . . . . . .    29#
SECSIZE. . . . . . . . . . . . .    29#
SECSZ. . . . . . . . . . . . . .    37#
SECTC. . . . . . . . . . . . . .    29#
SECTN. . . . . . . . . . . . . .    29#
SECTORS. . . . . . . . . . . . .    42#
SECTRK . . . . . . . . . . . . .    48#
SECTSZ . . . . . . . . . . . . .    29#
SEEK . . . . . . . . . . . . . .    29#
SEEKC. . . . . . . . . . . . . .    29#
SEEKC2 . . . . . . . . . . . . .    29#
SERR . . . . . . . . . . . . . .    29#    320
SETPAR . . . . . . . . . . . . .   305     310    315    390#
SETPAR1. . . . . . . . . . . . .   404     408#
SINIT. . . . . . . . . . . . . .    29#
SSCTF. . . . . . . . . . . . . .    29#
SSZ128 . . . . . . . . . . . . .    29#
SSZ1K. . . . . . . . . . . . . .    29#
SSZ256 . . . . . . . . . . . . .    29#
SSZ512 . . . . . . . . . . . . .    29#
START. . . . . . . . . . . . . .   124#    161
STATUS . . . . . . . . . . . . .   116#    156    180    233    241    335
STEPR0 . . . . . . . . . . . . .    29#
STEPR1 . . . . . . . . . . . . .    29#
STEPR2 . . . . . . . . . . . . .    29#
STEPR3 . . . . . . . . . . . . .    29#
STEPR4 . . . . . . . . . . . . .    29#
STEPR5 . . . . . . . . . . . . .    29#
STEPR6 . . . . . . . . . . . . .    29#
STEPR7 . . . . . . . . . . . . .    29#
STEPR8 . . . . . . . . . . . . .    29#
STEPR9 . . . . . . . . . . . . .    29#
STEPRA . . . . . . . . . . . . .    29#
STEPRB . . . . . . . . . . . . .    29#
STEPRC . . . . . . . . . . . . .    29#
STEPRD . . . . . . . . . . . . .    29#
```

```
 Hard Disk driver

 Symbol Cross Reference                   (# is definition)      Cref-5

STEPRE . . . . . . . . . . . . .    29#
STEPRF . . . . . . . . . . . . .    29#
STEPR_OFF. . . . . . . . . . . .    29#
ST_AST_OFF . . . . . . . . . . .    29#

TEMP_SEC . . . . . . . . . . . .    22#    411
TEMP_TRK . . . . . . . . . . . .    22#    410
TRANS. . . . . . . . . . . . . .   122#    174    260
TRK02. . . . . . . . . . . . . .    29#
TRK0E. . . . . . . . . . . . . .    29#
TRUE . . . . . . . . . . . . . .    29#     29

UNIT . . . . . . . . . . . . . .   114#    158
UP_BAT . . . . . . . . . . . . .    21#    337

WGATE. . . . . . . . . . . . . .    29#
WINI_OFFSET. . . . . . . . . . .    29#
WINI_SEG . . . . . . . . . . . .    29#
WPC_OFF. . . . . . . . . . . . .    29#
WPRCMP . . . . . . . . . . . . .    29#
WPRSNT . . . . . . . . . . . . .    29#
WTVERR . . . . . . . . . . . . .    29#

XBB_BO . . . . . . . . . . . . .    29#
XBB_BS . . . . . . . . . . . . .    29#
XBB_CN . . . . . . . . . . . . .    29#
XBB_DISKF. . . . . . . . . . . .    29#
XBB_DSN. . . . . . . . . . . . .    29#
XBB_ERRC . . . . . . . . . . . .    29#
XBB_LDN. . . . . . . . . . . . .    29#
XBB_SC . . . . . . . . . . . . .    29#
XBB_SN . . . . . . . . . . . . .    29#
XCPRSNT. . . . . . . . . . . . .    29#
XLT_F. . . . . . . . . . . . . .    20#    391    407
```

```
                                                PAGE 60,132
                                                TITLE   Hard Disk I/O Control Functions
                                                NAME    HIOCTL

                                        ;
                                        ;       COMPANY CONFIDENTIAL
                                        ;       Copyright (C) 1983 Digital Equipment Corporation
                                        ;       All rights reserved.
                                        ;
                                        ;       Hard Disk I/O Control Functions
                                        ;       08/30/83

                                        CGROUP  GROUP   CODE
      0000                              CODE    SEGMENT BYTE PUBLIC 'CODE'
                                        ASSUME  CS:CGROUP, DS:CGROUP, ES:CGROUP, SS:CGROUP

                                                EXTRN   HREAD:NEAR, HWRITE:NEAR, HWRITEV:NEAR
                                                EXTRN   HFORMAT:NEAR, HINIT:NEAR, HMCHK:NEAR
                                                EXTRN   HVDISK:NEAR, INTHDL:NEAR, DSK_INIT:NEAR
                                                EXTRN   XLT_F:BYTE, RDXLT:BYTE
                                                EXTRN   GETFTN:NEAR, GETFTNO:NEAR, GETPEA:NEAR, CLTPN:NEAR
                                                EXTRN   PARTITION:WORD

                                                PUBLIC  HIOCTL

                                                .LIST


                                                SUBTTL  Hard disk I/O Control functions
```

Hard disk I/O Control functions

```
                                                PAGE
                                        ;************************************************************
                                        ;
                                        ;       HIOCTL - Hard disk I/O Control functions
                                        ;
                                        ;************************************************************

      0000                              HIOCTL:
                                        ;XB_DISK:                               ; CP/M label
      0000  E8 0079 R                           CALL    GET_REG                 ; get all information
      0003  80 FC FF                            CMP     AH,0FFH                 ; check logical or physical operation
      0006  75 08                               JNZ     XBDSK1

                                        ;       physical disk operation

      0008  80 FF 40                            CMP     BH,40H                  ; check floppy or Winnie
      000B  7C 0B                               JL      XB_FLOPPY               ; it's floppy
      000D  EB 0D 90                            JMP     W_DISK                  ; it's Wini

                                        ;       logical disk operation

      0010  80 FC 04                    XBDSK1: CMP     AH,HDISK                ; check floppy or Winnie
      0013  7C 03                               JL      XB_FLOPPY
      0015  EB 05 90                            JMP     W_DISK                  ; disk is Winnie

                                        ;       hard disk driver will not do the hard work for floppy

      0018                              XB_FLOPPY:
      0018  B8 0005                             MOV     AX,DISKDF               ; set error flag to disk deffective
      001B  C3                                  RET
```

```
                                           PAGE
                                ;*****************************************************
                                ;
                                ;          W_DISK - Wini XBIOS functions
                                ;
                                ;          ENTRY:
                                ;                  AL = disk function
                                ;                  AH = MS-DOS logical disk drive #
                                ;                     = OFFH if physical operation
                                ;                  BL = sector #
                                ;                  BH = high 4 nibbles = drive #
                                ;                       low  4 nibbles = side or surface #
                                ;                  DX = track # if logical, cylinder # if physical
                                ;                  CX = sector count
                                ;                  DI = buffer offset
                                ;                  ES = buffer segment
                                ;          EXIT:   AX = error code defined in Wini disk driver spec
                                ;
                                ;*****************************************************

001C                            W_DISK:
001C  C6 06 0000 E 00                    MOV     BYTE PTR XLT_F,0         ; assume physical operation
0021  80 FC FF                           CMP     AH,OFFH             ; check physical or logical
0024  74 26                              JZ      WDISK2              ; it's physical

                                ;          logical disk operation

0026  50                                 PUSH    AX                  ; save function #
0027  53                                 PUSH    BX                  ; save sector #
0028  8B F2                              MOV     SI,DX               ; save logical track #
002A  83 FE 02                           CMP     SI,2                ; check if first 2 tracks?
002D  72 05                              JB      WDISK1
002F  C6 06 0000 E 01                    MOV     BYTE PTR XLT_F,1         ; set logical flag
0034                            WDISK1:
0034  8A C4                              MOV     AL,AH               ; get MS-DOS drive #
0036  E8 0000 E                          CALL    GETFTNO             ; get partition first track #
0039  5B                                 POP     BX                  ; restore sector #
003A  74 24                              JZ      WDISK10             ; error if drive not exist
003C  03 D6                              ADD     DX,SI               ; add start track #
003E  F6 06 0000 E FF                    TEST    XLT_F,OFFH          ; IF WE'RE NOT SUPPOSED TO SKEW,
0043  74 03                              JZ      WDISK3              ; DONT
0045  E8 0065 R                          CALL    DO_XLT              ; XLT the first sector #
0048                            WDISK3:
                                ;        CALL    DO_PASX             ; check if bad sector
0048  E8 0000 E                          CALL    CLTPN               ; convert logical track to physical

004B  58                                 POP     AX                  ; restore function #

                                ;          physical disk operation

004C                            WDISK2:
004C  80 E7 OF                           AND     BH,OFH              ; physical only drive O exist
004F  98                                 CBW
0050  D1 EO                              SHL     AX,1                ; word offset
0052  8B FO                              MOV     SI,AX
```

```
0054  FF 94 006D R                       CALL    WORD PTR W_DISKF[SI]     ; do it
0058  C5 3E 0095 R                       LDS     DI,TEMP_PTR         ; get pointer back
005C  89 45 OC                           MOV     WORD PTR [DI.XBB_ERRC],AX        ; set error code
005F  C3                                 RET

0060                            WDISK10:                             ; logical drive not exist
0060  58                                 POP     AX
0061  B8 0005                            MOV     AX,DISKDF           ; set error flag to disk deffective
0064  C3                                 RET

                                ;          DO_XLT - do translate for the first sector number
                                ;
                                ;          ENTRY:  BL = logical sector number **base 1
                                ;          EXIT:   BL = sector number after skewing
                                ;          USE:    BX

0065                            DO_XLT:
0065  32 FF                              XOR     BH,BH               ; clear high byte
0067  4B                                 DEC     BX                  ; make it base O
0068  8A 9F 0000 E                       MOV     BL,BYTE PTR RDXLT[BX]    ; set logical sector #
006C  C3                                 RET


006D  0000 E                    W_DISKF DW       HREAD
006F  0000 E                             DW      HWRITE
0071  0000 E                             DW      HWRITEV
0073  0000 E                             DW      HFORMAT
0075  0000 E                             DW      HMCHK
0077  0000 E                             DW      HVDISK
```

```
                                  Hard disk I/O Control functions

                                                PAGE
                                        ;*****************************************************
                                        ;
                                        ;       GET_REG - get all block information into registers
                                        ;
                                        ;       ENTRY:  ES:DI = block pointer
                                        ;       EXIT:   AL = disk function
                                        ;               AH = MS-DOS logical disk drive #
                                        ;                  = OFFH if physical operation
                                        ;               BL = sector #
                                        ;               BH = high 4 nibbles = physical drive #
                                        ;                    low  4 nibbles = side or surface #
                                        ;               DX = track or cylinder #
                                        ;               CX = sector count
                                        ;               DI = buffer offset
                                        ;               ES = buffer segment
                                        ;
                                        ;*****************************************************

      0079                              GET_REG:
      0079  89 3E 0095 R                        MOV     WORD PTR [TEMP_PTR],DI          ; save pointer
      007D  8C 06 0097 R                        MOV     WORD PTR [TEMP_PTR+2],ES

      0081  26: 8B 05                           MOV     AX,ES:XBB_DISKF[DI]     ; get logical disk # and disk function
      0084  26: 8B 5D 02                        MOV     BX,ES:XBB_SN[DI]        ; get drive/surface # and sector #
      0088  26: 8B 55 04                        MOV     DX,ES:XBB_CN[DI]        ; get track (cylinder) #
      008C  26: 8B 4D 06                        MOV     CX,ES:XBB_SC[DI]        ; get sector count
      0090  26: C4 7D 08                        LES     DI,ES:DWORD PTR XBB_BO[DI]      ; get buffer offset
      0094  C3                                  RET

      0095  00 00 00 00                 TEMP_PTR        DD      0

      0099                              CODE    ENDS
                                                END
```

```
Segments and groups:

                N a m e                         Size    align   combine class

CGROUP . . . . . . . . . . . . . .              GROUP
  CODE . . . . . . . . . . . . . . .            0099    BYTE    PUBLIC  'CODE'

Symbols:

                N a m e                         Type    Value   Attr

ABRTC. . . . . . . . . . . . . . .              Number  0004
AST_OFF. . . . . . . . . . . . . .              Number  0026
ATRETRY. . . . . . . . . . . . . .              Number  0000
BADBD. . . . . . . . . . . . . . .              Number  0080
BAT_OFF. . . . . . . . . . . . . .              Number  0012
BID_AST. . . . . . . . . . . . . .              Number  0005
BID_BAT. . . . . . . . . . . . . .              Number  0002
BID_BOT. . . . . . . . . . . . . .              Number  0008
BID_DPD. . . . . . . . . . . . . .              Number  0003
BID_HOM. . . . . . . . . . . . . .              Number  0001
BID_OSN. . . . . . . . . . . . . .              Number  0004
BID_PAS. . . . . . . . . . . . . .              Number  0006
BID_PBT. . . . . . . . . . . . . .              Number  0007
BID_UKN. . . . . . . . . . . . . .              Number  0000
BIT0 . . . . . . . . . . . . . . .              Number  0001
BIT1 . . . . . . . . . . . . . . .              Number  0002
BIT15. . . . . . . . . . . . . . .              Number  8000
BIT2 . . . . . . . . . . . . . . .              Number  0004
BIT3 . . . . . . . . . . . . . . .              Number  0008
BIT4 . . . . . . . . . . . . . . .              Number  0010
BIT5 . . . . . . . . . . . . . . .              Number  0020
BIT6 . . . . . . . . . . . . . . .              Number  0040
BIT7 . . . . . . . . . . . . . . .              Number  0080
BK_EC_OFF. . . . . . . . . . . . .              Number  0008
BK_LBN_OFF . . . . . . . . . . . .              Number  0003
BK_MEC_OFF . . . . . . . . . . . .              Number  0006
BOOT_OFF . . . . . . . . . . . . .              Number  0021
CBSY . . . . . . . . . . . . . . .              Number  0080
CBSY2. . . . . . . . . . . . . . .              Number  0001
CLRIL. . . . . . . . . . . . . . .              Number  0004
CLRSPL . . . . . . . . . . . . . .              Number  0008
CLTPN. . . . . . . . . . . . . . .              L NEAR  0000    CODE    External
CMDIP. . . . . . . . . . . . . . .              Number  0002
CPM8680. . . . . . . . . . . . . .              Number  0001
CR . . . . . . . . . . . . . . . .              Number  000D
CRCER. . . . . . . . . . . . . . .              Number  0040
CTRL_Q . . . . . . . . . . . . . .              Number  0011
CTRL_S . . . . . . . . . . . . . .              Number  0013
CTRL_Z . . . . . . . . . . . . . .              Number  001A
CYLDH. . . . . . . . . . . . . . .              Number  0065
CYLDL. . . . . . . . . . . . . . .              Number  0064
DAMNF. . . . . . . . . . . . . . .              Number  0001
DATA . . . . . . . . . . . . . . .              Number  0060
DATARQ . . . . . . . . . . . . . .              Number  0008
```

```
DISKDF . . . . . . . . . . . . . .      Number  0005
DO_XLT . . . . . . . . . . . . . .      L NEAR  0065      CODE
DPDE_FLAG. . . . . . . . . . . . .      Number  0000
DPDE_FTN . . . . . . . . . . . . .      Number  000C
DPDE_INIT. . . . . . . . . . . . .      Number  00F0
DPDE_LTN . . . . . . . . . . . . .      Number  000E
DPDE_LU. . . . . . . . . . . . . .      Number  0001
DPDE_OFF . . . . . . . . . . . . .      Number  0020
DPDE_OST . . . . . . . . . . . . .      Number  000B
DPDE_PN. . . . . . . . . . . . . .      Number  0002
DPDE_PON . . . . . . . . . . . . .      Number  000A
DPD_OFF. . . . . . . . . . . . . .      Number  0017
DPD_START_OFF. . . . . . . . . . .      Number  0020
DRDY . . . . . . . . . . . . . . .      Number  0040
DRDY2. . . . . . . . . . . . . . .      Number  0040
DRIVE0 . . . . . . . . . . . . . .      Number  0000
DRIVE1 . . . . . . . . . . . . . .      Number  0008
DRIVE2 . . . . . . . . . . . . . .      Number  0010
DRIVE3 . . . . . . . . . . . . . .      Number  0018
DRIVEDF. . . . . . . . . . . . . .      Number  0003
DRIVEM . . . . . . . . . . . . . .      Number  0018
DRVSEL . . . . . . . . . . . . . .      Number  0001
DSK_INIT . . . . . . . . . . . . .      L NEAR  0000      CODE      External
DSTAT0 . . . . . . . . . . . . . .      Number  0068
DSTAT1 . . . . . . . . . . . . . .      Number  0069
DWRTF. . . . . . . . . . . . . . .      Number  0020
DWRTF2 . . . . . . . . . . . . . .      Number  0020
ERRF . . . . . . . . . . . . . . .      Number  0001
ERROR. . . . . . . . . . . . . . .      Number  0061
FALSE. . . . . . . . . . . . . . .      Number  0000
FMTTRK . . . . . . . . . . . . . .      Number  0050
GETFTN . . . . . . . . . . . . . .      L NEAR  0000      CODE      External
GETFTNO. . . . . . . . . . . . . .      L NEAR  0000      CODE      External
GETPEA . . . . . . . . . . . . . .      L NEAR  0000      CODE      External
GET_REG. . . . . . . . . . . . . .      L NEAR  0079      CODE
HDIR . . . . . . . . . . . . . . .      Number  0008
HDISK. . . . . . . . . . . . . . .      Number  0004
HEAD0. . . . . . . . . . . . . . .      Number  0000
HEAD02 . . . . . . . . . . . . . .      Number  0000
HEAD1. . . . . . . . . . . . . . .      Number  0001
HEAD12 . . . . . . . . . . . . . .      Number  0002
HEAD2. . . . . . . . . . . . . . .      Number  0002
HEAD22 . . . . . . . . . . . . . .      Number  0004
HEAD3. . . . . . . . . . . . . . .      Number  0003
HEAD32 . . . . . . . . . . . . . .      Number  0006
HEADM. . . . . . . . . . . . . . .      Number  0007
HEADM2 . . . . . . . . . . . . . .      Number  000E
HFORMAT. . . . . . . . . . . . . .      L NEAR  0000      CODE      External
HINIT. . . . . . . . . . . . . . .      L NEAR  0000      CODE      External
HIOCTL . . . . . . . . . . . . . .      L NEAR  0000      CODE      Global
HIVEC_A. . . . . . . . . . . . . .      Number  0114
HIVEC_B. . . . . . . . . . . . . .      Number  0294
HMCHK. . . . . . . . . . . . . . .      L NEAR  0000      CODE      External
HREAD. . . . . . . . . . . . . . .      L NEAR  0000      CODE      External
HVDISK . . . . . . . . . . . . . .      L NEAR  0000      CODE      External
```

```
HWRITE . . . . . . . . . . . . . .      L NEAR  0000      CODE      External
HWRITEV. . . . . . . . . . . . . .      L NEAR  0000      CODE      External
IDBIT. . . . . . . . . . . . . . .      Number  00E0
IDBITV . . . . . . . . . . . . . .      Number  00A0
IDNF . . . . . . . . . . . . . . .      Number  0010
INTHDL . . . . . . . . . . . . . .      L NEAR  0000      CODE      External
LF . . . . . . . . . . . . . . . .      Number  000A
LINDEX . . . . . . . . . . . . . .      Number  0002
LSTEPP . . . . . . . . . . . . . .      Number  0004
MSCTF. . . . . . . . . . . . . . .      Number  0004
MSDOS. . . . . . . . . . . . . . .      Number  0002
NCYLD_OFF. . . . . . . . . . . . .      Number  0040
NT_AST_OFF . . . . . . . . . . . .      Number  002D
NUMHEAD. . . . . . . . . . . . . .      Number  0004
OSN_OFF. . . . . . . . . . . . . .      Number  001C
PARTITION. . . . . . . . . . . . .      V WORD  0000      CODE      External
PART_FTN . . . . . . . . . . . . .      Number  0000
PART_LTN . . . . . . . . . . . . .      Number  0002
PART_PAS . . . . . . . . . . . . .      Number  0004
PASE_OFF . . . . . . . . . . . . .      Number  0005
PAS_ASN. . . . . . . . . . . . . .      Number  0004
PAS_ATN. . . . . . . . . . . . . .      Number  0003
PAS_BSN. . . . . . . . . . . . . .      Number  0002
PAS_BTN. . . . . . . . . . . . . .      Number  0000
PRE_READ . . . . . . . . . . . . .      Alias   FALSE
RDBASE . . . . . . . . . . . . . .      Number  0060
RDCMD. . . . . . . . . . . . . . .      Number  0067
RDCMD2 . . . . . . . . . . . . . .      Number  0068
RDINTF . . . . . . . . . . . . . .      Number  0008
RDREAD . . . . . . . . . . . . . .      Number  0020
RDSTAT . . . . . . . . . . . . . .      Number  0067
RDWRITE. . . . . . . . . . . . . .      Number  0030
RDXLT. . . . . . . . . . . . . . .      V BYTE  0000      CODE      External
RESTOR . . . . . . . . . . . . . .      Number  0010
SBUFR. . . . . . . . . . . . . . .      Number  0001
SCANID . . . . . . . . . . . . . .      Number  0040
SCTEXT . . . . . . . . . . . . . .      Number  0080
SDH. . . . . . . . . . . . . . . .      Number  0066
SECPTRK. . . . . . . . . . . . . .      Number  0010
SECSIZE. . . . . . . . . . . . . .      Number  0200
SECTC. . . . . . . . . . . . . . .      Number  0062
SECTN. . . . . . . . . . . . . . .      Number  0063
SECTSZ . . . . . . . . . . . . . .      Number  0060
SEEK . . . . . . . . . . . . . . .      Number  0070
SEEKC. . . . . . . . . . . . . . .      Number  0010
SEEKC2 . . . . . . . . . . . . . .      Number  0010
SERR . . . . . . . . . . . . . . .      Number  000F
SINIT. . . . . . . . . . . . . . .      Number  0002
SSCTF. . . . . . . . . . . . . . .      Number  0000
SSZ128 . . . . . . . . . . . . . .      Number  0060
SSZ1K. . . . . . . . . . . . . . .      Number  0040
SSZ256 . . . . . . . . . . . . . .      Number  0000
SSZ512 . . . . . . . . . . . . . .      Number  0020
STEPR0 . . . . . . . . . . . . . .      Number  0000
STEPR1 . . . . . . . . . . . . . .      Number  0001
```

```
STEPR2 . . . . . . . . . . . . . .      Number  0002
STEPR3 . . . . . . . . . . . . . .      Number  0003
STEPR4 . . . . . . . . . . . . . .      Number  0004
STEPR5 . . . . . . . . . . . . . .      Number  0005
STEPR6 . . . . . . . . . . . . . .      Number  0006
STEPR7 . . . . . . . . . . . . . .      Number  0007
STEPR8 . . . . . . . . . . . . . .      Number  0008
STEPR9 . . . . . . . . . . . . . .      Number  0009
STEPRA . . . . . . . . . . . . . .      Number  000A
STEPRB . . . . . . . . . . . . . .      Number  000B
STEPRC . . . . . . . . . . . . . .      Number  000C
STEPRD . . . . . . . . . . . . . .      Number  000D
STEPRE . . . . . . . . . . . . . .      Number  000E
STEPRF . . . . . . . . . . . . . .      Number  000F
STEPR_OFF. . . . . . . . . . . . .      Number  004C
ST_AST_OFF . . . . . . . . . . . .      Number  002B
TEMP_PTR . . . . . . . . . . . . .      L DWORD 0095      CODE
TRKO2. . . . . . . . . . . . . . .      Number  0080
TRKOE. . . . . . . . . . . . . . .      Number  0002
TRUE . . . . . . . . . . . . . . .      Number  - 0001
WDISK1 . . . . . . . . . . . . . .      L NEAR  0034      CODE
WDISK10. . . . . . . . . . . . . .      L NEAR  0060      CODE
WDISK2 . . . . . . . . . . . . . .      L NEAR  004C      CODE
WDISK3 . . . . . . . . . . . . . .      L NEAR  0048      CODE
WGATE. . . . . . . . . . . . . . .      Number  0010
WINI_OFFSET. . . . . . . . . . . .      Number  0094
WINI_SEG . . . . . . . . . . . . .      Number  0096
WPC_OFF. . . . . . . . . . . . . .      Number  004A
WPRCMP . . . . . . . . . . . . . .      Number  0061
WPRSNT . . . . . . . . . . . . . .      Number  0001
WTVERR . . . . . . . . . . . . . .      Number  000D
W_DISK . . . . . . . . . . . . . .      L NEAR  001C      CODE
W_DISKF. . . . . . . . . . . . . .      L WORD  006D      CODE
XBB_BO . . . . . . . . . . . . . .      Number  0008
XBB_BS . . . . . . . . . . . . . .      Number  000A
XBB_CN . . . . . . . . . . . . . .      Number  0004
XBB_DISKF. . . . . . . . . . . . .      Number  0000
XBB_DSN. . . . . . . . . . . . . .      Number  0003
XBB_ERRC . . . . . . . . . . . . .      Number  000C
XBB_LDN. . . . . . . . . . . . . .      Number  0001
XBB_SC . . . . . . . . . . . . . .      Number  0006
XBB_SN . . . . . . . . . . . . . .      Number  0002
XBDSK1 . . . . . . . . . . . . . .      L NEAR  0010      CODE
XB_FLOPPY. . . . . . . . . . . . .      L NEAR  0018      CODE
XCPRSNT. . . . . . . . . . . . . .      Number  0002
XLT_F. . . . . . . . . . . . . . .      V BYTE  0000      CODE     External

Warning Severe
Errors  Errors
0       0
```

```
ABRTC. . . . . . . . . . . . . . .     26#
AST_OFF. . . . . . . . . . . . . .     26#
ATRETRY. . . . . . . . . . . . . .     26#

BADBD. . . . . . . . . . . . . . .     26#
BAT_OFF. . . . . . . . . . . . . .     26#
BID_AST. . . . . . . . . . . . . .     26#
BID_BAT. . . . . . . . . . . . . .     26#
BID_BOT. . . . . . . . . . . . . .     26#
BID_DPD. . . . . . . . . . . . . .     26#
BID_HOM. . . . . . . . . . . . . .     26#
BID_OSN. . . . . . . . . . . . . .     26#
BID_PAS. . . . . . . . . . . . . .     26#
BID_PBT. . . . . . . . . . . . . .     26#
BID_UKN. . . . . . . . . . . . . .     26#
BIT0 . . . . . . . . . . . . . . .     26#
BIT1 . . . . . . . . . . . . . . .     26#
BIT15. . . . . . . . . . . . . . .     26#
BIT2 . . . . . . . . . . . . . . .     26#
BIT3 . . . . . . . . . . . . . . .     26#
BIT4 . . . . . . . . . . . . . . .     26#
BIT5 . . . . . . . . . . . . . . .     26#
BIT6 . . . . . . . . . . . . . . .     26#
BIT7 . . . . . . . . . . . . . . .     26#
BK_EC_OFF. . . . . . . . . . . . .     26#
BK_LBN_OFF . . . . . . . . . . . .     26#
BK_MEC_OFF . . . . . . . . . . . .     26#
BOOT_OFF . . . . . . . . . . . . .     26#

CBSY . . . . . . . . . . . . . . .     26#
CBSY2. . . . . . . . . . . . . . .     26#
CGROUP . . . . . . . . . . . . . .     13      15      15      15      15
CLRIL. . . . . . . . . . . . . . .     26#
CLRSPL . . . . . . . . . . . . . .     26#
CLTPN. . . . . . . . . . . . . . .     21#    104
CMDIP. . . . . . . . . . . . . . .     26#
CODE . . . . . . . . . . . . . . .     13      14#     14     176
CPM8680. . . . . . . . . . . . . .     26#
CR . . . . . . . . . . . . . . . .     26#
CRCER. . . . . . . . . . . . . . .     26#
CTRL_Q . . . . . . . . . . . . . .     26#
CTRL_S . . . . . . . . . . . . . .     26#
CTRL_Z . . . . . . . . . . . . . .     26#
CYLDH. . . . . . . . . . . . . . .     26#
CYLDL. . . . . . . . . . . . . . .     26#

DAMNF. . . . . . . . . . . . . . .     26#
DATA . . . . . . . . . . . . . . .     26#
DATARQ . . . . . . . . . . . . . .     26#
DISKDF . . . . . . . . . . . . . .     26#     58     122
DO_XLT . . . . . . . . . . . . . .    101     131#
DPDE_FLAG. . . . . . . . . . . . .     26#
DPDE_FTN . . . . . . . . . . . . .     26#
DPDE_INIT. . . . . . . . . . . . .     26#
DPDE_LTN . . . . . . . . . . . . .     26#
DPDE_LU. . . . . . . . . . . . . .     26#
DPDE_OFF . . . . . . . . . . . . .     26#
```

```
 Hard Disk I/O Control Functions

 Symbol Cross Reference                          (# is definition)       Cref-2

DPDE_OST . . . . . . . . . . . . .    26#
DPDE_PN. . . . . . . . . . . . . .    26#
DPDE_PON . . . . . . . . . . . . .    26#
DPD_OFF. . . . . . . . . . . . . .    26#
DPD_START_OFF. . . . . . . . . . .    26#
DRDY . . . . . . . . . . . . . . .    26#
DRDY2. . . . . . . . . . . . . . .    26#
DRIVE0 . . . . . . . . . . . . . .    26#
DRIVE1 . . . . . . . . . . . . . .    26#
DRIVE2 . . . . . . . . . . . . . .    26#
DRIVE3 . . . . . . . . . . . . . .    26#
DRIVEDF. . . . . . . . . . . . . .    26#
DRIVEM . . . . . . . . . . . . . .    26#
DRVSEL . . . . . . . . . . . . . .    26#
DSK_INIT . . . . . . . . . . . . .    19#
DSTAT0 . . . . . . . . . . . . . .    26#
DSTAT1 . . . . . . . . . . . . . .    26#
DWRTF. . . . . . . . . . . . . . .    26#
DWRTF2 . . . . . . . . . . . . . .    26#

ERRF . . . . . . . . . . . . . . .    26#
ERROR. . . . . . . . . . . . . . .    26#

FALSE. . . . . . . . . . . . . . .    26#     26
FMTTRK . . . . . . . . . . . . . .    26#

GETFTN . . . . . . . . . . . . . .    21#
GETFTNO. . . . . . . . . . . . . .    21#     95
GETPEA . . . . . . . . . . . . . .    21#
GET_REG. . . . . . . . . . . . . .    39     163#

HDIR . . . . . . . . . . . . . . .    26#
HDISK. . . . . . . . . . . . . . .    26#     51
HEAD0. . . . . . . . . . . . . . .    26#
HEAD02 . . . . . . . . . . . . . .    26#
HEAD1. . . . . . . . . . . . . . .    26#
HEAD12 . . . . . . . . . . . . . .    26#
HEAD2. . . . . . . . . . . . . . .    26#
HEAD22 . . . . . . . . . . . . . .    26#
HEAD3. . . . . . . . . . . . . . .    26#
HEAD32 . . . . . . . . . . . . . .    26#
HEADM. . . . . . . . . . . . . . .    26#
HEADM2 . . . . . . . . . . . . . .    26#
HFORMAT. . . . . . . . . . . . . .    18#    141
HINIT. . . . . . . . . . . . . . .    18#
HIOCTL . . . . . . . . . . . . . .    24      37#
HIVEC_A. . . . . . . . . . . . . .    26#
HIVEC_B. . . . . . . . . . . . . .    26#
HMCHK. . . . . . . . . . . . . . .    18#    142
HREAD. . . . . . . . . . . . . . .    17#    138
HVDISK . . . . . . . . . . . . . .    19#    143
HWRITE . . . . . . . . . . . . . .    17#    139
HWRITEV. . . . . . . . . . . . . .    17#    140

IDBIT. . . . . . . . . . . . . . .    26#
IDBITV . . . . . . . . . . . . . .    26#
IDNF . . . . . . . . . . . . . . .    26#
```

```
 Hard Disk I/O Control Functions

 Symbol Cross Reference                          (# is definition)       Cref-3

INTHDL . . . . . . . . . . . . . .    19#

LF . . . . . . . . . . . . . . . .    26#
LINDEX . . . . . . . . . . . . . .    26#
LSTEPP . . . . . . . . . . . . . .    26#

MSCTF. . . . . . . . . . . . . . .    26#
MSDOS. . . . . . . . . . . . . . .    26#

NCYLD_OFF. . . . . . . . . . . . .    26#
NT_AST_OFF . . . . . . . . . . . .    26#
NUMHEAD. . . . . . . . . . . . . .    26#

OSN_OFF. . . . . . . . . . . . . .    26#

PARTITION. . . . . . . . . . . . .    22#
PART_FTN . . . . . . . . . . . . .    26#
PART_LTN . . . . . . . . . . . . .    26#
PART_PAS . . . . . . . . . . . . .    26#
PASE_OFF . . . . . . . . . . . . .    26#
PAS_ASN. . . . . . . . . . . . . .    26#
PAS_ATN. . . . . . . . . . . . . .    26#
PAS_BSN. . . . . . . . . . . . . .    26#
PAS_BTN. . . . . . . . . . . . . .    26#
PRE_READ . . . . . . . . . . . . .    26#

RDBASE . . . . . . . . . . . . . .    26#     26      26      26      26      26      26      26      26      26      26      26      26      26
RDCMD. . . . . . . . . . . . . . .    26#
RDCMD2 . . . . . . . . . . . . . .    26#
RDINTF . . . . . . . . . . . . . .    26#
RDREAD . . . . . . . . . . . . . .    26#
RDSTAT . . . . . . . . . . . . . .    26#
RDWRITE. . . . . . . . . . . . . .    26#
RDXLT. . . . . . . . . . . . . . .    20#    134
RESTOR . . . . . . . . . . . . . .    26#

SBUFR. . . . . . . . . . . . . . .    26#
SCANID . . . . . . . . . . . . . .    26#
SCTEXT . . . . . . . . . . . . . .    26#
SDH. . . . . . . . . . . . . . . .    26#
SECPTRK. . . . . . . . . . . . . .    26#
SECSIZE. . . . . . . . . . . . . .    26#
SECTC. . . . . . . . . . . . . . .    26#
SECTN. . . . . . . . . . . . . . .    26#
SECTSZ . . . . . . . . . . . . . .    26#
SEEK . . . . . . . . . . . . . . .    26#
SEEKC. . . . . . . . . . . . . . .    26#
SEEKC2 . . . . . . . . . . . . . .    26#
SERR . . . . . . . . . . . . . . .    26#
SINIT. . . . . . . . . . . . . . .    26#
SSCTF. . . . . . . . . . . . . . .    26#
SSZ128 . . . . . . . . . . . . . .    26#
SSZ1K. . . . . . . . . . . . . . .    26#
SSZ256 . . . . . . . . . . . . . .    26#
SSZ512 . . . . . . . . . . . . . .    26#
STEPR0 . . . . . . . . . . . . . .    26#
STEPR1 . . . . . . . . . . . . . .    26#
```

```
 Hard Disk I/O Control Functions

 Symbol Cross Reference                    (# is definition)       Cref-4

STEPR2 . . . . . . . . . . . . . .        26#
STEPR3 . . . . . . . . . . . . . .        26#
STEPR4 . . . . . . . . . . . . . .        26#
STEPR5 . . . . . . . . . . . . . .        26#
STEPR6 . . . . . . . . . . . . . .        26#
STEPR7 . . . . . . . . . . . . . .        26#
STEPR8 . . . . . . . . . . . . . .        26#
STEPR9 . . . . . . . . . . . . . .        26#
STEPRA . . . . . . . . . . . . . .        26#
STEPRB . . . . . . . . . . . . . .        26#
STEPRC . . . . . . . . . . . . . .        26#
STEPRD . . . . . . . . . . . . . .        26#
STEPRE . . . . . . . . . . . . . .        26#
STEPRF . . . . . . . . . . . . . .        26#
STEPR_OFF. . . . . . . . . . . . .        26#
ST_AST_OFF . . . . . . . . . . . .        26#

TEMP_PTR . . . . . . . . . . . . .       116     164     165     174#
TRKO2. . . . . . . . . . . . . . .        26#
TRKOE. . . . . . . . . . . . . . .        26#
TRUE . . . . . . . . . . . . . . .        26#     26

WDISK1 . . . . . . . . . . . . . .        91      93#
WDISK10. . . . . . . . . . . . . .        97     120#
WDISK2 . . . . . . . . . . . . . .        83     110#
WDISK3 . . . . . . . . . . . . . .       100     102#
WGATE. . . . . . . . . . . . . . .        26#
WINI_OFFSET. . . . . . . . . . . .        26#
WINI_SEG . . . . . . . . . . . . .        26#
WPC_OFF. . . . . . . . . . . . . .        26#
WPRCMP . . . . . . . . . . . . . .        26#
WPRSNT . . . . . . . . . . . . . .        26#
WTVERR . . . . . . . . . . . . . .        26#
W_DISK . . . . . . . . . . . . . .        47      53      80#
W_DISKF. . . . . . . . . . . . . .       115     138#

XBB_BO . . . . . . . . . . . . . .        26#    171
XBB_BS . . . . . . . . . . . . . .        26#
XBB_CN . . . . . . . . . . . . . .        26#    169
XBB_DISKF. . . . . . . . . . . . .        26#    167
XBB_DSN. . . . . . . . . . . . . .        26#
XBB_ERRC . . . . . . . . . . . . .        26#    117
XBB_LDN. . . . . . . . . . . . . .        26#
XBB_SC . . . . . . . . . . . . . .        26#    170
XBB_SN . . . . . . . . . . . . . .        26#    168
XBDSK1 . . . . . . . . . . . . . .        41      51#
XB_FLOPPY. . . . . . . . . . . . .        46      52      57#
XCPRSNT. . . . . . . . . . . . . .        26#
XLT_F. . . . . . . . . . . . . . .        20#     81      92      99
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE     1-1
Hard Disk Driver


1                                                  PAGE    60,132
2                                                  TITLE   Hard Disk Driver
3                                                  NAME    HDD
4
5                                          ;       08/30/83
6
7                                          CGROUP  GROUP   CODE
8           0000                           CODE    SEGMENT BYTE PUBLIC 'CODE'
9                                          ASSUME  CS:CGROUP, DS:CGROUP, ES:CGROUP, SS:CGROUP
10
11                                                 EXTRN   RDXLT:BYTE
12
13                                                 PUBLIC  HREAD   ; - read sector(s)
14                                                 PUBLIC  HWRITE  ; - write sector(s)
15                                                 PUBLIC  HWRITEV ; - write sector(s) with verify
16                                                 PUBLIC  HFORMAT ; - format track(s)
17                                                 PUBLIC  HINIT   ; - init hard disk drive
18                                                 PUBLIC  HMCHK   ; - media check
19                                                 PUBLIC  HVDISK  ; - verify disk
20                                                 PUBLIC  INTHDL  ; - hard disk interrupt handler
21                                                 PUBLIC  XLT_F, CLTPN, MAXTRK,STEPRATE, PRECOMP
22
23                                                 .LIST
24
25          0000  0098                     MAXTRK          DW      152             ; max track #
26          0002  06                       STEPRATE        DB      STEPR6          ; step rate value
27          0003  0020                     PRECOMP         DW      128/4           ; write pre-comp value
28
29          0005  00                       RETNS   DB      0               ; # of sectors do retry
30          0006  00                       RETRIES DB      0               ; retry count
31
32          0007  0000                     ERRCODE DW      0               ; error code save area
33          0009  00                       WRTWVF  DB      0               ; write with verify flag
34
35          000A  00                       XLT_F   DB      0               ; translate table flag
36
37                                             ENDIF
```

```
                                                PAGE
                                        ;       Hard Disk routines are:
                                        ;
                                        ;       1. HREAD - read sector(s)
                                        ;       2. HWRITE - write sector(s)
                                        ;       3. HWRITEV - write sector(s) with verify
                                        ;       4. HFORMAT - format track(s)
                                        ;       5. HINIT - init hard disk drive
                                        ;       6. HMCHK - media check
                                        ;       7. HVDISK - verify disk
                                        ;       8. INTHDL - hard disk interrupt handler
                                        ;
                                        ;       Other support routines are:
                                        ;
                                        ;       1. RETRY - error recovery after a read
                                        ;       2. HOMET - restore head
                                        ;       3. MOVEMX - move head to max. track
                                        ;       4. MOVEH - move head
                                        ;       5. LDTSKF - load hard disk task file
                                        ;       6. OUTCMD - output command and wait for command completion
                                        ;       7. ADVTNS - advance to next logical sector *** this is OS dependent
                                        ;       8. ADVTNT - advance to next logical track
                                        ;       9. DELAY - delay 0.5 ms
                                        ;
                                        ;
                                        ;       Default values for all the variables are set for the RD51/RD50
                                        ;       Winchester drive. They are:
                                        ;
                                        ;       SECPTRK = 16            ; sector per track
                                        ;       SECSIZE = 512           ; sector size
                                        ;       NUMHEAD = 4             ; number of surfaces per cylinder
                                        ;       (RDXLT) =               ; CP/M skew table with skew factor 7
                                        ;                                  this is the only item here that OS
                                        ;                                  dependent
                                        ;
                                        ;       Values has to be loaded for different drive:
                                        ;
                                        ;               RD51            RD50
                                        ;       MAXTRK = 305            152     ; max. cylinder number
                                        ;       STEPRATE = 0            6       ; step rate value
                                        ;       PRECOMP = 200/4 = 50    128/4   ; write precomp value
```

```
                                                PAGE
                                        ;************************************************************
                                        ;
                                        ;       HREAD - HARD DISK READ SECTOR
                                        ;
                                        ;       ENTRY:
                                        ;               BH = surface/drive #
                                        ;                       BIT 0-3 surface #
                                        ;                       BIT 4-7 drive #
                                        ;               BL = sector #
                                        ;               CX = sector #
                                        ;               DX = cylinder #
                                        ;               ES:DI = buffer start address
                                        ;       EXIT:
                                        ;               Z flag is set if error
                                        ;               AX = ERROR STATUS
                                        ;                       AL = 0, NO ERROR
                                        ;                          = 1, ERROR
                                        ;                          = 3, DRIVE DEFFECTIVE
                                        ;                          = 5, DISK DEFFECTIVE
                                        ;                          = 0DH, WRITE VERIFY ERROR
                                        ;                          = 0FH, SOFT ERROR
                                        ;                       AH, BIT 0 = DAM NOT FOUND
                                        ;                           BIT 1 = TRACK 0 ERROR
                                        ;                           BIT 2 = ABORTED COMMAND
                                        ;                           BIT 4 = ID NOT FOUND
                                        ;                           BIT 6 = CRC ERROR DATA FIELD
                                        ;                           BIT 7 = BAD BLOCK DETECT
                                        ;               (RETNS) = NUMBER OF SECTORS PERFORMED THE RETRY
                                        ;
                                        ;************************************************************

        000B                            HREAD:
        000B  FC                                CLD                             ; set forward flag
        000C  C6 06 0005 R 00                   MOV     BYTE PTR RETNS,0        ; no any sector do retry yet

                                            ENDIF


                                                read sector(s) loop

        0011                            HREAD5:
        0011  E8 0105 R                         CALL    LDTSKF                  ; load task file
        0014  75 37                             JNZ     HREAD2                  ; error if pass limit
        0016  B0 20                             MOV     AL,RDREAD+ATRETRY       ; output read command
        0018  E8 014F R                         CALL    OUTCMD
        001B                            HREAD10:
        001B  0A C0                             OR      AL,AL                   ; error?
        001D  74 07                             JZ      HREAD3                  ;  skip if no error
        001F  E8 01F0 R                         CALL    RETRY                   ; error recovery
        0022  0A C0                             OR      AL,AL                   ; still have error?
        0024  75 27                             JNZ     HREAD2                  ;  abort at this point
        0026                            HREAD3:
        0026  52                                PUSH    DX                      ; save cylinder #
        0027  51                                PUSH    CX                      ; save sector count
```

```
0028  BA 0080                          MOV     DX,OFFSET DATA          ; get data port addr
002B  B9 0100                          MOV     CX,SECSIZE/2            ; get sector size in words
002E                           HREAD1:                                 ; es:di has buffer addr
002E  EC                               IN      AL,DX                   ; (8) get data byte
002F  8A E0                            MOV     AH,AL                   ; (2) save to high byte
0031  EC                               IN      AL,DX                   ; (8) get next byte
0032  86 C4                            XCHG    AL,AH                   ; (4) make into correct order
0034  AB                               STOSW                           ; (15) save into buffer
0035  E2 F7                            LOOP    HREAD1                  ; (17) loop for whole sector

0037  59                               POP     CX                      ; restore sector count
0038  5A                               POP     DX                      ; restore cylinder #

0039  B0 01                            MOV     AL,SBUFR                ; strobe buffer ready
003B  E6 68                            OUT     RDCMD2,AL

003D  E8 018F R                        CALL    ADVTNS                  ; advance to next sector
0040  E2 CF                            LOOP    HREAD5                  ; loop for multi_sector

                                   ENDIF

0042  33 C0                            XOR     AX,AX                   ; assume no error
0044  F6 06 0005 R FF                  TEST    BYTE PTR RETNS,0FFH     ; retry before?
0049  74 02                            JZ      HREAD2
004B  B0 0F                            MOV     AL,SERR                 ; set soft error
004D                           HREAD2:
                                   IF  NOT     PRE_READ
004D  EB 62 90                         JMP     LED_OFF                 ; this is the way to handle LED
                                   ENDIF                               ;   if no pre_read

0050  C3                               RET
```

```
                                       PAGE
                               ;****************************************************************
                               ;
                               ;       HWRITE - hard disk write sector (without verify)
                               ;       HWRITEV - write sector with verify
                               ;
                               ;       ENTRY:
                               ;               BH = surface/drive number
                               ;                       BIT 0-3 surface #
                               ;                       BIT 4-7 drive #
                               ;               BL = sector #
                               ;               CX = sector count
                               ;               DX = cylinder #
                               ;               ES:DI = buffer start address
                               ;       EXIT:
                               ;               Z flag is set if error
                               ;               AX = ERROR STATUS
                               ;                       AL = 0, NO ERROR
                               ;                          = 1, ERROR
                               ;                          = 3, DRIVE DEFFECTIVE
                               ;                          = 5, DISK DEFFECTIVE
                               ;                          = 0DH, WRITE VERIFY ERROR
                               ;                          = 0FH, SOFT ERROR
                               ;                       AH, BIT 0 = DAM NOT FOUND
                               ;                           BIT 1 = TRACK 0 ERROR
                               ;                           BIT 2 = ABORTED COMMAND
                               ;                           BIT 4 = ID NOT FOUND
                               ;                           BIT 6 = CRC ERROR DATA FIELD
                               ;                           BIT 7 = BAD BLOCK DETECT
                               ;
                               ;****************************************************************

0051                           HWRITEV:
0051  C6 06 0009 R 01                  MOV     BYTE PTR WRTWVF,1       ; SET WRITE WITH VERIFY FLAG
0056  EB 05                            JMP     SHORT HWRITE1

0058                           HWRITE:
0058  C6 06 0009 R 00                  MOV     BYTE PTR WRTWVF,0       ; CLEAR WRITE WITH VERIFY FLAG
005D                           HWRITE1:
005D  FC                               CLD                             ; SET FORWARD FLAG

                                   ENDIF

005E  8B F7                            MOV     SI,DI                   ; SI = BUFFER ADDR

                               ;       write sector loop

0060                           HWRITE5:
0060  E8 0105 R                        CALL    LDTSKF                  ; LOAD TASK FILE
0063  75 4C                            JNZ     HWRITE2                 ; error if pass limit
0065  B0 30                            MOV     AL,RDWRITE+ATRETRY      ; OUTPUT WRITE CMD
0067  E8 014F R                        CALL    OUTCMD
006A  0A C0                            OR      AL,AL                   ; ERROR?
006C  75 43                            JNZ     HWRITE2                 ; ABORT IF ERROR
```

```
008E  52                               PUSH    DX                    ; SAVE cylinder #
008F  51                               PUSH    CX                    ; SAVE SECTOR COUNT
0070  BA 0060                          MOV     DX,OFFSET DATA        ; GET DATA PORT ADDR
0073  B9 0100                          MOV     CX,SECSIZE/2          ; GET SECTOR SIZE in words
0076  1E                               PUSH    DS                    ; SAVE DS
0077  06                               PUSH    ES                    ; DS=ES
0078  1F                               POP     DS
0079                         HWRITE21:                               ; DS:SI HAS BUFFER ADDR
0079  AD                               LODSW                         ; (16) GET DATA FROM BUFFER
007A  EE                               OUT     DX,AL                 ; (8) OUTPUT DATA TO RD
007B  8A C4                            MOV     AL,AH                 ; (2) GET HIGH BYTE
007D  EE                               OUT     DX,AL                 ; (8)
007E  E2 F9                            LOOP    HWRITE21              ; (17) LOOP FOR WHOLE SECTOR

0080  1F                               POP     DS                    ; GET DS BACK
0081  59                               POP     CX                    ; RESTORE SECTOR COUNT
0082  5A                               POP     DX                    ; RESTORE cylinder #

0083  2E: C6 06 01E7 R 00              MOV     BYTE PTR CS:IRQDRQ,0  ; CLEAR INT FLAG
0089  B0 01                            MOV     AL,SBUFR              ; STROBE BUFFER READY
008B  E6 68                            OUT     RDCMD2,AL
008D  E8 0157 R                        CALL    OUTCMD1               ; WAIT UNTIL RD GETS IT
0090  0A C0                            OR      AL,AL                 ; ERROR?
0092  75 1D                            JNZ     HWRITE2               ; ABORT IF ERROR

                                       check for write with verify

0094  F6 06 0009 R FF                  TEST    BYTE PTR WRTWVF,0FFH
0099  74 0F                            JZ      HWRITE11              ; NO VERIFY

009B  B0 20                            MOV     AL,RDREAD+ATRETRY     ; OUTPUT READ COMMAND
009D  E8 014F R                        CALL    OUTCMD                ; TASK FILE SHOULD CONTAIN THE SAME
                                                                     ; cylinder/surface/SECTOR INFORMATION
00A0  0A C0                            OR      AL,AL                 ; ERROR?
00A2  B0 0D                            MOV     AL,WTVERR             ; ASSUME WRITE VERIFY ERROR
00A4  75 0B                            JNZ     HWRITE2               ;  SKIP IF NO ERROR

00A6  B0 01                            MOV     AL,SBUFR              ; STROBE BUFFER READY
00A8  E6 68                            OUT     RDCMD2,AL
00AA                         HWRITE11:
00AA  E8 018F R                        CALL    ADVTNS                ; ADVANCE TO NEXT SECTOR
00AD  E2 B1                            LOOP    HWRITE5               ; LOOP FOR MULTI_SECTOR

00AF  33 C0                            XOR     AX,AX                 ; SET NO ERROR STATUS
00B1                         HWRITE2:
                             ;         CALL    LED_OFF               ; wini blink
                             ;         RET



00B1                         LED_OFF:
00B1  50                               PUSH    AX                    ; save
00B2  B0 38                            MOV     AL,SSZ512+DRIVE3      ; de-select drive
00B4  E6 66                            OUT     SDH,AL
00B6  58                               POP     AX
00B7  C3                               RET
```

```
                                       PAGE
                             ;*****************************************************************
                             ;
                             ;         HFORMAT - HARD DISK TRACK FORMAT ROUTINE
                             ;
                             ;         ENTRY:
                             ;                 BH = DRIVE/surface NUMBER
                             ;                 DX = cylinder #
                             ;                 CX = TRACK COUNT
                             ;                 ES:DI = FORMAT BUFFER
                             ;
                             ;         This format routine is used to format (CX) tracks.
                             ;         ES:DI holds additional parameter information.
                             ;         Each sector requires a 2 bytes sequence. The first
                             ;         byte designates whether a bad bloock mark is to be
                             ;         recorded in the sector's ID field. A '00'H is normal;
                             ;         a '80'H indicates a bad block mark for that sector.
                             ;         The second byte indicates the logical sector number
                             ;         to be recorded.
                             ;         ES:DI expecting total number of bytes:
                             ;
                             ;         2 * 16 * (number of track to be formatted)
                             ;
                             ;         EXIT:
                             ;                 Z flag is set if error
                             ;
                             ;*****************************************************************

00B8                         HFORMAT:
00B8  FC                               CLD                           ; SET FORWARD FLAG

                                  ENDIF

00B9  8B F7                            MOV     SI,DI                 ; GET BUFFER ADDR

                             ;         format track loop

00BB                         HFORMAT5:
00BB  B3 28                            MOV     BL,40                 ; # of bytes for gaps
00BD  E8 0105 R                        CALL    LDTSKF                ; LOAD TASK FILE
00C0  75 3A                            JNZ     HFORMAT2              ; error if pass limit
00C2  B0 10                            MOV     AL,SECPTRK            ; SET # OF SECTORS PER TRACK
00C4  E6 62                            OUT     SECTC,AL

00C6  B0 50                            MOV     AL,FMTTRK             ; OUTPUT WRITE CMD
00C8  E8 014F R                        CALL    OUTCMD
00CB  0A C0                            OR      AL,AL                 ; ERROR?
00CD  75 2D                            JNZ     HFORMAT2              ; ABORT IF ERROR

00CF  52                               PUSH    DX                    ; SAVE cylinder #
00D0  51                               PUSH    CX                    ; SAVE SECTOR COUNT
00D1  BA 0060                          MOV     DX,OFFSET DATA        ; GET PORT ADDR
00D4  B9 0010                          MOV     CX,SECPTRK            ; GET # OF SECTORS PER TRACK
00D7  1E                               PUSH    DS                    ; SAVE DS
00D8  06                               PUSH    ES                    ; DS=ES
```

```
      00D9   1F                                  POP     DS
      00DA                                HFORMAT1:                                  ; DS:SI HAS BUFFER ADDR
      00DA   AD                                  LODSW                               ; (16) GET DATA FROM BUFFER
      00DB   EE                                  OUT     DX,AL                       ; (8) OUTPUT BAD BLOCK MARK
      00DC   8A C4                               MOV     AL,AH                       ; (2) GET DATA FROM BUFFER
      00DE   EE                                  OUT     DX,AL                       ; (8) OUTPUT LOGICAL SECTOR #
      00DF   E2 F9                               LOOP    HFORMAT1                    ; (17) LOOP FOR ALL SECTOR

      00E1   1F                                  POP     DS                          ; GET DS BACK
      00E2   59                                  POP     CX                          ; RESTORE SECTOR COUNT
      00E3   5A                                  POP     DX                          ; RESTORE cylinder #

      00E4   2E: C6 06 01E7 R 00                 MOV     BYTE PTR CS:IRQDRQ,0        ; CLEAR INT FLAG
      00EA   B0 01                               MOV     AL,SBUFR                    ; STROBE BUFFER READY
      00EC   E6 68                               OUT     RDCMD2,AL
      00EE   E8 0157 R                           CALL    OUTCMD1                     ; WAIT UNTIL RD GETS IT
      00F1   0A C0                               OR      AL,AL                       ; ERROR?
      00F3   75 07                               JNZ     HFORMAT2                    ; ABORT IF ERROR

      00F5   E8 01BD R                           CALL    ADVTNT                      ; ADVANCE TO NEXT TRACK
      00F8   E2 C1                               LOOP    HFORMAT5                    ; LOOP FOR MULTI_TRACK FORMAT

      00FA   33 C0                               XOR     AX,AX                       ; SET NO ERROR STATUS
      00FC                                HFORMAT2:
      00FC   EB B3                               JMP     LED_OFF                     ; blink and RET
```

```
                                                 PAGE
                                          ;**************************************************************
                                          ;
                                          ;      HMCHK - MEDIA CHECK
                                          ;
                                          ;      EXIT:   AL = 0, NO ERROR
                                          ;              AH = 10, RD51 ID
                                          ;
                                          ;**************************************************************

      00FE                                HMCHK:
      00FE   B8 0A00                             MOV     AX,0A00H                    ; return with RD51 ID
      0101   C3                                  RET


                                          ;**************************************************************
                                          ;
                                          ;      HVDISK - VERIFY DISK
                                          ;
                                          ;      EXIT:   AL = DISK DEFFECTIVE ERROR VALUE
                                          ;
                                          ;**************************************************************

      0102                                HVDISK:
      0102   B0 05                               MOV     AL,DISKDF                   ; RETURN WITH DISK
      0104   C3                                  RET                                 ; DEFFECTIVE ERROR
```

```
                                                PAGE
                                        ;*************************************************************
                                        ;
                                        ;       LDTSKF - load 'TASK FILE', controller registers 2 - 6
                                        ;
                                        ;       ENTRY:
                                        ;               BH = drive/surface #
                                        ;               BL = sector #
                                        ;               DX = cylinder #
                                        ;               (MAXTRK) = max track #
                                        ;       EXIT:   AX = 0 and Z flag is set if no error
                                        ;               AX = 1001H (ID not found error) and Z flag is cleared
                                        ;                       if track # > (MAXTRK)
                                        ;
                                        ;*************************************************************

 0105                                   LDTSKF:
 0105  B8 1001                                  MOV     AX,1001H                ; load error code
 0108  3B 16 0000 R                             CMP     DX,MAXTRK               ; check pass limit
 010C  77 18                                    JA      LDTSKF1                 ; error if >
 010E  8A C2                                    MOV     AL,DL
 0110  E6 64                                    OUT     CYLDL,AL                ; low cylinder
 0112  8A C6                                    MOV     AL,DH
 0114  E6 65                                    OUT     CYLDH,AL                ; high cylinder

 0116  B0 20                                    MOV     AL,SSZ512               ; get sector size
 0118  0A C7                                    OR      AL,BH                   ; or with surface #, assume drive0
 011A  E6 66                                    OUT     SDH,AL

 011C  8A C3                                    MOV     AL,BL                   ; get sector #
 011E  E6 63                                    OUT     SECTN,AL

 0120  B0 01                                    MOV     AL,1                    ; do 1 sector
 0122  E6 62                                    OUT     SECTC,AL
 0124  33 C0                                    XOR     AX,AX
 0126                                   LDTSKF1:
 0126  23 C0                                    AND     AX,AX                   ; set Z flag
 0128  C3                                       RET
```

```
                                                PAGE
                                        ;*************************************************************
                                        ;
                                        ;       HINIT - hard disk init routine
                                        ;
                                        ;       ENTRY:  Interrupt vector must be set
                                        ;       EXIT:   AX = error status
                                        ;
                                        ;*************************************************************

 0129                                   HINIT:
 0129  FC                                       CLD                             ; set forward direction
 012A  FB                                       STI                             ;TRY IT WITH INTERRUPTS ON
                                            ENDIF

 012B  B0 02                                    MOV     AL,SINIT                ; set software init
 012D  E6 68                                    OUT     RDCMD2,AL
 012F  E8 01E8 R                                CALL    DELAY                   ; allow 0.5 ms

 0132  A1 0003 R                                MOV     AX,PRECOMP              ; init write pre-comp value
 0135  E6 61                                    OUT     WPRCMP,AL

 0137  0C 20                                    OR      AL,DRIVE0+SSZ512+HEAD0  ; set size/drive/surface register
 0139  E6 66                                    OUT     SDH,AL

 013B  BB 0001                                  MOV     BX,1                    ; sector 1
 013E  BA 0004                                  MOV     DX,4
 0141  E8 0243 R                                CALL    MOVE04                  ; move head to track 0

 0144  A0 0002 R                                MOV     AL,STEPRATE             ; set step-rate and
 0147  04 10                                    ADD     AL,RESTOR               ;  restore
 0149  E8 014F R                                CALL    OUTCMD                  ; output the restore cmd
 014C  E9 00B1 R                                JMP     LED_OFF                 ; turn off LED and RET
```

```
                                             PAGE
                                     ;***************************************************************
                                     ;
                                     ;       OUTCMD - OUTPUT COMMAND TO WINNIE CONTROLLER
                                     ;
                                     ;       ENTRY:  AL = COMMAND
                                     ;       EXIT:   AX = ERROR STATUS
                                     ;       USED:   AX
                                     ;
                                     ;***************************************************************

014F                                 OUTCMD:
C14F  2E: C6 06 01E7 R 00                    MOV     BYTE PTR CS:IRQDRQ,0    ; CLEAR INT FLAG
0155  E6 67                                  OUT     RDCMD,AL                ; OUTPUT COMMAND
0157                                 OUTCMD1:
0157  53                                     PUSH    BX                      ; SAVE REGISTERS
0158  51                                     PUSH    CX

0159  B3 20                                  MOV     BL,4*8                  ; ABOUT 8 SECONDS TIME OUT
015B                                 OUTCMD2:
015B  B9 807F                                MOV     CX,32895                ; 32895*38 CLOCKS*0.2US= 250MS
015E                                 OUTCMD11:
015E  2E: F6 06 01E7 R FF                    TEST    BYTE PTR CS:IRQDRQ,OFFH ; WAIT FOR INT, 17 CLOCKS
0164  75 18                                  JNZ     OUTCMD4                 ; JMP IF INT, 4 CLOCKS
0166  E2 F6                                  LOOP    OUTCMD11                ; 17 CLOCKS
0168  FE CB                                  DEC     BL
016A  75 EF                                  JNZ     OUTCMD2
016C                                 OUTCMD3:
016C  B0 03                                  MOV     AL,DRIVEDF              ; SET HARDWARE ERROR FLAG
016E                                 OUTCMD6:
016E  8A E0                                  MOV     AH,AL                   ; SAVE MAIN ERROR STATUS
0170  B0 01                                  MOV     AL,SBUFR                ; STROBE BUFFER READY
0172  E6 68                                  OUT     RDCMD2,AL
0174  E4 61                                  IN      AL,ERROR                ; READ RD ERROR STATUS
0176  E4 61                                  IN      AL,ERROR                ; (THIS IS NOT BUG!!, WAIT UNTIL
0178  E4 61                                  IN      AL,ERROR                ; ERROR REGISTER IS VALID)
017A  86 C4                                  XCHG    AL,AH                   ; FORM ERROR STATUS
017C  EB 0E                                  JMP     SHORT OUTCMD5
017E                                 OUTCMD4:
017E  E4 67                                  IN      AL,RDSTAT               ; GET STATUS
0180  24 20                                  AND     AL,DWRTF                ; CHECK WRITE FAULT
0182  75 E8                                  JNZ     OUTCMD3
0184  E4 67                                  IN      AL,RDSTAT               ; GET STATUS AGAIN
0186  24 01                                  AND     AL,ERRF                 ; CHECK ERROR
0188  75 E4                                  JNZ     OUTCMD6                 ; ERROR
018A  33 C0                                  XOR     AX,AX                   ; NO ERROR
018C                                 OUTCMD5:
018C  59                                     POP     CX                      ; RESTORE REGISTERS
018D  5B                                     POP     BX
018E  C3                                     RET
```

```
                                             PAGE
                                     ;***************************************************************
                                     ;
                                     ;       ADVTNS - advance to next sector
                                     ;
                                     ;       ENTRY:  BH = surface #
                                     ;               BL = sector #
                                     ;               DX = cylinder #
                                     ;               (RDXLT) = translate table
                                     ;               (XLT_F) = 0 if NOT USE translate table
                                     ;                       <> 0 if USE
                                     ;       USED:   AX
                                     ;
                                     ;       *** NOTE:       1. This routine is OS dependent
                                     ;                       2. (XLT_F) must be set before enter this
                                     ;                          routine. (This flag will fix the logical
                                     ;                          not logical and physical not physical problem.)
                                     ;                          Actually only routines HREAD, HWRITE, and
                                     ;                          HWRITEV calls this routine.
                                     ;
                                     ;***************************************************************

018F                                 ADVTNS:
018F  F6 06 000A R FF                        TEST    BYTE PTR XLT_F,OFFH     ; check if using translate table
0194  75 0B                                  JNZ     ADVTNS3                 ; yes
0196  FE C3                                  INC     BL                      ; just set to next sector #
0198  80 FB 10                               CMP     BL,SECPTRK              ; pass boundary?
019B  7E 2A                                  JLE     ADVT1                   ; no
019D  B3 01                                  MOV     BL,1                    ; yes, reset sector #
019F  EB 1C                                  JMP     SHORT ADVTNT            ; and update surface/cylinder #

01A1                                 ADVTNS3:
01A1  51                                     PUSH    CX                      ; save registers
01A2  57                                     PUSH    DI
01A3  06                                     PUSH    ES

01A4  0E                                     PUSH    CS                      ; ES = CS
01A5  07                                     POP     ES
01A6  BF 0000 E                              MOV     DI,OFFSET RDXLT         ; get translate table
01A9  B9 0010                                MOV     CX,SECPTRK              ; get sectors per track
01AC  8A C3                                  MOV     AL,BL                   ; get sector #
01AE  F2/ AE                                 REPNZ   SCASB                   ; check in XLT

01B0  8A 1D                                  MOV     BL,BYTE PTR [DI]        ; get next sector #
01B2  07                                     POP     ES                      ; restore registers
01B3  5F                                     POP     DI
01B4  E3 02                                  JCXZ    ADVT2                   ; if not in XLT
01B6  59                                     POP     CX
01B7  C3                                     RET
01B8                                 ADVT2:
01B8  59                                     POP     CX
01B9  8A 1E 0000 E                           MOV     BL,RDXLT                ; get first sector #
                                     ;       JMP     ADVTNT

```

```
                            PAGE
                    ;************************************************************
                    ;
                    ;       ADVTNT - advance to next track
                    ;
                    ;       ENTRY:  BH = surface #
                    ;               DX = cylinder #
                    ;
                    ;************************************************************

01BD                ADVTNT:
01BD  FE C7                 INC     BH                      ; bump surface #
01BF  80 FF 04              CMP     BH,NUMHEAD              ; check pass limit
01C2  7C 03                 JL      ADVT1
01C4  32 FF                 XOR     BH,BH                   ; set surface # =0
01C6  42                    INC     DX                      ; bump cylinder #
01C7                ADVT1:
01C7  C3                    RET
```

```
                            PAGE
                    ;************************************************************
                    ;
                    ;       CLTPN - Convert Logical Track # to Physical
                    ;               surface/cylinder Number
                    ;
                    ;       ENTRY:  DX = logical track #
                    ;       EXIT:   BH = surface #
                    ;               DX = cylinder #
                    ;       USE:    NONE
                    ;
                    ;************************************************************

01C8                CLTPN:
01C8  8A FA                 MOV     BH,DL           ; get low track byte
01CA  80 E7 03              AND     BH,03H          ; keep surface only
01CD  D1 EA                 SHR     DX,1            ; get rid of surface bits
01CF  D1 EA                 SHR     DX,1
01D1  C3                    RET


                    ;************************************************************
                    ;
                    ;       CPTLN - Convert Physical surface/cylinder Number
                    ;               to Logical Track Number
                    ;
                    ;       ENTRY:  BH = surface #
                    ;               DX = cylinder #
                    ;       EXIT:   DX = logical track #
                    ;       USE:    AX
                    ;
                    ;************************************************************

01D2                CPTLN:
01D2  D1 E2                 SHL     DX,1            ; make room for surface #
01D4  D1 E2                 SHL     DX,1
01D6  33 C0                 XOR     AX,AX
01D8  8A C7                 MOV     AL,BH           ; get surface #
01DA  03 D0                 ADD     DX,AX           ; add surface #
01DC  C3                    RET
```

```
                                                PAGE
                                        ;*************************************************************
                                        ;
                                        ;       INTHDL - INTERRUPT HANDLER
                                        ;
                                        ;       ENTRY:  NONE
                                        ;       EXIT:   NONE
                                        ;
                                        ;*************************************************************

        01DD                            INTHDL:
        01DD  50                                PUSH    AX
        01DE  2E: FE 06 01E7 R                  INC     BYTE PTR CS:IRQDRQ      ; ACKOWNLEDGE INT
        01E3  E4 67                             IN      AL,RDSTAT               ; CLEAR INT

                                            ENDIF

        01E5  58                                POP     AX
        01E6  CF                                IRET


        01E7                            IRQDRQ  LABEL   BYTE                    ; INTERRUPT FLAG !!!
        01E7  00                                DB      0                       ; THIS BYTE BETTER GO WITH CODE_SEG
                                                                                ; SINCE, DS IS UNKNOWN WHEN INTERRUPT

                                        ;*************************************************************
                                        ;
                                        ;       DELAY - DELAY ABOUT 0.5 MS
                                        ;
                                        ;       ENTRY:  NONE
                                        ;       EXIT:   NONE
                                        ;
                                        ;*************************************************************

        01E8                            DELAY:
        01E8  51                                PUSH    CX
        01E9  B9 0094                           MOV     CX,148                  ; LOOP COUNT
        01EC  E2 FE                     DELAY1: LOOP    DELAY1                  ; 17 CLOCKS * 0.2 US
        01EE  59                                POP     CX
        01EF  C3                                RET
```

```
                                                PAGE
                                        ;*************************************************************
                                        ;
                                        ;       RETRY - error recovery
                                        ;
                                        ;       ENTRY:  AX = original error status
                                        ;               BX = drive/surface/sector
                                        ;               DX = cylinder #
                                        ;       EXIT:   AX = original error status
                                        ;               (RETNS) updates
                                        ;
                                        ;*************************************************************

        01F0                            RETRY:
        01F0  A3 0007 R                         MOV     ERRCODE,AX              ; SAVE THE ORIGINAL ERROR
        01F3  FE 06 0005 R                      INC     BYTE PTR RETNS          ; UPDATE RETRY #
        01F7  BE 0006 R                         MOV     SI,OFFSET RETRIES       ; RESET LOCAL RETRY COUNT
        01FA  C6 04 00                          MOV     BYTE PTR [SI],0
        01FD                            RETRY1:
        01FD  8A 04                             MOV     AL,[SI]                 ; GET RETRY COUNT
        01FF  24 03                             AND     AL,00000011B            ; TIME TO DO NEW TRICKS? (EVERY 4)
        0201  75 10                             JNZ     RETRY2                  ;  SKIP THE TRICK IF NO
        0203  8B 2C                             MOV     BP,[SI]                 ; GET THE RETRY COUNT AGAIN
        0205  81 E5 00FC                        AND     BP,11111100B            ; ONLY KEEP THE BITS WE WANT
        0209  D1 FD                             SAR     BP,1                    ; SHIFT TO WORD OFFSET
        020B  2E: FF 96 0224 R                  CALL    CS:MOVEHR[BP]           ; PERFORM THE TRICK
        0210  E8 01E8 R                         CALL    DELAY                   ; WAIT FOR HEAD TO SETTLE

                                        ;       READ SECTOR RETRY

        0213                            RETRY2:
        0213  E8 0105 R                         CALL    LDTSKF                  ; RELOAD REGISTERS
        0216  B0 20                             MOV     AL,RDREAD+ATRETRY       ; ISSUE THE READ COMMAND
        0218  E8 014F R                         CALL    OUTCMD
        021B  0A C0                             OR      AL,AL                   ; ERROR?
        021D  74 04                             JZ      RETRY3                  ;  NO ERROR, FINALLY
        021F  FE 04                             INC     BYTE PTR [SI]           ; 1 MORE RETRY
        0221  EB DA                             JMP     SHORT RETRY1            ; TRY MORE
        0223                            RETRY3:
        0223  C3                                RET
```

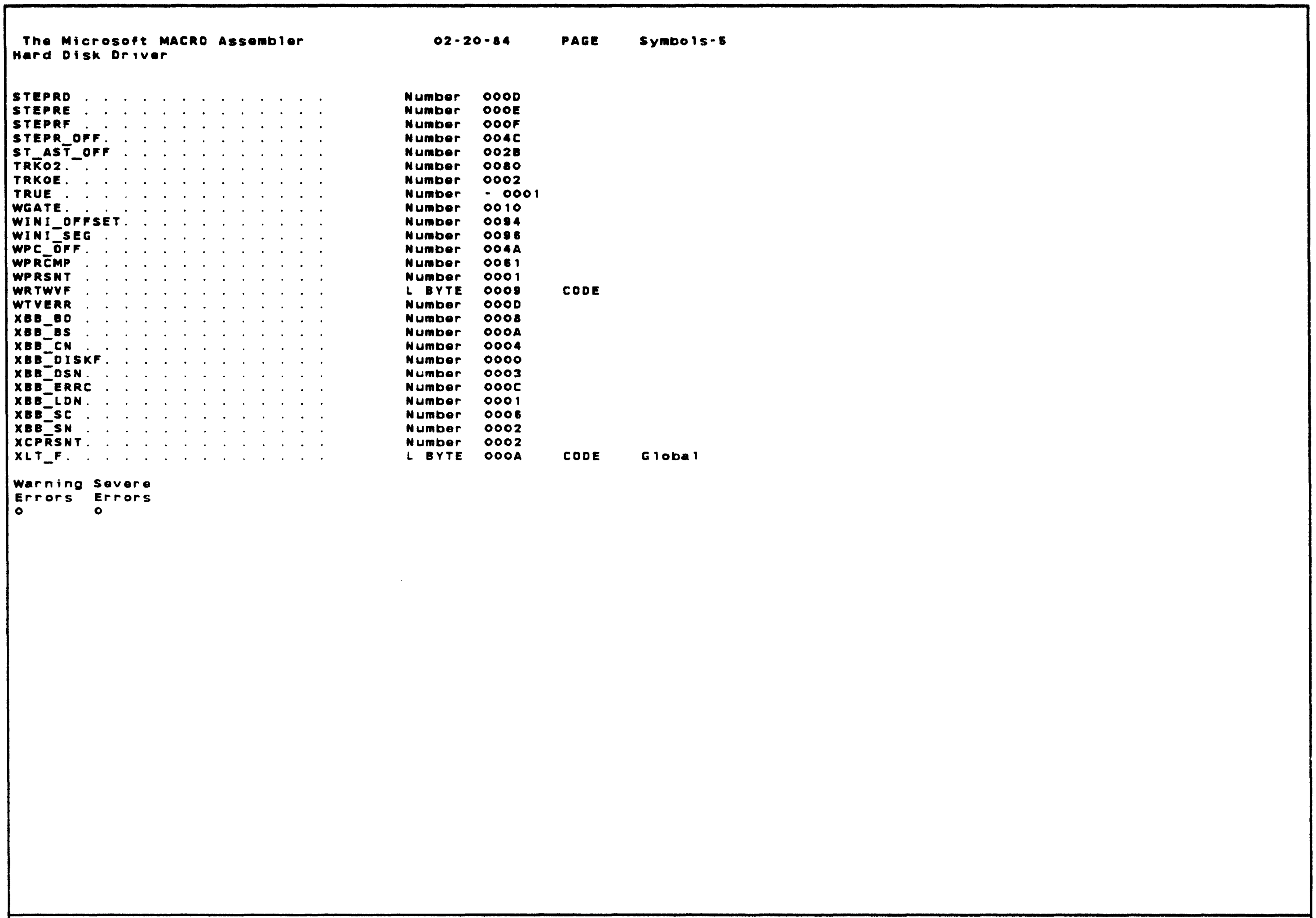
```
                                        PAGE
                                ;*************************************************************
                                ;
                                ;       MOVEHR - MOVE HEAD ROUTINES
                                ;
                                ;*************************************************************

 0224                           MOVEHR  LABEL   WORD
 0224  0264 R                           DW      SAME                    ; FIRST ROUND STAY ON SAME cylinder
 0226  0243 R                           DW      MOVEO4                  ; OUT 4 cylinder
 0228  0233 R                           DW      MOVEI4                  ; IN 4 cylinders
 022A  0258 R                           DW      HOMET                   ; DO A RESTORE
 022C  0230 R                           DW      MOVEMX                  ; MOVE TO INNER cylinder
 022E  0260 R                           DW      GIVEUP                  ; TIME TO GIVE UP


                                ;       MOVE TO INNER cylinder

 0230                           MOVEMX:
 0230  52                               PUSH    DX                      ; SAVE cylinder #
 0231  EB 0A                            JMP     SHORT MVI41             ; GO TO LOAD MAX cylinder # & MOVE HEAD


                                ;       MOVE IN 4 cylinderS

 0233                           MOVEI4:
 0233  52                               PUSH    DX                      ; SAVE cylinder #
 0234  83 C2 04                         ADD     DX,4                    ; ADD 4 cylinderS
 0237  3B 16 0000 R                     CMP     DX,MAXTRK               ; CHECK PASS MAX cylinder #
 023B  72 0E                            JC      MOVEH                   ; IF < MAX, THEN PERFORM MOVE HEAD
 023D                           MVI41:
 023D  8B 16 0000 R                     MOV     DX,MAXTRK               ; LOAD THE MAX cylinder #
 0241  EB 08                            JMP     SHORT MOVEH             ; PERFORM MOVE HEAD


                                ;       MOVE OUT 4 cylinderS

 0243                           MOVEO4:
 0243  52                               PUSH    DX                      ; SAVE cylinder #
 0244  83 EA 04                         SUB     DX,4                    ; SUB 4 cylinderS
 0247  79 02                            JNS     MOVEH                   ; < 0 ?
 0249  33 D2                            XOR     DX,DX                   ; JUST USE 0
                                ;       JMP     MOVEH                   ; MERGE TO MOVE HEAD


                                ;       MOVEH - MOVE HEAD

 024B                           MOVEH:
 024B  E8 0105 R                        CALL    LDTSKF                  ; LOAD REGISTERS (cylinder #)
 024E  A0 0002 R                        MOV     AL,STEPRATE             ; GET STEP RATE and issue seek
 0251  04 70                            ADD     AL,SEEK
 0253  E8 014F R                        CALL    OUTCMD
 0256  5A                               POP     DX                      ; RESTORE OLD cylinder #
 0257  C3                               RET

```

```

                                ;       PERFORM RESTORE

 0258                           HOMET:
 0258  A0 0002 R                        MOV     AL,STEPRATE             ; GET STEP RATE
 025B  04 10                            ADD     AL,RESTOR               ; ISSUE RESTORE COMMAND
 025D  E9 014F R                        JMP     OUTCMD


                                ;       GIVE UP AFTER SO MANY RETRIES

 0260                           GIVEUP:
 0260  58                               POP     AX                      ; GET RID OF RETURN ADDR
 0261  A1 0007 R                        MOV     AX,ERRCODE              ; RETURN THE ORIGINAL ERROR
 0264                           SAME:
 0264  C3                               RET

 0265                           CODE    ENDS
                                        END
 The Microsoft MACRO Assembler                02-20-84      PAGE     Symbols-1
Hard Disk Driver

Segments and groups:

                N a m e                      Size     align    combine class

CGROUP . . . . . . . . . . . . . .           GROUP
  CODE . . . . . . . . . . . . . . .         0265     BYTE     PUBLIC  'CODE'

Symbols:

                N a m e                      Type     Value    Attr

ABRTC. . . . . . . . . . . . . . .           Number   0004
ADVT1. . . . . . . . . . . . . . .           L NEAR   01C7     CODE
ADVT2. . . . . . . . . . . . . . .           L NEAR   01B8     CODE
ADVTNS . . . . . . . . . . . . . .           L NEAR   018F     CODE
ADVTNS3. . . . . . . . . . . . . .           L NEAR   01A1     CODE
ADVTNT . . . . . . . . . . . . . .           L NEAR   01BD     CODE
AST_OFF. . . . . . . . . . . . . .           Number   0026
ATRETRY. . . . . . . . . . . . . .           Number   0000
BADBD. . . . . . . . . . . . . . .           Number   0080
BAT_OFF. . . . . . . . . . . . . .           Number   0012
BID_AST. . . . . . . . . . . . . .           Number   0005
BID_BAT. . . . . . . . . . . . . .           Number   0002
BID_BOT. . . . . . . . . . . . . .           Number   0008
BID_DPD. . . . . . . . . . . . . .           Number   0003
BID_HOM. . . . . . . . . . . . . .           Number   0001
BID_OSN. . . . . . . . . . . . . .           Number   0004
BID_PAS. . . . . . . . . . . . . .           Number   0006
BID_PBT. . . . . . . . . . . . . .           Number   0007
BID_UKN. . . . . . . . . . . . . .           Number   0000
BIT0 . . . . . . . . . . . . . . .           Number   0001
BIT1 . . . . . . . . . . . . . . .           Number   0002
BIT15. . . . . . . . . . . . . . .           Number   8000
BIT2 . . . . . . . . . . . . . . .           Number   0004
BIT3 . . . . . . . . . . . . . . .           Number   0008
BIT4 . . . . . . . . . . . . . . .           Number   0010
BIT5 . . . . . . . . . . . . . . .           Number   0020
BIT6 . . . . . . . . . . . . . . .           Number   0040
BIT7 . . . . . . . . . . . . . . .           Number   0080
BK_EC_OFF. . . . . . . . . . . . .           Number   0008
BK_LBN_OFF . . . . . . . . . . . .           Number   0003
BK_MEC_OFF . . . . . . . . . . . .           Number   0006
BOOT_OFF . . . . . . . . . . . . .           Number   0021
CBSY . . . . . . . . . . . . . . .           Number   0080
CBSY2. . . . . . . . . . . . . . .           Number   0001
CLRIL. . . . . . . . . . . . . . .           Number   0004
CLRSPL . . . . . . . . . . . . . .           Number   0008
CLTPN. . . . . . . . . . . . . . .           L NEAR   01C8     CODE     Global
CMDIP. . . . . . . . . . . . . . .           Number   0002
CPM8680. . . . . . . . . . . . . .           Number   0001
CPTLN. . . . . . . . . . . . . . .           L NEAR   01D2     CODE
CR . . . . . . . . . . . . . . . .           Number   000D
CRCER. . . . . . . . . . . . . . .           Number   0040
CTRL_Q . . . . . . . . . . . . . .           Number   0011
CTRL_S . . . . . . . . . . . . . .           Number   0013
```

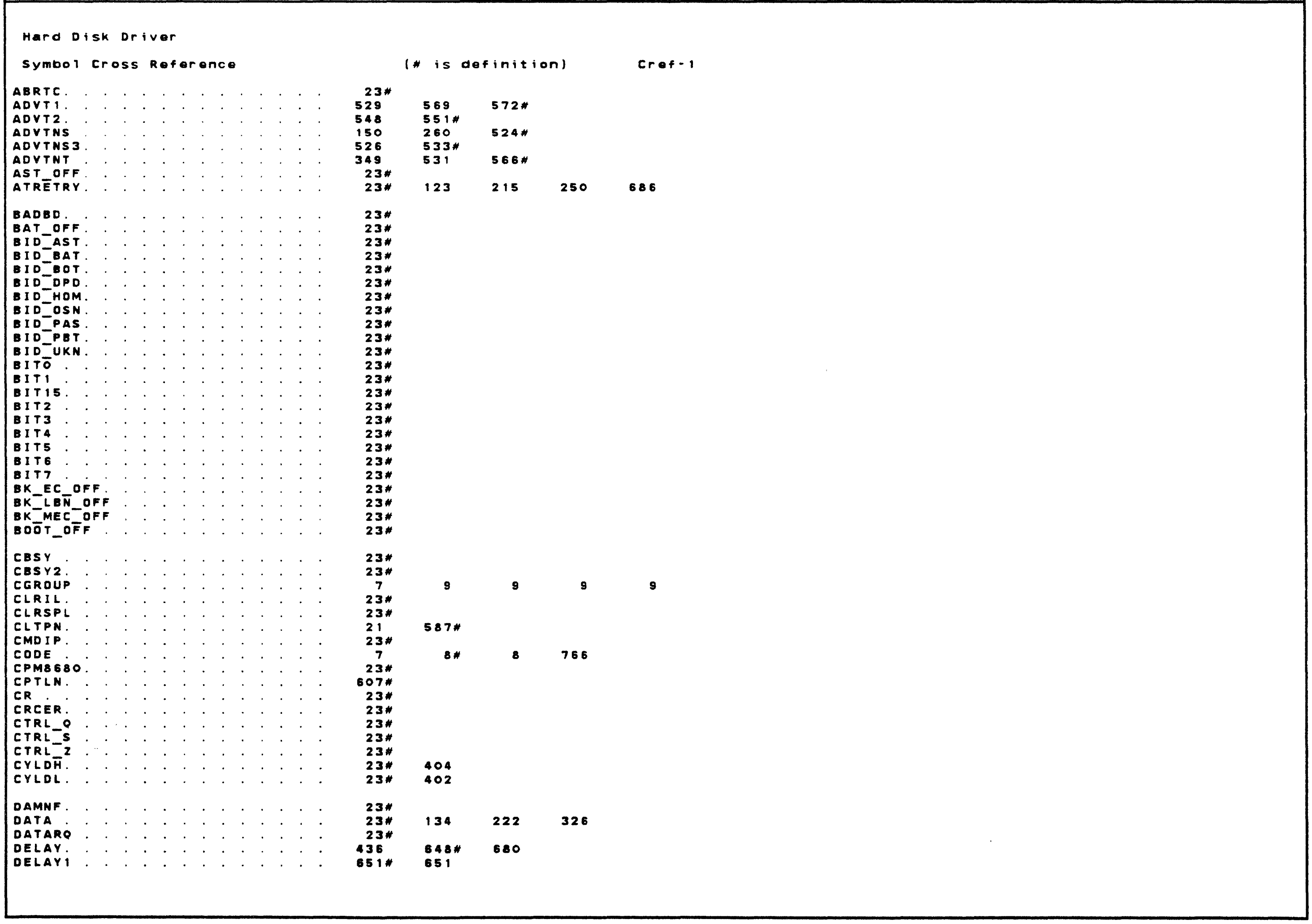
```
 The Microsoft MACRO Assembler                02-20-84      PAGE     Symbols-2
Hard Disk Driver


CTRL_Z . . . . . . . . . . . . . .           Number   001A
CYLDH. . . . . . . . . . . . . . .           Number   0065
CYLDL. . . . . . . . . . . . . . .           Number   0064
DAMNF. . . . . . . . . . . . . . .           Number   0001
DATA . . . . . . . . . . . . . . .           Number   0060
DATARQ . . . . . . . . . . . . . .           Number   0008
DELAY. . . . . . . . . . . . . . .           L NEAR   01E8     CODE
DELAY1 . . . . . . . . . . . . . .           L NEAR   01EC     CODE
DISKDF . . . . . . . . . . . . . .           Number   0005
DPDE_FLAG. . . . . . . . . . . . .           Number   0000
DPDE_FTN . . . . . . . . . . . . .           Number   000C
DPDE_INIT. . . . . . . . . . . . .           Number   00F0
DPDE_LTN . . . . . . . . . . . . .           Number   000E
DPDE_LU. . . . . . . . . . . . . .           Number   0001
DPDE_OFF . . . . . . . . . . . . .           Number   0020
DPDE_OST . . . . . . . . . . . . .           Number   000B
DPDE_PN. . . . . . . . . . . . . .           Number   0002
DPDE_PON . . . . . . . . . . . . .           Number   000A
DPD_OFF. . . . . . . . . . . . . .           Number   0017
DPD_START_OFF. . . . . . . . . . .           Number   0020
DRDY . . . . . . . . . . . . . . .           Number   0040
DRDY2. . . . . . . . . . . . . . .           Number   0040
DRIVE0 . . . . . . . . . . . . . .           Number   0000
DRIVE1 . . . . . . . . . . . . . .           Number   0008
DRIVE2 . . . . . . . . . . . . . .           Number   0010
DRIVE3 . . . . . . . . . . . . . .           Number   0018
DRIVEDF. . . . . . . . . . . . . .           Number   0003
DRIVEM . . . . . . . . . . . . . .           Number   0018
DRVSEL . . . . . . . . . . . . . .           Number   0001
DSTAT0 . . . . . . . . . . . . . .           Number   0068
DSTAT1 . . . . . . . . . . . . . .           Number   0069
DWRTF. . . . . . . . . . . . . . .           Number   0020
DWRTF2 . . . . . . . . . . . . . .           Number   0020
ERRCODE. . . . . . . . . . . . . .           L WORD   0007     CODE
ERRF . . . . . . . . . . . . . . .           Number   0001
ERROR. . . . . . . . . . . . . . .           Number   0061
FALSE. . . . . . . . . . . . . . .           Number   0000
FMTTRK . . . . . . . . . . . . . .           Number   0050
GIVEUP . . . . . . . . . . . . . .           L NEAR   0260     CODE
HDIR . . . . . . . . . . . . . . .           Number   0008
HDISK. . . . . . . . . . . . . . .           Number   0004
HEAD0. . . . . . . . . . . . . . .           Number   0000
HEAD02 . . . . . . . . . . . . . .           Number   0000
HEAD1. . . . . . . . . . . . . . .           Number   0001
HEAD12 . . . . . . . . . . . . . .           Number   0002
HEAD2. . . . . . . . . . . . . . .           Number   0002
HEAD22 . . . . . . . . . . . . . .           Number   0004
HEAD3. . . . . . . . . . . . . . .           Number   0003
HEAD32 . . . . . . . . . . . . . .           Number   0006
HEADM. . . . . . . . . . . . . . .           Number   0007
HEADM2 . . . . . . . . . . . . . .           Number   000E
HFORMAT. . . . . . . . . . . . . .           L NEAR   0088     CODE     Global
HFORMAT1 . . . . . . . . . . . . .           L NEAR   00DA     CODE
HFORMAT2 . . . . . . . . . . . . .           L NEAR   00FC     CODE
HFORMAT5 . . . . . . . . . . . . .           L NEAR   008B     CODE
```

```
HINIT. . . . . . . . . . . . .   L NEAR   0129    CODE     Global
HIVEC_A. . . . . . . . . . . .   Number   0114
HIVEC_B. . . . . . . . . . . .   Number   0294
HMCHK. . . . . . . . . . . . .   L NEAR   00FE    CODE     Global
HOMET. . . . . . . . . . . . .   L NEAR   0258    CODE
HREAD. . . . . . . . . . . . .   L NEAR   000B    CODE     Global
HREAD1 . . . . . . . . . . . .   L NEAR   002E    CODE
HREAD10. . . . . . . . . . . .   L NEAR   001B    CODE
HREAD2 . . . . . . . . . . . .   L NEAR   004D    CODE
HREAD3 . . . . . . . . . . . .   L NEAR   0026    CODE
HREAD5 . . . . . . . . . . . .   L NEAR   0011    CODE
HVDISK . . . . . . . . . . . .   L NEAR   0102    CODE     Global
HWRITE . . . . . . . . . . . .   L NEAR   0058    CODE     Global
HWRITE1. . . . . . . . . . . .   L NEAR   005D    CODE
HWRITE11 . . . . . . . . . . .   L NEAR   00AA    CODE
HWRITE2. . . . . . . . . . . .   L NEAR   00B1    CODE
HWRITE21 . . . . . . . . . . .   L NEAR   0079    CODE
HWRITE5. . . . . . . . . . . .   L NEAR   0060    CODE
HWRITEV. . . . . . . . . . . .   L NEAR   0051    CODE     Global
IDBIT. . . . . . . . . . . . .   Number   00E0
IDBITV . . . . . . . . . . . .   Number   00A0
IDNF . . . . . . . . . . . . .   Number   0010
INTHDL . . . . . . . . . . . .   L NEAR   01DD    CODE     Global
IRQDRQ . . . . . . . . . . . .   L BYTE   01E7    CODE
LDTSKF . . . . . . . . . . . .   L NEAR   0105    CODE
LDTSKF1. . . . . . . . . . . .   L NEAR   0126    CODE
LED_OFF. . . . . . . . . . . .   L NEAR   00B1    CODE
LF . . . . . . . . . . . . . .   Number   000A
LINDEX . . . . . . . . . . . .   Number   0002
LSTEPP . . . . . . . . . . . .   Number   0004
MAXTRK . . . . . . . . . . . .   L WORD   0000    CODE     Global
MOVEH. . . . . . . . . . . . .   L NEAR   024B    CODE
MOVEHR . . . . . . . . . . . .   L WORD   0224    CODE
MOVEI4 . . . . . . . . . . . .   L NEAR   0233    CODE
MOVEMX . . . . . . . . . . . .   L NEAR   0230    CODE
MOVEO4 . . . . . . . . . . . .   L NEAR   0243    CODE
MSCTF. . . . . . . . . . . . .   Number   0004
MSDOS. . . . . . . . . . . . .   Number   0002
MVI41. . . . . . . . . . . . .   L NEAR   023D    CODE
NCYLD_OFF. . . . . . . . . . .   Number   0040
NT_AST_OFF . . . . . . . . . .   Number   002D
NUMHEAD. . . . . . . . . . . .   Number   0004
OSN_OFF. . . . . . . . . . . .   Number   001C
OUTCMD . . . . . . . . . . . .   L NEAR   014F    CODE
OUTCMD1. . . . . . . . . . . .   L NEAR   0157    CODE
OUTCMD11 . . . . . . . . . . .   L NEAR   015E    CODE
OUTCMD2. . . . . . . . . . . .   L NEAR   015B    CODE
OUTCMD3. . . . . . . . . . . .   L NEAR   016C    CODE
OUTCMD4. . . . . . . . . . . .   L NEAR   017E    CODE
OUTCMD5. . . . . . . . . . . .   L NEAR   018C    CODE
OUTCMD6. . . . . . . . . . . .   L NEAR   016E    CODE
PART_FTN . . . . . . . . . . .   Number   0000
PART_LTN . . . . . . . . . . .   Number   0002
PART_PAS . . . . . . . . . . .   Number   0004
PASE_OFF . . . . . . . . . . .   Number   0005
```

```
PAS_ASN. . . . . . . . . . . .   Number   0004
PAS_ATN. . . . . . . . . . . .   Number   0003
PAS_BSN. . . . . . . . . . . .   Number   0002
PAS_BTN. . . . . . . . . . . .   Number   0000
PRECOMP. . . . . . . . . . . .   L WORD   0003    CODE     Global
PRE_READ . . . . . . . . . . .   Alias    FALSE
RDBASE . . . . . . . . . . . .   Number   0060
RDCMD. . . . . . . . . . . . .   Number   0067
RDCMD2 . . . . . . . . . . . .   Number   0068
RDINTF . . . . . . . . . . . .   Number   0008
RDREAD . . . . . . . . . . . .   Number   0020
RDSTAT . . . . . . . . . . . .   Number   0067
RDWRITE. . . . . . . . . . . .   Number   0030
RDXLT. . . . . . . . . . . . .   V BYTE   0000    CODE     External
RESTOR . . . . . . . . . . . .   Number   0010
RETNS. . . . . . . . . . . . .   L BYTE   0005    CODE
RETRIES. . . . . . . . . . . .   L BYTE   0006    CODE
RETRY. . . . . . . . . . . . .   L NEAR   01F0    CODE
RETRY1 . . . . . . . . . . . .   L NEAR   01FD    CODE
RETRY2 . . . . . . . . . . . .   L NEAR   0213    CODE
RETRY3 . . . . . . . . . . . .   L NEAR   0223    CODE
SAME . . . . . . . . . . . . .   L NEAR   0264    CODE
SBUFR. . . . . . . . . . . . .   Number   0001
SCANID . . . . . . . . . . . .   Number   0040
SCTEXT . . . . . . . . . . . .   Number   0080
SDH. . . . . . . . . . . . . .   Number   0066
SECPTRK. . . . . . . . . . . .   Number   0010
SECSIZE. . . . . . . . . . . .   Number   0200
SECTC. . . . . . . . . . . . .   Number   0062
SECTN. . . . . . . . . . . . .   Number   0063
SECTSZ . . . . . . . . . . . .   Number   0060
SEEK . . . . . . . . . . . . .   Number   0070
SEEKC. . . . . . . . . . . . .   Number   0010
SEEKC2 . . . . . . . . . . . .   Number   0010
SERR . . . . . . . . . . . . .   Number   000F
SINIT. . . . . . . . . . . . .   Number   0002
SSCTF. . . . . . . . . . . . .   Number   0000
SSZ128 . . . . . . . . . . . .   Number   0060
SSZ1K. . . . . . . . . . . . .   Number   0040
SSZ256 . . . . . . . . . . . .   Number   0000
SSZ512 . . . . . . . . . . . .   Number   0020
STEPR0 . . . . . . . . . . . .   Number   0000
STEPR1 . . . . . . . . . . . .   Number   0001
STEPR2 . . . . . . . . . . . .   Number   0002
STEPR3 . . . . . . . . . . . .   Number   0003
STEPR4 . . . . . . . . . . . .   Number   0004
STEPR5 . . . . . . . . . . . .   Number   0005
STEPR6 . . . . . . . . . . . .   Number   0006
STEPR7 . . . . . . . . . . . .   Number   0007
STEPR8 . . . . . . . . . . . .   Number   0008
STEPR9 . . . . . . . . . . . .   Number   0009
STEPRA . . . . . . . . . . . .   Number   000A
STEPRATE . . . . . . . . . . .   L BYTE   0002    CODE     Global
STEPRB . . . . . . . . . . . .   Number   000B
STEPRC . . . . . . . . . . . .   Number   000C
```



```
 The Microsoft MACRO Assembler              02-20-84      PAGE     Symbols-5
Hard Disk Driver


STEPRD . . . . . . . . . . . . . .          Number   000D
STEPRE . . . . . . . . . . . . . .          Number   000E
STEPRF . . . . . . . . . . . . . .          Number   000F
STEPR_OFF. . . . . . . . . . . . .          Number   004C
ST_AST_OFF . . . . . . . . . . . .          Number   002B
TRKO2. . . . . . . . . . . . . . .          Number   0080
TRKOE. . . . . . . . . . . . . . .          Number   0002
TRUE . . . . . . . . . . . . . . .          Number   - 0001
WGATE. . . . . . . . . . . . . . .          Number   0010
WINI_OFFSET. . . . . . . . . . . .          Number   0094
WINI_SEG . . . . . . . . . . . . .          Number   0096
WPC_OFF. . . . . . . . . . . . . .          Number   004A
WPRCMP . . . . . . . . . . . . . .          Number   0061
WPRSNT . . . . . . . . . . . . . .          Number   0001
WRTWVF . . . . . . . . . . . . . .          L BYTE   0009     CODE
WTVERR . . . . . . . . . . . . . .          Number   000D
XBB_BD . . . . . . . . . . . . . .          Number   0008
XBB_BS . . . . . . . . . . . . . .          Number   000A
XBB_CN . . . . . . . . . . . . . .          Number   0004
XBB_DISKF. . . . . . . . . . . . .          Number   0000
XBB_DSN. . . . . . . . . . . . . .          Number   0003
XBB_ERRC . . . . . . . . . . . . .          Number   000C
XBB_LDN. . . . . . . . . . . . . .          Number   0001
XBB_SC . . . . . . . . . . . . . .          Number   0006
XBB_SN . . . . . . . . . . . . . .          Number   0002
XCPRSNT. . . . . . . . . . . . . .          Number   0002
XLT_F. . . . . . . . . . . . . . .          L BYTE   000A     CODE     Global

Warning Severe
Errors  Errors
0       0
```



```
 Hard Disk Driver

 Symbol Cross Reference                    (# is definition)        Cref-1

ABRTC. . . . . . . . . . . . . . .    23#
ADVT1. . . . . . . . . . . . . . .   529     569     572#
ADVT2. . . . . . . . . . . . . . .   548     551#
ADVTNS . . . . . . . . . . . . . .   150     260     524#
ADVTNS3. . . . . . . . . . . . . .   526     533#
ADVTNT . . . . . . . . . . . . . .   349     531     566#
AST_OFF. . . . . . . . . . . . . .    23#
ATRETRY. . . . . . . . . . . . . .    23#    123     215     250     686

BADBD. . . . . . . . . . . . . . .    23#
BAT_OFF. . . . . . . . . . . . . .    23#
BID_AST. . . . . . . . . . . . . .    23#
BID_BAT. . . . . . . . . . . . . .    23#
BID_BOT. . . . . . . . . . . . . .    23#
BID_DPD. . . . . . . . . . . . . .    23#
BID_HOM. . . . . . . . . . . . . .    23#
BID_OSN. . . . . . . . . . . . . .    23#
BID_PAS. . . . . . . . . . . . . .    23#
BID_PBT. . . . . . . . . . . . . .    23#
BID_UKN. . . . . . . . . . . . . .    23#
BIT0 . . . . . . . . . . . . . . .    23#
BIT1 . . . . . . . . . . . . . . .    23#
BIT15. . . . . . . . . . . . . . .    23#
BIT2 . . . . . . . . . . . . . . .    23#
BIT3 . . . . . . . . . . . . . . .    23#
BIT4 . . . . . . . . . . . . . . .    23#
BIT5 . . . . . . . . . . . . . . .    23#
BIT6 . . . . . . . . . . . . . . .    23#
BIT7 . . . . . . . . . . . . . . .    23#
BK_EC_OFF. . . . . . . . . . . . .    23#
BK_LBN_OFF . . . . . . . . . . . .    23#
BK_MEC_OFF . . . . . . . . . . . .    23#
BOOT_OFF . . . . . . . . . . . . .    23#

CBSY . . . . . . . . . . . . . . .    23#
CBSY2. . . . . . . . . . . . . . .    23#
CGROUP . . . . . . . . . . . . . .     7       9       9       9       9
CLRIL. . . . . . . . . . . . . . .    23#
CLRSPL . . . . . . . . . . . . . .    23#
CLTPN. . . . . . . . . . . . . . .    21     587#
CMDIP. . . . . . . . . . . . . . .    23#
CODE . . . . . . . . . . . . . . .     7       8#      8     766
CPM8680. . . . . . . . . . . . . .    23#
CPTLN. . . . . . . . . . . . . . .   607#
CR . . . . . . . . . . . . . . . .    23#
CRCER. . . . . . . . . . . . . . .    23#
CTRL_Q . . . . . . . . . . . . . .    23#
CTRL_S . . . . . . . . . . . . . .    23#
CTRL_Z . . . . . . . . . . . . . .    23#
CYLDH. . . . . . . . . . . . . . .    23#    404
CYLDL. . . . . . . . . . . . . . .    23#    402

DAMNF. . . . . . . . . . . . . . .    23#
DATA . . . . . . . . . . . . . . .    23#    134     222     326
DATARQ . . . . . . . . . . . . . .    23#
DELAY. . . . . . . . . . . . . . .   436     648#    680
DELAY1 . . . . . . . . . . . . . .   651#    651
```

```
 Hard Disk Driver

 Symbol Cross Reference                        (# is definition)       Cref-2

DISKDF . . . . . . . . . . . . . .    23#    379
DPDE_FLAG. . . . . . . . . . . . .    23#
DPDE_FTN . . . . . . . . . . . . .    23#
DPDE_INIT. . . . . . . . . . . . .    23#
DPDE_LTN . . . . . . . . . . . . .    23#
DPDE_LU. . . . . . . . . . . . . .    23#
DPDE_OFF . . . . . . . . . . . . .    23#
DPDE_OST . . . . . . . . . . . . .    23#
DPDE_PN. . . . . . . . . . . . . .    23#
DPDE_PON . . . . . . . . . . . . .    23#
DPD_OFF. . . . . . . . . . . . . .    23#
DPD_START_OFF. . . . . . . . . . .    23#
DRDY . . . . . . . . . . . . . . .    23#
DRDY2. . . . . . . . . . . . . . .    23#
DRIVE0 . . . . . . . . . . . . . .    23#    441
DRIVE1 . . . . . . . . . . . . . .    23#
DRIVE2 . . . . . . . . . . . . . .    23#
DRIVE3 . . . . . . . . . . . . . .    23#    271
DRIVEDF. . . . . . . . . . . . . .    23#    480
DRIVEM . . . . . . . . . . . . . .    23#
DRVSEL . . . . . . . . . . . . . .    23#
DSTAT0 . . . . . . . . . . . . . .    23#
DSTAT1 . . . . . . . . . . . . . .    23#
DWRTF. . . . . . . . . . . . . . .    23#    492
DWRTF2 . . . . . . . . . . . . . .    23#

ERRCODE. . . . . . . . . . . . . .    32#    668     762
ERRF . . . . . . . . . . . . . . .    23#    495
ERROR. . . . . . . . . . . . . . .    23#    485     486     487

FALSE. . . . . . . . . . . . . . .    23#     23
FMTTRK . . . . . . . . . . . . . .    23#    319

GIVEUP . . . . . . . . . . . . . .   707     760#

HDIR . . . . . . . . . . . . . . .    23#
HDISK. . . . . . . . . . . . . . .    23#
HEAD0. . . . . . . . . . . . . . .    23#    441
HEAD02 . . . . . . . . . . . . . .    23#
HEAD1. . . . . . . . . . . . . . .    23#
HEAD12 . . . . . . . . . . . . . .    23#
HEAD2. . . . . . . . . . . . . . .    23#
HEAD22 . . . . . . . . . . . . . .    23#
HEAD3. . . . . . . . . . . . . . .    23#
HEAD32 . . . . . . . . . . . . . .    23#
HEADM. . . . . . . . . . . . . . .    23#
HEADM2 . . . . . . . . . . . . . .    23#
HFORMAT. . . . . . . . . . . . . .    16     303#
HFORMAT1 . . . . . . . . . . . . .   331#    336
HFORMAT2 . . . . . . . . . . . . .   315     322     347     353#
HFORMAT5 . . . . . . . . . . . . .   312#    350
HINIT. . . . . . . . . . . . . . .    17     429#
HIVEC_A. . . . . . . . . . . . . .    23#
HIVEC_B. . . . . . . . . . . . . .    23#
HMCHK. . . . . . . . . . . . . . .    18     365#
HOMET. . . . . . . . . . . . . . .   705     752#
HREAD. . . . . . . . . . . . . . .    13     111#
```

```
 Hard Disk Driver

 Symbol Cross Reference                        (# is definition)       Cref-3

HREAD1 . . . . . . . . . . . . . .   136#    142
HREAD10. . . . . . . . . . . . . .   125#
HREAD2 . . . . . . . . . . . . . .   122     130     157     159#
HREAD3 . . . . . . . . . . . . . .   127     131#
HREAD5 . . . . . . . . . . . . . .   120#    151
HVDISK . . . . . . . . . . . . . .    19     378#
HWRITE . . . . . . . . . . . . . .    14     201#
HWRITE1. . . . . . . . . . . . . .   199     203#
HWRITE11 . . . . . . . . . . . . .   248     259#
HWRITE2. . . . . . . . . . . . . .   214     218     243     255     264#
HWRITE21 . . . . . . . . . . . . .   227#    232
HWRITE5. . . . . . . . . . . . . .   212#    261
HWRITEV. . . . . . . . . . . . . .    15     197#

IDBIT. . . . . . . . . . . . . . .    23#
IDBITV . . . . . . . . . . . . . .    23#
IDNF . . . . . . . . . . . . . . .    23#
INTHDL . . . . . . . . . . . . . .    20     624#
IRQDRQ . . . . . . . . . . . . . .   238     342     464     474     626     635#

LDTSKF . . . . . . . . . . . . . .   121     213     314     397#    685     742
LDTSKF1. . . . . . . . . . . . . .   400     416#
LED_OFF. . . . . . . . . . . . . .   161     269#    354     451
LF . . . . . . . . . . . . . . . .    23#
LINDEX . . . . . . . . . . . . . .    23#
LSTEPP . . . . . . . . . . . . . .    23#

MAXTRK . . . . . . . . . . . . . .    21      25#    399     722     725
MOVEH. . . . . . . . . . . . . . .   723     726     734     741#
MOVEHR . . . . . . . . . . . . . .   679     701#
MOVEI4 . . . . . . . . . . . . . .   704     719#
MOVEMX . . . . . . . . . . . . . .   706     712#
MOVE04 . . . . . . . . . . . . . .   446     703     731#
MSCTF. . . . . . . . . . . . . . .    23#
MSDOS. . . . . . . . . . . . . . .    23#
MVI41. . . . . . . . . . . . . . .   714     724#

NCYLD_OFF. . . . . . . . . . . . .    23#
NT_AST_OFF . . . . . . . . . . . .    23#
NUMHEAD. . . . . . . . . . . . . .    23#    568

OSN_OFF. . . . . . . . . . . . . .    23#
OUTCMD . . . . . . . . . . . . . .   124     216     251     320     450     463#    687     745     755
OUTCMD1. . . . . . . . . . . . . .   241     345     466#
OUTCMD11 . . . . . . . . . . . . .   473#    476
OUTCMD2. . . . . . . . . . . . . .   471#    478
OUTCMD3. . . . . . . . . . . . . .   479#    493
OUTCMD4. . . . . . . . . . . . . .   475     490#
OUTCMD5. . . . . . . . . . . . . .   489     498#
OUTCMD6. . . . . . . . . . . . . .   481#    496

PART_FTN . . . . . . . . . . . . .    23#
PART_LTN . . . . . . . . . . . . .    23#
PART_PAS . . . . . . . . . . . . .    23#
PASE_OFF . . . . . . . . . . . . .    23#
PAS_ASN. . . . . . . . . . . . . .    23#
PAS_ATN. . . . . . . . . . . . . .    23#
```

```
 Hard Disk Driver

 Symbol Cross Reference                      (# is definition)       Cref-4

PAS_BSN. . . . . . . . . . . . . .    23#
PAS_BTN. . . . . . . . . . . . . .    23#
PRECOMP. . . . . . . . . . . . . .    21      27#    438
PRE_READ . . . . . . . . . . . . .    23#     37     115     153     160     206     306     432     629

RDBASE . . . . . . . . . . . . . .    23#     23      23      23      23      23      23      23      23      23      23      23      23      23
RDCMD. . . . . . . . . . . . . . .    23#    465
RDCMD2 . . . . . . . . . . . . . .    23#    148     240     258     344     435     484
RDINTF . . . . . . . . . . . . . .    23#
RDREAD . . . . . . . . . . . . . .    23#    123     250     686
RDSTAT . . . . . . . . . . . . . .    23#    491     494     627
RDWRITE. . . . . . . . . . . . . .    23#    215
RDXLT. . . . . . . . . . . . . . .    11#    540     553
RESTOR . . . . . . . . . . . . . .    23#    449     754
RETNS. . . . . . . . . . . . . . .    29#    113     156     669
RETRIES. . . . . . . . . . . . . .    30#    670
RETRY. . . . . . . . . . . . . . .   128     667#
RETRY1 . . . . . . . . . . . . . .   672#    691
RETRY2 . . . . . . . . . . . . . .   675     684#
RETRY3 . . . . . . . . . . . . . .   689     692#

SAME . . . . . . . . . . . . . . .   702     763#
SBUFR. . . . . . . . . . . . . . .    23#    147     239     257     343     483
SCANID . . . . . . . . . . . . . .    23#
SCTEXT . . . . . . . . . . . . . .    23#
SDH. . . . . . . . . . . . . . . .    23#    272     408     442
SECPTRK. . . . . . . . . . . . . .    23#    316     327     528     541
SECSIZE. . . . . . . . . . . . . .    23#    135     223
SECTC. . . . . . . . . . . . . . .    23#    317     414
SECTN. . . . . . . . . . . . . . .    23#    411
SECTSZ . . . . . . . . . . . . . .    23#
SEEK . . . . . . . . . . . . . . .    23#    744
SEEKC. . . . . . . . . . . . . . .    23#
SEEKC2 . . . . . . . . . . . . . .    23#
SERR . . . . . . . . . . . . . . .    23#    158
SINIT. . . . . . . . . . . . . . .    23#    434
SSCTF. . . . . . . . . . . . . . .    23#
SSZ128 . . . . . . . . . . . . . .    23#
SSZ1K. . . . . . . . . . . . . . .    23#
SSZ256 . . . . . . . . . . . . . .    23#
SSZ512 . . . . . . . . . . . . . .    23#    271     406     441
STEPRO . . . . . . . . . . . . . .    23#
STEPR1 . . . . . . . . . . . . . .    23#
STEPR2 . . . . . . . . . . . . . .    23#
STEPR3 . . . . . . . . . . . . . .    23#
STEPR4 . . . . . . . . . . . . . .    23#
STEPR5 . . . . . . . . . . . . . .    23#
STEPR6 . . . . . . . . . . . . . .    23#     26
STEPR7 . . . . . . . . . . . . . .    23#
STEPR8 . . . . . . . . . . . . . .    23#
STEPR9 . . . . . . . . . . . . . .    23#
STEPRA . . . . . . . . . . . . . .    23#
STEPRATE . . . . . . . . . . . . .    21      26#    448     743     753
STEPRB . . . . . . . . . . . . . .    23#
STEPRC . . . . . . . . . . . . . .    23#
STEPRD . . . . . . . . . . . . . .    23#
STEPRE . . . . . . . . . . . . . .    23#
```

```
 Hard Disk Driver

 Symbol Cross Reference                      (# is definition)       Cref-5

STEPRF . . . . . . . . . . . . . .    23#
STEPR_OFF. . . . . . . . . . . . .    23#
ST_AST_OFF . . . . . . . . . . . .    23#

TRKO2. . . . . . . . . . . . . . .    23#
TRKOE. . . . . . . . . . . . . . .    23#
TRUE . . . . . . . . . . . . . . .    23#     23

WGATE. . . . . . . . . . . . . . .    23#
WINI_OFFSET. . . . . . . . . . . .    23#
WINI_SEG . . . . . . . . . . . . .    23#
WPC_OFF. . . . . . . . . . . . . .    23#
WPRCMP . . . . . . . . . . . . . .    23#    439
WPRSNT . . . . . . . . . . . . . .    23#
WRTWVF . . . . . . . . . . . . . .    33#    198     202     247
WTVERR . . . . . . . . . . . . . .    23#    254

XBB_BO . . . . . . . . . . . . . .    23#
XBB_BS . . . . . . . . . . . . . .    23#
XBB_CN . . . . . . . . . . . . . .    23#
XBB_DISKF. . . . . . . . . . . . .    23#
XBB_DSN. . . . . . . . . . . . . .    23#
XBB_ERRC . . . . . . . . . . . . .    23#
XBB_LDN. . . . . . . . . . . . . .    23#
XBB_SC . . . . . . . . . . . . . .    23#
XBB_SN . . . . . . . . . . . . . .    23#
XCPRSNT. . . . . . . . . . . . . .    23#
XLT_F. . . . . . . . . . . . . . .    21      35#    525
```

```
                                        PAGE 60,132
                                        TITLE   Hard Disk Initialization
                                        NAME    HINIT
                                ;
                                ;       COMPANY CONFIDENTIAL
                                ;       Copyright (C) 1983 Digital Equipment Corporation
                                ;       All rights reserved.
                                ;

                                ;       09/22/83

                                CGROUP  GROUP   CODE
        0000                    CODE    SEGMENT BYTE PUBLIC 'CODE'
                                ASSUME  CS:CGROUP, DS:CGROUP, ES:CGROUP, SS:CGROUP

                                        EXTRN   HREAD:NEAR, HWRITE:NEAR, HWRITEV:NEAR
                                        EXTRN   HFORMAT:NEAR, HINIT:NEAR, HMCHK:NEAR
                                        EXTRN   HVDISK:NEAR, INTHDL:NEAR, PSTR:NEAR
                                        EXTRN   INI_TAB:WORD, PTRSAVE:DWORD, HDSKO:BYTE
                                        EXTRN   PARTITION:WORD
                                        EXTRN   GETFTN:NEAR, GETPEA:NEAR, CLTPN:NEAR
                                        EXTRN   XLT_F:BYTE, MAXTRK:WORD, STEPRATE:BYTE, PRECOMP:WORD
                                        EXTRN   BUFFER:WORD, PART_SIZE:ABS
                                        EXTRN   XOPTION:BYTE

                                        PUBLIC  DSK_INIT, RHOME, NDRV, HNFMT
                                        PUBLIC  UP_BAT, TEMP_TRK, TEMP_SEC, HD_INIT

                                        .LIST

        = 0004                  MNPARTS EQU     4

```

```
                                        PAGE
                                ;Define offsets for io data packet

                                IODAT   STRUC
        0000  ??                CMDLEN  DB      ?               ;LENGTH OF THIS COMMAND
        0001  ??                UNIT    DB      ?               ;SUB UNIT SPECIFIER
        0002  ??                CMD     DB      ?               ;COMMAND CODE
        0003  ????              STATUSW DW      ?               ;STATUS
        0005      08 [                  DB      8 DUP (?)
                        ??
                             ]

        000D  ??                MEDIA   DB      ?               ;MEDIA DESCRIPTOR
        000E  ????????          TRANS   DD      ?               ;TRANSFER ADDRESS
        0012  ????              COUNT   DW      ?               ;COUNT OF BLOCKS OR CHARACTERS
        0014  ????              START   DW      ?               ;FIRST BLOCK TO TRANSFER
        0016                    IODAT   ENDS

                                        SUBTTL  Common Drive parameter block definitions
```

```
                                         PAGE
                                DBP      STRUC

                                ;-------  Start of Drive Parameter Block.

0000  ????                      SECSZ    DW      ?              ;Sector size in bytes.                (dpb)
0002  ??                        ALLOC    DB      ?              ;Number of sectors per alloc. block.   (dpb)
0003  ????                      RESSEC   DW      ?              ;Reserved sectors.                    (dpb)
0005  ??                        FATS     DB      ?              ;Number of FAT's.                     (dpb)
0006  ????                      MAXDIR   DW      ?              ;Number of root directory entries.    (dpb)
0008  ????                      SECTORS  DW      ?              ;Number of sectors per diskette.      (dpb)
000A  ??                        MEDIAID  DB      ?              ;Media byte ID.                       (dpb)
000B  ????                      FATSEC   DW      ?              ;Number of FAT Sectors.               (dpb)

                                ;-------  End of Drive Parameter Block.

000D  ????                      SECTRK   DW      ?              ;Number of Sectors per track.
000F  ????                      HEADS    DW      ?              ;Number of heads per cylinder.
0011  ????                      HIDDEN   DW      ?              ,Number of hidden sectors.

0013                            DBP      ENDS

= 0013                          DPB_OFF  EQU     19             ;Size of DPB

                                         SUBTTL  Hard Disk Drive initalization routine.
```

```
                                         PAGE

                                         PUBLIC  HD_INIT                  ;Entry from BIOSINIT

0000                            HD_INIT:
0000  C6 06 00E2 R 00 90                 MOV     NDRV,0                   ;Initially no drives
0006  E8 0000 E                          CALL    HINIT                    ; init hard disk

0009  0E                                 PUSH    CS                       ; set buffer addr for read home track
000A  07                                 POP     ES
000B  BF 0000 E                          MOV     DI,OFFSET BUFFER
000E  E8 00E3 R                          CALL    RHOME                    ; read home
0011  22 C0                              AND     AL,AL                    ; check any error
0013  74 05                              JZ      DINIT2

0015  BB 0042 R                          MOV     BX,OFFSET HNFMT          ; print error message
0018  EB 0D                              JMP     SHORT DINITX

001A  E8 0000 E                 DINIT2:  CALL    HINIT                    ; re_init hard disk
001D  A0 00E2 R                          MOV     AL,NDRV                  ; get # of drives present
0020  0A C0                              OR      AL,AL                    ; see if any msdos partitions are here
0022  75 08                              JNZ     DINITY
0024  BB 0082 R                          MOV     BX,OFFSET NOMSDOS        ;IF NOT, TELL THE USER
0027  E8 0000 E                 DINITX:  CALL    PSTR
002A  33 C0                              XOR     AX,AX
002C  C3                        DINITY:  RET                              ;RETURN # OF DRIVES

                                ; DEVICE INIT ENTRY (HARD INIT ALREADY DONE, JUST RETURN INFO)

002D                            DSK_INIT:
002D  8A 26 00E2 R                       MOV     AH,NDRV
0031  2E: C5 1E 0000 E                   LDS     BX,CS:[PTRSAVE]
0036  88 67 0D                           MOV     BYTE PTR [BX.MEDIA],AH   ; # of drives for MSDOS

0039  C7 47 12 0000 E                    MOV     WORD PTR [BX.COUNT],OFFSET INI_TAB
003E  8C 4F 14                           MOV     WORD PTR [BX.COUNT+2],CS

0041  C3                                 RET

0042  0D 0A 57 41 52 4E         HNFMT    DB      CR,LF,'WARNING: The hard disk is not formatted '
      49 4E 47 3A 20 54
      68 65 20 68 61 72
      64 20 64 69 73 6B
      20 69 73 20 6E 6F
      74 20 66 6F 72 6D
      61 74 74 65 64 20
006C  6F 72 20 6E 6F 74                  DB      'or not partitioned!'
      20 70 61 72 74 69
      74 69 6F 6E 65 64
      21
007F  0D 0A 00                           DB      CR,LF,0

0082  0D 0A 54 68 65 72         NOMSDOS  DB      CR,LF,'There are no MS-DOS partitions on '
      65 20 61 72 65 20
      6E 6F 20 4D 53 2D
```

```
        44 4F 53 20 70 61
        72 74 69 74 69 6F
        6E 73 20 6F 6E 20
  00A6  74 68 65 20 68 61          DB      'the hard disk.',CR,LF,0
        72 64 20 64 69 73
        6B 2E 0D 0A 00

                                   SUBTTL  Winchester HOME block reader & DPE/DPD builder
```

```
                                    PAGE
                           ; must be the same order as the equates

  00B7  48 4F 4D 00        BM_HOM  DB      'HOM',0
  00BB  42 41 54 00        BM_BAT  DB      'BAT',0
  00BF  44 50 44 00        BM_DPD  DB      'DPD',0
  00C3  4F 53 4E 00        BM_OSN  DB      'OSN',0
  00C7  41 53 54 00        BM_AST  DB      'AST',0
  00CB  50 41 53 00        BM_PAS  DB      'PAS',0

  00CF                     HOMEER          LABEL   WORD            ; home track error word
  00CF  00                 HT_ERR_C        DB      0               ; home track error code
  00D0  00                 HT_ERR_ID       DB      0               ; home track error block ID

  00D1  0000               ST_AST          DW      0               ; start track # for AST
  00D3  00                 NT_AST          DB      0               ; # of tracks for AST
  00D4  0000               BAT_TN          DW      0               ; BAT track #
  00D6  00                 BAT_SN          DB      0               ; BAT sector #
  00D7  0000               BAT_LEN         DW      0               ; BAT length
  00D9  00                 BK_LBN          DB      0               ; logical block #
  00DA  0000               BK_MEC          DW      0               ; max entry count
  00DC  0000               BK_EC           DW      0               ; entry count

  00DE  0000               TEMP            DW      0
  00E0  0000               SAVE_SEC        DW      0
  00E2  00                 NDRV            DB      0               ; # of drivers
```

```
                                                  PAGE
                                        ;************************************************************
                                        ;
                                        ;         RHOME - Read 'HOM' block and other necessary blocks to build
                                        ;                 the logical partition and alternate sectors.
                                        ;                 More information can be obtained from V2 spec. DDS.MEM
                                        ;
                                        ;         This routine assumes that the RD controller board is installed.
                                        ;
                                        ;         ENTRY:  ES:DI = sector buffer start address.
                                        ;                           It must contain one sector memory (512 bytes).
                                        ;                           Since there are more than 1 sector read in this
                                        ;                           routine, the sector buffer start address, DI,
                                        ;                           will be saved in (TEMP).
                                        ;         EXIT:
                                        ;                 AX = home track error word
                                        ;                       (AL = error code, and AH = error block)
                                        ;                           AL = 0, no error
                                        ;                           AL = 1, sector read error (hard error)
                                        ;                           AL = 2, block ID not found
                                        ;                           AL = 3, block checksum error
                                        ;                           AL = 4, no partitions
                                        ;                       and
                                        ;                           AH = 0, unknown area
                                        ;                           AH = 1, 'HOM' block
                                        ;                           AH = 2, 'BAT' block
                                        ;                           AH = 3, 'DPD' block
                                        ;                           AH = 4, 'OSN' block
                                        ;                           AH = 5, 'AST' block
                                        ;                           AH = 6, 'PAS' block
                                        ;                           AH = 7, PRE-BOOT block
                                        ;                           AH = 8, BOOT block
                                        ;
                                        ;                 WORD HOMEER = AX, home track error word.
                                        ;                                     which can be examined later.
                                        ;                 WORD ST_AST = start track # for alternate sector area
                                        ;                 BYTE NT_AST = # of tracks reserved for alternate sector area
                                        ;
                                        ;************************************************************

00E3                                    RHOME:
00E3  FC                                          CLD                             ; set direction forward
00E4  89 3E 00DE R                                MOV     TEMP,DI                 ; save buffer start addr
00E8  33 C0                                       XOR     AX,AX
00EA  A3 00CF R                                   MOV     HOMEER,AX               ; init error word

00ED  B3 02                                       MOV     BL,2                    ; sector 2
00EF  33 D2                                       XOR     DX,DX                   ; track 0
00F1  B4 01                                       MOV     AH,BID_HOM              ; HOM ID
00F3  E8 0178 R                                   CALL    READINB                 ; read HOM block
00F6  75 42                                       JNZ     RHOM4                   ; abort if error

00F8  26: 8B 44 40                                MOV     AX,ES:NCYLD_OFF[SI]     ; get # of cylinders
00FC  48                                          DEC     AX
00FD  A3 0000 E                                   MOV     MAXTRK,AX               ; save max track #
```

```
0100  26: 8B 44 4A                                MOV     AX,ES:WPC_OFF[SI]       ; get write pre_comp value
0104  A3 0000 E                                   MOV     PRECOMP,AX              ; save pre_comp
0107  26: 8A 44 4C                                MOV     AL,ES:STEPR_OFF[SI]     ; get step rate value
010B  A2 0000 E                                   MOV     STEPRATE,AL             ; save step rate


010E  80 3E 00D9 R 00                             CMP     BK_LBN,0                ; check partitioned
0113  B8 0104                                     MOV     AX,104H                 ; assume no partitions
0116  75 22                                       JNZ     RHOM4

0118  26: 8B 54 17                                MOV     DX,WORD PTR ES:DPD_OFF[SI]      ; get DPD track number
011C  26: 8A 5C 19                                MOV     BL,BYTE PTR ES:DPD_OFF+2[SI]    ; get sector #
0120  26: 8B 4C 1A                                MOV     CX,WORD PTR ES:DPD_OFF+3[SI]    ; get length
0124  23 C9                                       AND     CX,CX                   ; check any entry?
0126  B8 0104                                     MOV     AX,104H                 ; assume no partitions
0129  74 0F                                       JZ      RHOM4

012B  B4 03                                       MOV     AH,BID_DPD              ; set block ID
012D  E8 0178 R                                   CALL    READINB                 ; read DPD block
0130  75 08                                       JNZ     RHOM4                   ; abort if error

0132  E8 013E R                                   CALL    BHPAR                   ; build hard disk partition
0135  E8 01F0 R                                   CALL    BHDRV                   ; build hard disk drive DPE & DPB
0138  33 C0                                       XOR     AX,AX                   ; no error
013A                                    RHOM4:
013A  A3 00CF R                                   MOV     HOMEER,AX               ; save home track error word
013D  C3                                          RET
```

Winchester HOME block reader & DPE/DPD builder

```
                                      PAGE
                              ;************************************************************
                              ;
                              ;       BHPAR - Build Hard disk PARtition first/last track #
                              ;
                              ;       ENTRY:  none
                              ;       EXIT:
                              ;
                              ;************************************************************

013E                          BHPAR:
013E  8B 36 00DE R                    MOV     SI,TEMP                 ; get buffer addr

0142  8B 0E 00DC R                    MOV     CX,BK_EC                ; get entry count
0146  23 C9                           AND     CX,CX                   ; any entry?
0148  74 30                           JZ      BHPAR4                  ; done if no entry
014A  83 C6 20                        ADD     SI,OFFSET DPD_START_OFF ; point to entry start addr
014D                          BHPAR1:
014D  26: 80 3C F0                    CMP     BYTE PTR ES:DPDE_FLAG[SI],DPDE_INIT     ; check initialized?
0151  75 22                           JNZ     BHPAR5                          ; skip if not
0153  26: 80 7C 0B 02                 CMP     BYTE PTR ES:DPDE_OST[SI],MSDOS  ; check this OS type?
0158  75 1B                           JNZ     BHPAR5                          ; skip this if wrong OS

015A  A0 00E2 R                       MOV     AL,NDRV                         ; get # of drives so far
015D  3C 04                           CMP     AL,MNPARTS                      ; check max
015F  74 19                           JZ      BHPAR4                          ; done if max drives reached
0161  FE 06 00E2 R                    INC     BYTE PTR NDRV                   ; update # of drives
0165  E8 0000 E                       CALL    GETPEA                          ; get partition entry addr

0168  26: 8B 44 0C                    MOV     AX,WORD PTR ES:DPDE_FTN[SI]     ; get first track #
016C  89 07                           MOV     WORD PTR PART_FTN[BX],AX        ; save to partition table
016E  26: 8B 44 0E                    MOV     AX,WORD PTR ES:DPDE_LTN[SI]     ; get last track #
0172  89 47 02                        MOV     WORD PTR PART_LTN[BX],AX        ; save to partition table
0175  83 C6 20                BHPAR5: ADD     SI,OFFSET DPDE_OFF              ; set to next DPD entry
0178  E2 D3                           LOOP    BHPAR1                          ; loop for all entries
017A  C3                      BHPAR4: RET
```

Winchester HOME block reader & DPE/DPD builder

```
                                      PAGE
                              ;************************************************************
                              ;
                              ;       READIN - This routine read in a Wini sector into the buffer area.
                              ;                It also checks the correct block ID and checksum.
                              ;                Some useful data are also being colected.
                              ;
                              ;       ENTRY:
                              ;               AH = block ID
                              ;               BL = sector number
                              ;               DX = logical track number
                              ;               ES:(TEMP) = sector buffer start addr
                              ;       EXIT:
                              ;               AX = error word defined in RHOME routine
                              ;               SI = (TEMP)
                              ;               Z flag is set if error
                              ;               BYTE BK_LBN = logical block number, byte 3
                              ;               WORD BK_MEC = max entry count, byte 6-7
                              ;               WORD BK_EC  = entry count, byte 8-9
                              ;
                              ;************************************************************

017B                          READINB:
017B  53                              PUSH    BX                      ; save sector #
017C  52                              PUSH    DX                      ; save track #
017D  E8 0187 R                       CALL    READIN
0180  5A                              POP     DX
0181  5B                              POP     BX
0182  75 01                           JNZ     READINB1
0184  C3                              RET

0185                          READINB1:
0185  42                              INC     DX                      ; try to read the alternate block
0186  42                              INC     DX
                              ;       JMP     READIN

0187                          READIN:
0187  88 26 00D0 R                    MOV     HT_ERR_ID,AH            ; save block ID
018B  E8 0000 E                       CALL    CLTPN                   ; convert to physical #
018E  8B 3E 00DE R                    MOV     DI,TEMP                 ; get buffer start addr
0192  B9 0001                         MOV     CX,1                    ; set sector count
0195  C6 06 0000 E 00                 MOV     BYTE PTR XLT_F,0        ; set physical flag
019A  E8 0000 E                       CALL    HREAD                   ; read sector
019D  B0 01                           MOV     AL,1                    ; assume hard error
019F  75 4A                           JNZ     READIN10

                              ;       check correct block ID

01A1  8B 3E 00DE R                    MOV     DI,TEMP                 ; get buffer addr
01A5  A0 00D0 R                       MOV     AL,HT_ERR_ID            ; get block ID
01A8  FE C8                           DEC     AL
01AA  98                              CBW
01AB  D1 E0                           SHL     AX,1                    ; *4
01AD  D1 E0                           SHL     AX,1
01AF  BE 00B7 R                       MOV     SI,OFFSET BM_HOM        ; get string start addr
```

```
01B2  03 F0                               ADD     SI,AX
01B4  E8 0251 R                           CALL    COMPS
01B7  B0 02                               MOV     AL,2                    ; assume ID not found
01B9  75 30                               JNZ     READIN10

                              ;           check correct check sum

01BB  8B 36 00DE R                        MOV     SI,TEMP                 ; get buffer start addr
01BF  B9 0100                             MOV     CX,512/2                ; get sector count
01C2  33 DB                               XOR     BX,BX                   ; init check sum
01C4                          READIN3:
01C4  26: AD                              LODS    ES:WORD PTR [SI]        ; get data
01C6  03 D8                               ADD     BX,AX                   ; add to check sum
01C8  E2 FA                               LOOP    READIN3                 ; loop for whole sector
01CA  23 DB                               AND     BX,BX                   ; checksum=0?
01CC  B0 03                               MOV     AL,3                    ; assume check sum error
01CE  75 1B                               JNZ     READIN10

                              ;           save some useful information

01D0  8B 36 00DE R                        MOV     SI,TEMP                 ; get buffer addr
01D4  26: 8A 44 03                        MOV     AL,BYTE PTR ES:BK_LBN_OFF[SI]   ; get logical block number
01D8  A2 00D9 R                           MOV     BK_LBN,AL               ; save it
01DB  26: 8B 44 06                        MOV     AX,WORD PTR ES:BK_MEC_OFF[SI]   ; get max entry count
01DF  A3 00DA R                           MOV     BK_MEC,AX               ; save it
01E2  26: 8B 44 08                        MOV     AX,WORD PTR ES:BK_EC_OFF[SI]    ; get entry count
01E6  A3 00DC R                           MOV     BK_EC,AX                ; save it
01E9  32 C0                               XOR     AL,AL                   ; no error
01EB                          READIN10:
01EB  8A 26 00D0 R                        MOV     AH,HT_ERR_ID            ; restore ID
01EF  C3                                  RET
```

```
                                          PAGE
                              ;*******************************************************
                              ;
                              ;           BHDRV - build hard disk logical drives
                              ;
                              ;           ENTRY:  (PARTITION) should contain first & last track
                              ;                           number for all drives. if any one of them
                              ;                           is not present, the first track number should
                              ;                           contain a 0.
                              ;           EXIT:   DPB are built
                              ;           USES:   ALL
                              ;
                              ;*******************************************************

01F0                          BHDRV:
01F0  BF 0000 E                           MOV     DI,OFFSET PARTITION
01F3  BE 0000 E                           MOV     SI,OFFSET HDSKO
01F6  B9 0004                             MOV     CX,MNPARTS              ; do all partitions
01F9                          BHDRV2:
01F9  51                                  PUSH    CX
01FA  8B 1D                               MOV     BX,PART_FTN[DI]         ; get first track #
01FG  8B 4D 02                            MOV     CX,PART_LTN[DI]         ; get last track #
01FF  23 DB                               AND     BX,BX                   ; check this partition present
0201  74 05                               JZ      BHDRV1                  ; skip if not

0203  56                                  PUSH    SI                      ; save DPE
0204  E8 0213 R                           CALL    BDPB                    ; build this DPE & DPB
0207  5E                                  POP     SI                      ; restore DPE
0208                          BHDRV1:
0208  81 C7 0000 E                        ADD     DI,PART_SIZE            ; set to next partition
020C  83 C6 13                            ADD     SI,DPB_OFF
020F  59                                  POP     CX                      ; restore partition count
0210  E2 E7                               LOOP    BHDRV2                  ; loop until all partitions

0212  C3                                  RET

                              ;*******************************************************
                              ;
                              ;           BDPB - Build Disk Parameter Block Fields
                              ;                   for Winchester logical disk drive
                              ;
                              ;           ETNRY:  BX = first track #
                              ;                   CX = last track #
                              ;                   SI = DPE addr
                              ;           EXIT:
                              ;
                              ;*******************************************************

0213                          BDPB:
0213  2B CB                               SUB     CX,BX
0215  41                                  INC     CX              ; total tracks
0216  D1 E1                               SHL     CX,1
0218  D1 E1                               SHL     CX,1
021A  D1 E1                               SHL     CX,1
021C  D1 E1                               SHL     CX,1            ; total sectors
```

```
0221E  89 4C 08                 MOV     WORD PTR SECTORS[SI],CX
0221   BD 0002                  MOV     BP,2                ; assume 1 sectors per FAT
0224                    BDPB1:
0224   8B C1                    MOV     AX,CX               ; get total sectors
0226   2D 0030                  SUB     AX,32+16            ; 2 reserved tracks+16 sectors for DIR
0229   2B C5                    SUB     AX,BP               ; - FAT = total sectors for data
022B   D1 E8                    SHR     AX,1                ; / 4 sectors per cluster
022D   D1 E8                    SHR     AX,1
022F   8B D8                    MOV     BX,AX               ; *1.5
0231   D1 E8                    SHR     AX,1                ; /2
0233   03 C3                    ADD     AX,BX
0235   05 0004                  ADD     AX,4                ; +3 and 1 may be overflow
0238   BB 0200                  MOV     BX,512
023B   33 D2                    XOR     DX,DX               ;THIS MAY FIX A BUG PIS
023D   F7 F3                    DIV     BX
023F   25 00FF                  AND     AX,OFFH             ; don't remove this instruction
0242   40                       INC     AX
0243   03 C0                    ADD     AX,AX               ; 2 copies of FAT
0245   3B C5                    CMP     AX,BP               ; enough sector for FAT
0247   8B E8                    MOV     BP,AX
0249   7C D9                    JL      BDPB1
024B   D1 E8                    SHR     AX,1                ; 1 FAT
024D   89 44 0B                 MOV     WORD PTR FATSEC[SI],AX
0250   C3                       RET
```

```
                                PAGE
                        ;***************************************************************
                        ;
                        ;       COMPS - compare string
                        ;
                        ;       ENTRY:  DS:SI = message to compare end with 0
                        ;               ES:DI = buffer addr
                        ;       EXIT:   Z FLAG is set if equal
                        ;       USES:   AL, SI, DI
                        ;
                        ;***************************************************************

0251                    COMPS:
0251                    COMPS2:
0251   AC                       LODSB                           ; get byte
0252   22 C0                    AND     AL,AL                   ; end of string?
0254   74 03                    JZ      COMPS1                  ;  done, if yes
0256   AE                       SCASB                           ; same byte?
0257   74 F8                    JZ      COMPS2                  ; if equal, keep going
0259                    COMPS1:
0259   C3                       RET
```

```
                                        PAGE
025A                        UP_BAT:
025A  8C C8                             MOV     AX,CS                   ;SETUP ES:DS:
025C  8E D8                             MOV     DS,AX
025E  8E C0                             MOV     ES,AX
0260  BF 0000 E                         MOV     DI,OFFSET BUFFER        ; set up working buffer ES:DI
0263  8B 16 02F2 R                      MOV     DX,TEMP_TRK             ; get bad track #
0267  8A 1E 02F4 R                      MOV     BL,TEMP_SEC             ;  and sector #
                            ;           JMP     WT_BAD_B

                            ;*************************************************************
                            ;
                            ;           WT_BAD_B - write bad bit into BAT
                            ;
                            ;           ENTRY:
                            ;                   DX = track #
                            ;                   BL = sector #
                            ;                   ES:DI = buffer
                            ;           NOTE: all BAT must stay in the same track
                            ;
                            ;*************************************************************

026B                        WT_BAD_B:
026B  89 1E 00E0 R                      MOV     SAVE_SEC,BX             ; save sector #
026F  89 3E 00DE R                      MOV     TEMP,DI                 ; save buffer addr
0273  8B C2                             MOV     AX,DX                   ; get bad track #
0275  8B 16 00D4 R                      MOV     DX,BAT_TN               ; get BAT track #
0279  8A 1E 00D6 R                      MOV     BL,BAT_SN               ; get BAT sector #
027D  8B 0E 00D7 R                      MOV     CX,BAT_LEN              ; get length
0281                        WT_BADB1:
0281  3D 00FA                           CMP     AX,250                  ; check in this block
0284  72 08                             JB      WT_BADB4                ;  by compare with entry size
0286  FE C3                             INC     BL                      ; set to next sector
0288  2D 00FA                           SUB     AX,250
028B  E2 F4                             LOOP    WT_BADB1
028D  C3                                RET                             ; can't find it, don't update BAT
028E                        WT_BADB4:
028E  52                                PUSH    DX                      ; save track #
028F  53                                PUSH    BX                      ; save sector #
0290  50                                PUSH    AX                      ; save track offset in BAT
0291  B4 02                             MOV     AH,BID_BAT              ; set ID
0293  E8 017B R                         CALL    READINB                 ; read the sector
0296  58                                POP     AX                      ; restore offset
0297  75 56                             JNZ     WT_BADB5                ; if error don't update BAT

0299  8B 3E 00DE R                      MOV     DI,TEMP                 ; get buffer addr
029D  8B F7                             MOV     SI,DI                   ; get buffer addr
029F  D1 E0                             SHL     AX,1                    ; word offset
02A1  03 F0                             ADD     SI,AX
02A3  8B 0E 00E0 R                      MOV     CX,SAVE_SEC             ; get sector #
02A7  49                                DEC     CX                      ; base 0
02A8  B8 0001                           MOV     AX,1                    ; set 1 bit
02AB  D3 E0                             SHL     AX,CL                   ; shift into position
02AD  26: 09 44 0C                      OR      ES:12[SI],AX            ; set the bit
02B1  26: C7 45 04 0000                 MOV     ES:WORD PTR 4[DI],0     ; clear checksum
```

```
02B7  33 DB                             XOR     BX,BX                   ; init new checksum
02B9  8B F7                             MOV     SI,DI                   ; get buffer offset
02BB  B9 0100                           MOV     CX,512/2                ; # of words in buffer
02BE                        WT_BCLP:
02BE  26: AD                            LODS    ES:WORD PTR [SI]        ; get a word
02C0  03 D8                             ADD     BX,AX                   ; accumulate result
02C2  E2 FA                             LOOP    WT_BCLP                 ; do'em all

02C4  F7 DB                             NEG     BX                      ; compliment
02C6  26: 89 5D 04                      MOV     ES:4[DI],BX             ; store new checksum
02CA  5B                                POP     BX                      ; restore sector
02CB  5A                                POP     DX                      ;  and track
02CC  B9 0001                           MOV     CX,1                    ; write 1 sector
02CF  53                                PUSH    BX                      ; Save for 2nd write
02D0  52                                PUSH    DX
02D1  57                                PUSH    DI
02D2  51                                PUSH    CX
02D3  E8 0000 E                         CALL    CLTPN                   ; convert to physical #
02D6  C6 06 0000 E 00                   MOV     BYTE PTR XLT_F,0        ; set translate flag
02DB  E8 0000 E                         CALL    HWRITE                  ; update the bad bit
02DE  59                                POP     CX                      ; restore write params
02DF  5F                                POP     DI
02E0  5A                                POP     DX
02E1  5B                                POP     BX
02E2  42                                INC     DX                      ; Step to 2nd copy of BAT
02E3  42                                INC     DX
02E4  E8 0000 E                         CALL    CLTPN                   ; convert to physical #
02E7  C6 06 0000 E 00                   MOV     BYTE PTR XLT_F,0        ; set translate flag
02EC  E9 0000 E                         JMP     HWRITE                  ; update the bad bit and RETURN

02EF                        WT_BADB5:
02EF  58                                POP     AX
02F0  58                                POP     AX
02F1  C3                                RET                             ; clean stack and exit

02F2  0000                  TEMP_TRK    DW      0
02F4  00                    TEMP_SEC    DB      0

02F5                        CODE        ENDS
                                        END
```

```
Structures and records:

                N a m e                 Width   # fields
                                        Shift   Width    Mask     Initial

DBP. . . . . . . . . . . . . . . .      0013    0008
  SECSZ. . . . . . . . . . . . . .      0000
  ALLOC. . . . . . . . . . . . . .      0002
  RESSEC . . . . . . . . . . . . .      0003
  FATS . . . . . . . . . . . . . .      0005
  MAXDIR . . . . . . . . . . . . .      0006
  SECTORS. . . . . . . . . . . . .      0008
  MEDIAID. . . . . . . . . . . . .      000A
  FATSEC . . . . . . . . . . . . .      000B
  SECTRK . . . . . . . . . . . . .      000D
  HEADS. . . . . . . . . . . . . .      000F
  HIDDEN . . . . . . . . . . . . .      0011
IODAT. . . . . . . . . . . . . . .      0016    0009
  CMDLEN . . . . . . . . . . . . .      0000
  UNIT . . . . . . . . . . . . . .      0001
  CMD. . . . . . . . . . . . . . .      0002
  STATUSW. . . . . . . . . . . . .      0003
  MEDIA. . . . . . . . . . . . . .      000D
  TRANS. . . . . . . . . . . . . .      000E
  COUNT. . . . . . . . . . . . . .      0012
  START. . . . . . . . . . . . . .      0014

Segments and groups:

                N a m e                 Size    align    combine class

CGROUP . . . . . . . . . . . . . .      GROUP
  CODE . . . . . . . . . . . . . .      02F5    BYTE     PUBLIC  'CODE'

Symbols:

                N a m e                 Type    Value    Attr

ABRTC. . . . . . . . . . . . . . .      Number  0004
AST_OFF. . . . . . . . . . . . . .      Number  0026
ATRETRY. . . . . . . . . . . . . .      Number  0000
BADBD. . . . . . . . . . . . . . .      Number  0080
BAT_LEN. . . . . . . . . . . . . .      L WORD  00D7     CODE
BAT_OFF. . . . . . . . . . . . . .      Number  0012
BAT_SN . . . . . . . . . . . . . .      L BYTE  00D6     CODE
BAT_TN . . . . . . . . . . . . . .      L WORD  00D4     CODE
BDPB . . . . . . . . . . . . . . .      L NEAR  0213     CODE
BDPB1. . . . . . . . . . . . . . .      L NEAR  0224     CODE
BHDRV. . . . . . . . . . . . . . .      L NEAR  01F0     CODE
BHDRV1 . . . . . . . . . . . . . .      L NEAR  0208     CODE
BHDRV2 . . . . . . . . . . . . . .      L NEAR  01F9     CODE
BHPAR. . . . . . . . . . . . . . .      L NEAR  013E     CODE
BHPAR1 . . . . . . . . . . . . . .      L NEAR  014D     CODE
BHPAR4 . . . . . . . . . . . . . .      L NEAR  017A     CODE
BHPAR5 . . . . . . . . . . . . . .      L NEAR  0175     CODE
BID_AST. . . . . . . . . . . . . .      Number  0005
```

```
BID_BAT. . . . . . . . . . . . . .      Number  0002
BID_BOT. . . . . . . . . . . . . .      Number  0008
BID_DPD. . . . . . . . . . . . . .      Number  0003
BID_HOM. . . . . . . . . . . . . .      Number  0001
BID_OSN. . . . . . . . . . . . . .      Number  0004
BID_PAS. . . . . . . . . . . . . .      Number  0006
BID_PBT. . . . . . . . . . . . . .      Number  0007
BID_UKN. . . . . . . . . . . . . .      Number  0000
BIT0 . . . . . . . . . . . . . . .      Number  0001
BIT1 . . . . . . . . . . . . . . .      Number  0002
BIT15. . . . . . . . . . . . . . .      Number  8000
BIT2 . . . . . . . . . . . . . . .      Number  0004
BIT3 . . . . . . . . . . . . . . .      Number  0008
BIT4 . . . . . . . . . . . . . . .      Number  0010
BIT5 . . . . . . . . . . . . . . .      Number  0020
BIT6 . . . . . . . . . . . . . . .      Number  0040
BIT7 . . . . . . . . . . . . . . .      Number  0080
BK_EC. . . . . . . . . . . . . . .      L WORD  00DC     CODE
BK_EC_OFF. . . . . . . . . . . . .      Number  0008
BK_LBN . . . . . . . . . . . . . .      L BYTE  00D9     CODE
BK_LBN_OFF . . . . . . . . . . . .      Number  0003
BK_MEC . . . . . . . . . . . . . .      L WORD  00DA     CODE
BK_MEC_OFF . . . . . . . . . . . .      Number  0006
BM_AST . . . . . . . . . . . . . .      L BYTE  00C7     CODE
BM_BAT . . . . . . . . . . . . . .      L BYTE  00BB     CODE
BM_DPD . . . . . . . . . . . . . .      L BYTE  00BF     CODE
BM_HOM . . . . . . . . . . . . . .      L BYTE  00B7     CODE
BM_OSN . . . . . . . . . . . . . .      L BYTE  00C3     CODE
BM_PAS . . . . . . . . . . . . . .      L BYTE  00CB     CODE
BOOT_OFF . . . . . . . . . . . . .      Number  0021
BUFFER . . . . . . . . . . . . . .      V WORD  0000     CODE     External
CBSY . . . . . . . . . . . . . . .      Number  0080
CBSY2. . . . . . . . . . . . . . .      Number  0001
CLRIL. . . . . . . . . . . . . . .      Number  0004
CLRSPL . . . . . . . . . . . . . .      Number  0008
CLTPN. . . . . . . . . . . . . . .      L NEAR  0000     CODE     External
CMDIP. . . . . . . . . . . . . . .      Number  0002
COMPS. . . . . . . . . . . . . . .      L NEAR  0251     CODE
COMPS1 . . . . . . . . . . . . . .      L NEAR  0259     CODE
COMPS2 . . . . . . . . . . . . . .      L NEAR  0251     CODE
CPM8680. . . . . . . . . . . . . .      Number  0001
CR . . . . . . . . . . . . . . . .      Number  000D
CRCER. . . . . . . . . . . . . . .      Number  0040
CTRL_Q . . . . . . . . . . . . . .      Number  0011
CTRL_S . . . . . . . . . . . . . .      Number  0013
CTRL_Z . . . . . . . . . . . . . .      Number  001A
CYLDH. . . . . . . . . . . . . . .      Number  0065
CYLDL. . . . . . . . . . . . . . .      Number  0064
DAMNF. . . . . . . . . . . . . . .      Number  0001
DATA . . . . . . . . . . . . . . .      Number  0060
DATARQ . . . . . . . . . . . . . .      Number  0008
DINIT2 . . . . . . . . . . . . . .      L NEAR  001A     CODE
DINITX . . . . . . . . . . . . . .      L NEAR  0027     CODE
DINITY . . . . . . . . . . . . . .      L NEAR  002C     CODE
DISKDF . . . . . . . . . . . . . .      Number  0005
```

```
DPB_OFF. . . . . . . . . . . . .        Number   0013
DPDE_FLAG. . . . . . . . . . . .        Number   0000
DPDE_FTN . . . . . . . . . . . .        Number   000C
DPDE_INIT. . . . . . . . . . . .        Number   00F0
DPDE_LTN . . . . . . . . . . . .        Number   000E
DPDE_LU. . . . . . . . . . . . .        Number   0001
DPDE_OFF . . . . . . . . . . . .        Number   0020
DPDE_OST . . . . . . . . . . . .        Number   000B
DPDE_PN. . . . . . . . . . . . .        Number   0002
DPDE_PON . . . . . . . . . . . .        Number   000A
DPD_OFF. . . . . . . . . . . . .        Number   0017
DPD_START_OFF. . . . . . . . . .        Number   0020
DRDY . . . . . . . . . . . . . .        Number   0040
DRDY2. . . . . . . . . . . . . .        Number   0040
DRIVE0 . . . . . . . . . . . . .        Number   0000
DRIVE1 . . . . . . . . . . . . .        Number   0008
DRIVE2 . . . . . . . . . . . . .        Number   0010
DRIVE3 . . . . . . . . . . . . .        Number   0018
DRIVEDF. . . . . . . . . . . . .        Number   0003
DRIVEM . . . . . . . . . . . . .        Number   0018
DRVSEL . . . . . . . . . . . . .        Number   0001
DSK_INIT . . . . . . . . . . . .        L NEAR   002D      CODE      Global
DSTAT0 . . . . . . . . . . . . .        Number   0068
DSTAT1 . . . . . . . . . . . . .        Number   0069
DWRTF. . . . . . . . . . . . . .        Number   0020
DWRTF2 . . . . . . . . . . . . .        Numbor   0020
ERRF . . . . . . . . . . . . . .        Number   0001
ERROR. . . . . . . . . . . . . .        Number   0061
FALSE. . . . . . . . . . . . . .        Number   0000
FMTTRK . . . . . . . . . . . . .        Number   0050
GETFTN . . . . . . . . . . . . .        L NEAR   0000      CODE      External
GETPEA . . . . . . . . . . . . .        L NEAR   0000      CODE      External
HDIR . . . . . . . . . . . . . .        Number   0008
HDISK. . . . . . . . . . . . . .        Number   0004
HDSK0. . . . . . . . . . . . . .        V BYTE   0000      CODE      External
HD_INIT. . . . . . . . . . . . .        L NEAR   0000      CODE      Global
HEAD0. . . . . . . . . . . . . .        Number   0000
HEAD02 . . . . . . . . . . . . .        Number   0000
HEAD1. . . . . . . . . . . . . .        Number   0001
HEAD12 . . . . . . . . . . . . .        Number   0002
HEAD2. . . . . . . . . . . . . .        Number   0002
HEAD22 . . . . . . . . . . . . .        Number   0004
HEAD3. . . . . . . . . . . . . .        Number   0003
HEAD32 . . . . . . . . . . . . .        Number   0006
HEADM. . . . . . . . . . . . . .        Number   0007
HEADM2 . . . . . . . . . . . . .        Number   000E
HFORMAT. . . . . . . . . . . . .        L NEAR   0000      CODE      External
HINIT. . . . . . . . . . . . . .        L NEAR   0000      CODE      External
HIVEC_A. . . . . . . . . . . . .        Number   0114
HIVEC_B. . . . . . . . . . . . .        Number   0294
HMCHK. . . . . . . . . . . . . .        L NEAR   0000      CODE      External
HNFMT. . . . . . . . . . . . . .        L BYTE   0042      CODE      Global
HOMEER . . . . . . . . . . . . .        L WORD   00CF      CODE
HREAD. . . . . . . . . . . . . .        L NEAR   0000      CODE      External
HT_ERR_C . . . . . . . . . . . .        L BYTE   00CF      CODE
```

```
HT_ERR_ID. . . . . . . . . . . .        L BYTE   00D0      CODE
HVDISK . . . . . . . . . . . . .        L NEAR   0000      CODE      External
HWRITE . . . . . . . . . . . . .        L NEAR   0000      CODE      External
HWRITEV. . . . . . . . . . . . .        L NEAR   0000      CODE      External
IDBIT. . . . . . . . . . . . . .        Number   00E0
IDBITV . . . . . . . . . . . . .        Number   00A0
IDNF . . . . . . . . . . . . . .        Number   0010
INI_TAB. . . . . . . . . . . . .        V WORD   0000      CODE      External
INTHDL . . . . . . . . . . . . .        L NEAR   0000      CODE      External
LF . . . . . . . . . . . . . . .        Number   000A
LINDEX . . . . . . . . . . . . .        Number   0002
LSTEPP . . . . . . . . . . . . .        Number   0004
MAXTRK . . . . . . . . . . . . .        V WORD   0000      CODE      External
MNPARTS. . . . . . . . . . . . .        Number   0004
MSCTF. . . . . . . . . . . . . .        Number   0004
MSDOS. . . . . . . . . . . . . .        Number   0002
NCYLD_OFF. . . . . . . . . . . .        Number   0040
NDRV . . . . . . . . . . . . . .        L BYTE   00E2      CODE      Global
NOMSDOS. . . . . . . . . . . . .        L BYTE   0082      CODE
NT_AST . . . . . . . . . . . . .        L BYTE   00D3      CODE
NT_AST_OFF . . . . . . . . . . .        Number   002D
NUMHEAD. . . . . . . . . . . . .        Number   0004
OSN_OFF. . . . . . . . . . . . .        Number   001C
PARTITION. . . . . . . . . . . .        V WORD   0000      CODE      External
PART_FTN . . . . . . . . . . . .        Number   0000
PART_LTN . . . . . . . . . . . .        Number   0002
PART_PAS . . . . . . . . . . . .        Number   0004
PART_SIZE. . . . . . . . . . . .        Number   - 0000              External
PASE_OFF . . . . . . . . . . . .        Number   0005
PAS_ASN. . . . . . . . . . . . .        Number   0004
PAS_ATN. . . . . . . . . . . . .        Number   0003
PAS_BSN. . . . . . . . . . . . .        Number   0002
PAS_BTN. . . . . . . . . . . . .        Number   0000
PRECOMP. . . . . . . . . . . . .        V WORD   0000      CODE      External
PRE_READ . . . . . . . . . . . .        Alias    FALSE
PSTR . . . . . . . . . . . . . .        L NEAR   0000      CODE      External
PTRSAVE. . . . . . . . . . . . .        V DWORD  0000      CODE      External
RDBASE . . . . . . . . . . . . .        Number   0060
RDCMD. . . . . . . . . . . . . .        Number   0067
RDCMD2 . . . . . . . . . . . . .        Number   0068
RDINTF . . . . . . . . . . . . .        Number   0008
RDREAD . . . . . . . . . . . . .        Number   0020
RDSTAT . . . . . . . . . . . . .        Number   0067
RDWRITE. . . . . . . . . . . . .        Number   0030
READIN . . . . . . . . . . . . .        L NEAR   0187      CODE
READIN10 . . . . . . . . . . . .        L NEAR   01EB      CODE
READIN3. . . . . . . . . . . . .        L NEAR   01C4      CODE
READINB. . . . . . . . . . . . .        L NEAR   017B      CODE
READINB1 . . . . . . . . . . . .        L NEAR   0185      CODE
RESTOR . . . . . . . . . . . . .        Number   0010
RHOM4. . . . . . . . . . . . . .        L NEAR   013A      CODE
RHOME. . . . . . . . . . . . . .        L NEAR   00E3      CODE      Global
SAVE_SEC . . . . . . . . . . . .        L WORD   00E0      CODE
SBUFR. . . . . . . . . . . . . .        Number   0001
SCANID . . . . . . . . . . . . .        Number   0040
```

```
 The Microsoft MACRO Assembler               02-20-84      PAGE     Symbols-5
Hard Disk Initialization


SCTEXT . . . . . . . . . . . . . .           Number   0080
SDH. . . . . . . . . . . . . . . .           Number   0066
SECPTRK. . . . . . . . . . . . . .           Number   0010
SECSIZE. . . . . . . . . . . . . .           Number   0200
SECTC. . . . . . . . . . . . . . .           Number   0062
SECTN. . . . . . . . . . . . . . .           Number   0063
SECTSZ . . . . . . . . . . . . . .           Number   0080
SEEK . . . . . . . . . . . . . . .           Number   0070
SEEKC. . . . . . . . . . . . . . .           Number   0010
SEEKC2 . . . . . . . . . . . . . .           Number   0010
SERR . . . . . . . . . . . . . . .           Number   000F
SINIT. . . . . . . . . . . . . . .           Number   0002
SSCTF. . . . . . . . . . . . . . .           Number   0000
SSZ128 . . . . . . . . . . . . . .           Number   0080
SSZ1K. . . . . . . . . . . . . . .           Number   0040
SSZ256 . . . . . . . . . . . . . .           Number   0000
SSZ512 . . . . . . . . . . . . . .           Number   0020
STEPR0 . . . . . . . . . . . . . .           Number   0000
STEPR1 . . . . . . . . . . . . . .           Number   0001
STEPR2 . . . . . . . . . . . . . .           Number   0002
STEPR3 . . . . . . . . . . . . . .           Number   0003
STEPR4 . . . . . . . . . . . . . .           Number   0004
STEPR5 . . . . . . . . . . . . . .           Number   0005
STEPR6 . . . . . . . . . . . . . .           Number   0006
STEPR7 . . . . . . . . . . . . . .           Number   0007
STEPR8 . . . . . . . . . . . . . .           Number   0008
STEPR9 . . . . . . . . . . . . . .           Number   0009
STEPRA . . . . . . . . . . . . . .           Number   000A
STEPRATE . . . . . . . . . . . . .           V BYTE   0000      CODE     External
STEPRB . . . . . . . . . . . . . .           Number   000B
STEPRC . . . . . . . . . . . . . .           Number   000C
STEPRD . . . . . . . . . . . . . .           Number   000D
STEPRE . . . . . . . . . . . . . .           Number   000E
STEPRF . . . . . . . . . . . . . .           Number   000F
STEPR_OFF. . . . . . . . . . . . .           Number   004C
ST_AST . . . . . . . . . . . . . .           L WORD   00D1      CODE
ST_AST_OFF . . . . . . . . . . . .           Number   002B
TEMP . . . . . . . . . . . . . . .           L WORD   00DE      CODE
TEMP_SEC . . . . . . . . . . . . .           L BYTE   02F4      CODE     Global
TEMP_TRK . . . . . . . . . . . . .           L WORD   02F2      CODE     Global
TRK02. . . . . . . . . . . . . . .           Number   0080
TRK0E. . . . . . . . . . . . . . .           Number   0002
TRUE . . . . . . . . . . . . . . .           Number   - 0001
UP_BAT . . . . . . . . . . . . . .           L NEAR   025A      CODE     Global
WGATE. . . . . . . . . . . . . . .           Number   0010
WINI_OFFSET. . . . . . . . . . . .           Number   0094
WINI_SEG . . . . . . . . . . . . .           Number   0096
WPC_OFF. . . . . . . . . . . . . .           Number   004A
WPRCMP . . . . . . . . . . . . . .           Number   0061
WPRSNT . . . . . . . . . . . . . .           Number   0001
WTVERR . . . . . . . . . . . . . .           Number   000D
WT_BADB1 . . . . . . . . . . . . .           L NEAR   0281      CODE
WT_BADB4 . . . . . . . . . . . . .           L NEAR   028E      CODE
WT_BADB5 . . . . . . . . . . . . .           L NEAR   02EF      CODE
WT_BAD_B . . . . . . . . . . . . .           L NEAR   026B      CODE
```

```
 The Microsoft MACRO Assembler               02-20-84      PAGE     Symbols-6
Hard Disk Initialization


WT_BCLP. . . . . . . . . . . . . .           L NEAR   02BE      CODE
XBB_BO . . . . . . . . . . . . . .           Number   0008
XBB_BS . . . . . . . . . . . . . .           Number   000A
XBB_CN . . . . . . . . . . . . . .           Number   0004
XBB_DISKF. . . . . . . . . . . . .           Number   0000
XBB_DSN. . . . . . . . . . . . . .           Number   0003
XBB_ERRC . . . . . . . . . . . . .           Number   000C
XBB_LDN. . . . . . . . . . . . . .           Number   0001
XBB_SC . . . . . . . . . . . . . .           Number   0006
XBB_SN . . . . . . . . . . . . . .           Number   0002
XCPRSNT. . . . . . . . . . . . . .           Number   0002
XLT_F. . . . . . . . . . . . . . .           V BYTE   0000      CODE     External
XOPTION. . . . . . . . . . . . . .           V BYTE   0000      CODE     External

Warning Severe
Errors  Errors
0       0
```

```
 Hard Disk Initialization

 Symbol Cross Reference                      (# is definition)      Cref-1

ABRTC. . . . . . . . . . . . . . . .    29#
ALLOC. . . . . . . . . . . . . . . .    58#
AST_OFF. . . . . . . . . . . . . . .    29#
ATRETRY. . . . . . . . . . . . . . .    29#

BADBD. . . . . . . . . . . . . . . .    29#
BAT_LEN. . . . . . . . . . . . . . .   158#   499
BAT_OFF. . . . . . . . . . . . . . .    29#
BAT_SN . . . . . . . . . . . . . . .   157#   498
BAT_TN . . . . . . . . . . . . . . .   156#   497
BDPB . . . . . . . . . . . . . . . .   396    418#
BDPB1. . . . . . . . . . . . . . . .   427#   445
BHDRV. . . . . . . . . . . . . . . .   243    384#
BHDRV1 . . . . . . . . . . . . . . .   393    398#
BHDRV2 . . . . . . . . . . . . . . .   388#   402
BHPAR. . . . . . . . . . . . . . . .   242    258#
BHPAR1 . . . . . . . . . . . . . . .   285#   282
BHPAR4 . . . . . . . . . . . . . . .   263    273    283#
BHPAR5 . . . . . . . . . . . . . . .   267    269    281#
BID_AST. . . . . . . . . . . . . . .    29#
BID_BAT. . . . . . . . . . . . . . .    29#   511
BID_BOT. . . . . . . . . . . . . . .    29#
BID_DPD. . . . . . . . . . . . . . .    29#   238
BID_HOM. . . . . . . . . . . . . . .    29#   214
BID_OSN. . . . . . . . . . . . . . .    29#
BID_PAS. . . . . . . . . . . . . . .    29#
BID_PBT. . . . . . . . . . . . . . .    29#
BID_UKN. . . . . . . . . . . . . . .    29#
BIT0 . . . . . . . . . . . . . . . .    29#
BIT1 . . . . . . . . . . . . . . . .    29#
BIT15. . . . . . . . . . . . . . . .    29#
BIT2 . . . . . . . . . . . . . . . .    29#
BIT3 . . . . . . . . . . . . . . . .    29#
BIT4 . . . . . . . . . . . . . . . .    29#
BIT5 . . . . . . . . . . . . . . . .    29#
BIT6 . . . . . . . . . . . . . . . .    29#
BIT7 . . . . . . . . . . . . . . . .    29#
BK_EC. . . . . . . . . . . . . . . .   161#   261    365
BK_EC_OFF. . . . . . . . . . . . . .    29#   364
BK_LBN . . . . . . . . . . . . . . .   159#   227    361
BK_LBN_OFF . . . . . . . . . . . . .    29#   360
BK_MEC . . . . . . . . . . . . . . .   160#   363
BK_MEC_OFF . . . . . . . . . . . . .    29#   362
BM_AST . . . . . . . . . . . . . . .   147#
BM_BAT . . . . . . . . . . . . . . .   144#
BM_DPD . . . . . . . . . . . . . . .   145#
BM_HOM . . . . . . . . . . . . . . .   143#   338
BM_OSN . . . . . . . . . . . . . . .   146#
BM_PAS . . . . . . . . . . . . . . .   148#
BOOT_OFF . . . . . . . . . . . . . .    29#
BUFFER . . . . . . . . . . . . . . .    23#    87    476

CBSY . . . . . . . . . . . . . . . .    29#
CBSY2. . . . . . . . . . . . . . . .    29#
CGROUP . . . . . . . . . . . . . . .    12      14     14     14     14
CLRIL. . . . . . . . . . . . . . . .    29#
CLRSPL . . . . . . . . . . . . . . .    29#
```

```
 Hard Disk Initialization

 Symbol Cross Reference                      (# is definition)      Cref-2

CLTPN. . . . . . . . . . . . . . . .    21#   322    543    552
CMD. . . . . . . . . . . . . . . . .    39#
CMDIP. . . . . . . . . . . . . . . .    29#
CMDLEN . . . . . . . . . . . . . . .    37#
CODE . . . . . . . . . . . . . . . .    12     13#     13    564
COMPS. . . . . . . . . . . . . . . .   340    461#
COMPS1 . . . . . . . . . . . . . . .   465    468#
COMPS2 . . . . . . . . . . . . . . .   462#   467
COUNT. . . . . . . . . . . . . . . .    47#   111    112
CPM8680. . . . . . . . . . . . . . .    29#
CR . . . . . . . . . . . . . . . . .    29#   116    127    129    137
CRCER. . . . . . . . . . . . . . . .    29#
CTRL_Q . . . . . . . . . . . . . . .    29#
CTRL_S . . . . . . . . . . . . . . .    29#
CTRL_Z . . . . . . . . . . . . . . .    29#
CYLDH. . . . . . . . . . . . . . . .    29#
CYLDL. . . . . . . . . . . . . . . .    29#

DAMNF. . . . . . . . . . . . . . . .    29#
DATA . . . . . . . . . . . . . . . .    29#
DATARQ . . . . . . . . . . . . . . .    29#
DBP. . . . . . . . . . . . . . . . .    53#    72
DINIT2 . . . . . . . . . . . . . . .    90     95#
DINITX . . . . . . . . . . . . . . .    83    100#
DINITY . . . . . . . . . . . . . . .    98    102#
DISKDF . . . . . . . . . . . . . . .    29#
DPB_OFF. . . . . . . . . . . . . . .    74#   400
DPDE_FLAG. . . . . . . . . . . . . .    29#   266
DPDE_FTN . . . . . . . . . . . . . .    29#   277
DPDE_INIT. . . . . . . . . . . . . .    29#   266
DPDE_LTN . . . . . . . . . . . . . .    29#   279
DPDE_LU. . . . . . . . . . . . . . .    29#
DPDE_OFF . . . . . . . . . . . . . .    29#   281
DPDE_OST . . . . . . . . . . . . . .    29#   268
DPDE_PN. . . . . . . . . . . . . . .    29#
DPDE_PON . . . . . . . . . . . . . .    29#
DPD_OFF. . . . . . . . . . . . . . .    29#   231    232    233
DPD_START_OFF. . . . . . . . . . . .    29#   264
DRDY . . . . . . . . . . . . . . . .    29#
DRDY2. . . . . . . . . . . . . . . .    29#
DRIVE0 . . . . . . . . . . . . . . .    29#
DRIVE1 . . . . . . . . . . . . . . .    29#
DRIVE2 . . . . . . . . . . . . . . .    29#
DRIVE3 . . . . . . . . . . . . . . .    29#
DRIVEDF. . . . . . . . . . . . . . .    29#
DRIVEM . . . . . . . . . . . . . . .    29#
DRVSEL . . . . . . . . . . . . . . .    29#
DSK_INIT . . . . . . . . . . . . . .    26    106#
DSTAT0 . . . . . . . . . . . . . . .    29#
DSTAT1 . . . . . . . . . . . . . . .    29#
DWRTF. . . . . . . . . . . . . . . .    29#
DWRTF2 . . . . . . . . . . . . . . .    29#

ERRF . . . . . . . . . . . . . . . .    29#
ERROR. . . . . . . . . . . . . . . .    29#

FALSE. . . . . . . . . . . . . . . .    29#    29
```

```
 Hard Disk Initialization

 Symbol Cross Reference                    (# is definition)        Cref-3

FATS . . . . . . . . . . . . . . .    60#
FATSEC . . . . . . . . . . . . . .    64#   447
FMTTRK . . . . . . . . . . . . . .    29#

GETFTN . . . . . . . . . . . . . .    21#
GETPEA . . . . . . . . . . . . . .    21#   275

HDIR . . . . . . . . . . . . . . .    29#
HDISK. . . . . . . . . . . . . . .    29#
HDSKO. . . . . . . . . . . . . . .    19#   386
HD_INIT. . . . . . . . . . . . . .    27     79      81#
HEADO. . . . . . . . . . . . . . .    29#
HEADO2 . . . . . . . . . . . . . .    29#
HEAD1. . . . . . . . . . . . . . .    29#
HEAD12 . . . . . . . . . . . . . .    29#
HEAD2. . . . . . . . . . . . . . .    29#
HEAD22 . . . . . . . . . . . . . .    29#
HEAD3. . . . . . . . . . . . . . .    29#
HEAD32 . . . . . . . . . . . . . .    29#
HEADM. . . . . . . . . . . . . . .    29#
HEADM2 . . . . . . . . . . . . . .    29#
HEADS. . . . . . . . . . . . . . .    69#
HFORMAT. . . . . . . . . . . . . .    17#
HIDDEN . . . . . . . . . . . . . .    70#
HINIT. . . . . . . . . . . . . . .    17#    83      95
HIVEC_A. . . . . . . . . . . . . .    29#
HIVEC_B. . . . . . . . . . . . . .    29#
HMCHK. . . . . . . . . . . . . . .    17#
HNFMT. . . . . . . . . . . . . . .    26     92     116#
HOMEER . . . . . . . . . . . . . .   150#   210     246
HREAD. . . . . . . . . . . . . . .    16#   326
HT_ERR_C . . . . . . . . . . . . .   151#
HT_ERR_ID. . . . . . . . . . . . .   152#   321     333     368
HVDISK . . . . . . . . . . . . . .    18#
HWRITE . . . . . . . . . . . . . .    16#   545     554
HWRITEV. . . . . . . . . . . . . .    16#

IDBIT. . . . . . . . . . . . . . .    29#
IDBITV . . . . . . . . . . . . . .    29#
IDNF . . . . . . . . . . . . . . .    29#
INI_TAB. . . . . . . . . . . . . .    19#   111
INTHDL . . . . . . . . . . . . . .    18#
IODAT. . . . . . . . . . . . . . .    36#    49

LF . . . . . . . . . . . . . . . .    29#   116     127     129     137
LINDEX . . . . . . . . . . . . . .    29#
LSTEPP . . . . . . . . . . . . . .    29#

MAXDIR . . . . . . . . . . . . . .    61#
MAXTRK . . . . . . . . . . . . . .    22#   220
MEDIA. . . . . . . . . . . . . . .    45#   109
MEDIAID. . . . . . . . . . . . . .    63#
MNPARTS. . . . . . . . . . . . . .    31#   272     387
MSCTF. . . . . . . . . . . . . . .    29#
MSDOS. . . . . . . . . . . . . . .    29#   268

NCYLD_OFF. . . . . . . . . . . . .    29#   218
```

```
 Hard Disk Initialization

 Symbol Cross Reference                    (# is definition)        Cref-4

NDRV . . . . . . . . . . . . . . .    26     82      96     107     165#    271     274
NOMSDOS. . . . . . . . . . . . . .    99    129#
NT_AST . . . . . . . . . . . . . .   155#
NT_AST_OFF . . . . . . . . . . . .    29#
NUMHEAD. . . . . . . . . . . . . .    29#

OSN_OFF. . . . . . . . . . . . . .    29#

PARTITION. . . . . . . . . . . . .    20#   385
PART_FTN . . . . . . . . . . . . .    29#   278     390
PART_LTN . . . . . . . . . . . . .    29#   280     391
PART_PAS . . . . . . . . . . . . .    29#
PART_SIZE. . . . . . . . . . . . .    23#   399
PASE_OFF . . . . . . . . . . . . .    29#
PAS_ASN. . . . . . . . . . . . . .    29#
PAS_ATN. . . . . . . . . . . . . .    29#
PAS_BSN. . . . . . . . . . . . . .    29#
PAS_BTN. . . . . . . . . . . . . .    29#
PRECOMP. . . . . . . . . . . . . .    22#   222
PRE_READ . . . . . . . . . . . . .    29#
PSTR . . . . . . . . . . . . . . .    18#   100
PTRSAVE. . . . . . . . . . . . . .    19#   108

RDBASE . . . . . . . . . . . . . .    29#    29      29      29      29      29      29      29      29      29      29      29      29      29
RDCMD. . . . . . . . . . . . . . .    29#
RDCMD2 . . . . . . . . . . . . . .    29#
RDINTF . . . . . . . . . . . . . .    29#
RDREAD . . . . . . . . . . . . . .    29#
RDSTAT . . . . . . . . . . . . . .    29#
RDWRITE. . . . . . . . . . . . . .    29#
READIN . . . . . . . . . . . . . .   309    320#
READIN10 . . . . . . . . . . . . .   328    342     355     367#
READIN3. . . . . . . . . . . . . .   349#   352
READINB. . . . . . . . . . . . . .   215    239     306#    512
READINB1 . . . . . . . . . . . . .   312    315#
RESSEC . . . . . . . . . . . . . .    59#
RESTOR . . . . . . . . . . . . . .    29#
RHOM4. . . . . . . . . . . . . . .   216    229     236     240     245#
RHOME. . . . . . . . . . . . . . .    26     88     206#

SAVE_SEC . . . . . . . . . . . . .   164#   494     520
SBUFR. . . . . . . . . . . . . . .    29#
SCANID . . . . . . . . . . . . . .    29#
SCTEXT . . . . . . . . . . . . . .    29#
SDH. . . . . . . . . . . . . . . .    29#
SECPTRK. . . . . . . . . . . . . .    29#
SECSIZE. . . . . . . . . . . . . .    29#
SECSZ. . . . . . . . . . . . . . .    57#
SECTC. . . . . . . . . . . . . . .    29#
SECTN. . . . . . . . . . . . . . .    29#
SECTORS. . . . . . . . . . . . . .    62#   425
SECTRK . . . . . . . . . . . . . .    68#
SECTSZ . . . . . . . . . . . . . .    29#
SEEK . . . . . . . . . . . . . . .    29#
SEEKC. . . . . . . . . . . . . . .    29#
SEEKC2 . . . . . . . . . . . . . .    29#
SERR . . . . . . . . . . . . . . .    29#
```

```
SINIT. . . . . . . . . . . . . . .    29#
SSCTF. . . . . . . . . . . . . . .    29#
SSZ128 . . . . . . . . . . . . . .    29#
SSZ1K. . . . . . . . . . . . . . .    29#
SSZ256 . . . . . . . . . . . . . .    29#
SSZ512 . . . . . . . . . . . . . .    29#
START. . . . . . . . . . . . . . .    48#
STATUSW. . . . . . . . . . . . . .    40#
STEPR0 . . . . . . . . . . . . . .    29#
STEPR1 . . . . . . . . . . . . . .    29#
STEPR2 . . . . . . . . . . . . . .    29#
STEPR3 . . . . . . . . . . . . . .    29#
STEPR4 . . . . . . . . . . . . . .    29#
STEPR5 . . . . . . . . . . . . . .    29#
STEPR6 . . . . . . . . . . . . . .    29#
STEPR7 . . . . . . . . . . . . . .    29#
STEPR8 . . . . . . . . . . . . . .    29#
STEPR9 . . . . . . . . . . . . . .    29#
STEPRA . . . . . . . . . . . . . .    29#
STEPRATE . . . . . . . . . . . . .    22#    224
STEPRB . . . . . . . . . . . . . .    29#
STEPRC . . . . . . . . . . . . . .    29#
STEPRD . . . . . . . . . . . . . .    29#
STEPRE . . . . . . . . . . . . . .    29#
STEPRF . . . . . . . . . . . . . .    29#
STEPR_OFF. . . . . . . . . . . . .    29#    223
ST_AST . . . . . . . . . . . . . .   154#
ST_AST_OFF . . . . . . . . . . . .    29#

TEMP . . . . . . . . . . . . . . .   163#    208     259     323     332     346     359     495     516
TEMP_SEC . . . . . . . . . . . . .    27     478     562#
TEMP_TRK . . . . . . . . . . . . .    27     477     561#
TRANS. . . . . . . . . . . . . . .    46#
TRK02. . . . . . . . . . . . . . .    29#
TRK0E. . . . . . . . . . . . . . .    29#
TRUE . . . . . . . . . . . . . . .    29#     29

UNIT . . . . . . . . . . . . . . .    38#
UP_BAT . . . . . . . . . . . . . .    27     472#

WGATE. . . . . . . . . . . . . . .    29#
WINI_OFFSET. . . . . . . . . . . .    29#
WINI_SEG . . . . . . . . . . . . .    29#
WPC_OFF. . . . . . . . . . . . . .    29#    221
WPRCMP . . . . . . . . . . . . . .    29#
WPRSNT . . . . . . . . . . . . . .    29#
WTVERR . . . . . . . . . . . . . .    29#
WT_BADB1 . . . . . . . . . . . . .   500#    505
WT_BADB4 . . . . . . . . . . . . .   502     507#
WT_BADB5 . . . . . . . . . . . . .   514     556#
WT_BAD_B . . . . . . . . . . . . .   493#
WT_BCLP. . . . . . . . . . . . . .   529#    532

XBB_BD . . . . . . . . . . . . . .    29#
XBB_BS . . . . . . . . . . . . . .    29#
XBB_CN . . . . . . . . . . . . . .    29#
XBB_DISKF. . . . . . . . . . . . .    29#
```

```
XBB_DSN. . . . . . . . . . . . . .    29#
XBB_ERRC . . . . . . . . . . . . .    29#
XBB_LDN. . . . . . . . . . . . . .    29#
XBB_SC . . . . . . . . . . . . . .    29#
XBB_SN . . . . . . . . . . . . . .    29#
XCPRSNT. . . . . . . . . . . . . .    29#
XLT_F. . . . . . . . . . . . . . .    22#    325     544     553
XOPTION. . . . . . . . . . . . . .    24#
```

```
 The Microsoft MACRO Assembler                    02-20-84      PAGE     1-1
Character IO for MSDOS 2.00

                                                         PAGE    60,132
                                                         TITLE   Character IO for MSDOS 2.00
                                                         NAME    CHARACT
                                                ;
                                                ;        COMPANY CONFIDENTIAL
                                                ;        Copyright (C) 1983 Digital Equipment Corporation
                                                ;        All rights reserved.
                                                ;
                                                ;        10/04/83

                                                CGROUP   GROUP CODE
        0000                                    CODE     SEGMENT BYTE PUBLIC 'CODE'
                                                ASSUME   CS:CGROUP, DS:CGROUP, ES:CGROUP

                                                         PUBLIC  CON, AUX, PRN, CLK, AUX2
                                                         PUBLIC  CLKISR, INTRET, CLK_60_50, CLK_16_20, CLK_ADJ, TICKER
                                                         PUBLIC  OUTHEX, PSTR, FASTCON

                                                         EXTRN   RUPT7201:NEAR, INITCOM:NEAR
                                                         EXTRN   PRG7201:NEAR, DFLT7201:NEAR, DFLTBUF:NEAR, RDSETUP:NEAR
                                                         EXTRN   RCVENA:NEAR, RCVDIS:NEAR, INSTAT:NEAR, INCHAR:NEAR
                                                         EXTRN   GETCHAR:NEAR, OUTSTAT:NEAR, OUTCHAR:NEAR
                                                         EXTRN   PUTCHAR:NEAR, OUTNOW:NEAR, RDMODEM:NEAR
                                                         EXTRN   SETMODEM:NEAR, BRKON:NEAR, BRKOFF:NEAR
                                                         EXTRN   RCVINT:NEAR, RCVCANCEL:NEAR, XMTMTY:NEAR
                                                         EXTRN   XMTCANCEL:NEAR, HACK7201:NEAR, VECT7201:NEAR
                                                         EXTRN   STATCHG:NEAR, STCANCEL:NEAR

                                                         EXTRN   STRATEGY:NEAR,PTRSAVE:DWORD
                                                         EXTRN   DSKTMR:BYTE
```

```
 The Microsoft MACRO Assembler                    02-20-84      PAGE     8-1
Character IO for MSDOS 2.00

                                                         .LIST

        = EE00                                  SYSRAM   EQU     0EE00H             ; SYSRAM segment
        = 0FB7                                  ESCSTR   EQU     0FB7H              ; 8 bit char buffer
        = 0FCC                                  STRINX   EQU     0FCCH              ; string index
        = 0014                                  ESCOFF   EQU     STRINX-ESCSTR-1
```

```
                                PAGE

                      ;         Device command dispatch tables.

0000  00B5 R          CONTBL:   DW      RETURN          ;0  not used
0002  00B5 R                    DW      RETURN          ;1  not used
0004  00B5 R                    DW      RETURN          ;2  not used
0006  00AC R                    DW      CMDERR          ;3  reserved
0008  00ED R                    DW      CONIN           ;4  read
000A  00C0 R                    DW      CONINXDR        ;5  non-d. read
000C  00E3 R                    DW      CONISTAT        ;6  input status
000E  0114 R                    DW      LCONFLUSH       ;7  flush buff
0010  00FA R                    DW      CONOUT          ;8  write
0012  00FA R                    DW      CONOUT          ;9  write/ver
0014  00B5 R                    DW      RETURN          ;10 write stat
0016  00B5 R                    DW      RETURN          ;11 not used
0018  00B5 R                    DW      RETURN          ;12 not used

001A  00B5 R          CLKTBL:   DW      RETURN          ;0  not used
001C  00B5 R                    DW      RETURN          ;1  not used
001E  00B5 R                    DW      RETURN          ;2  not used
0020  00AC R                    DW      CMDERR          ;3  reserved
0022  02AF R                    DW      GETTIME         ;4  read
0024  00B6 R                    DW      BUSY            ;5  read non-d
0026  00B5 R                    DW      RETURN          ;6  not used
0028  00B5 R                    DW      RETURN          ;7  not used
002A  0296 R                    DW      SETTIME         ;8  write
002C  0296 R                    DW      SETTIME         ;9  write/ver
002E  00B5 R                    DW      RETURN          ;10 not used
0030  00B5 R                    DW      RETURN          ;11 not used
0032  00B5 R                    DW      RETURN          ;12 not used
0034                  XCOM_TBL:
0034  004E R                    DW      CINIT           ;0  - Initialize Driver.
0036  00B5 R                    DW      RETURN          ;1  - Return current media code.
0038  00B5 R                    DW      RETURN          ;2  - Get Bios Parameter Block.
003A  0193 R                    DW      XC_IOCTL        ;3  - IOCTL
003C  0146 R                    DW      XC_IN           ;4  - read.
003E  016F R                    DW      XC_NDIN         ;5  - Non-d input
0040  00B5 R                    DW      RETURN          ;6  - Return status. (Not used)
0042  00B5 R                    DW      RETURN          ;7  - Flush input buffer. (Not used.)
0044  0159 R                    DW      XC_OUT          ;8  - write.
0046  0159 R                    DW      XC_OUT          ;9  - Block write with verify.
0048  0138 R                    DW      XC_OSTAT        ;10 - Return output status.
004A  00B5 R                    DW      RETURN          ;11 - Flush output buffer. (Not used.)
004C  0193 R                    DW      XC_IOCTL        ;12 - IOCTL
```

```
                                PAGE

004E                  CINIT:
004E  80 3E 0137 R 01           CMP     BYTE PTR XC_DEV,1       ; check first time
0053  75 03                     JNZ     CINIT1
0055  E9 0000 E                 JMP     INITCOM                 ; only init once
0058  C3              CINIT1:   RET


                      ;         IO device BIOS interface

0059                  AUX2:
0059  2E: C6 06 0137 R 03       MOV     BYTE PTR CS:XC_DEV,3    ; device extended comm port
005F  EB 0E                     JMP     SHORT XCOMM
0061                  PRN:
0061  2E: C6 06 0137 R 02       MOV     BYTE PTR CS:XC_DEV,2    ; device printer port
0067  EB 06                     JMP     SHORT XCOMM
0069                  AUX:
0069  2E: C6 06 0137 R 01       MOV     BYTE PTR CS:XC_DEV,1    ; device comm port
006F                  XCOMM:
006F  56                        PUSH    SI
0070  BE 0034 R                 MOV     SI,OFFSET XCOM_TBL
0073  EB 0A                     JMP     SHORT ENTER

0075  56              CLK:      PUSH    SI
0076  BE 001A R                 MOV     SI,OFFSET CLKTBL
0079  EB 04                     JMP     SHORT ENTER

007B  56              CON:      PUSH    SI
007C  BE 0000 R                 MOV     SI,OFFSET CONTBL
                      ;         JMP     SHORT ENTER



                      ;         Common command dispatch. Pick up useful things
                      ;         and dispatch to the right routine within the table.

007F                  ENTER:
007F                  DISPATCH PROC FAR
007F  50                        PUSH    AX
0080  53                        PUSH    BX
0081  51                        PUSH    CX
0082  52                        PUSH    DX
0083  57                        PUSH    DI
0084  1E                        PUSH    DS
0085  06                        PUSH    ES

0086  2E: C5 1E 0000 E          LDS     BX,CS:PTRSAVE           ; get data pkt,
008B  C7 47 03 0100             MOV     WORD PTR [BX.STATUS],100H       ; set done bit
0090  8B 4F 12                  MOV     CX,[BX.COUNT]           ; CX = count
0093  C4 7F 0E                  LES     DI,[BX.TRANS]           ; ES:DI = buffer addr
0096  8A 47 02                  MOV     AL,[BX.CMD]             ; AL = command
0099  98                        CBW                             ; make it word
009A  03 F0                     ADD     SI,AX                   ; add offset
009C  03 F0                     ADD     SI,AX
```

```
009E  0E                              PUSH    CS                    ; DS = CS
009F  1F                              POP     DS
00A0  FC                              CLD                           ; set forward direction
00A1  FF 14                           CALL    WORD PTR [SI]         ; perform func
00A3  07                              POP     ES
00A4  1F                              POP     DS
00A5  5F                              POP     DI
00A6  5A                              POP     DX
00A7  59                              POP     CX
00A8  5B                              POP     BX
00A9  58                              POP     AX
00AA  5E                              POP     SI
00AB  CB                              RET
00AC                          DISPATCH ENDP


                              ;       set command error.

00AC  C5 1E 0000 E            CMDERR: LDS     BX,PTRSAVE
00B0  81 4F 03 8003                   OR      WORD PTR [BX.STATUS],8003H
00B5  C3                      RETURN: RET


                              ;       Set BUSY.

00B6  C5 1E 0000 E            BUSY:   LDS     BX,PTRSAVE
00BA  81 4F 03 0200                   OR      WORD PTR [BX.STATUS],200H
00BF  C3                              RET
```

```
                                      PAGE

                              ;       Non destructive console read. If char is
                              ;       available, return it, else return busy set.

00C0                          CONINXDR:
00C0  BF 0004                         MOV     DI,4                  ; Level 2 console status
00C3  CD 18                           INT     ROM
00C5  22 C9                           AND     CL,CL                 ; Check any char
00C7  74 ED                           JZ      BUSY                  ; None - return busy
00C9  B8 EE00                         MOV     AX,OFFSET SYSRAM      ; set SYSRAM segment
00CC  8E C0                           MOV     ES,AX
00CE  BE 0FB8                         MOV     SI,OFFSET ESCSTR+1    ; string addr
00D1  26: 8A 44 14                    MOV     AL,ES:BYTE PTR ESCOFF[SI]       ; get string index
00D5  98                              CBW                           ; make it word
00D6  03 F0                           ADD     SI,AX
00D8  26: 8A 04                       MOV     AL,ES:[SI]            ; get char from buffer
00DB  C5 3E 0000 E                    LDS     DI,PTRSAVE            ; get pkt ptr,
00DF  88 45 0D                        MOV     [DI.MEDIA],AL         ;  return it
00E2  C3                              RET

                              ;       console input status

00E3                          CONISTAT:
00E3  BF 0004                         MOV     DI,4                  ; Level 2 console status
00E6  CD 18                           INT     ROM
00E8  22 C9                           AND     CL,CL                 ; Check any char
00EA  74 CA                           JZ      BUSY                  ; None - return busy
00EC  C3                              RET

                              ;       console input

00ED                          CONIN:
00ED  06                              PUSH    ES
00EE  57                              PUSH    DI
00EF  51                              PUSH    CX
00F0  E8 010A R                       CALL    LCONIN
00F3  59                              POP     CX
00F4  5F                              POP     DI
00F5  07                              POP     ES
00F6  AA                              STOSB
00F7  E2 F4                           LOOP    CONIN
00F9  C3                              RET

                              ;       Write character(s) to the console.

00FA                          CONOUT:
00FA  06                              PUSH    ES
00FB  1F                              POP     DS
00FC  8B F7                           MOV     SI,DI           ; get pointer into DS:DI
00FE                          CONOUT1:
00FE  51                              PUSH    CX
00FF  56                              PUSH    SI
0100  AC                              LODSB                   ; get output char
0101  33 FF                           XOR     DI,DI           ; do console output
```

```
0103  CD 18                             INT     ROM
0105  5E                                POP     SI
0106  59                                POP     CX
0107  E2 F5                             LOOP    CONOUT1
0109  C3                                RET


                                ;       CONSOLE input, wait if not any
                                ;
                                ;       EXIT:   AL = character
                                ;       USE:    ALL

010A                            LCONIN:
010A  BF 0002                           MOV     DI,2
010D  CD 18                             INT     ROM
010F  22 C9                             AND     CL,CL           ; check status
0111  74 F7                             JZ      LCONIN          ; wait
0113  C3                                RET

                                ;       Flush the console input buffer.

0114                            LCONFLUSH:
0114  BF 0002                           MOV     DI,2            ; level 2 console in
0117  CD 18                             INT     ROM
0119  22 C9                             AND     CL,CL           ; any character?
011B  75 F7                             JNZ     LCONFLUSH       ; keep flush if still have character
011D  C3                                RET


                                ;       Output a character from AL to the console.
                                ;
                                ;       USE: none

011E                            LCONOUT:
011E  50                                PUSH    AX
011F  53                                PUSH    BX
0120  51                                PUSH    CX
0121  52                                PUSH    DX
0122  56                                PUSH    SI
0123  57                                PUSH    DI
0124  55                                PUSH    BP
0125  06                                PUSH    ES
0126  33 FF                             XOR     DI,DI
0128  CD 18                             INT     ROM
012A  07                                POP     ES
012B  5D                                POP     BP
012C  5F                                POP     DI
012D  5E                                POP     SI
012E  5A                                POP     DX
012F  59                                POP     CX
0130  5B                                POP     BX
0131  58                                POP     AX
0132  C3                                RET

                                ;       fast console output interrupt service routine
```

```

0133                            FASTCON:
0133  E8 011E R                         CALL    LCONOUT
0136  CF                                IRET
```

```
                                        PAGE

                                ;       AUX, PRN, & AUX2 common routines

0137  00                        XC_DEV  DB      0

                                ;       comm output status

0138                            XC_OSTAT:
0138  8A 2E 0137 R                      MOV     CH,XC_DEV               ; get device #
013C  E8 0000 E                         CALL    OUTSTAT
013F  22 C0                             AND     AL,AL                   ; xmt ready?
0141  75 15                             JNZ     XC_IN2
0143  E9 00B6 R                         JMP     BUSY

                                ;       comm input character

0146                            XC_IN:
0146  57                                PUSH    DI
0147  51                                PUSH    CX
0148  06                                PUSH    ES
0149  8A 2E 0137 R                      MOV     CH,XC_DEV               ; get device #
014D  E8 0000 E                         CALL    GETCHAR                 ; go to get character
0150  8A C1                             MOV     AL,CL
0152  07                                POP     ES
0153  59                                POP     CX
0154  5F                                POP     DI
0155  AA                                STOSB                           ; save in buffer
0156  E2 EE                             LOOP    XC_IN
0158                            XC_IN2:
0158  C3                                RET

                                ;       comm output character

0159                            XC_OUT:
0159  26: 8A 05                         MOV     AL,ES:[DI]              ; get character
015C  47                                INC     DI
015D  57                                PUSH    DI
015E  06                                PUSH    ES
015F  51                                PUSH    CX
0160  8A D0                             MOV     DL,AL
0162  8A 2E 0137 R                      MOV     CH,XC_DEV
0166  E8 0000 E                         CALL    PUTCHAR
0169  59                                POP     CX
016A  07                                POP     ES
016B  5F                                POP     DI
016C  E2 EB                             LOOP    XC_OUT
016E  C3                                RET

                                ;       comm non destructive input

016F                            XC_NDIN:
016F  8A 2E 0137 R                      MOV     CH,XC_DEV               ; get device #
0173  E8 0184 R                         CALL    COM_NDIN                ; get it
0176  75 09                             JNZ     XC_NDIN1                ; check if any input
```

```
0178  C4 3E 0000 E                      LES     DI,PTRSAVE              ; get packet pointer
017C  26: 88 4D 0D                      MOV     BYTE PTR ES:[DI.MEDIA],CL       ; save character in packet
0180  C3                                RET
0181                            XC_NDIN1:
0181  E9 00B6 R                         JMP     BUSY


                                ;       Non-destructed input

0184                            COM_NDIN:
0184  E8 0000 E                         CALL    INSTAT                  ; check input status
0187  3C FF                             CMP     AL,0FFH                 ; any character?
0189  75 F6                             JNZ     XC_NDIN1                ; return if no
018B  26: 8B 77 06                      MOV     SI,ES:RBOUT[BX]         ; get buffer address
018F  26: 8B 0C                         MOV     CX,ES:0[SI]             ; get character and status
0192                            COM_NDIN1:
0192  C3                                RET
```

```
                                                PAGE
                                        XCIOCP  STRUC

        0000  00                        XC_FUN  DB      0               ; function #
        0001  00                        XC_RC   DB      0               ; return code
        0002  00                        XC_CHAR DB      0               ; character
        0003  00                        XC_STAT DB      0               ; character status
        0004  00                        XC_BUF  DB      0               ; CCB or buffer

        0005                            XCIOCP  ENDS

                                        ;       XC_IOCTL
                                        ;
                                        ;       ENTRY:  ES:DI = packet pointer
                                        ;
                                        ;       byte 0  = function #
                                        ;       byte 1  = return code
                                        ;       byte 2  = character
                                        ;       byte 3  = character status
                                        ;       byte 4 to byte n = buffer

        0193                            XC_IOCTL:
        0193  8A 2E 0137 R                      MOV     CH,XC_DEV               ; CH = device #
        0197  BB 0004                           MOV     BX,OFFSET XC_BUF        ; BX = buffer address
        019A  03 DF                             ADD     BX,DI                   ; add packet offset
        019C  8C C2                             MOV     DX,ES                   ; DX = segment

        019E  26: 8A 05                         MOV     AL,ES:[DI]              ; get function #
        01A1  98                                CBW                             ; make it word
        01A2  BE 01AB R                         MOV     SI,OFFSET XIOC_TBL      ; get offset
        01A5  03 F0                             ADD     SI,AX                   ; add word offset
        01A7  03 F0                             ADD     SI,AX
        01A9  FF 24                             JMP     WORD PTR [SI]           ; perform function

        01AB                            XIOC_TBL:
        01AB  01DF R 01EC R 01EF R              DW      FUN0,FUN1,FUN2,FUN3
              01F2 R
        01B3  01FB R 01FE R 0201 R              DW      FUN4,FUN5,FUN6
        01B9  0208 R 0216 R 0221 R              DW      FUN7,FUN8,FUN9
        01BF  0228 R 0235 R 023C R              DW      FUN10,FUN11,FUN12
        01C5  0243 R 024A R 0251 R              DW      FUN13,FUN14,FUN15
        01CB  0254 R 0257 R 0260 R              DW      FUN16,FUN17,FUN18
        01D1  0263 R 026D R 0270 R              DW      FUN19,FUN20,FUN21
        01D7  0278 R 027B R 0280 R              DW      FUN22,FUN23,FUN24
        01DD  0283 R                            DW      FUN25
```

```
                                                PAGE

        01DF                            FUN0:
        01DF  8B CB                             MOV     CX,BX
        01E1  06                                PUSH    ES
        01E2  57                                PUSH    DI
        01E3  E8 0000 E                         CALL    PRG7201
        01E6                            FUN_RC:
        01E6  5F                                POP     DI
        01E7  1F                                POP     DS
        01E8  88 45 01                          MOV     [DI.XC_RC],AL           ; return function code
        01EB  C3                                RET

        01EC                            FUN1:
        01EC  E9 0000 E                         JMP     DFLT7201
        01EF                            FUN2:
        01EF  E9 0000 E                         JMP     DFLTBUF
        01F2                            FUN3:
        01F2  8B CB                             MOV     CX,BX
        01F4  06                                PUSH    ES
        01F5  57                                PUSH    DI
        01F6  E8 0000 E                         CALL    RDSETUP
        01F9  EB EB                             JMP     SHORT FUN_RC
        01FB                            FUN4:
        01FB  E9 0000 E                         JMP     RCVENA
        01FE                            FUN5:
        01FE  E9 0000 E                         JMP     RCVDIS

        0201                            FUN6:
        0201  06                                PUSH    ES
        0202  57                                PUSH    DI
        0203  E8 0000 E                         CALL    INSTAT
        0206  EB DE                             JMP     SHORT FUN_RC
        0208                            FUN7:
        0208  06                                PUSH    ES
        0209  57                                PUSH    DI
        020A  E8 0000 E                         CALL    INCHAR
        020D                            FUN_CHAR:
        020D  5F                                POP     DI
        020E  1F                                POP     DS
        020F  88 45 01                          MOV     [DI.XC_RC],AL                   ; return function code
        0212  89 4D 02                          MOV     WORD PTR [DI.XC_CHAR],CX        ; return char and status
        0215  C3                                RET
        0216                            FUN8:
        0216  06                                PUSH    ES
        0217  57                                PUSH    DI
        0218  E8 0000 E                         CALL    GETCHAR
        021B                            FUN_AX:
        021B  5F                                POP     DI
        021C  1F                                POP     DS
        021D  89 45 02                          MOV     WORD PTR [DI.XC_CHAR],AX        ; return char and status
        0220  C3                                RET
        0221                            FUN9:
        0221  06                                PUSH    ES
        0222  57                                PUSH    DI
```

```
0223  E8 0000 E                      CALL    OUTSTAT
0226  EB BE                          JMP     SHORT FUN_RC
0228                         FUN10:
0228  26: 8A 55 02                   MOV     DL,ES:[DI.XC_CHAR]             ; get char
022C  06                             PUSH    ES
022D  57                             PUSH    DI
022E  E8 0000 E                      CALL    OUTCHAR
0231  8A C1                          MOV     AL,CL                          ; get return code
0233  EB B1                          JMP     SHORT FUN_RC
0235                         FUN11:
0235  26: 8A 55 02                   MOV     DL,ES:[DI.XC_CHAR]
0239  E9 0000 E                      JMP     PUTCHAR
023C                         FUN12:
023C  26: 8A 55 02                   MOV     DL,ES:[DI.XC_CHAR]
0240  E9 0000 E                      JMP     OUTNOW
0243                         FUN13:
0243  06                             PUSH    ES
0244  57                             PUSH    DI
0245  E8 0000 E                      CALL    RDMODEM
0248  EB D1                          JMP     FUN_AX
024A                         FUN14:
024A  06                             PUSH    ES
024B  57                             PUSH    DI
024C  E8 0000 E                      CALL    SETMODEM
024F  EB 95                          JMP     SHORT FUN_RC
0251                         FUN15:
0251  E9 0000 E                      JMP     BRKON
0254                         FUN16:
0254  E9 0000 E                      JMP     BRKOFF
0257                         FUN17:
0257  8B CB                          MOV     CX,BX
0259  06                             PUSH    ES
025A  57                             PUSH    DI
025B  E8 0000 E                      CALL    RCVINT
025E  EB 86                          JMP     SHORT FUN_RC
0260                         FUN18:
0260  E9 0000 E                      JMP     RCVCANCEL
0263                         FUN19:
0263  8B CB                          MOV     CX,BX
0265  06                             PUSH    ES
0266  57                             PUSH    DI
0267  E8 0000 E                      CALL    XMTMTY
026A  E9 01E6 R                      JMP     FUN_RC
026D                         FUN20:
026D  E9 0000 E                      JMP     XMTCANCEL
0270                         FUN21:
0270  8B CB                          MOV     CX,BX
0272  E8 0000 E                      CALL    HACK7201
0275  E9 01E6 R                      JMP     FUN_RC
0278                         FUN22:
0278  E9 0000 E                      JMP     VECT7201
027B                         FUN23:
027B  8B CB                          MOV     CX,BX
027D  E9 0000 E                      JMP     STATCHG
0280                         FUN24:
```

```
0280  E9 0000 E                      JMP     STCANCEL
0283                         FUN25:
0283  06                             PUSH    ES
0284  57                             PUSH    DI
0285  E8 0184 R                      CALL    COM_NDIN
0288  EB 83                          JMP     FUN_CHAR
```

```
                                            PAGE

                                    ;       Time and Date Data Area.
                                    ;       Initialized time and date.

                                    ;***** don't change the order
028A  0559                          DAYS    DW      1096+31+28+31+30+31+30+31+31+30 ; Oct 1, 1983
028C  00                            MINUTES DB      0
028D  00                            HOURS   DB      0
028E  00                            TICKS_H DB      0               ; high byte for 0.01 sec
028F  00                            SECONDS DB      0
0290  00                            TICKS_L DB      0               ; low byte for 0.01 sec
                                    ;***** don't change the order

0291  3C                            TICKER          DB      60      ;1/60 or 1/50 counter
0292  3C                            CLK_60_50       DB      60      ;60/50 Hz constant set by ioinit
0293  01AA                          CLK_16_20       DW      426     ;1/60.0502*256*100=426 or 1/49.9465*256*100=512
0295  FD                            CLK_ADJ         DB      -3      ;(60-60.052)*60=-3 or (50-49.9465)*60=+3


                                    ;       SETTIME - set time
                                    ;
                                    ;       ENTRY:  ES:DI = time source buffer

0296                                SETTIME:
0296  BE 028A R                             MOV     SI,OFFSET DAYS
0299  1E                                    PUSH    DS              ; XCHG DS,ES
029A  06                                    PUSH    ES
029B  1F                                    POP     DS
029C  07                                    POP     ES
029D  87 F7                                 XCHG    SI,DI
029F                                SETTIME1:
029F  B9 0003                               MOV     CX,3
02A2  F3/ A5                                REP     MOVSW
02A4  0E                                    PUSH    CS              ;RESTORE DS:
02A5  1F                                    POP     DS
02A6  8A 0E 0292 R                          MOV     CL,CLK_60_50    ;RESET COUNTER/TIMER
02AA  88 0E 0291 R                          MOV     TICKER,CL
02AE  C3                                    RET

                                    ;       GETTIME - get time
                                    ;
                                    ;       ENTRY:  ES:DI = time destination buffer

02AF                                GETTIME:
02AF  BE 028A R                             MOV     SI,OFFSET DAYS
02B2  B9 0003                               MOV     CX,3
02B5  F3/ A5                                REP     MOVSW
02B7  C3                                    RET

```

```
                                            PAGE

                                    ;       clock interrupt service. We have a 16.66 ms clock.

02B8                                CLKISR:
02B8  1E                                    PUSH    DS              ;DS = CS
02B9  0E                                    PUSH    CS
02BA  1F                                    POP     DS

                                    ;       NOTE: the 60/50 ticks/second constant(CLK_16_20) and
                                    ;       adjustment value (CLK_ADJ) must be set by ioinit, default is 60 Hz.

02BB  50                                    PUSH    AX
02BC  53                                    PUSH    BX
02BD  8A 26 028E R                          MOV     AH,TICKS_H      ; get ticks high byte
02C1  A0 0290 R                             MOV     AL,TICKS_L      ; get ticks low byte
02C4  03 06 0293 R                          ADD     AX,CLK_16_20    ; add constant for 0.01 sec
02C8  88 26 028E R                          MOV     TICKS_H,AH      ; save value
02CC  A2 0290 R                             MOV     TICKS_L,AL      ; save value

02CF  FE 0E 0291 R                          DEC     TICKER          ; count down ticker
02D3  75 4E                                 JNZ     CLKRET          ; time yet?
02D5  A0 0000 E                             MOV     AL,DSKTMR       ; do disk timeout processing
02D8  FE C8                                 DEC     AL
02DA  78 03                                 JS      OV              ; jump if timer already expired
02DC  A2 0000 E                             MOV     DSKTMR,AL       ; set new value
02DF  A0 0292 R                     OV:     MOV     AL,CLK_60_50    ; get 60/50 Hz constant
02E2  A2 0291 R                             MOV     TICKER,AL       ; reset the ticker
02E5  B8 003C                               MOV     AX,60           ; AH=0, AL=60 used as constant
02E8  88 26 028E R                          MOV     TICKS_H,AH      ; reset 0.01 sec count to 0

02EC  FE 06 028F R                          INC     SECONDS
02F0  38 06 028F R                          CMP     SECONDS,AL      ; AL=60
02F4  72 2D                                 JB      CLKRET
02F6  88 26 028F R                          MOV     SECONDS,AH      ; SECONDS=0
02FA  8A 1E 0295 R                          MOV     BL,CLK_ADJ      ; get adjust value
02FE  28 1E 0291 R                          SUB     TICKER,BL       ; adjust ticker every minute
0302  FE 06 028C R                          INC     MINUTES
0306  38 06 028C R                          CMP     MINUTES,AL      ; AL=60
030A  72 17                                 JB      CLKRET
030C  88 26 028C R                          MOV     MINUTES,AH      ; MINUTES=0
0310  FE 06 028D R                          INC     HOURS
0314  80 3E 028D R 18                       CMP     HOURS,24
0319  72 08                                 JB      CLKRET
031B  88 26 028D R                          MOV     HOURS,AH        ; HOURS=0
031F  FF 06 028A R                          INC     DAYS
0323                                CLKRET:
0323  CD 64                                 INT     PROFILE         ; for user interface
0325  5B                                    POP     BX
0326  58                                    POP     AX
0327  1F                                    POP     DS
0328  CF                            INTRET: IRET
```

```
 The Microsoft MACRO Assembler               02-20-84      PAGE     8-16
Character IO for MSDOS 2.00


                                                   PAGE

                                         ;         type bx as hex, followed by a space.

        0329  8A C7                      outhex:  mov      al,bh             ; most first
        032B  E8 0339 R                           call     out2
        032E  8A C3                               mov      al,bl             ; do least
        0330  E8 0339 R                           call     out2
        0333  B0 20                               mov      al,' '
        0335  E8 011E R                           call     lconout
        0338  C3                                  ret

                                         ;         type al as hex.

        0339  50                         out2:    push     ax
        033A  51                                  push     cx
        033B  B1 04                               mov      cl,4
        033D  D2 E8                               shr      al,cl
        033F  59                                  pop      cx
        0340  E8 0344 R                           call     out1h
        0343  58                                  pop      ax

                                         ;         type lsnybble of al.

        0344  24 0F                      out1h:   and      al,0fh            ;only ls 4 bits
        0346  0C 30                               or       al,'0'            ; make ascii
        0348  3C 3A                               cmp      al,'9'+1
        034A  72 02                               jb       out1
        034C  04 27                               add      al,'a'-'9'-1      ; 0-9,al-f
        034E  E8 011E R                  out1:    call     lconout
        0351  C3                         ohr:     ret

                                         ;         Type a null terminated string.

        0352  2E: 8A 07                  pstr:    mov      al,cs:[bx]
        0355  43                                  inc      bx
        0356  3C 00                               cmp      al,0
        0358  74 F7                               je       ohr
        035A  E8 011E R                           call     lconout
        035D  EB F3                               jmp      pstr

        035F                             CODE     ENDS
                                                  END
```

```
 The Microsoft MACRO Assembler               02-20-84      PAGE     Symbols-1
Character IO for MSDOS 2.00


Macros:

                N a m e                  Length

CALRET . . . . . . . . . . . . .         0001
SEQ. . . . . . . . . . . . . . .         0002
TABLE. . . . . . . . . . . . . .         0002

Structures and records:

                N a m e                  Width    # fields
                                         Shift    Width    Mask     Initial

IODATA . . . . . . . . . . . . .         0016     0009
  CMDLEN . . . . . . . . . . . . .       0000
  UNIT . . . . . . . . . . . . . .       0001
  CMD. . . . . . . . . . . . . . .       0002
  STATUS . . . . . . . . . . . . .       0003
  MEDIA. . . . . . . . . . . . . .       000D
  TRANS. . . . . . . . . . . . . .       000E
  COUNT. . . . . . . . . . . . . .       0012
  START. . . . . . . . . . . . . .       0014
XCIOCP . . . . . . . . . . . . .         0005     0005
  XC_FUN . . . . . . . . . . . . .       0000
  XC_RC. . . . . . . . . . . . . .       0001
  XC_CHAR. . . . . . . . . . . . .       0002
  XC_STAT. . . . . . . . . . . . .       0003
  XC_BUF . . . . . . . . . . . . .       0004

Segments and groups:

                N a m e                  Size     align    combine class

CGROUP . . . . . . . . . . . . .         GROUP
  CODE . . . . . . . . . . . . . .       035F     BYTE     PUBLIC  'CODE'

Symbols:

                N a m e                  Type     Value    Attr

A7201. . . . . . . . . . . . . .         Number   0040
ABAUD. . . . . . . . . . . . . .         Number   000E
AMODEM . . . . . . . . . . . . .         Number   0002
APPXOF . . . . . . . . . . . . .         Alias    BIT7
AS7201 . . . . . . . . . . . . .         Number   0042
ASCSUB . . . . . . . . . . . . .         Number   001A
AUX. . . . . . . . . . . . . . .         L NEAR   0069     CODE     Global
AUX2 . . . . . . . . . . . . . .         L NEAR   0059     CODE     Global
AUXDP. . . . . . . . . . . . . .         Number   0040
AUXP . . . . . . . . . . . . . .         Number   0042
AUX_PRN. . . . . . . . . . . . .         Number   0044
B7201. . . . . . . . . . . . . .         Number   0041
BAKGRND. . . . . . . . . . . . .         Number   24AF
BAUDMAX. . . . . . . . . . . . .         Number   0011
BBAUD. . . . . . . . . . . . . .         Number   0006
BIT0 . . . . . . . . . . . . . .         Number   0001
```

```
BIT1 . . . . . . . . . . . . . . .      Number    0002
BIT2 . . . . . . . . . . . . . . .      Number    0004
BIT3 . . . . . . . . . . . . . . .      Number    0008
BIT4 . . . . . . . . . . . . . . .      Number    0010
BIT5 . . . . . . . . . . . . . . .      Number    0020
BIT6 . . . . . . . . . . . . . . .      Number    0040
BIT7 . . . . . . . . . . . . . . .      Number    0080
BMODEM . . . . . . . . . . . . . .      Number    00FF
BRKERR . . . . . . . . . . . . . .      Alias     BIT7
BRKOFF . . . . . . . . . . . . . .      L NEAR    0000      CODE      External
BRKON. . . . . . . . . . . . . . .      L NEAR    0000      CODE      External
BS7201 . . . . . . . . . . . . . .      Number    0043
BUSY . . . . . . . . . . . . . . .      L NEAR    00B6      CODE
CBDATB . . . . . . . . . . . . . .      Number    0003
CBLEN. . . . . . . . . . . . . . .      Number    0011
CBMODE . . . . . . . . . . . . . .      Number    0001
CBPORT . . . . . . . . . . . . . .      Number    0000
CBPRTY . . . . . . . . . . . . . .      Number    0004
CBRADR . . . . . . . . . . . . . .      Number    000D
CBRCVB . . . . . . . . . . . . . .      Number    0005
CBRXOF . . . . . . . . . . . . . .      Number    0009
CBSIZE . . . . . . . . . . . . . .      Number    000B
CBSTART. . . . . . . . . . . . . .      L BYTE    0000      CODE
CBSTPB . . . . . . . . . . . . . .      Number    0002
CBTXB. . . . . . . . . . . . . . .      Number    0006
CBTXOF . . . . . . . . . . . . . .      Number    000A
CBXOFF . . . . . . . . . . . . . .      Number    0008
CBXON. . . . . . . . . . . . . . .      Number    0007
CINIT. . . . . . . . . . . . . . .      L NEAR    004E      CODE
CINIT1 . . . . . . . . . . . . . .      L NEAR    0058      CODE
CLK. . . . . . . . . . . . . . . .      L NEAR    0075      CODE      Global
CLK16X . . . . . . . . . . . . . .      Number    0040
CLKISR . . . . . . . . . . . . . .      L NEAR    0288      CODE      Global
CLKRET . . . . . . . . . . . . . .      L NEAR    0323      CODE
CLKTBL . . . . . . . . . . . . . .      L NEAR    001A      CODE
CLK_16_20. . . . . . . . . . . . .      L WORD    0293      CODE      Global
CLK_60_50. . . . . . . . . . . . .      L BYTE    0292      CODE      Global
CLK_ADJ. . . . . . . . . . . . . .      L BYTE    0295      CODE      Global
CLK_INT. . . . . . . . . . . . . .      Number    002C
CMDERR . . . . . . . . . . . . . .      L NEAR    00AC      CODE
COMDMA . . . . . . . . . . . . . .      Number    0043
COM_NDIN . . . . . . . . . . . . .      L NEAR    0184      CODE
COM_NDIN1. . . . . . . . . . . . .      L NEAR    0192      CODE
CON. . . . . . . . . . . . . . . .      L NEAR    007B      CODE      Global
CONIN. . . . . . . . . . . . . . .      L NEAR    00ED      CODE
CONINXDR . . . . . . . . . . . . .      L NEAR    00C0      CODE
CONISTAT . . . . . . . . . . . . .      L NEAR    00E3      CODE
CONOUT . . . . . . . . . . . . . .      L NEAR    00FA      CODE
CONOUT1. . . . . . . . . . . . . .      L NEAR    00FE      CODE
CONTBL . . . . . . . . . . . . . .      L NEAR    0000      CODE
CR1. . . . . . . . . . . . . . . .      Number    0001
CR17201. . . . . . . . . . . . . .      Number    0015
CR1TXBE. . . . . . . . . . . . . .      Number    0002
CR2. . . . . . . . . . . . . . . .      Number    0002
CR27201. . . . . . . . . . . . . .      Number    0010
```

```
CR3. . . . . . . . . . . . . . . .      Number    0003
CR4. . . . . . . . . . . . . . . .      Number    0004
CR5. . . . . . . . . . . . . . . .      Number    0005
CTS. . . . . . . . . . . . . . . .      Alias     BIT3
DATMAX . . . . . . . . . . . . . .      Number    0006
DAYS . . . . . . . . . . . . . . .      L WORD    028A      CODE
DC1. . . . . . . . . . . . . . . .      Number    0011
DC3. . . . . . . . . . . . . . . .      Number    0013
DEVCOM . . . . . . . . . . . . . .      Number    0001
DEVMAX . . . . . . . . . . . . . .      Number    0003
DEVPRT . . . . . . . . . . . . . .      Number    0002
DEVXCOM. . . . . . . . . . . . . .      Number    0003
DFLT7201 . . . . . . . . . . . . .      L NEAR    0000      CODE      External
DFLTBUF. . . . . . . . . . . . . .      L NEAR    0000      CODE      External
DISPATCH . . . . . . . . . . . . .      F PROC    007F      CODE      Length =002D
DSKIO. . . . . . . . . . . . . . .      Number    0065
DSKTMR . . . . . . . . . . . . . .      V BYTE    0000      CODE      External
DSR. . . . . . . . . . . . . . . .      Alias     BIT2
DTR. . . . . . . . . . . . . . . .      Alias     BIT2
ENDTXBE. . . . . . . . . . . . . .      Number    0028
ENTER. . . . . . . . . . . . . . .      L NEAR    007F      CODE
EOI7201. . . . . . . . . . . . . .      Number    0038
ERR7201. . . . . . . . . . . . . .      Number    0030
ESCOFF . . . . . . . . . . . . . .      Number    0014
ESCSTR . . . . . . . . . . . . . .      Number    0FB7
EXCOM. . . . . . . . . . . . . . .      Number    0045
FASTCON. . . . . . . . . . . . . .      L NEAR    0133      CODE      Global
FRMERR . . . . . . . . . . . . . .      Alias     BIT6
FUN0 . . . . . . . . . . . . . . .      L NEAR    01DF      CODE
FUN1 . . . . . . . . . . . . . . .      L NEAR    01EC      CODE
FUN10. . . . . . . . . . . . . . .      L NEAR    0228      CODE
FUN11. . . . . . . . . . . . . . .      L NEAR    0235      CODE
FUN12. . . . . . . . . . . . . . .      L NEAR    023C      CODE
FUN13. . . . . . . . . . . . . . .      L NEAR    0243      CODE
FUN14. . . . . . . . . . . . . . .      L NEAR    024A      CODE
FUN15. . . . . . . . . . . . . . .      L NEAR    0251      CODE
FUN16. . . . . . . . . . . . . . .      L NEAR    0254      CODE
FUN17. . . . . . . . . . . . . . .      L NEAR    0257      CODE
FUN18. . . . . . . . . . . . . . .      L NEAR    0260      CODE
FUN19. . . . . . . . . . . . . . .      L NEAR    0263      CODE
FUN2 . . . . . . . . . . . . . . .      L NEAR    01EF      CODE
FUN20. . . . . . . . . . . . . . .      L NEAR    026D      CODE
FUN21. . . . . . . . . . . . . . .      L NEAR    0270      CODE
FUN22. . . . . . . . . . . . . . .      L NEAR    0278      CODE
FUN23. . . . . . . . . . . . . . .      L NEAR    027B      CODE
FUN24. . . . . . . . . . . . . . .      L NEAR    0280      CODE
FUN25. . . . . . . . . . . . . . .      L NEAR    0283      CODE
FUN3 . . . . . . . . . . . . . . .      L NEAR    01F2      CODE
FUN4 . . . . . . . . . . . . . . .      L NEAR    01FB      CODE
FUN5 . . . . . . . . . . . . . . .      L NEAR    01FE      CODE
FUN6 . . . . . . . . . . . . . . .      L NEAR    0201      CODE
FUN7 . . . . . . . . . . . . . . .      L NEAR    0208      CODE
FUN8 . . . . . . . . . . . . . . .      L NEAR    0216      CODE
FUN9 . . . . . . . . . . . . . . .      L NEAR    0221      CODE
FUN_AX . . . . . . . . . . . . . .      L NEAR    021B      CODE
```

```
 The Microsoft MACRO Assembler                02-20-84      PAGE     Symbols-4
Character IO for MSDOS 2.00


FUN_CHAR . . . . . . . . . . .           L NEAR   020D      CODE
FUN_RC . . . . . . . . . . . .           L NEAR   01E6      CODE
GETCHAR. . . . . . . . . . . .           L NEAR   0000      CODE      External
GETTIME. . . . . . . . . . . .           L NEAR   02AF      CODE
HACK7201 . . . . . . . . . . .           L NEAR   0000      CODE      External
HOURS. . . . . . . . . . . . .           L BYTE   028D      CODE
INCHAR . . . . . . . . . . . .           L NEAR   0000      CODE      External
INITCOM. . . . . . . . . . . .           L NEAR   0000      CODE      External
INSTAT . . . . . . . . . . . .           L NEAR   0000      CODE      External
INTRET . . . . . . . . . . . .           L NEAR   0328      CODE      Global
KDP. . . . . . . . . . . . . .           Number   0010
KSP. . . . . . . . . . . . . .           Number   0011
LCONFLUSH. . . . . . . . . . .           L NEAR   0114      CODE
LCONIN . . . . . . . . . . . .           L NEAR   010A      CODE
LCONOUT. . . . . . . . . . . .           L NEAR   011E      CODE
MINUTES. . . . . . . . . . . .           L BYTE   028C      CODE
MODEMAX. . . . . . . . . . . .           Number   0002
OHR. . . . . . . . . . . . . .           L NEAR   0351      CODE
OUT1 . . . . . . . . . . . . .           L NEAR   034E      CODE
OUT1H. . . . . . . . . . . . .           L NEAR   0344      CODE
OUT2 . . . . . . . . . . . . .           L NEAR   0339      CODE
OUTCHAR. . . . . . . . . . . .           L NEAR   0000      CODE      External
OUTHEX . . . . . . . . . . . .           L NEAR   0329      CODE      Global
OUTNOW . . . . . . . . . . . .           L NEAR   0000      CODE      External
OUTSTAT. . . . . . . . . . . .           L NEAR   0000      CODE      External
OV . . . . . . . . . . . . . .           L NEAR   02DF      CODE
OVRERR . . . . . . . . . . . .           Alias    BIT5
PARERR . . . . . . . . . . . .           Alias    BIT4
PC7201 . . . . . . . . . . . .           Number   0010
PCBAUD . . . . . . . . . . . .           Number   0008
PCCR1. . . . . . . . . . . . .           Number   000B
PCCR2. . . . . . . . . . . . .           Number   000C
PCCR3. . . . . . . . . . . . .           Number   000D
PCCR4. . . . . . . . . . . . .           Number   000E
PCCR5. . . . . . . . . . . . .           Number   000F
PCDATB . . . . . . . . . . . .           Number   0029
PCDFLT . . . . . . . . . . . .           Number   0004
PCFAIL . . . . . . . . . . . .           Number   0026
PCFLAG . . . . . . . . . . . .           Number   0014
PCID . . . . . . . . . . . . .           Number   0000
PCLEN. . . . . . . . . . . . .           Number   0037
PCMASK . . . . . . . . . . . .           Number   0015
PCMODE . . . . . . . . . . . .           Number   0027
PCMODM . . . . . . . . . . . .           Number   0006
PCPRTY . . . . . . . . . . . .           Number   002A
PCRADR . . . . . . . . . . . .           Number   0033
PCRATE . . . . . . . . . . . .           Number   000A
PCRCVA . . . . . . . . . . . .           Number   0018
PCRCVB . . . . . . . . . . . .           Number   002B
PCRCVF . . . . . . . . . . . .           Number   0017
PCRXOF . . . . . . . . . . . .           Number   002F
PCS7201. . . . . . . . . . . .           Number   0012
PCSIZE . . . . . . . . . . . .           Number   0031
PCSTART. . . . . . . . . . . .           L BYTE   0000      CODE
PCSTAT . . . . . . . . . . . .           Number   0003
```

```
 The Microsoft MACRO Assembler                02-20-84      PAGE     Symbols-5
Character IO for MSDOS 2.00


PCSTCA . . . . . . . . . . . .           Number   0022
PCSTCF . . . . . . . . . . . .           Number   0021
PCSTPB . . . . . . . . . . . .           Number   0028
PCTXB. . . . . . . . . . . . .           Number   002C
PCTXOF . . . . . . . . . . . .           Number   0030
PCXMTA . . . . . . . . . . . .           Number   001D
PCXMTF . . . . . . . . . . . .           Number   001C
PCXOFF . . . . . . . . . . . .           Number   002E
PCXON. . . . . . . . . . . . .           Number   002D
PRG7201. . . . . . . . . . . .           L NEAR   0000      CODE      External
PRN. . . . . . . . . . . . . .           L NEAR   0061      CODE      Global
PRNDP. . . . . . . . . . . . .           Number   0041
PRNP . . . . . . . . . . . . .           Number   0043
PROFILE. . . . . . . . . . . .           Number   0064
PRTYMAX. . . . . . . . . . . .           Number   0003
PSTR . . . . . . . . . . . . .           L NEAR   0352      CODE      Global
PTRSAVE. . . . . . . . . . . .           V DWORD  0000      CODE      External
PUTCHAR. . . . . . . . . . . .           L NEAR   0000      CODE      External
RBCOUNT. . . . . . . . . . . .           Number   0000
RBDFLT . . . . . . . . . . . .           Number   0020
RBHEAD . . . . . . . . . . . .           Number   000C
RBIN . . . . . . . . . . . . .           Number   0004
RBLEN. . . . . . . . . . . . .           Number   0010
RBMAX. . . . . . . . . . . . .           Number   0002
RBOUT. . . . . . . . . . . . .           Number   0006
RBSTART. . . . . . . . . . . .           L BYTE   0000      CODE
RBTAIL . . . . . . . . . . . .           Number   000E
RBXOFF . . . . . . . . . . . .           Number   0008
RBXON. . . . . . . . . . . . .           Number   000A
RCVBRK . . . . . . . . . . . .           Alias    BIT0
RCVCANCEL. . . . . . . . . . .           L NEAR   0000      CODE      External
RCVDIS . . . . . . . . . . . .           L NEAR   0000      CODE      External
RCVENA . . . . . . . . . . . .           L NEAR   0000      CODE      External
RCVINT . . . . . . . . . . . .           L NEAR   0000      CODE      External
RCVOFF . . . . . . . . . . . .           Alias    BIT5
RCVXOF . . . . . . . . . . . .           Alias    BIT6
RDMODEM. . . . . . . . . . . .           L NEAR   0000      CODE      External
RDSETUP. . . . . . . . . . . .           L NEAR   0000      CODE      External
RETURN . . . . . . . . . . . .           L NEAR   00B5      CODE
RI . . . . . . . . . . . . . .           Alias    BIT0
RLSD . . . . . . . . . . . . .           Alias    BIT4
ROM. . . . . . . . . . . . . .           Number   0018
RST7201. . . . . . . . . . . .           Number   0018
RTS. . . . . . . . . . . . . .           Alias    BIT3
RUPT7201 . . . . . . . . . . .           L NEAR   0000      CODE      External
SECONDS. . . . . . . . . . . .           L BYTE   028F      CODE
SETMODEM . . . . . . . . . . .           L NEAR   0000      CODE      External
SETTIME. . . . . . . . . . . .           L NEAR   0296      CODE
SETTIME1 . . . . . . . . . . .           L NEAR   029F      CODE
SPARE41. . . . . . . . . . . .           Number   0041
SPARE42. . . . . . . . . . . .           Number   0042
SPDI . . . . . . . . . . . . .           Alias    BIT1
SPSEL. . . . . . . . . . . . .           Alias    BIT0
SR1. . . . . . . . . . . . . .           Number   0001
SR2. . . . . . . . . . . . . .           Number   0002
```

```
 The Microsoft MACRO Assembler                02-20-84     PAGE     Symbols-6
Character IO for MSDOS 2.00


SR3. . . . . . . . . . . . . . . .           Number  0003
SRLSD. . . . . . . . . . . . . . .           Alias   BIT1
SRTS . . . . . . . . . . . . . . .           Alias   BIT1
STATCHG. . . . . . . . . . . . . .           L NEAR  0000     CODE     External
STCANCEL . . . . . . . . . . . . .           L NEAR  0000     CODE     External
STPBMAX. . . . . . . . . . . . . .           Number  0003
STRATEGY . . . . . . . . . . . . .           L NEAR  0000     CODE     External
STRINX . . . . . . . . . . . . . .           Number  0FCC
SX7201 . . . . . . . . . . . . . .           Number  0010
SYSRAM . . . . . . . . . . . . . .           Number  EE00
TICKER . . . . . . . . . . . . . .           L BYTE  0291     CODE     Global
TICKS_H. . . . . . . . . . . . . .           L BYTE  028E     CODE
TICKS_L. . . . . . . . . . . . . .           L BYTE  0290     CODE
UART . . . . . . . . . . . . . . .           Number  0046
VECT7201 . . . . . . . . . . . . .           L NEAR  0000     CODE     External
VERT . . . . . . . . . . . . . . .           Number  0040
WPRSNT . . . . . . . . . . . . . .           Number  0001
XA7201 . . . . . . . . . . . . . .           Number  0028
XABAUD . . . . . . . . . . . . . .           Number  0021
XAMODEM. . . . . . . . . . . . . .           Number  00FF
XAS7201. . . . . . . . . . . . . .           Number  002A
XB7201 . . . . . . . . . . . . . .           Number  0029
XBMODEM. . . . . . . . . . . . . .           Number  0002
XBS7201. . . . . . . . . . . . . .           Number  002B
XCOMM. . . . . . . . . . . . . . .           L NEAR  006F     CODE
XCOM_TBL . . . . . . . . . . . . .           L NEAR  0034     CODE
XCPRSNT. . . . . . . . . . . . . .           Number  0002
XC_DEV . . . . . . . . . . . . . .           L BYTE  0137     CODE
XC_IN. . . . . . . . . . . . . . .           L NEAR  0146     CODE
XC_IN2 . . . . . . . . . . . . . .           L NEAR  0158     CODE
XC_IOCTL . . . . . . . . . . . . .           L NEAR  0193     CODE
XC_NDIN. . . . . . . . . . . . . .           L NEAR  016F     CODE
XC_NDIN1 . . . . . . . . . . . . .           L NEAR  0181     CODE
XC_OSTAT . . . . . . . . . . . . .           L NEAR  0138     CODE
XC_OUT . . . . . . . . . . . . . .           L NEAR  0159     CODE
XIOC_TBL . . . . . . . . . . . . .           L NEAR  01AB     CODE
XMTBRK . . . . . . . . . . . . . .           Alias   BIT3
XMTCANCEL. . . . . . . . . . . . .           L NEAR  0000     CODE     External
XMTMTY . . . . . . . . . . . . . .           L NEAR  0000     CODE     External
XMTXOF . . . . . . . . . . . . . .           Alias   BIT2
XMTXON . . . . . . . . . . . . . .           Alias   BIT1
Z80. . . . . . . . . . . . . . . .           Number  0047
?N . . . . . . . . . . . . . . . .           Number  0010

Warning Severe
Errors  Errors
0       0
```

```
 Character IO for MSDOS 2.00

 Symbol Cross Reference                  (# is definition)       Cref-1

?N . . . . . . . . . . . . . . . .      31#   31    31#   31    31#   31    31#   31    31#   31    31#   31    31#   31
                                        31#   31    31#   31    31#   31    31#   31    31#   31    31#   31    31#   31
                                        31#   31    31    31#   31    31#   31    31#   31    31#   31    31#   31    31#
                                        31    31#   31    31#   31    31#   31    31#   31    31#   31    31#   31    31#
                                        31    31#   31    31#   31    31#   31    31#   31    31#   31    31#   31    31#
                                        31    31#   31    31#   31    31#   31    31#   31    31#   31    31#   31    31#
                                        31    31#   31    31#   31    31#   31    31#   31    31#   31    31#   31    31#
                                        31    31#   31    31#   31#   31    31#   31    31#   31    31#   31    31#   31
                                        31#   31    31#   31    31#   31    31#   31    31#   31

A7201. . . . . . . . . . . . . . .      31#
ABAUD. . . . . . . . . . . . . . .      31#
AMODEM . . . . . . . . . . . . . .      31#
APPXOF . . . . . . . . . . . . . .      31#
AS7201 . . . . . . . . . . . . . .      31#
ASCSUB . . . . . . . . . . . . . .      31#
AUX. . . . . . . . . . . . . . . .      15    99#
AUX2 . . . . . . . . . . . . . . .      15    93#
AUXDP. . . . . . . . . . . . . . .      31#
AUXP . . . . . . . . . . . . . . .      31#
AUX_PRN. . . . . . . . . . . . . .      31#

B7201. . . . . . . . . . . . . . .      31#
BAKGRND. . . . . . . . . . . . . .      31#
BAUDMAX. . . . . . . . . . . . . .      31#
BBAUD. . . . . . . . . . . . . . .      31#
BIT0 . . . . . . . . . . . . . . .      31#   31    31    31
BIT1 . . . . . . . . . . . . . . .      31#   31    31    31    31
BIT2 . . . . . . . . . . . . . . .      31#   31    31    31
BIT3 . . . . . . . . . . . . . . .      31#   31    31
BIT4 . . . . . . . . . . . . . . .      31#   31    31
BIT5 . . . . . . . . . . . . . . .      31#   31    31
BIT6 . . . . . . . . . . . . . . .      31#   31    31
BIT7 . . . . . . . . . . . . . . .      31#   31
BMODEM . . . . . . . . . . . . . .      31#
BRKERR . . . . . . . . . . . . . .      31#
BRKOFF . . . . . . . . . . . . . .      24#   479
BRKON. . . . . . . . . . . . . . .      24#   477
BS7201 . . . . . . . . . . . . . .      31#
BUSY . . . . . . . . . . . . . . .      60    162#  174   192   292   338

CBDATB . . . . . . . . . . . . . .      31#
CBLEN. . . . . . . . . . . . . . .      31#
CBMODE . . . . . . . . . . . . . .      31#
CBPORT . . . . . . . . . . . . . .      31#
CBPRTY . . . . . . . . . . . . . .      31#
CBRADR . . . . . . . . . . . . . .      31#
CBRCVB . . . . . . . . . . . . . .      31#
CBRXOF . . . . . . . . . . . . . .      31#
CBSIZE . . . . . . . . . . . . . .      31#
CBSTART. . . . . . . . . . . . . .      31#
CBSTPB . . . . . . . . . . . . . .      31#
CBTXB. . . . . . . . . . . . . . .      31#
CBTXOF . . . . . . . . . . . . . .      31#
CBXOFF . . . . . . . . . . . . . .      31#
CBXON. . . . . . . . . . . . . . .      31#
CGROUP . . . . . . . . . . . . . .      11    13    13    13
```

```
 Character IO for MSDOS 2.00

 Symbol Cross Reference                          (# is definition)       Cref-2

CINIT. . . . . . . . . . . . . . .       89      84#
CINIT1 . . . . . . . . . . . . . .       86      88#
CLK. . . . . . . . . . . . . . . .       15     106#
CLK16X . . . . . . . . . . . . . .       31#
CLKISR . . . . . . . . . . . . . .       16     566#
CLKRET . . . . . . . . . . . . . .      583     595     601     605     608#
CLKTBL . . . . . . . . . . . . . .       55#    107
CLK_16_20. . . . . . . . . . . . .       16     528#    578
CLK_60_50. . . . . . . . . . . . .       16     527#    548     588
CLK_ADJ. . . . . . . . . . . . . .       16     529#    597
CLK_INT. . . . . . . . . . . . . .       31#
CMD. . . . . . . . . . . . . . . .       31#    133
CMDERR . . . . . . . . . . . . . .       44      58     155#
CMDLEN . . . . . . . . . . . . . .       31#
CODE . . . . . . . . . . . . . . .       11      12#     12     655
COMDMA . . . . . . . . . . . . . .       31#
COM_NDIN . . . . . . . . . . . . .      332     343#    510
COM_NDIN1. . . . . . . . . . . . .      349#
CON. . . . . . . . . . . . . . . .       15     110#
CONIN. . . . . . . . . . . . . . .       45     197#    206
CONINXDR . . . . . . . . . . . . .       46     170#
CONISTAT . . . . . . . . . . . . .       47     188#
CONOUT . . . . . . . . . . . . . .       49      50     211#
CONOUT1. . . . . . . . . . . . . .      215#    223
CONTBL . . . . . . . . . . . . . .       41#    111
COUNT. . . . . . . . . . . . . . .       31#    131
CR1. . . . . . . . . . . . . . . .       31#
CR17201. . . . . . . . . . . . . .       31#
CR1TXBE. . . . . . . . . . . . . .       31#
CR2. . . . . . . . . . . . . . . .       31#
CR27201. . . . . . . . . . . . . .       31#
CR3. . . . . . . . . . . . . . . .       31#
CR4. . . . . . . . . . . . . . . .       31#
CR5. . . . . . . . . . . . . . . .       31#
CTS. . . . . . . . . . . . . . . .       31#

DATMAX . . . . . . . . . . . . . .       31#
DAYS . . . . . . . . . . . . . . .      518#    537     557     607
DC1. . . . . . . . . . . . . . . .       31#
DC3. . . . . . . . . . . . . . . .       31#
DEVCOM . . . . . . . . . . . . . .       31#
DEVMAX . . . . . . . . . . . . . .       31#
DEVPRT . . . . . . . . . . . . . .       31#
DEVXCOM. . . . . . . . . . . . . .       31#
DFLT7201 . . . . . . . . . . . . .       20#    410
DFLTBUF. . . . . . . . . . . . . .       20#    412
DISPATCH . . . . . . . . . . . . .      120#    150
DSKIO. . . . . . . . . . . . . . .       31#
DSKTMR . . . . . . . . . . . . . .       30#    584     587
DSR. . . . . . . . . . . . . . . .       31#
DTR. . . . . . . . . . . . . . . .       31#

ENDTXBE. . . . . . . . . . . . . .       31#
ENTER. . . . . . . . . . . . . . .      104     108     119#
EOI7201. . . . . . . . . . . . . .       31#
ERR7201. . . . . . . . . . . . . .       31#
ESCOFF . . . . . . . . . . . . . .       36#    178
```

```
 Character IO for MSDOS 2.00

 Symbol Cross Reference                          (# is definition)       Cref-3

ESCSTR . . . . . . . . . . . . . .       34#     36     177
EXCOM. . . . . . . . . . . . . . .       31#

FASTCON. . . . . . . . . . . . . .       17     276#
FRMERR . . . . . . . . . . . . . .       31#
FUNO . . . . . . . . . . . . . . .      386     398#
FUN1 . . . . . . . . . . . . . . .      386     409#
FUN10. . . . . . . . . . . . . . .      390     453#
FUN11. . . . . . . . . . . . . . .      390     460#
FUN12. . . . . . . . . . . . . . .      390     463#
FUN13. . . . . . . . . . . . . . .      391     466#
FUN14. . . . . . . . . . . . . . .      391     471#
FUN15. . . . . . . . . . . . . . .      391     476#
FUN16. . . . . . . . . . . . . . .      392     478#
FUN17. . . . . . . . . . . . . . .      392     480#
FUN18. . . . . . . . . . . . . . .      392     486#
FUN19. . . . . . . . . . . . . . .      393     488#
FUN2 . . . . . . . . . . . . . . .      386     411#
FUN20. . . . . . . . . . . . . . .      393     494#
FUN21. . . . . . . . . . . . . . .      393     496#
FUN22. . . . . . . . . . . . . . .      394     500#
FUN23. . . . . . . . . . . . . . .      394     502#
FUN24. . . . . . . . . . . . . . .      394     505#
FUN25. . . . . . . . . . . . . . .      395     507#
FUN3 . . . . . . . . . . . . . . .      386     413#
FUN4 . . . . . . . . . . . . . . .      388     419#
FUN5 . . . . . . . . . . . . . . .      388     421#
FUN6 . . . . . . . . . . . . . . .      388     424#
FUN7 . . . . . . . . . . . . . . .      389     429#
FUN8 . . . . . . . . . . . . . . .      389     439#
FUN9 . . . . . . . . . . . . . . .      389     448#
FUN_AX . . . . . . . . . . . . . .      443#    470
FUN_CHAR . . . . . . . . . . . . .      433#    511
FUN_RC . . . . . . . . . . . . . .      403#    418     428     452     459     475     485     493     499

GETCHAR. . . . . . . . . . . . . .       22#    301     442
GETTIME. . . . . . . . . . . . . .       59     556#

HACK7201 . . . . . . . . . . . . .       26#    498
HOURS. . . . . . . . . . . . . . .      520#    603     604     606

INCHAR . . . . . . . . . . . . . .       21#    432
INITCOM. . . . . . . . . . . . . .       19#     87
INSTAT . . . . . . . . . . . . . .       21#    344     427
INTRET . . . . . . . . . . . . . .       16     613#
IODATA . . . . . . . . . . . . . .       31#     31

KDP. . . . . . . . . . . . . . . .       31#
KSP. . . . . . . . . . . . . . . .       31#

LCONFLUSH. . . . . . . . . . . . .       48     241#    245
LCONIN . . . . . . . . . . . . . .      201     232#    236
LCONOUT. . . . . . . . . . . . . .      253#    277     623     643     652

MEDIA. . . . . . . . . . . . . . .       31#    183     335
MINUTES. . . . . . . . . . . . . .      519#    599     600     602
MODEMAX. . . . . . . . . . . . . .       31#
```

```
Character IO for MSDOS 2.00

Symbol Cross Reference                    (# is definition)      Cref-4


OHR. . . . . . . . . . . . . . .     644#    651
OUT1 . . . . . . . . . . . . . .     641     643#
OUT1H. . . . . . . . . . . . . .     633     638#
OUT2 . . . . . . . . . . . . . .     619     621     628#
OUTCHAR. . . . . . . . . . . . .      22#    457
OUTHEX . . . . . . . . . . . . .      17     618#
OUTNOW . . . . . . . . . . . . .      23#    465
OUTSTAT. . . . . . . . . . . . .      22#    289     451
OV . . . . . . . . . . . . . . .     586     588#
OVRERR . . . . . . . . . . . . .      31#

PARERR . . . . . . . . . . . . .      31#
PC7201 . . . . . . . . . . . . .      31#
PCBAUD . . . . . . . . . . . . .      31#
PCCR1. . . . . . . . . . . . . .      31#
PCCR2. . . . . . . . . . . . . .      31#
PCCR3. . . . . . . . . . . . . .      31#
PCCR4. . . . . . . . . . . . . .      31#
PCCR5. . . . . . . . . . . . . .      31#
PCDATB . . . . . . . . . . . . .      31#
PCDFLT . . . . . . . . . . . . .      31#
PCFAIL . . . . . . . . . . . . .      31#
PCFLAG . . . . . . . . . . . . .      31#
PCID . . . . . . . . . . . . . .      31#
PCLEN. . . . . . . . . . . . . .      31#
PCMASK . . . . . . . . . . . . .      31#
PCMODE . . . . . . . . . . . . .      31#
PCMODM . . . . . . . . . . . . .      31#
PCPRTY . . . . . . . . . . . . .      31#
PCRADR . . . . . . . . . . . . .      31#
PCRATE . . . . . . . . . . . . .      31#
PCRCVA . . . . . . . . . . . . .      31#
PCRCVB . . . . . . . . . . . . .      31#
PCRCVF . . . . . . . . . . . . .      31#
PCRXOF . . . . . . . . . . . . .      31#
PCS7201. . . . . . . . . . . . .      31#
PCSIZE . . . . . . . . . . . . .      31#
PCSTART. . . . . . . . . . . . .      31#
PCSTAT . . . . . . . . . . . . .      31#
PCSTCA . . . . . . . . . . . . .      31#
PCSTCF . . . . . . . . . . . . .      31#
PCSTPB . . . . . . . . . . . . .      31#
PCTXB. . . . . . . . . . . . . .      31#
PCTXOF . . . . . . . . . . . . .      31#
PCXMTA . . . . . . . . . . . . .      31#
PCXMTF . . . . . . . . . . . . .      31#
PCXOFF . . . . . . . . . . . . .      31#
PCXON. . . . . . . . . . . . . .      31#
PRG7201. . . . . . . . . . . . .      20#    402
PRN. . . . . . . . . . . . . . .      15      96#
PRNDP. . . . . . . . . . . . . .      31#
PRNP . . . . . . . . . . . . . .      31#
PROFILE. . . . . . . . . . . . .      31#    609
PRTYMAX. . . . . . . . . . . . .      31#
PSTR . . . . . . . . . . . . . .      17     648#    653
PTRSAVE. . . . . . . . . . . . .      29#    129     155     162     182     334
```

```
Character IO for MSDOS 2.00

Symbol Cross Reference                    (# is definition)      Cref-5

PUTCHAR. . . . . . . . . . . . .      23#    321     462

RBCOUNT. . . . . . . . . . . . .      31#
RBDFLT . . . . . . . . . . . . .      31#
RBHEAD . . . . . . . . . . . . .      31#
RBIN . . . . . . . . . . . . . .      31#
RBLEN. . . . . . . . . . . . . .      31#
RBMAX. . . . . . . . . . . . . .      31#
RBOUT. . . . . . . . . . . . . .      31#    347
RBSTART. . . . . . . . . . . . .      31#
RBTAIL . . . . . . . . . . . . .      31#
RBXOFF . . . . . . . . . . . . .      31#
RBXON. . . . . . . . . . . . . .      31#
RCVBRK . . . . . . . . . . . . .      31#
RCVCANCEL. . . . . . . . . . . .      25#    487
RCVDIS . . . . . . . . . . . . .      21#    422
RCVENA . . . . . . . . . . . . .      21#    420
RCVINT . . . . . . . . . . . . .      25#    484
RCVOFF . . . . . . . . . . . . .      31#
RCVXOF . . . . . . . . . . . . .      31#
RDMODEM. . . . . . . . . . . . .      23#    469
RDSETUP. . . . . . . . . . . . .      20#    417
RETURN . . . . . . . . . . . . .      41      42      43      51      52      53      55      56      57      61      62      65      66      67
                                      70      71      75      76      80     157#

RI . . . . . . . . . . . . . . .      31#
RLSD . . . . . . . . . . . . . .      31#
ROM. . . . . . . . . . . . . . .      31#    172     190     220     234     243     263
RST7201. . . . . . . . . . . . .      31#
RTS. . . . . . . . . . . . . . .      31#
RUPT7201 . . . . . . . . . . . .      19#

SECONDS. . . . . . . . . . . . .     522#    593     594     596
SEQ. . . . . . . . . . . . . . .      31      31      31      31      31      31      31      31      31      31      31      31      31      31
                                      31      31      31      31      31      31      31      31      31      31      31      31      31      31
                                      31      31      31      31      31      31      31      31      31      31      31      31      31      31
                                      31      31      31      31      31      31      31      31      31      31      31      31      31      31
                                      31      31

SETMODEM . . . . . . . . . . . .      24#    474
SETTIME. . . . . . . . . . . . .      63      64     536#
SETTIME1 . . . . . . . . . . . .     543#
SPARE41. . . . . . . . . . . . .      31#
SPARE42. . . . . . . . . . . . .      31#
SPDI . . . . . . . . . . . . . .      31#
SPSEL. . . . . . . . . . . . . .      31#
SR1. . . . . . . . . . . . . . .      31#
SR2. . . . . . . . . . . . . . .      31#
SR3. . . . . . . . . . . . . . .      31#
SRLSD. . . . . . . . . . . . . .      31#
SRTS . . . . . . . . . . . . . .      31#
START. . . . . . . . . . . . . .      31#
STATCHG. . . . . . . . . . . . .      27#    504
STATUS . . . . . . . . . . . . .      31#    130     156     163
STCANCEL . . . . . . . . . . . .      27#    506
STPBMAX. . . . . . . . . . . . .      31#
STRATEGY . . . . . . . . . . . .      29#
```

```
 Character IO for MSDOS 2.00

 Symbol Cross Reference                              (# is definition)       Cref-6

STRINX . . . . . . . . . . . . . . . .    35#     36
SX7201 . . . . . . . . . . . . . . . .    31#
SYSRAM . . . . . . . . . . . . . . . .    33#    175

TABLE. . . . . . . . . . . . . . . . .    31      31      31
TICKER . . . . . . . . . . . . . . . .    16     526#    549     582     589     598
TICKS_H. . . . . . . . . . . . . . . .   521#    576     579     591
TICKS_L. . . . . . . . . . . . . . . .   523#    577     580
TRANS. . . . . . . . . . . . . . . . .    31#    132

UART . . . . . . . . . . . . . . . . .    31#
UNIT . . . . . . . . . . . . . . . . .    31#

VECT7201 . . . . . . . . . . . . . . .    28#    501
VERT . . . . . . . . . . . . . . . . .    31#

WPRSNT . . . . . . . . . . . . . . . .    31#

XA7201 . . . . . . . . . . . . . . . .    31#
XABAUD . . . . . . . . . . . . . . . .    31#
XAMODEM. . . . . . . . . . . . . . . .    31#
XAS7201. . . . . . . . . . . . . . . .    31#
XB7201 . . . . . . . . . . . . . . . .    31#
XBMODEM. . . . . . . . . . . . . . . .    31#
XBS7201. . . . . . . . . . . . . . . .    31#
XCIOCP . . . . . . . . . . . . . . . .   352#    360
XCOMM. . . . . . . . . . . . . . . . .    95      98     101#
XCOM_TBL . . . . . . . . . . . . . . .    68#    103
XCPRSNT. . . . . . . . . . . . . . . .    31#
XC_BUF . . . . . . . . . . . . . . . .   358#    374
XC_CHAR. . . . . . . . . . . . . . . .   356#    437     446     454     461     464
XC_DEV . . . . . . . . . . . . . . . .    85      94      97     100     283#    288     300     320     331     373
XC_FUN . . . . . . . . . . . . . . . .   354#
XC_IN. . . . . . . . . . . . . . . . .    73     296#    307
XC_IN2 . . . . . . . . . . . . . . . .   291     308#
XC_IOCTL . . . . . . . . . . . . . . .    72      81     372#
XC_NDIN. . . . . . . . . . . . . . . .    74     330#
XC_NDIN1 . . . . . . . . . . . . . . .   333     337#    346
XC_OSTAT . . . . . . . . . . . . . . .    79     287#
XC_OUT . . . . . . . . . . . . . . . .    77      78     313#    325
XC_RC. . . . . . . . . . . . . . . . .   355#    406     436
XC_STAT. . . . . . . . . . . . . . . .   357#
XIOC_TBL . . . . . . . . . . . . . . .   380     385#
XMTBRK . . . . . . . . . . . . . . . .    31#
XMTCANCEL. . . . . . . . . . . . . . .    26#    495
XMTMTY . . . . . . . . . . . . . . . .    25#    492
XMTXOF . . . . . . . . . . . . . . . .    31#
XMTXON . . . . . . . . . . . . . . . .    31#

Z80. . . . . . . . . . . . . . . . . .    31#
```

```
 The Microsoft MACRO Assembler               02-20-84      PAGE     1-1
COMTAB


1                                                       PAGE    ,132
2                                                       TITLE   COMTAB
3                                                       SUBTTL  TABLES & CONTROL BLOCKS FOR COMM PORTION OF VERSION 2 BIOS
4
5                                               ;
6                                               ;
7                                               ;       COMPANY CONFIDENTIAL
8                                               ;       Copyright (C) 1982, 1983 Digital Equipment Corporation
9                                               ;       All rights reserved.
10                                              ;
11                                              ;
12                                              ;       COMBIOS.A86     version /V00-01/        APRIL 11, 1983  DEVELOPMENT
13                                              ;
14                                              ;                       version /V00-02/        JUN 02, 1983
15                                              ;                               change default values for optional comm
16                                              ;                               port to be the same as comm port.
17                                              ;
18                                              ;
19                                              ;       SECT    CODE_SEG,REL
20                                              ;
21                                              CGROUP  GROUP   CODE
22      0000                                    CODE    SEGMENT BYTE PUBLIC 'CODE'
23                                              ASSUME  CS:CGROUP, DS:CGROUP, ES:CGROUP, SS:CGROUP
24                                              ;
25                                              ;       NLIST
```

```
                                        .LIST
                                ;       LIST
                                ;       NMLIST                  ; do not list macro expansion


                                        PUBLIC  PCCOM           ; port control block for comm port
                                        PUBLIC  PCXCOM          ; port control block for extended comm
                                        PUBLIC  PCPRT           ; port control block for printer port

                                        PUBLIC  DFLTCOM         ; default comm control block for comm
                                        PUBLIC  DFLTPRT         ; default comm control block for printer port
                                        PUBLIC  DFLTXCOM        ; default comm control block for optional comm
                                        PUBLIC  NVMCOM          ; NVM comm control block for comm
                                        PUBLIC  NVMPRT          ; NVM comm control block for printer port
                                        PUBLIC  NVMXCOM         ; NVM comm control block for optional comm
                                        PUBLIC  DATBCOM         ; NVM -> CCB values for comm data bits
                                        PUBLIC  PRTYCOM         ; NVM -> CCB values for comm PARITY bit
                                        PUBLIC  PRTBAUD         ; NVM -> CCB values for printer baud
                                        PUBLIC  PRTYPRT         ; NVM -> CCB values for printer parity
                                        PUBLIC  DATBPRT         ; NVM -> CCB values for printer data bits


                                        PUBLIC  XBAUD1          ; translation table from Comm control block
                                        PUBLIC  XBAUD2          ;       to value to be programmed to generator

                                        PUBLIC  HVECTOR         ; the hack vector table of 4 double word
                                                                ; pointers that points to user service routines
```

```
                                ;---------------------------------------------------------------
                                ;
                                ;                       C O M T A B
                                ;
                                ;       This file contains all the tables and control blocks for the
                                ; Communication part of the BIOS.
                                ;
                                ;---------------------------------------------------------------

                                ;               --- PORT CONTROL BLOCKS -----
                                ;
                                ; These blocks contain all the information kept on each port.
                                ;

0000                            HVECTOR LABEL   WORD
0000    0000 0000                       DW      0,0                     ; port A on the mother board
0004    0000 0000                       DW      0,0                     ; port B on the mother board
0008    0000 0000                       DW      0,0                     ; port B on the optional comm board
000C    0000 0000                       DW      0,0                     ; port A on the optional comm board

0010                            PCCOM   LABEL   BYTE                    ; PORT CONTROL BLOCK FOR COMM
0010    41 55 58                        DB      'AUX'                   ; device name
0013    00                              DB      0                       ; status
0014    01E1 R                          DW      OFFSET NVMCOM           ; address of default device settings
0016    0002                            DW      2H                      ; port address for modem signals
0018    0006                            DW      06H                     ; baud rate generator address
001A    00                              DB      0H                      ; 9600 baud
001B    00 00 00 00 00                  DB      0,0,0,0,0               ; image of 5 control registers
0020    0040                            DW      40H                     ; 7201 data register address
0022    0042                            DW      42H                     ; port address of status/control reg
0024    00                              DB      0                       ; FLAGS
0025    FF00                            DW      0FF00H                  ; mask for databits/parity
0027    00                              DB      0                       ; receive char interrupt service flag
0028    0000                            DW      0                       ; address of receive character
002A    0000                            DW      0                       ;       interrupt service routine
002C    00                              DB      0                       ; transmit buffer empty service flag
002D    0000                            DW      0                       ; address of transmit buffer empty
002F    0000                            DW      0                       ;       service routine
0031    00                              DB      0                       ; status change service flag
0032    0000                            DW      0                       ; address of status change
0034    0000                            DW      0                       ;       service routine
0036    00                              DB      0                       ; count of # times harware failed

0037       11 [                         DB      CBLEN dup (0)           ; length of comm control block
                  00
                        ]

0048    004A R                          DW      $+2                     ; (last part of PORT CONTROL BLOCK)
004A       50 [                         DB      RBLEN+2*RBDFLT DUP (0)  ;
                  00
                        ]
```

```



009A                              PCPRT   LABEL   BYTE                  ; PORT CONTROL BLOCK FOR PRINTER
009A  4C 53 54                            DB      'LST'                 ; device name
009D  00                                  DB      0                     ; status
009E  01F2 R                              DW      OFFSET NVMPRT         ; address of default device settings
00A0  FFFF                                DW      0FFFFH                ; port address for modem signals
00A2  000E                                DW      0EH                   ; baud rate generator address
00A4  00                                  DB      0H                    ; 9600 baud
00A5  00 00 00 00 00                      DB      0,0,0,0,0             ; image of 5 control registers
00AA  0041                                DW      41H                   ; 7201 data register address
00AC  0043                                DW      43H                   ; port address of status/control reg
00AE  00                                  DB      0                     ; FLAGS
00AF  FF00                                DW      0FF00H                ; mask for databits/parity
00B1  00                                  DB      0                     ; receive char interrupt service flag
00B2  0000                                DW      0                     ; address of receive character
00B4  0000                                DW      0                     ;       interrupt service routine
00B6  00                                  DB      0                     ; transmit buffer empty service flag
00B7  0000                                DW      0                     ; address of transmit buffer empty
00B9  0000                                DW      0                     ;       service routine
00BB  00                                  DB      0                     ; status change service flag
00BC  0000                                DW      0                     ; address of status change
00BE  0000                                DW      0                     ;       service routine
00C0  00                                  DB      0                     ; count of # times harware failed

00C1      11 [                            DB      CBLEN DUP (0)         ; length of comm control block
                00
                       ]

00D2  00D4 R                              DW      $+2                   ; (last part of PORT CONTROL BLOCK)
00D4      50 [                            DB      RBLEN+2*RBDFLT DUP (0) ;
                00
                       ]




0124                              PCXCOM  LABEL   BYTE                  ; PORT CONTROL BLOCK FOR EXTENDED COMM
0124  55 43 31                            DB      'UC1'                 ; device name
0127  00                                  DB      0                     ; status
0128  0203 R                              DW      OFFSET NVMXCOM        ; address of default device settings
012A  0020                                DW      20H                   ; port address for modem signals
012C  0021                                DW      21H                   ; baud rate generator address
012E  EE                                  DB      0EEH                  ; 9600 baud
012F      05 [                            DB      5 DUP (0)             ; image of 5 control registers
                00
                       ]

0134  0029                                DW      29H                   ; 7201 data register address
0136  002B                                DW      2BH                   ; port address of status/control reg
0138  00                                  DB      0                     ; FLAGS
```

```
        0139  FF00                      DW      OFF00H                  ; mask for databits/parity
        013B  00                        DB      0                       ; receive char interrupt service flag
        013C  0000                      DW      0                       ; address of receive character
        013E  0000                      DW      0                       ;       interrupt service routine
        0140  00                        DB      0                       ; transmit buffer empty service flag
        0141  0000                      DW      0                       ; address of transmit buffer empty
        0143  0000                      DW      0                       ;       service routine
        0145  00                        DB      0                       ; status change service flag
        0146  0000                      DW      0                       ; address of status change
        0148  0000                      DW      0                       ;       service routine
        014A  00                        DB      0                       ; count of # times harware failed

        014B     11 [                   DB      CBLEN DUP (0)           ; length of comm control block
                        00
                              ]

        015C  015E R                    DW      $+2                     ; (last part of PORT CONTROL BLOCK)
        015E     50 [                   DB      RBLEN+2*RBDFLT DUP (0)  ;
                        00
                              ]

```

```


                                ;-------------------------------------------------------------------
                                ;
                                ; Default Communication Control Blocks (CCB) for each port.
                                ;
                                ; These CCB's define the default settings for each port. The reason
                                ; that it is kept in this type of data structure is that the application
                                ; porgram communicates with the BIOS to reprogram the devices in these
                                ; CCB's. A seperate one is kept for each port so that customising
                                ; can be easily accomplished after the product is delivered.
                                ;
                                ; Note the following:
                                ;
                                ;       1 - In order for the BIOS to function as defined, the state of
                                ;           the 7201's must be known completely. The reason is that the
                                ;           application can ask for certain variables to be reprogrammed
                                ;           without reprogramming the whole 7201. This requires the values
                                ;           that were programmed into the 7201 to be kept in memory.
                                ;           However, on power up, the firmware already programmed the 7201
                                ;           and has not kept a copy of the control registers. This leaves
                                ;           only two alternatives, either guess what values were programmed by
                                ;           reading NVM, or reprogram the devices with a new set of values
                                ;           which may be different from what is in NVM.
                                ;
                                ;           The problem with reading the NVM are as follows:
                                ;
                                ;               a) BIOS become hardware dependent.
                                ;               b) Code is generated is large and awkward because
                                ;                  of the way NVM is set up (VT102 hsa some funny rules
                                ;                  about implied NVM variables; eg there is no variable
                                ;                  the number of data bits for the printer, it is implied
                                ;                  from the baud rate specified!).
                                ;
                                ;       The approach taken here is to define a set of different defaults,
                                ; independent of the NVM. If this becomes undesirable, a utility program
                                ; will be written to fill these data structures with values deduced from
                                ; reading the NVM.
                                ;
                                ;-------------------------------------------------------------------
```

```


01AE                     DFLTCOM LABEL   BYTE        ; default table for Comm
01AE  01                         DB      1           ; port number for comm
01AF  01                         DB      1           ; mode - data leads only
01B0  01                         DB      1           ; one stop bit
01B1  05                         DB      5           ; 7 Space data bits
01B2  01                         DB      1           ; no parity
01B3  10                         DB      16          ; 9600 receive baud rate
01B4  10                         DB      16          ; 9600 transmit baud rate
01B5  11                         DB      DC1         ; xon character to be used
01B6  13                         DB      DC3         ; xoff charcter to be used
01B7  02                         DB      2           ; receive auto XON/XOFF enabled
01B8  02                         DB      2           ; transmit auto XON/XOFF enabled
01B9  0000                       DW      0           ; no newly defined receive buffer
01BB  0000                       DW      0           ;         .
01BD  0000                       DW      0           ;         .


01BF                     DFLTPRT LABEL   BYTE        ; default table for printer
01BF  02                         DB      2           ; port number for printer
01C0  01                         DB      1           ; mode - data leads only
01C1  01                         DB      1           ; one stop bit
01C2  04                         DB      4           ; 8 data bits
01C3  03                         DB      3           ; No parity
01C4  0E                         DB      14          ; 4800 receive baud rate
01C5  0E                         DB      14          ; 4800 transmit baud rate
01C6  11                         DB      DC1         ; xon character to be used
01C7  13                         DB      DC3         ; xoff charcter to be used
01C8  02                         DB      2           ; receive auto XON/XOFF enabled
01C9  02                         DB      2           ; transmit auto XON/XOFF enabled
01CA  0000                       DW      0           ; no newly defined receive buffer
01CC  0000                       DW      0           ;         .
01CE  0000                       DW      0           ;         .

01D0                     DFLTXCOM LABEL  BYTE        ; default table for extended Comm
01D0  03                         DB      3           ; port number for x comm
01D1  01                         DB      1           ; mode - data leads only
01D2  01                         DB      1           ; one stop bit
01D3  05                         DB      5           ; 7 space data bits
01D4  00                         DB      0           ; no parity change
01D5  10                         DB      16          ; 9600 receive baud rate
01D6  10                         DB      16          ; 9600 transmit baud rate
01D7  11                         DB      DC1         ; xon character to be used
01D8  13                         DB      DC3         ; xoff charcter to be used
01D9  02                         DB      2           ; receive auto XON/XOFF enabled
01DA  02                         DB      2           ; transmit auto XON/XOFF enabled
01DB  0000                       DW      0           ; no newly defined receive buffer
01DD  0000                       DW      0           ;         .
01DF  0000                       DW      0           ;         .
```

```


01E1                     NVMCOM LABEL    BYTE        ; default table for Comm
01E1  01                         DB      1           ; port number for comm
01E2  01                         DB      1           ; mode - data leads only
01E3  01                         DB      1           ; one stop bit
01E4  05                         DB      5           ; 7 Space data bits
01E5  01                         DB      1           ; no parity
01E6  10                         DB      16          ; 9600 receive baud rate
01E7  10                         DB      16          ; 9600 transmit baud rate
01E8  11                         DB      DC1         ; xon character to be used
01E9  13                         DB      DC3         ; xoff charcter to be used
01EA  02                         DB      2           ; receive auto XON/XOFF enabled
01EB  02                         DB      2           ; transmit auto XON/XOFF enabled
01EC  0000                       DW      0           ; no newly defined receive buffer
01EE  0000                       DW      0           ;         .
01F0  0000                       DW      0           ;         .


01F2                     NVMPRT LABEL    BYTE        ; default table for printer
01F2  02                         DB      2           ; port number for printer
01F3  01                         DB      1           ; mode - data leads only
01F4  01                         DB      1           ; one stop bit
01F5  04                         DB      4           ; 8 data bits
01F6  01                         DB      1           ; No parity
01F7  0E                         DB      14          ; 4800 receive baud rate
01F8  0E                         DB      14          ; 4800 transmit baud rate
01F9  11                         DB      DC1         ; xon character to be used
01FA  13                         DB      DC3         ; xoff charcter to be used
01FB  02                         DB      2           ; receive auto XON/XOFF enabled
01FC  02                         DB      2           ; transmit auto XON/XOFF enabled
01FD  0000                       DW      0           ; no newly defined receive buffer
01FF  0000                       DW      0           ;         .
0201  0000                       DW      0           ;         .

0203                     NVMXCOM LABEL   BYTE        ; default table for extended Comm
0203  03                         DB      3           ; port number for x comm
0204  01                         DB      1           ; mode - data leads only
0205  01                         DB      1           ; one stop bit
0206  05                         DB      5           ; 7 space data bits
0207  00                         DB      0           ; no parity change
0208  10                         DB      16          ; 9600 receive baud rate
0209  10                         DB      16          ; 9600 transmit baud rate
020A  11                         DB      DC1         ; xon character to be used
020B  13                         DB      DC3         ; xoff charcter to be used
020C  02                         DB      2           ; receive auto XON/XOFF enabled
020D  02                         DB      2           ; transmit auto XON/XOFF enabled
020E  0000                       DW      0           ; no newly defined receive buffer
0210  0000                       DW      0           ;         .
0212  0000                       DW      0           ;         .

0214                     DATBCOM LABEL   BYTE        ; NVM -> CCB values for comm data bits
0214                     DATBPRT LABEL   BYTE        ; NVM -> CCB values for printer data bits
```

```
0214  03 03 03 06 05 04                 DB      3,3,3,6,5,4,4,4
      04 04

021C                            PRTYCOM LABEL   BYTE            ; NVM -> CCB values for comm PARITY bit
021C                            PRTYPRT LABEL   BYTE            ; NVM -> CCB values for printer parity
021C  02 01 03 03 03 02                 DB      2,1,3,3,3,2,1,3
      01 03

0224                            PRTBAUD LABEL   BYTE            ; NVM -> CCB values for printer baud
0224  02 05 07 08 09 0C                 DB      2,5,7,8,9,12,14,16
      0E 10

```

```



                                ;
                                ; ----- BAUD RATE TRANSLATION TABLES. If the value is FF, the particular
                                ;       baud rate not supported for that device.
                                ;

022C                            XBAUD1  LABEL   BYTE                    ; table for COMM & XCOMM
022C  00 01 02 03 04 05                 DB      0,1,2,3,4,5,6,7,8,9,0AH,0BH,0CH,0DH,0FFH,0EH,0FH
      06 07 08 09 0A 0B
      0C 0D FF 0E 0F

023D                            XBAUD2  LABEL   BYTE                    ; table for printer
023D  FF 00 FF FF 01 FF                 DB      0FFH,0,0FFH,0FFH,1,0FFH,2,3,4,0FFH,0FFH,5,0FFH,6,0FFH,7,0FFH
      02 03 04 FF FF 05
      FF 06 FF 07 FF
024E                            CODE    ENDS
                                        END
```

```
 The Microsoft MACRO Assembler                02-20-84      PAGE     Symbols-1
COMTAB


Macros:

                N a m e                       Length

CALRET . . . . . . . . . . . . . .            0001
SEQ. . . . . . . . . . . . . . . .            0002
TABLE. . . . . . . . . . . . . . .            0002

Segments and groups:

                N a m e                       Size     align    combine class

CGROUP . . . . . . . . . . . . . .            GROUP
  CODE . . . . . . . . . . . . . . .          024E     BYTE     PUBLIC  'CODE'

Symbols:

                N a m e                       Type     Value    Attr

A7201. . . . . . . . . . . . . . .            Number   0040
ABAUD. . . . . . . . . . . . . . .            Number   000E
AMODEM . . . . . . . . . . . . . .            Number   0002
APPXOF . . . . . . . . . . . . . .            Alias    BIT7
AS7201 . . . . . . . . . . . . . .            Number   0042
ASCSUB . . . . . . . . . . . . . .            Number   001A
B7201. . . . . . . . . . . . . . .            Number   0041
BAKGRND. . . . . . . . . . . . . .            Number   24AF
BAUDMAX. . . . . . . . . . . . . .            Number   0011
BBAUD. . . . . . . . . . . . . . .            Number   0006
BIT0 . . . . . . . . . . . . . . .            Number   0001
BIT1 . . . . . . . . . . . . . . .            Number   0002
BIT2 . . . . . . . . . . . . . . .            Number   0004
BIT3 . . . . . . . . . . . . . . .            Number   0008
BIT4 . . . . . . . . . . . . . . .            Number   0010
BIT5 . . . . . . . . . . . . . . .            Number   0020
BIT6 . . . . . . . . . . . . . . .            Number   0040
BIT7 . . . . . . . . . . . . . . .            Number   0080
BMODEM . . . . . . . . . . . . . .            Number   00FF
BRKERR . . . . . . . . . . . . . .            Alias    BIT7
BS7201 . . . . . . . . . . . . . .            Number   0043
CBDATB . . . . . . . . . . . . . .            Number   0003
CBLEN. . . . . . . . . . . . . . .            Number   0011
CBMODE . . . . . . . . . . . . . .            Number   0001
CBPORT . . . . . . . . . . . . . .            Number   0000
CBPRTY . . . . . . . . . . . . . .            Number   0004
CBRADR . . . . . . . . . . . . . .            Number   000D
CBRCVB . . . . . . . . . . . . . .            Number   0005
CBRXOF . . . . . . . . . . . . . .            Number   0009
CBSIZE . . . . . . . . . . . . . .            Number   000B
CBSTART. . . . . . . . . . . . . .            L BYTE   0000     CODE
CBSTPB . . . . . . . . . . . . . .            Number   0002
CBTXB. . . . . . . . . . . . . . .            Number   0006
CBTXOF . . . . . . . . . . . . . .            Number   000A
```

```
 The Microsoft MACRO Assembler                02-20-84      PAGE     Symbols-2
COMTAB


CBXOFF . . . . . . . . . . . . . .            Number   0008
CBXON. . . . . . . . . . . . . . .            Number   0007
CLK16X . . . . . . . . . . . . . .            Number   0040
CR1. . . . . . . . . . . . . . . .            Number   0001
CR17201. . . . . . . . . . . . . .            Number   0015
CR1TXBE. . . . . . . . . . . . . .            Number   0002
CR2. . . . . . . . . . . . . . . .            Number   0002
CR27201. . . . . . . . . . . . . .            Number   0010
CR3. . . . . . . . . . . . . . . .            Number   0003
CR4. . . . . . . . . . . . . . . .            Number   0004
CR5. . . . . . . . . . . . . . . .            Number   0005
CTS. . . . . . . . . . . . . . . .            Alias    BIT3
DATBCOM. . . . . . . . . . . . . .            L BYTE   0214     CODE     Global
DATBPRT. . . . . . . . . . . . . .            L BYTE   0214     CODE     Global
DATMAX . . . . . . . . . . . . . .            Number   0006
DC1. . . . . . . . . . . . . . . .            Number   0011
DC3. . . . . . . . . . . . . . . .            Number   0013
DEVCOM . . . . . . . . . . . . . .            Number   0001
DEVMAX . . . . . . . . . . . . . .            Number   0003
DEVPRT . . . . . . . . . . . . . .            Number   0002
DEVXCOM. . . . . . . . . . . . . .            Number   0003
DFLTCOM. . . . . . . . . . . . . .            L BYTE   01AE     CODE     Global
DFLTPRT. . . . . . . . . . . . . .            L BYTE   01BF     CODE     Global
DFLTXCOM . . . . . . . . . . . . .            L BYTE   01D0     CODE     Global
DSR. . . . . . . . . . . . . . . .            Alias    BIT2
DTR. . . . . . . . . . . . . . . .            Alias    BIT2
ENDTXBE. . . . . . . . . . . . . .            Number   0028
EOI7201. . . . . . . . . . . . . .            Number   0038
ERR7201. . . . . . . . . . . . . .            Number   0030
FRMERR . . . . . . . . . . . . . .            Alias    BIT6
HVECTOR. . . . . . . . . . . . . .            L WORD   0000     CODE     Global
MODEMAX. . . . . . . . . . . . . .            Number   0002
NVMCOM . . . . . . . . . . . . . .            L BYTE   01E1     CODE     Global
NVMPRT . . . . . . . . . . . . . .            L BYTE   01F2     CODE     Global
NVMXCOM. . . . . . . . . . . . . .            L BYTE   0203     CODE     Global
OVRERR . . . . . . . . . . . . . .            Alias    BIT5
PARERR . . . . . . . . . . . . . .            Alias    BIT4
PC7201 . . . . . . . . . . . . . .            Number   0010
PCBAUD . . . . . . . . . . . . . .            Number   0008
PCCOM. . . . . . . . . . . . . . .            L BYTE   0010     CODE     Global
PCCR1. . . . . . . . . . . . . . .            Number   000B
PCCR2. . . . . . . . . . . . . . .            Number   000C
PCCR3. . . . . . . . . . . . . . .            Number   000D
PCCR4. . . . . . . . . . . . . . .            Number   000E
PCCR5. . . . . . . . . . . . . . .            Number   000F
PCDATB . . . . . . . . . . . . . .            Number   0029
PCDFLT . . . . . . . . . . . . . .            Number   0004
PCFAIL . . . . . . . . . . . . . .            Number   0026
PCFLAG . . . . . . . . . . . . . .            Number   0014
PCID . . . . . . . . . . . . . . .            Number   0000
PCLEN. . . . . . . . . . . . . . .            Number   0037
PCMASK . . . . . . . . . . . . . .            Number   0015
PCMODE . . . . . . . . . . . . . .            Number   0027
```

```
PCMODM . . . . . . . . . . . . . . .    Number  0006
PCPRT. . . . . . . . . . . . . . . .    L BYTE  009A    CODE    Global
PCPRTY . . . . . . . . . . . . . . .    Number  002A
PCRADR . . . . . . . . . . . . . . .    Number  0033
PCRATE . . . . . . . . . . . . . . .    Number  000A
PCRCVA . . . . . . . . . . . . . . .    Number  0018
PCRCVB . . . . . . . . . . . . . . .    Number  002B
PCRCVF . . . . . . . . . . . . . . .    Number  0017
PCRXOF . . . . . . . . . . . . . . .    Number  002F
PCS7201. . . . . . . . . . . . . . .    Number  0012
PCSIZE . . . . . . . . . . . . . . .    Number  0021
PCSTART. . . . . . . . . . . . . . .    L BYTE  0000    CODE
PCSTAT . . . . . . . . . . . . . . .    Number  0003
PCSTCA . . . . . . . . . . . . . . .    Number  0022
PCSTCF . . . . . . . . . . . . . . .    Number  0021
PCSTPB . . . . . . . . . . . . . . .    Number  0028
PCTXB. . . . . . . . . . . . . . . .    Number  002C
PCTXOF . . . . . . . . . . . . . . .    Number  0030
PCXCOM . . . . . . . . . . . . . . .    L BYTE  0124    CODE    Global
PCXMTA . . . . . . . . . . . . . . .    Number  001D
PCXMTF . . . . . . . . . . . . . . .    Number  001C
PCXOFF . . . . . . . . . . . . . . .    Number  002E
PCXON. . . . . . . . . . . . . . . .    Number  002D
PRTBAUD. . . . . . . . . . . . . . .    L BYTE  0224    CODE    Global
PRTYCOM. . . . . . . . . . . . . . .    L BYTE  021C    CODE    Global
PRTYMAX. . . . . . . . . . . . . . .    Number  0003
PRTYPRT. . . . . . . . . . . . . . .    L BYTE  021C    CODE    Global
RBCOUNT. . . . . . . . . . . . . . .    Number  0000
RBDFLT . . . . . . . . . . . . . . .    Number  0020
RBHEAD . . . . . . . . . . . . . . .    Number  000C
RBIN . . . . . . . . . . . . . . . .    Number  0004
RBLEN. . . . . . . . . . . . . . . .    Number  0010
RBMAX. . . . . . . . . . . . . . . .    Number  0002
RBOUT. . . . . . . . . . . . . . . .    Number  0006
RBSTART. . . . . . . . . . . . . . .    L BYTE  0000    CODE
RBTAIL . . . . . . . . . . . . . . .    Number  000E
RBXOFF . . . . . . . . . . . . . . .    Number  0008
RBXON. . . . . . . . . . . . . . . .    Number  000A
RCVBRK . . . . . . . . . . . . . . .    Alias   BIT0
RCVOFF . . . . . . . . . . . . . . .    Alias   BIT5
RCVXOF . . . . . . . . . . . . . . .    Alias   BIT6
RI . . . . . . . . . . . . . . . . .    Alias   BIT0
RLSD . . . . . . . . . . . . . . . .    Alias   BIT4
RST7201. . . . . . . . . . . . . . .    Number  0018
RTS. . . . . . . . . . . . . . . . .    Alias   BIT3
SPDI . . . . . . . . . . . . . . . .    Alias   BIT1
SPSEL. . . . . . . . . . . . . . . .    Alias   BIT0
SR1. . . . . . . . . . . . . . . . .    Number  0001
SR2. . . . . . . . . . . . . . . . .    Number  0002
SR3. . . . . . . . . . . . . . . . .    Number  0003
SRLSD. . . . . . . . . . . . . . . .    Alias   BIT1
SRTS . . . . . . . . . . . . . . . .    Alias   BIT1
STPBMAX. . . . . . . . . . . . . . .    Number  0003
```

```
SX7201 . . . . . . . . . . . . . . .    Number  0010
XA7201 . . . . . . . . . . . . . . .    Number  0028
XABAUD . . . . . . . . . . . . . . .    Number  0021
XAMODEM. . . . . . . . . . . . . . .    Number  00FF
XAS7201. . . . . . . . . . . . . . .    Number  002A
XB7201 . . . . . . . . . . . . . . .    Number  0029
XBAUD1 . . . . . . . . . . . . . . .    L BYTE  022C    CODE    Global
XBAUD2 . . . . . . . . . . . . . . .    L BYTE  023D    CODE    Global
XBMODEM. . . . . . . . . . . . . . .    Number  0002
XBS7201. . . . . . . . . . . . . . .    Number  002B
XMTBRK . . . . . . . . . . . . . . .    Alias   BIT3
XMTXOF . . . . . . . . . . . . . . .    Alias   BIT2
XMTXON . . . . . . . . . . . . . . .    Alias   BIT1
?N . . . . . . . . . . . . . . . . .    Number  0010

Warning Severe
Errors  Errors
0       0
```

```
COMTAB

Symbol Cross Reference                  (# is definition)       Cref-1

?N . . . . . . . . . . . . . . . .     26#     26      26#     26      26#     26      26#     26      26#     26      26#     26      26#     26
                                       26#     26      26#     26      26#     26      26#     26      26#     26      26#     26      26#     26
                                       26#     26      26      26#     26      26#     26      26#     26      26#     26      26#     26      26#
                                       26      26#     26      26#     26      26#     26      26#     26      26#     26      26#     26      26#
                                       26      26#     26      26#     26      26#     26      26#     26      26#     26      26#     26      26#
                                       26      26#     26      26#     26      26#     26      26#     26      26#     26      26#     26      26#
                                       26      26#     26      26#     26      26#     26      26#     26      26#     26      26#     26      26#
                                       26      26#     26      26#     26#     26      26#     26      26#     26      26#     26      26#     26
                                       26#     26      26#     26      26#     26      26#     26      26#     26

A7201. . . . . . . . . . . . . . .     26#
ABAUD. . . . . . . . . . . . . . .     26#
AMODEM . . . . . . . . . . . . . .     26#
APPXOF . . . . . . . . . . . . . .     26#
AS7201 . . . . . . . . . . . . . .     26#
ASCSUB . . . . . . . . . . . . . .     26#

B7201. . . . . . . . . . . . . . .     26#
BAKGRND. . . . . . . . . . . . . .     26#
BAUDMAX. . . . . . . . . . . . . .     26#
BBAUD. . . . . . . . . . . . . . .     26#
BIT0 . . . . . . . . . . . . . . .     26#     26      26      26
BIT1 . . . . . . . . . . . . . . .     26#     26      26      26      26
BIT2 . . . . . . . . . . . . . . .     26#     26      26      26
BIT3 . . . . . . . . . . . . . . .     26#     26      26
BIT4 . . . . . . . . . . . . . . .     26#     26      26
BIT5 . . . . . . . . . . . . . . .     26#     26      26
BIT6 . . . . . . . . . . . . . . .     26#     26      26
BIT7 . . . . . . . . . . . . . . .     26#     26
BMODEM . . . . . . . . . . . . . .     26#
BRKERR . . . . . . . . . . . . . .     26#
BS7201 . . . . . . . . . . . . . .     26#

CBDATB . . . . . . . . . . . . . .     26#
CBLEN. . . . . . . . . . . . . . .     26#     98      133     172
CBMODE . . . . . . . . . . . . . .     26#
CBPORT . . . . . . . . . . . . . .     26#
CBPRTY . . . . . . . . . . . . . .     26#
CBRADR . . . . . . . . . . . . . .     26#
CBRCVB . . . . . . . . . . . . . .     26#
CBRXOF . . . . . . . . . . . . . .     26#
CBSIZE . . . . . . . . . . . . . .     26#
CBSTART. . . . . . . . . . . . . .     26#
CBSTPB . . . . . . . . . . . . . .     26#
CBTXB. . . . . . . . . . . . . . .     26#
CBTXOF . . . . . . . . . . . . . .     26#
CBXOFF . . . . . . . . . . . . . .     26#
CBXON. . . . . . . . . . . . . . .     26#
CGROUP . . . . . . . . . . . . . .     21      23      23      23      23
CLK16X . . . . . . . . . . . . . .     26#
CODE . . . . . . . . . . . . . . .     21      22#     22      352
CR1. . . . . . . . . . . . . . . .     26#
CR17201. . . . . . . . . . . . . .     26#
CR1TXBE. . . . . . . . . . . . . .     26#
CR2. . . . . . . . . . . . . . . .     26#
```

```
COMTAB

Symbol Cross Reference                  (# is definition)       Cref-2

CR27201. . . . . . . . . . . . . .     26#
CR3. . . . . . . . . . . . . . . .     26#
CR4. . . . . . . . . . . . . . . .     26#
CR5. . . . . . . . . . . . . . . .     26#
CTS. . . . . . . . . . . . . . . .     26#

DATBCOM. . . . . . . . . . . . . .     41      322#
DATBPRT. . . . . . . . . . . . . .     45      323#
DATMAX . . . . . . . . . . . . . .     26#
DC1. . . . . . . . . . . . . . . .     26#     231     248     264     281     298     314
DC3. . . . . . . . . . . . . . . .     26#     232     249     265     282     299     315
DEVCOM . . . . . . . . . . . . . .     26#
DEVMAX . . . . . . . . . . . . . .     26#
DEVPRT . . . . . . . . . . . . . .     26#
DEVXCOM. . . . . . . . . . . . . .     26#
DFLTCOM. . . . . . . . . . . . . .     35      223#
DFLTPRT. . . . . . . . . . . . . .     36      240#
DFLTXCOM . . . . . . . . . . . . .     37      256#
DSR. . . . . . . . . . . . . . . .     26#
DTR. . . . . . . . . . . . . . . .     26#

ENDTXBE. . . . . . . . . . . . . .     26#
EOI7201. . . . . . . . . . . . . .     26#
ERR7201. . . . . . . . . . . . . .     26#

FRMERR . . . . . . . . . . . . . .     26#

HVECTOR. . . . . . . . . . . . . .     51      69#

MODEMAX. . . . . . . . . . . . . .     26#

NVMCOM . . . . . . . . . . . . . .     38      78      273#
NVMPRT . . . . . . . . . . . . . .     39      113     290#
NVMXCOM. . . . . . . . . . . . . .     40      149     306#

OVRERR . . . . . . . . . . . . . .     26#

PARERR . . . . . . . . . . . . . .     26#
PC7201 . . . . . . . . . . . . . .     26#
PCBAUD . . . . . . . . . . . . . .     26#
PCCOM. . . . . . . . . . . . . . .     31      75#
PCCR1. . . . . . . . . . . . . . .     26#
PCCR2. . . . . . . . . . . . . . .     26#
PCCR3. . . . . . . . . . . . . . .     26#
PCCR4. . . . . . . . . . . . . . .     26#
PCCR5. . . . . . . . . . . . . . .     26#
PCDATB . . . . . . . . . . . . . .     26#
PCDFLT . . . . . . . . . . . . . .     26#
PCFAIL . . . . . . . . . . . . . .     26#
PCFLAG . . . . . . . . . . . . . .     26#
PCID . . . . . . . . . . . . . . .     26#
PCLEN. . . . . . . . . . . . . . .     26#
PCMASK . . . . . . . . . . . . . .     26#
PCMODE . . . . . . . . . . . . . .     26#
PCMODM . . . . . . . . . . . . . .     26#
```

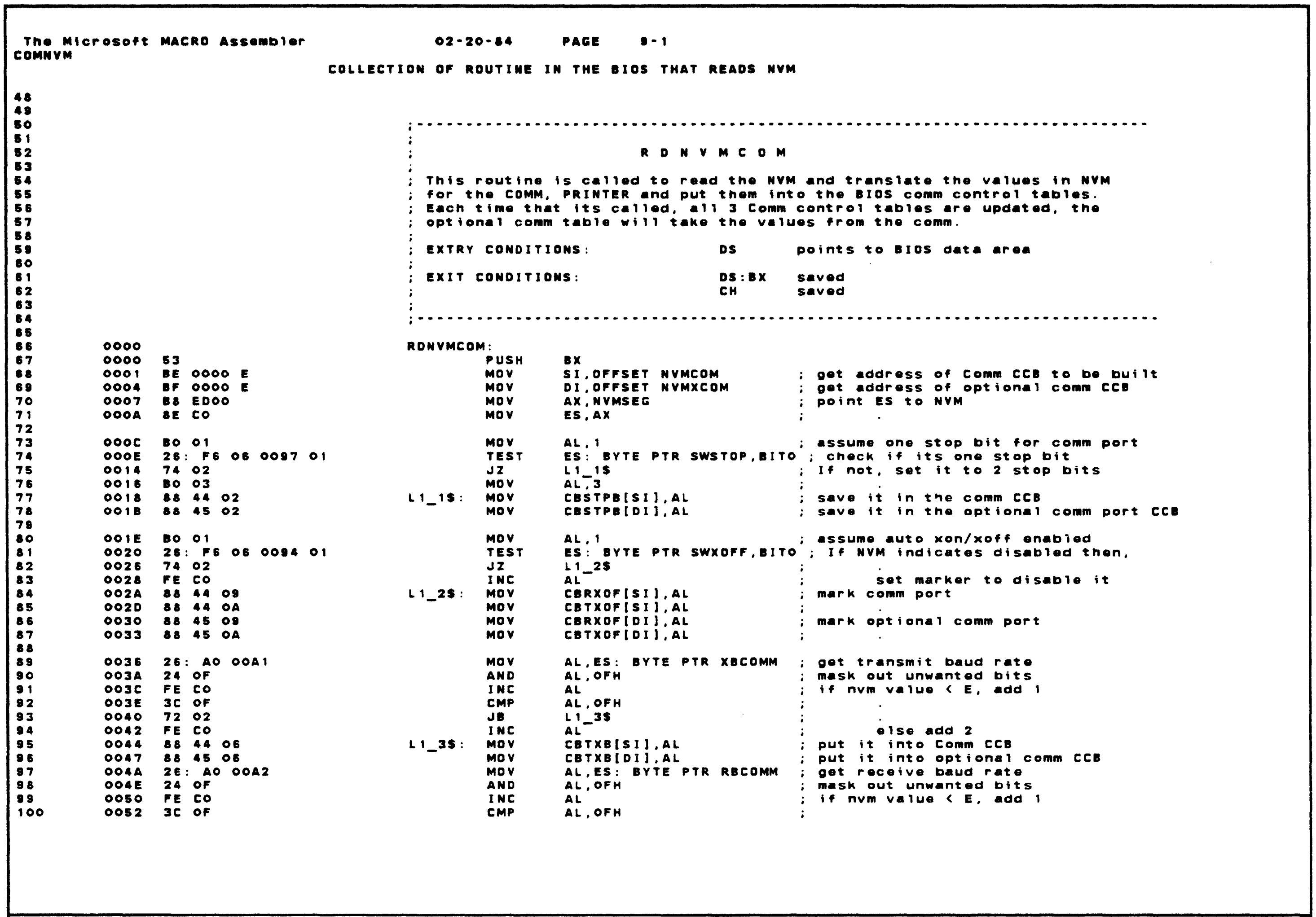
```
PCPRT. . . . . . . . . . . . . . . .   33     110#
PCPRTY . . . . . . . . . . . . . . .   26#
PCRADR . . . . . . . . . . . . . . .   26#
PCRATE . . . . . . . . . . . . . . .   26#
PCRCVA . . . . . . . . . . . . . . .   26#
PCRCVB . . . . . . . . . . . . . . .   26#
PCRCVF . . . . . . . . . . . . . . .   26#
PCRXOF . . . . . . . . . . . . . . .   26#
PCS7201. . . . . . . . . . . . . . .   26#
PCSIZE . . . . . . . . . . . . . . .   26#
PCSTART. . . . . . . . . . . . . . .   26#
PCSTAT . . . . . . . . . . . . . . .   26#
PCSTCA . . . . . . . . . . . . . . .   26#
PCSTCF . . . . . . . . . . . . . . .   26#
PCSTPB . . . . . . . . . . . . . . .   26#
PCTXB. . . . . . . . . . . . . . . .   26#
PCTXOF . . . . . . . . . . . . . . .   26#
PCXCOM . . . . . . . . . . . . . . .   32     146#
PCXMTA . . . . . . . . . . . . . . .   26#
PCXMTF . . . . . . . . . . . . . . .   26#
PCXOFF . . . . . . . . . . . . . . .   26#
PCXON. . . . . . . . . . . . . . . .   26#
PRTBAUD. . . . . . . . . . . . . . .   43     332#
PRTYCOM. . . . . . . . . . . . . . .   42     327#
PRTYMAX. . . . . . . . . . . . . . .   26#
PRTYPRT. . . . . . . . . . . . . . .   44     328#

RBCOUNT. . . . . . . . . . . . . . .   26#
RBDFLT . . . . . . . . . . . . . . .   26#    103     138     177
RBHEAD . . . . . . . . . . . . . . .   26#
RBIN . . . . . . . . . . . . . . . .   26#
RBLEN. . . . . . . . . . . . . . . .   26#    103     138     177
RBMAX. . . . . . . . . . . . . . . .   26#
RBOUT. . . . . . . . . . . . . . . .   26#
RBSTART. . . . . . . . . . . . . . .   26#
RBTAIL . . . . . . . . . . . . . . .   26#
RBXOFF . . . . . . . . . . . . . . .   26#
RBXON. . . . . . . . . . . . . . . .   26#
RCVBRK . . . . . . . . . . . . . . .   26#
RCVOFF . . . . . . . . . . . . . . .   26#
RCVXOF . . . . . . . . . . . . . . .   26#
RI . . . . . . . . . . . . . . . . .   26#
RLSD . . . . . . . . . . . . . . . .   26#
RST7201. . . . . . . . . . . . . . .   26#
RTS. . . . . . . . . . . . . . . . .   26#

SEQ. . . . . . . . . . . . . . . . .   26     26     26     26     26     26     26     26     26     26     26     26     26     26
                                       26     26     26     26     26     26     26     26     26     26     26     26     26     26
                                       26     26     26     26     26     26     26     26     26     26     26     26     26     26
                                       26     26     26     26     26     26     26     26     26     26     26     26     26     26
                                       26     26

SPDI . . . . . . . . . . . . . . . .   26#
SPSEL. . . . . . . . . . . . . . . .   26#
SR1. . . . . . . . . . . . . . . . .   26#
```

```
SR2. . . . . . . . . . . . . . . . .   26#
SR3. . . . . . . . . . . . . . . . .   26#
SRLSD. . . . . . . . . . . . . . . .   26#
SRTS . . . . . . . . . . . . . . . .   26#
STPBMAX. . . . . . . . . . . . . . .   26#
SX7201 . . . . . . . . . . . . . . .   26#

TABLE. . . . . . . . . . . . . . . .   26     26     26

XA7201 . . . . . . . . . . . . . . .   26#
XABAUD . . . . . . . . . . . . . . .   26#
XAMODEM. . . . . . . . . . . . . . .   26#
XAS7201. . . . . . . . . . . . . . .   26#
XB7201 . . . . . . . . . . . . . . .   26#
XBAUD1 . . . . . . . . . . . . . . .   48     343#
XBAUD2 . . . . . . . . . . . . . . .   49     348#
XBMODEM. . . . . . . . . . . . . . .   26#
XBS7201. . . . . . . . . . . . . . .   26#
XMTBRK . . . . . . . . . . . . . . .   26#
XMTXOF . . . . . . . . . . . . . . .   26#
XMTXON . . . . . . . . . . . . . . .   26#
```

```
                                        PAGE    ,132
                                        TITLE   COMNVM
                                        SUBTTL  COLLECTION OF ROUTINE IN THE BIOS THAT READS NVM

                                ;
                                ;
                                ;       COMPANY CONFIDENTIAL
                                ;       Copyright (C) 1982, 1983 Digital Equipment Corporation
                                ;       All rights reserved.
                                ;
                                ;
                                ;       READNVM.A86     version /V00-01/        JUN 07, 1983    DEVELOPMENT
                                ;
                                ;       SECT    CODE_SEG,REL

                                CGROUP  GROUP   CODE
        0000                    CODE    SEGMENT BYTE PUBLIC 'CODE'
                                ASSUME  CS:CGROUP, DS:CGROUP, ES:CGROUP, SS:CGROUP

                                ;       NLIST
```

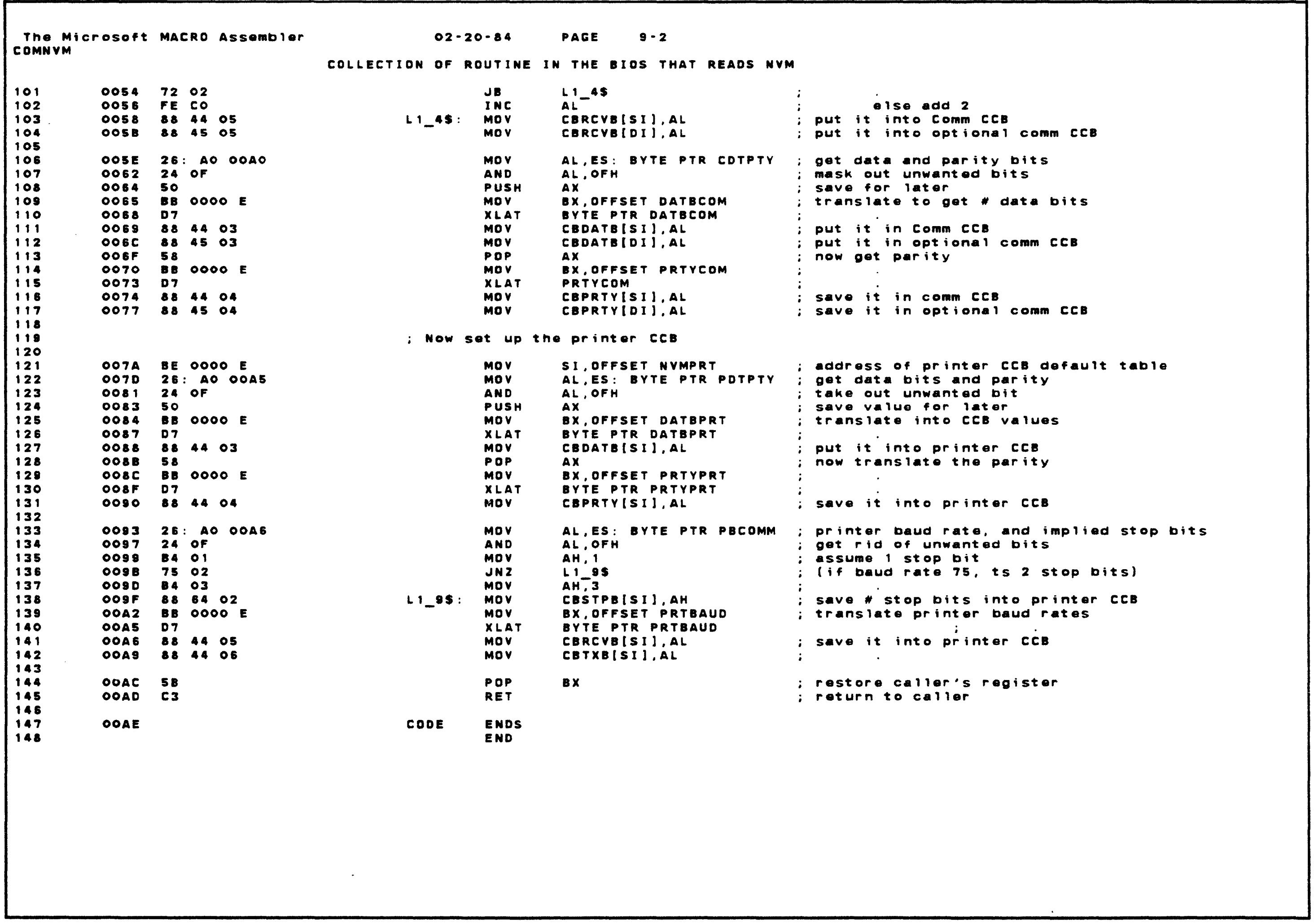
COLLECTION OF ROUTINE IN THE BIOS THAT READS NVM

```
                                        .LIST
                                ;       LIST
                                ;       NMLIST                  ; do not list macro expansion

                                        PUBLIC  RDNVMCOM        ; Read NVM for comm, printer default
                                                                ; optional comm, will take NVM COMM values

                                        EXTRN   NVMCOM:BYTE     ; CCB for COMM
                                        EXTRN   NVMXCOM:BYTE    ; CCB for optional comm
                                        EXTRN   NVMPRT:BYTE     ; CCB for printer

                                        EXTRN   DATBCOM:BYTE    ; NVM -> CCB values for comm data bits
                                        EXTRN   PRTYCOM:BYTE    ; NVM -> CCB values for comm parity
                                        EXTRN   PRTBAUD:BYTE    ; NVM -> CCB values for printer baud rates
                                        EXTRN   DATBPRT:BYTE    ; NVM -> CCB values for printer data bits
                                        EXTRN   PRTYPRT:BYTE    ; NVM -> CCB values for printer parity

        = ED00                  NVMSEG  EQU     0ED00H          ; base address for NVM

        = 0097                  SWSTOP  EQU     97H             ; NVM for # stop bits in COMM port
        = 0094                  SWXOFF  EQU     94H             ; NVM for auto enable xon/xoff for COMM
        = 00A0                  CDTPTY  EQU     0A0H            ; NVM for # data and parity bits for COMM
        = 00A1                  XBCOMM  EQU     0A1H            ; NVM transmit baud rate, for comm port
        = 00A2                  RBCOMM  EQU     0A2H            ; NVM for receive baud rate for comm port
        = 00A5                  PDTPTY  EQU     0A5H            ; NVM for # data and parity bit for printer
        = 00A6                  PBCOMM  EQU     0A6H            ; NVM transmit and receive baud for printer
```



COLLECTION OF ROUTINE IN THE BIOS THAT READS NVM

```


                                    ;----------------------------------------------------------------
                                    ;
                                    ;                          R D N V M C O M
                                    ;
                                    ; This routine is called to read the NVM and translate the values in NVM
                                    ; for the COMM, PRINTER and put them into the BIOS comm control tables.
                                    ; Each time that its called, all 3 Comm control tables are updated, the
                                    ; optional comm table will take the values from the comm.
                                    ;
                                    ; EXTRY CONDITIONS:              DS        points to BIOS data area
                                    ;
                                    ; EXIT CONDITIONS:               DS:BX     saved
                                    ;                                CH        saved
                                    ;
                                    ;----------------------------------------------------------------

0000                                RDNVMCOM:
0000  53                                      PUSH    BX
0001  BE 0000 E                               MOV     SI,OFFSET NVMCOM         ; get address of Comm CCB to be built
0004  BF 0000 E                               MOV     DI,OFFSET NVMXCOM        ; get address of optional comm CCB
0007  B8 ED00                                 MOV     AX,NVMSEG                ; point ES to NVM
000A  8E C0                                   MOV     ES,AX                    ;       .

000C  B0 01                                   MOV     AL,1                     ; assume one stop bit for comm port
000E  26: F6 06 0097 01                       TEST    ES: BYTE PTR SWSTOP,BIT0 ; check if its one stop bit
0014  74 02                                   JZ      L1_1$                    ; If not, set it to 2 stop bits
0016  B0 03                                   MOV     AL,3                     ;       .
0018  88 44 02                      L1_1$:    MOV     CBSTPB[SI],AL            ; save it in the comm CCB
001B  88 45 02                                MOV     CBSTPB[DI],AL            ; save it in the optional comm port CCB

001E  B0 01                                   MOV     AL,1                     ; assume auto xon/xoff enabled
0020  26: F6 06 0094 01                       TEST    ES: BYTE PTR SWXOFF,BIT0 ; If NVM indicates disabled then,
0026  74 02                                   JZ      L1_2$                    ;       .
0028  FE C0                                   INC     AL                       ;       set marker to disable it
002A  88 44 09                      L1_2$:    MOV     CBRXOF[SI],AL            ; mark comm port
002D  88 44 0A                                MOV     CBTXOF[SI],AL            ;       .
0030  88 45 09                                MOV     CBRXOF[DI],AL            ; mark optional comm port
0033  88 45 0A                                MOV     CBTXOF[DI],AL            ;       .

0036  26: A0 00A1                             MOV     AL,ES: BYTE PTR XBCOMM   ; get transmit baud rate
003A  24 0F                                   AND     AL,0FH                   ; mask out unwanted bits
003C  FE C0                                   INC     AL                       ; if nvm value < E, add 1
003E  3C 0F                                   CMP     AL,0FH                   ;       .
0040  72 02                                   JB      L1_3$                    ;       .
0042  FE C0                                   INC     AL                       ;       else add 2
0044  88 44 06                      L1_3$:    MOV     CBTXB[SI],AL             ; put it into Comm CCB
0047  88 45 06                                MOV     CBTXB[DI],AL             ; put it into optional comm CCB
004A  26: A0 00A2                             MOV     AL,ES: BYTE PTR RBCOMM   ; get receive baud rate
004E  24 0F                                   AND     AL,0FH                   ; mask out unwanted bits
0050  FE C0                                   INC     AL                       ; if nvm value < E, add 1
0052  3C 0F                                   CMP     AL,0FH                   ;
```



COLLECTION OF ROUTINE IN THE BIOS THAT READS NVM

```
0054  72 02                                   JB      L1_4$                    ;       .
0056  FE C0                                   INC     AL                       ;       else add 2
0058  88 44 05                      L1_4$:    MOV     CBRCVB[SI],AL            ; put it into Comm CCB
005B  88 45 05                                MOV     CBRCVB[DI],AL            ; put it into optional comm CCB

005E  26: A0 00A0                             MOV     AL,ES: BYTE PTR CDTPTY   ; get data and parity bits
0062  24 0F                                   AND     AL,0FH                   ; mask out unwanted bits
0064  50                                      PUSH    AX                       ; save for later
0065  BB 0000 E                               MOV     BX,OFFSET DATBCOM        ; translate to get # data bits
0068  D7                                      XLAT    BYTE PTR DATBCOM         ;       .
0069  88 44 03                                MOV     CBDATB[SI],AL            ; put it in Comm CCB
006C  88 45 03                                MOV     CBDATB[DI],AL            ; put it in optional comm CCB
006F  58                                      POP     AX                       ; now get parity
0070  BB 0000 E                               MOV     BX,OFFSET PRTYCOM        ;       .
0073  D7                                      XLAT    PRTYCOM                  ;       .
0074  88 44 04                                MOV     CBPRTY[SI],AL            ; save it in comm CCB
0077  88 45 04                                MOV     CBPRTY[DI],AL            ; save it in optional comm CCB

                                    ; Now set up the printer CCB

007A  BE 0000 E                               MOV     SI,OFFSET NVMPRT         ; address of printer CCB default table
007D  26: A0 00A5                             MOV     AL,ES: BYTE PTR PDTPTY   ; get data bits and parity
0081  24 0F                                   AND     AL,0FH                   ; take out unwanted bit
0083  50                                      PUSH    AX                       ; save value for later
0084  BB 0000 E                               MOV     BX,OFFSET DATBPRT        ; translate into CCB values
0087  D7                                      XLAT    BYTE PTR DATBPRT         ;       .
0088  88 44 03                                MOV     CBDATB[SI],AL            ; put it into printer CCB
008B  58                                      POP     AX                       ; now translate the parity
008C  BB 0000 E                               MOV     BX,OFFSET PRTYPRT        ;       .
008F  D7                                      XLAT    BYTE PTR PRTYPRT         ;       .
0090  88 44 04                                MOV     CBPRTY[SI],AL            ; save it into printer CCB

0093  26: A0 00A6                             MOV     AL,ES: BYTE PTR PBCOMM   ; printer baud rate, and implied stop bits
0097  24 0F                                   AND     AL,0FH                   ; get rid of unwanted bits
0099  B4 01                                   MOV     AH,1                     ; assume 1 stop bit
009B  75 02                                   JNZ     L1_9$                    ; (if baud rate 75, ts 2 stop bits)
009D  B4 03                                   MOV     AH,3                     ;       .
009F  88 64 02                      L1_9$:    MOV     CBSTPB[SI],AH            ; save # stop bits into printer CCB
00A2  BB 0000 E                               MOV     BX,OFFSET PRTBAUD        ; translate printer baud rates
00A5  D7                                      XLAT    BYTE PTR PRTBAUD         ;                  ;    .
00A6  88 44 05                                MOV     CBRCVB[SI],AL            ; save it into printer CCB
00A9  88 44 06                                MOV     CBTXB[SI],AL             ;       .

00AC  5B                                      POP     BX                       ; restore caller's register
00AD  C3                                      RET                              ; return to caller

00AE                                CODE      ENDS
                                              END
```

```
 The Microsoft MACRO Assembler            02-20-84    PAGE    Symbols-1
COMNVM


Macros:

                N a m e                     Length

CALRET . . . . . . . . . . .                0001
SEQ. . . . . . . . . . . . .                0002
TABLE. . . . . . . . . . . .                0002

Segments and groups:

                N a m e                     Size    align   combine class

CGROUP . . . . . . . . . . .                GROUP
  CODE . . . . . . . . . . . .              00AE    BYTE    PUBLIC  'CODE'

Symbols:

                N a m e                     Type     Value   Attr

A7201. . . . . . . . . . . .                Number   0040
ABAUD. . . . . . . . . . . .                Number   000E
AMODEM . . . . . . . . . . .                Number   0002
APPXOF . . . . . . . . . . .                Alias    BIT7
AS7201 . . . . . . . . . . .                Number   0042
ASCSUB . . . . . . . . . . .                Number   001A
B7201. . . . . . . . . . . .                Number   0041
BAKGRND. . . . . . . . . . .                Number   24AF
BAUDMAX. . . . . . . . . . .                Number   0011
BBAUD. . . . . . . . . . . .                Number   0006
BIT0 . . . . . . . . . . . .                Number   0001
BIT1 . . . . . . . . . . . .                Number   0002
BIT2 . . . . . . . . . . . .                Number   0004
BIT3 . . . . . . . . . . . .                Number   0008
BIT4 . . . . . . . . . . . .                Number   0010
BIT5 . . . . . . . . . . . .                Number   0020
BIT6 . . . . . . . . . . . .                Number   0040
BIT7 . . . . . . . . . . . .                Number   0080
BMODEM . . . . . . . . . . .                Number   00FF
BRKERR . . . . . . . . . . .                Alias    BIT7
BS7201 . . . . . . . . . . .                Number   0043
CBDATB . . . . . . . . . . .                Number   0003
CBLEN. . . . . . . . . . . .                Number   0011
CBMODE . . . . . . . . . . .                Number   0001
CBPORT . . . . . . . . . . .                Number   0000
CBPRTY . . . . . . . . . . .                Number   0004
CBRADR . . . . . . . . . . .                Number   000D
CBRCVB . . . . . . . . . . .                Number   0005
CBRXOF . . . . . . . . . . .                Number   0009
CBSIZE . . . . . . . . . . .                Number   000B
CBSTART. . . . . . . . . . .                L BYTE   0000    CODE
CBSTPB . . . . . . . . . . .                Number   0002
CBTXB. . . . . . . . . . . .                Number   0006
CBTXOF . . . . . . . . . . .                Number   000A
```

```
 The Microsoft MACRO Assembler            02-20-84    PAGE    Symbols-2
COMNVM


CBXOFF . . . . . . . . . . .                Number   0008
CBXON. . . . . . . . . . . .                Number   0007
CDTPTY . . . . . . . . . . .                Number   00A0
CLK16X . . . . . . . . . . .                Number   0040
CR1. . . . . . . . . . . . .                Number   0001
CR17201. . . . . . . . . . .                Number   0015
CR1TXBE. . . . . . . . . . .                Number   0002
CR2. . . . . . . . . . . . .                Number   0002
CR27201. . . . . . . . . . .                Number   0010
CR3. . . . . . . . . . . . .                Number   0003
CR4. . . . . . . . . . . . .                Number   0004
CR5. . . . . . . . . . . . .                Number   0005
CTS. . . . . . . . . . . . .                Alias    BIT3
DATBCOM. . . . . . . . . . .                V BYTE   0000    CODE    External
DATBPRT. . . . . . . . . . .                V BYTE   0000    CODE    External
DATMAX . . . . . . . . . . .                Number   0006
DC1. . . . . . . . . . . . .                Number   0011
DC3. . . . . . . . . . . . .                Number   0013
DEVCOM . . . . . . . . . . .                Number   0001
DEVMAX . . . . . . . . . . .                Number   0003
DEVPRT . . . . . . . . . . .                Number   0002
DEVXCOM. . . . . . . . . . .                Number   0003
DSR. . . . . . . . . . . . .                Alias    BIT2
DTR. . . . . . . . . . . . .                Alias    BIT2
ENDTXBE. . . . . . . . . . .                Number   0028
EOI7201. . . . . . . . . . .                Number   0038
ERR7201. . . . . . . . . . .                Number   0030
FRMERR . . . . . . . . . . .                Alias    BIT6
L1_1$. . . . . . . . . . . .                L NEAR   0018    CODE
L1_2$. . . . . . . . . . . .                L NEAR   002A    CODE
L1_3$. . . . . . . . . . . .                L NEAR   0044    CODE
L1_4$. . . . . . . . . . . .                L NEAR   0058    CODE
L1_9$. . . . . . . . . . . .                L NEAR   009F    CODE
MODEMAX. . . . . . . . . . .                Number   0002
NVMCOM . . . . . . . . . . .                V BYTE   0000    CODE    External
NVMPRT . . . . . . . . . . .                V BYTE   0000    CODE    External
NVMSEG . . . . . . . . . . .                Number   ED00
NVMXCOM. . . . . . . . . . .                V BYTE   0000    CODE    External
OVRERR . . . . . . . . . . .                Alias    BIT5
PARERR . . . . . . . . . . .                Alias    BIT4
PBCOMM . . . . . . . . . . .                Number   00A6
PC7201 . . . . . . . . . . .                Number   0010
PCBAUD . . . . . . . . . . .                Number   0008
PCCR1. . . . . . . . . . . .                Number   000B
PCCR2. . . . . . . . . . . .                Number   000C
PCCR3. . . . . . . . . . . .                Number   000D
PCCR4. . . . . . . . . . . .                Number   000E
PCCR5. . . . . . . . . . . .                Number   000F
PCDATB . . . . . . . . . . .                Number   0029
PCDFLT . . . . . . . . . . .                Number   0004
PCFAIL . . . . . . . . . . .                Number   0026
PCFLAG . . . . . . . . . . .                Number   0014
PCID . . . . . . . . . . . .                Number   0000
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE     Symbols-3
COMNVM


PCLEN. . . . . . . . . . . . . . . .        Number   0037
PCMASK . . . . . . . . . . . . . . .        Number   0015
PCMODE . . . . . . . . . . . . . . .        Number   0027
PCMODM . . . . . . . . . . . . . . .        Number   0006
PCPRTY . . . . . . . . . . . . . . .        Number   002A
PCRADR . . . . . . . . . . . . . . .        Number   0033
PCRATE . . . . . . . . . . . . . . .        Number   000A
PCRCVA . . . . . . . . . . . . . . .        Number   0018
PCRCVB . . . . . . . . . . . . . . .        Number   002B
PCRCVF . . . . . . . . . . . . . . .        Number   0017
PCRXOF . . . . . . . . . . . . . . .        Number   002F
PCS7201. . . . . . . . . . . . . . .        Number   0012
PCSIZE . . . . . . . . . . . . . . .        Number   0031
PCSTART. . . . . . . . . . . . . . .        L BYTE   0000     CODE
PCSTAT . . . . . . . . . . . . . . .        Number   0003
PCSTCA . . . . . . . . . . . . . . .        Number   0022
PCSTCF . . . . . . . . . . . . . . .        Number   0021
PCSTPB . . . . . . . . . . . . . . .        Number   0028
PCTXB. . . . . . . . . . . . . . . .        Number   002C
PCTXOF . . . . . . . . . . . . . . .        Number   0030
PCXMTA . . . . . . . . . . . . . . .        Number   001D
PCXMTF . . . . . . . . . . . . . . .        Number   001C
PCXOFF . . . . . . . . . . . . . . .        Number   002E
PCXON. . . . . . . . . . . . . . . .        Number   002D
PDTPTY . . . . . . . . . . . . . . .        Number   00A5
PRTBAUD. . . . . . . . . . . . . . .        V BYTE   0000     CODE     External
PRTYCOM. . . . . . . . . . . . . . .        V BYTE   0000     CODE     External
PRTYMAX. . . . . . . . . . . . . . .        Number   0003
PRTYPRT. . . . . . . . . . . . . . .        V BYTE   0000     CODE     External
RBCOMM . . . . . . . . . . . . . . .        Number   00A2
RBCOUNT. . . . . . . . . . . . . . .        Number   0000
RBDFLT . . . . . . . . . . . . . . .        Number   0020
RBHEAD . . . . . . . . . . . . . . .        Number   000C
RBIN . . . . . . . . . . . . . . . .        Number   0004
RBLEN. . . . . . . . . . . . . . . .        Number   0010
RBMAX. . . . . . . . . . . . . . . .        Number   0002
RBOUT. . . . . . . . . . . . . . . .        Number   0006
RBSTART. . . . . . . . . . . . . . .        L BYTE   0000     CODE
RBTAIL . . . . . . . . . . . . . . .        Number   000E
RBXOFF . . . . . . . . . . . . . . .        Number   0008
RBXON. . . . . . . . . . . . . . . .        Number   000A
RCVBRK . . . . . . . . . . . . . . .        Alias    BIT0
RCVOFF . . . . . . . . . . . . . . .        Alias    BIT5
RCVXOF . . . . . . . . . . . . . . .        Alias    BIT6
RDNVMCOM . . . . . . . . . . . . . .        L NEAR   0000     CODE     Global
RI . . . . . . . . . . . . . . . . .        Alias    BIT0
RLSD . . . . . . . . . . . . . . . .        Alias    BIT4
RST7201. . . . . . . . . . . . . . .        Number   0018
RTS. . . . . . . . . . . . . . . . .        Alias    BIT3
SPDI . . . . . . . . . . . . . . . .        Alias    BIT1
SPSEL. . . . . . . . . . . . . . . .        Alias    BIT0
SR1. . . . . . . . . . . . . . . . .        Number   0001
SR2. . . . . . . . . . . . . . . . .        Number   0002
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE     Symbols-4
COMNVM


SR3. . . . . . . . . . . . . . . . .        Number   0003
SRLSD. . . . . . . . . . . . . . . .        Alias    BIT1
SRTS . . . . . . . . . . . . . . . .        Alias    BIT1
STPBMAX. . . . . . . . . . . . . . .        Number   0003
SWSTOP . . . . . . . . . . . . . . .        Number   0097
SWXOFF . . . . . . . . . . . . . . .        Number   0094
SX7201 . . . . . . . . . . . . . . .        Number   0010
XA7201 . . . . . . . . . . . . . . .        Number   0028
XABAUD . . . . . . . . . . . . . . .        Number   0021
XAMODEM. . . . . . . . . . . . . . .        Number   00FF
XAS7201. . . . . . . . . . . . . . .        Number   002A
XB7201 . . . . . . . . . . . . . . .        Number   0029
XBCOMM . . . . . . . . . . . . . . .        Number   00A1
XBMODEM. . . . . . . . . . . . . . .        Number   0002
XBS7201. . . . . . . . . . . . . . .        Number   002B
XMTBRK . . . . . . . . . . . . . . .        Alias    BIT3
XMTXOF . . . . . . . . . . . . . . .        Alias    BIT2
XMTXON . . . . . . . . . . . . . . .        Alias    BIT1
?N . . . . . . . . . . . . . . . . .        Number   0010

Warning Severe
Errors  Errors
0       0
```

```
COMNVM

Symbol Cross Reference                   (# is definition)       Cref-1

?N . . . . . . . . . . . . . . . .  21#     21      21#     21      21#     21      21#     21      21#     21      21#     21      21#     21
                                    21#     21      21#     21      21#     21      21#     21      21#     21      21#     21      21#     21
                                    21#     21      21      21#     21      21#     21      21#     21      21#     21      21#     21      21#
                                    21      21#     21      21#     21      21#     21      21#     21      21#     21      21#     21      21#
                                    21      21#     21      21#     21      21#     21      21#     21      21#     21      21#     21      21#
                                    21      21#     21      21#     21      21#     21      21#     21      21#     21      21#     21      21#
                                    21      21#     21      21#     21      21#     21      21#     21      21#     21      21#     21      21#
                                    21      21#     21      21#     21#     21      21#     21      21#     21      21#     21      21#     21
                                    21#     21      21#     21      21#     21      21#     21      21#     21

A7201. . . . . . . . . . . . . . .  21#
ABAUD. . . . . . . . . . . . . . .  21#
AMODEM . . . . . . . . . . . . . .  21#
APPXOF . . . . . . . . . . . . . .  21#
AS7201 . . . . . . . . . . . . . .  21#
ASCSUB . . . . . . . . . . . . . .  21#

B7201. . . . . . . . . . . . . . .  21#
BAKGRND. . . . . . . . . . . . . .  21#
BAUDMAX. . . . . . . . . . . . . .  21#
BBAUD. . . . . . . . . . . . . . .  21#
BIT0 . . . . . . . . . . . . . . .  21#     21      21      21      74      81
BIT1 . . . . . . . . . . . . . . .  21#     21      21      21      21
BIT2 . . . . . . . . . . . . . . .  21#     21      21      21
BIT3 . . . . . . . . . . . . . . .  21#     21      21
BIT4 . . . . . . . . . . . . . . .  21#     21      21
BIT5 . . . . . . . . . . . . . . .  21#     21      21
BIT6 . . . . . . . . . . . . . . .  21#     21      21
BIT7 . . . . . . . . . . . . . . .  21#     21
BMODEM . . . . . . . . . . . . . .  21#
BRKERR . . . . . . . . . . . . . .  21#
BS7201 . . . . . . . . . . . . . .  21#

CBDATB . . . . . . . . . . . . . .  21#     111     112     127
CBLEN. . . . . . . . . . . . . . .  21#
CBMODE . . . . . . . . . . . . . .  21#
CBPORT . . . . . . . . . . . . . .  21#
CBPRTY . . . . . . . . . . . . . .  21#     116     117     131
CBRADR . . . . . . . . . . . . . .  21#
CBRCVB . . . . . . . . . . . . . .  21#     103     104     141
CBRXOF . . . . . . . . . . . . . .  21#     84      86
CBSIZE . . . . . . . . . . . . . .  21#
CBSTART. . . . . . . . . . . . . .  21#
CBSTPB . . . . . . . . . . . . . .  21#     77      78      138
CBTXB. . . . . . . . . . . . . . .  21#     95      96      142
CBTXOF . . . . . . . . . . . . . .  21#     85      87
CBXOFF . . . . . . . . . . . . . .  21#
CBXON. . . . . . . . . . . . . . .  21#
CDTPTY . . . . . . . . . . . . . .  42#     106
CGROUP . . . . . . . . . . . . . .  16      18      18      18      18
CLK16X . . . . . . . . . . . . . .  21#
CODE . . . . . . . . . . . . . . .  16      17#     17      147
CR1. . . . . . . . . . . . . . . .  21#
CR17201. . . . . . . . . . . . . .  21#
CR1TXBE. . . . . . . . . . . . . .  21#
```

```
COMNVM

Symbol Cross Reference                   (# is definition)       Cref-2

CR2. . . . . . . . . . . . . . . .  21#
CR27201. . . . . . . . . . . . . .  21#
CR3. . . . . . . . . . . . . . . .  21#
CR4. . . . . . . . . . . . . . . .  21#
CR5. . . . . . . . . . . . . . . .  21#
CTS. . . . . . . . . . . . . . . .  21#

DATBCOM. . . . . . . . . . . . . .  32#     109     110
DATBPRT. . . . . . . . . . . . . .  35#     125     126
DATMAX . . . . . . . . . . . . . .  21#
DC1. . . . . . . . . . . . . . . .  21#
DC3. . . . . . . . . . . . . . . .  21#
DEVCOM . . . . . . . . . . . . . .  21#
DEVMAX . . . . . . . . . . . . . .  21#
DEVPRT . . . . . . . . . . . . . .  21#
DEVXCOM. . . . . . . . . . . . . .  21#
DSR. . . . . . . . . . . . . . . .  21#
DTR. . . . . . . . . . . . . . . .  21#

ENDTXBE. . . . . . . . . . . . . .  21#
EOI7201. . . . . . . . . . . . . .  21#
ERR7201. . . . . . . . . . . . . .  21#

FRMERR . . . . . . . . . . . . . .  21#

L1_1$. . . . . . . . . . . . . . .  75      77#
L1_2$. . . . . . . . . . . . . . .  82      84#
L1_3$. . . . . . . . . . . . . . .  93      95#
L1_4$. . . . . . . . . . . . . . .  101     103#
L1_9$. . . . . . . . . . . . . . .  136     138#

MODEMAX. . . . . . . . . . . . . .  21#

NVMCOM . . . . . . . . . . . . . .  28#     68
NVMPRT . . . . . . . . . . . . . .  30#     121
NVMSEG . . . . . . . . . . . . . .  38#     70
NVMXCOM. . . . . . . . . . . . . .  29#     69

OVRERR . . . . . . . . . . . . . .  21#

PARERR . . . . . . . . . . . . . .  21#
PBCOMM . . . . . . . . . . . . . .  46#     133
PC7201 . . . . . . . . . . . . . .  21#
PCBAUD . . . . . . . . . . . . . .  21#
PCCR1. . . . . . . . . . . . . . .  21#
PCCR2. . . . . . . . . . . . . . .  21#
PCCR3. . . . . . . . . . . . . . .  21#
PCCR4. . . . . . . . . . . . . . .  21#
PCCR5. . . . . . . . . . . . . . .  21#
PCDATB . . . . . . . . . . . . . .  21#
PCDFLT . . . . . . . . . . . . . .  21#
PCFAIL . . . . . . . . . . . . . .  21#
PCFLAG . . . . . . . . . . . . . .  21#
PCID . . . . . . . . . . . . . . .  21#
PCLEN. . . . . . . . . . . . . . .  21#
```

```
COMNVM

Symbol Cross Reference                 (# is definition)       Cref-3

PCMASK . . . . . . . . . . . . . .     21#
PCMODE . . . . . . . . . . . . . .     21#
PCMODM . . . . . . . . . . . . . .     21#
PCPRTY . . . . . . . . . . . . . .     21#
PCRADR . . . . . . . . . . . . . .     21#
PCRATE . . . . . . . . . . . . . .     21#
PCRCVA . . . . . . . . . . . . . .     21#
PCRCVB . . . . . . . . . . . . . .     21#
PCRCVF . . . . . . . . . . . . . .     21#
PCRXOF . . . . . . . . . . . . . .     21#
PCS7201. . . . . . . . . . . . . .     21#
PCSIZE . . . . . . . . . . . . . .     21#
PCSTART. . . . . . . . . . . . . .     21#
PCSTAT . . . . . . . . . . . . . .     21#
PCSTCA . . . . . . . . . . . . . .     21#
PCSTCF . . . . . . . . . . . . . .     21#
PCSTPB . . . . . . . . . . . . . .     21#
PCTXB. . . . . . . . . . . . . . .     21#
PCTXOF . . . . . . . . . . . . . .     21#
PCXMTA . . . . . . . . . . . . . .     21#
PCXMTF . . . . . . . . . . . . . .     21#
PCXOFF . . . . . . . . . . . . . .     21#
PCXON. . . . . . . . . . . . . . .     21#
PDTPTY . . . . . . . . . . . . . .     45#     122
PRTBAUD. . . . . . . . . . . . . .     34#     139      140
PRTYCOM. . . . . . . . . . . . . .     33#     114      115
PRTYMAX. . . . . . . . . . . . . .     21#
PRTYPRT. . . . . . . . . . . . . .     36#     129      130

RBCOMM . . . . . . . . . . . . . .     44#      97
RBCOUNT. . . . . . . . . . . . . .     21#
RBDFLT . . . . . . . . . . . . . .     21#
RBHEAD . . . . . . . . . . . . . .     21#
RBIN . . . . . . . . . . . . . . .     21#
RBLEN. . . . . . . . . . . . . . .     21#
RBMAX. . . . . . . . . . . . . . .     21#
RBOUT. . . . . . . . . . . . . . .     21#
RBSTART. . . . . . . . . . . . . .     21#
RBTAIL . . . . . . . . . . . . . .     21#
RBXOFF . . . . . . . . . . . . . .     21#
RBXON. . . . . . . . . . . . . . .     21#
RCVBRK . . . . . . . . . . . . . .     21#
RCVOFF . . . . . . . . . . . . . .     21#
RCVXOF . . . . . . . . . . . . . .     21#
RDNVMCOM . . . . . . . . . . . . .     25       66#
RI . . . . . . . . . . . . . . . .     21#
RLSD . . . . . . . . . . . . . . .     21#
RST7201. . . . . . . . . . . . . .     21#
RTS. . . . . . . . . . . . . . . .     21#

SEQ. . . . . . . . . . . . . . . .     21      21       21       21       21       21       21       21       21       21       21       21       21       21
                                       21      21       21       21       21       21       21       21       21       21       21       21       21       21
                                       21      21       21       21       21       21       21       21       21       21       21       21       21       21
                                       21      21       21       21       21       21       21       21       21       21       21       21       21       21
                                       21      21
```

```
COMNVM

Symbol Cross Reference                 (# is definition)       Cref-4

SPDI . . . . . . . . . . . . . . .     21#
SPSEL. . . . . . . . . . . . . . .     21#
SR1. . . . . . . . . . . . . . . .     21#
SR2. . . . . . . . . . . . . . . .     21#
SR3. . . . . . . . . . . . . . . .     21#
SRLSD. . . . . . . . . . . . . . .     21#
SRTS . . . . . . . . . . . . . . .     21#
STPBMAX. . . . . . . . . . . . . .     21#
SWSTOP . . . . . . . . . . . . . .     40#     74
SWXOFF . . . . . . . . . . . . . .     41#     81
SX7201 . . . . . . . . . . . . . .     21#

TABLE. . . . . . . . . . . . . . .     21      21       21

XA7201 . . . . . . . . . . . . . .     21#
XABAUD . . . . . . . . . . . . . .     21#
XAMODEM. . . . . . . . . . . . . .     21#
XAS7201. . . . . . . . . . . . . .     21#
XB7201 . . . . . . . . . . . . . .     21#
XBCOMM . . . . . . . . . . . . . .     43#     89
XBMODEM. . . . . . . . . . . . . .     21#
XBS7201. . . . . . . . . . . . . .     21#
XMTBRK . . . . . . . . . . . . . .     21#
XMTXOF . . . . . . . . . . . . . .     21#
XMTXON . . . . . . . . . . . . . .     21#
```

```
        PAGE    ,132
        TITLE   COMBIOS
        SUBTTL  COMMUNICATION PORTION OF VERSION 2 BIOS
        NAME    COMBIOS

;
;
;       COMPANY CONFIDENTIAL
;       Copyright (C) 1982, 1983 Digital Equipment Corporation
;       All rights reserved.
;
;
;       COMBIOS.A86     version /V00-01/        APRIL 11, 1983  DEVELOPMENT
;
;                       version /V00-02/        May 25, 1983
;
;                       1 - altered the interfaces so that the comm
;                           control block does not have to be in user DS.
;                       2 - Take out the setting of the vectors in
;                           INITCOM.
;                       3 - Change GETPCB so that there is an error if
;                           application tries to access non existng optional
;                           comm board.
;
;                       version /V00-03/        Jun 01, 1983
;
;                       1 - Compensate for the difference in hardware for
;                           ports on the mother board and the optional comm
;                           board. The internal/external clock is differently
;                           for both 7201. This is compensated by always
;                           setting the modem signals to deasserted state
;                           on cold boot.
;
;                       2 - Fix the special receive condition for PORT B
;                           of both 7201.
;
;                       version /V00-04/        Jun 13,1983
;
;                       1 - Fix printer baud rate problem
;                       2 - Fix programming of 7M and 7S for all ports
;
;                       version /V00-05/        Jun 20, 1983
;
;                       1 - Add error handling to programming of devices
;                       2 - Do not enable interrupts in the driver
;                       3 - Mask out hi order bit before checking for
;                           xon/xoff type control characters.
;
;               NOTE :  this module was to begin with, interruptable, this
;                       may be desirable in the future, so the places where
;                       interrupts are enabled and disabled in the driver
;                       are just commented out with the following mark so
;                       that it can be easily changed :
```

```
;
;                      %1%
;
;                       6-23-83  Fix the interfaces to read and set modem
;                       signals.
;
;
;                       6-27-83
;                       Add the hack as requested.
;                       That is add BIOS call to take over 1/2 7201 !!
;
;                       7-01-83
;                       Add status change interrupt service calls
;
;------------------------------------------------------------------------
```

```


                                        ;       SECT    CODE_SEG,REL

                                        CGROUP  GROUP   CODE
         0000                           CODE    SEGMENT BYTE PUBLIC 'CODE'
                                        ASSUME  CS:CGROUP, DS:CGROUP, ES:CGROUP, SS:CGROUP

                                        ;       NLIST
```

```
                                                .LIST

                                                EXTRN   PCCOM:BYTE       ; port control block for comm port
                                                EXTRN   PCXCOM:BYTE      ; port control block for extended comm port
                                                EXTRN   PCPRT:BYTE       ; port control block for printer port

                                                EXTRN   DFLTCOM:BYTE     ; default comm control block for comm
                                                EXTRN   DFLTXCOM:BYTE    ; default comm control block for Xcomm
                                                EXTRN   DFLTPRT:BYTE     ; default comm control block for printer port

                                                EXTRN   XBAUD1:BYTE      ; baud rate translation table for Comm & Xcomm
                                                EXTRN   XBAUD2:BYTE      ; baud rate translation table for printer

                                                EXTRN   XOPTION:BYTE     ; this is zero, when optional comm port present
                                                EXTRN   RDNVMCOM:NEAR    ; read NVM and set defaults

                                                EXTRN   HVECTOR:WORD     ;addresses of the user defined service routines

                                                PUBLIC  RUPT7201         ; address of mother board 7201 interrupt
                                                PUBLIC  XRUPT7201        ; address of optional board 7201 interrupt
                                                PUBLIC  INITCOM          ; cold start initialize of all 7201
                                                PUBLIC  WINITCOM         ; warm start initialize of all 7201

                                                PUBLIC  PRG7201          ; reprogram 7201
                                                PUBLIC  DFLT7201         ; reprogram 7201 to default values
                                                PUBLIC  DFLTBUF          ; reset receive char buffer to default area
                                                PUBLIC  RDSETUP          ; read device setup information
                                                PUBLIC  RCVENA           ; receiver enable
                                                PUBLIC  RCVDIS           ; receiver disable
                                                PUBLIC  INSTAT           ; get input status
                                                PUBLIC  INCHAR           ; get input character
                                                PUBLIC  GETCHAR          ; get input char return when available
                                                PUBLIC  OUTSTAT          ; get output status
                                                PUBLIC  OUTCHAR          ; write character
                                                PUBLIC  PUTCHAR          ; write chartcer, return when sucessful
                                                PUBLIC  OUTNOW           ; write character immediately
                                                PUBLIC  RDMODEM          ; read modem signals
                                                PUBLIC  SETMODEM         ; set modem signals
                                                PUBLIC  BRKON            ; transmit break
                                                PUBLIC  BRKOFF           ; cease transmission of break
                                                PUBLIC  RCVINT           ; set receive character interrupt service
                                                PUBLIC  RCVCANCEL        ; cancel receive char interrupt service
                                                PUBLIC  XMTMTY           ; set transmit buffer empty service
                                                PUBLIC  XMTCANCEL        ; cancel transmit buffer empty service

                                                PUBLIC  HACK7201         ; setup 1/2 vector
                                                PUBLIC  VECT7201         ; clean up redirection vector

                                                PUBLIC  STATCHG          ; status change service call
                                                PUBLIC  STCANCEL         ; cancel status change service calls
```

```



                                      SUBTTL  INTERRUPT SERVICE ROUTINES
                               ;------------------------------------------------------------------

                               ;
                               ;       R U P T 7 2 0 1         &       X R U P T 7 2 0 1
                               ;
                               ; This is the interrupt handler routine for the 7201, on the mother
                               ; board, and the equivalent for the extended comm.
                               ;
                               ; These 2 routines are setup to share some common code.
                               ;
                               ; Status register 2 of the 7201 will indicate the reason for the
                               ; interrupt. This will be used as a table lookup to call the appropriate
                               ; service routine.
                               ;
                               ; Since the only difference between the 2 interrupt handlers are the
                               ; service routine table and the 7201 data port address, the two routines
                               ; can be setup to use some common code.
                               ;
                               ;
                               ; All the service routines (tables INT7201, and XINT7201) must follow the
                               ; the following interfaces:
                               ;
                               ; ON ENTRY TO THE SERVICE ROUTINES:
                               ;
                               ;             AX, BX, CX, DX, DS, ES, BP are saved and can be used in
                               ;                                         in any way.
                               ;
                               ;             DX contains the address of port A of the corresponding
                               ;                7201.
                               ;
                               ;             DS will point to the BIOS data area (ie same as CS)
                               ;
                               ;------------------------------------------------------------------
0000                           RUPT7201:                              ; comm service routine
0000  53                              PUSH    BX                      ; save registers
0001  52                              PUSH    DX                      ;       .
0002  BB 00EA R                       MOV     BX,OFFSET INT7201       ; get the service routine table
0005  BA 0043                         MOV     DX,BS7201               ; get 7201 port B, status/control reg
0008  EB 08                           JMP     SHORT RUPTCOMMON        ; > go to common table lookup portion

000A                           XRUPT7201:                             ; (Xtended comm service routine)
000A  53                              PUSH    BX                      ; save registers
000B  52                              PUSH    DX                      ;       .
000C  BB 00FA R                       MOV     BX,OFFSET XINT7201      ; get service routine table
000F  BA 002B                         MOV     DX,XBS7201              ; get 7201 port B, status/control reg
```

```



0012                           RUPTCOMMON:                            ; common code for both service routines
0012  50                              PUSH    AX                      ; save rest of registers needed
0013  1E                              PUSH    DS                      ;        .
0014  8C C8                           MOV     AX,CS                   ; set up DS to point to BIOS data area
0016  8E D8                           MOV     DS,AX                   ;        .
0018  89 26 00A6 R                    MOV     SAVE_SP,SP              ;Save stack pointers
001C  8C 16 00A8 R                    MOV     SAVE_SS,SS
0020  8E D0                           MOV     SS,AX                   ;Copy CS: => SS:
0022  BC 00EA R                       MOV     SP,OFFSET COMM_STACK    ;Setup local stack
0025  51                              PUSH    CX                      ;        .
0026  56                              PUSH    SI                      ;        .
0027  57                              PUSH    DI                      ;        .
0028  06                              PUSH    ES                      ;        .
0029  55                              PUSH    BP                      ;        .

002A                           NXTINT:
002A  33 C0                           XOR     AX,AX                   ; make sure certain idiot programs
002C  EE                              OUT     DX,AL                   ; don't mess reg pointer
002D  B0 02                           MOV     AL,2                    ; set up to read status register 2 B
002F  EE                              OUT     DX,AL                   ;        .(note AH still 0)
0030  EC                              IN      AL,DX                   ;        .

0031  3C 07                           CMP     AL,7                    ; Range check the value to make sure
0033  76 30                           JBE     NXTINT1                 ; (should be 0 to 7)

0035  83 FA 43                        CMP     DX,BS7201               ; check if its comm & printer 7201
0038  75 1A                           JNZ     L1_1$                   ; If so then
003A  BA 0042                         MOV     DX,OFFSET AS7201        ;        reset whole 7201
003D  E8 0904 R                       CALL    RESET7201               ;        .
0040  BB 0000 E                       MOV     BX,OFFSET PCCOM         ;        reprogram port A of 7201
0043  FE 47 26                        INC     BYTE PTR PCFAIL[BX]     ;        count up # hardware failures
0046  E8 0083 R                       CALL    CR7201                  ;        (put back images of 7201 reg)
0049  BB 0000 E                       MOV     BX,OFFSET PCPRT         ;        reprogram port B of 7201
004C  FE 47 26                        INC     BYTE PTR PCFAIL[BX]     ;        count up # hardware failures
004F  E8 0083 R                       CALL    CR7201                  ;        (put back images of 7201 reg)
0052  EB 1D                           JMP     SHORT INTEXIT           ; exit from interrupt

0054                           L1_1$:                                 ; (problem in optional comm port)
0054  BA 002A                         MOV     DX,OFFSET XAS7201       ; reset whole 7201
0057  E8 0904 R                       CALL    RESET7201               ;        .
005A  BB 0000 E                       MOV     BX,OFFSET PCXCOM        ; reprogram port A of 7201
005D  FE 47 26                        INC     BYTE PTR PCFAIL[BX]     ; count up # hardware failures
0060  E8 0083 R                       CALL    CR7201                  ; (put back images of 7201 reg)
0063  EB 0C                           JMP     SHORT INTEXIT           ; exit from interrupt

0065                           NXTINT1:
                               ; %1%  STI                             ; enable interrupts
0065  D1 E0                           SHL     AX,1                    ; multiply to use it as table offsets
0067  03 D8                           ADD     BX,AX                   ; find address of service routine
0069  52                              PUSH    DX                      ; save port B address
```

```
006A  FF 17                        CALL    WORD PTR [BX]       ; call service routine
006C  5A                           POP     DX                  ; restore port B address
006D  4A                           DEC     DX                  ; get address of status/control reg 2A
006E  B0 38                        MOV     AL,EOI7201          ; issue "end of interrupt" to 7201
0070  EE                           OUT     DX,AL               ; write it to control reg 0 (port A)

0071                       INTEXIT:
0071  5D                           POP     BP                  ; restore registers
0072  07                           POP     ES                  ;        .
0073  5F                           POP     DI                  ;        .
0074  5E                           POP     SI                  ;        .
0075  59                           POP     CX                  ;        .
0076  8E 16 00A8 R                 MOV     SS,SAVE_SS          ;Restore entry stack
007A  8B 26 00A6 R                 MOV     SP,SAVE_SP          ; ...
007E  1F                           POP     DS                  ;        .
007F  58                           POP     AX                  ;        .
0080  5A                           POP     DX                  ;        .
0081  5B                           POP     BX                  ;        .
0082  CF                           IRET                        ; all done, adios!

0083                       CR7201:
0083  8B 57 12                     MOV     DX,PCS7201[BX]      ; get control register of 7201
0086  8A 47 0E                     MOV     AL,PCCR4[BX]        ; do CR4 first
0089  B4 04                        MOV     AH,4                ;        .
008B  E8 0269 R                    CALL    WRITE7201           ;        .
008E  8A 47 0B                     MOV     AL,PCCR1[BX]        ; do CR1
0091  B4 01                        MOV     AH,1                ;        .
0093  E8 0269 R                    CALL    WRITE7201           ;        .
0096  8A 47 0F                     MOV     AL,PCCR5[BX]        ; do CR5
0099  B4 05                        MOV     AH,5                ;        .
009B  E8 0269 R                    CALL    WRITE7201           ;        .
009E  8A 47 0D                     MOV     AL,PCCR3[BX]        ; do CR3
00A1  B4 03                        MOV     AH,3                ;        .
                                   CALRET  WRITE7201           ;        .
00A3  E9 0269 R            +       JMP     WRITE7201

                           ;Internal stack and saved pointers

00A6  0000                 SAVE_SP DW      0
00A8  0000                 SAVE_SS DW      0

00AA     40 [                      DB 64 DUP (?)
                ??
                      ]

 = 00EA                    COMM_STACK      EQU $

```

```


                           ;---------------------------------------------------------------
                           ;
                           ;             I N T 7 2 0 1     &     X I N T 7 2 0 1
                           ;
                           ; This is a table of offsets for the interrupt service routines for
                           ; the NEC 7201 on the mother board, and the optional comm board.
                           ;
                           ; The offsets are set up to use the 7201 status register 2 (status
                           ; affect vector) so that service routines can be fastest reached with
                           ; the least code.
                           ;
                           ; NOTE : these 2 tables have to be contiguous
                           ;
                           ;----------------------------------------------------------------

00EA                       INT7201 LABEL   WORD
00EA  021F R                       DW      BTXEMT          ; Port B transmit buffer empty
00EC  028B R                       DW      BSTAT           ; Port B external status change
00EE  0143 R                       DW      BRXDAT          ; Port B receive character available
00F0  023E R                       DW      BRXSPC          ; Port B special receive condition
00F2  0215 R                       DW      ATXEMT          ; Port A transmit buffer empty
00F4  0295 R                       DW      ASTAT           ; Port A external status change
00F6  013E R                       DW      ARXDAT          ; Port A receive character available
00F8  0248 R                       DW      ARXSPC          ; Port A special receive condition

00FA                       XINT7201 LABEL  WORD
00FA  021A R                       DW      XBTXEMT         ; Port B transmit buffer empty
00FC  0290 R                       DW      XBSTAT          ; Port B external status change
00FE  0148 R                       DW      XBRXDAT         ; Port B receive character available
0100  0243 R                       DW      XBRXSPC         ; Port B special receive condition
0102  0276 R                       DW      XATXEMT         ; Port A transmit buffer empty
0104  027A R                       DW      XASTAT          ; Port A external status change
0106  0286 R                       DW      XARXDAT         ; Port A receive character available
0108  0281 R                       DW      XARXSPC         ; Port A special receive condition
```

```


                                  ;-----------------------------------------------------------------
                                  ;
                                  ;       DEFAULT TABLES OF SERVICE ROUTINES FOR THE 4 DEVICES
                                  ;
                                  ; PORTA, PORTB are for the 7201 on the mother board
                                  ; PORTXA, PORTXB, are for the 7201 on the optional comm port
                                  ;
                                  ;-----------------------------------------------------------------
 010A                             PORTB   LABEL   WORD
 010A  021F R                             DW      BTXEMT            ; Port B transmit buffer empty
 010C  028B R                             DW      BSTAT             ; Port B external status change
 010E  0143 R                             DW      BRXDAT            ; Port B receive character available
 0110  023E R                             DW      BRXSPC            ; Port B special receive condition

 0112                             PORTA   LABEL   WORD
 0112  0215 R                             DW      ATXEMT            ; Port A transmit buffer empty
 0114  0295 R                             DW      ASTAT             ; Port A external status change
 0116  013E R                             DW      ARXDAT            ; Port A receive character available
 0118  0248 R                             DW      ARXSPC            ; Port A special receive condition

 011A                             PORTXB  LABEL   WORD
 011A  021A R                             DW      XBTXEMT           ; Port B transmit buffer empty
 011C  0290 R                             DW      XBSTAT            ; Port B external status change
 011E  0148 R                             DW      XBRXDAT           ; Port B receive character available
 0120  0243 R                             DW      XBRXSPC           ; Port B special receive condition

 0122                             PORTXA  LABEL   WORD
 0122  0276 R                             DW      XATXEMT           ; Port A transmit buffer empty
 0124  027A R                             DW      XASTAT            ; Port A external status change
 0126  0286 R                             DW      XARXDAT           ; Port A receive character available
 0128  0281 R                             DW      XARXSPC           ; Port A special receive condition
```

```


 012A                             HACKA:
 012A  FF 1E 0000 E                       CALL    DWORD PTR HVECTOR
 012E  C3                                 RET

 012F                             HACKB:
 012F  FF 1E 0004 E                       CALL    DWORD PTR HVECTOR+4
 0133  C3                                 RET

 0134                             HACKXB:
 0134  FF 1E 0008 E                       CALL    DWORD PTR HVECTOR+8
 0138  C3                                 RET

 0139                             HACKXA:
 0139  FF 1E 000C E                       CALL    DWORD PTR HVECTOR+12
 013D  C3                                 RET

```

```


                            ;-----------------------------------------------------------------
                            ;
                            ;                        A R X D A T
                            ;
                            ; Receive character interrupt service routines.
                            ;
                            ; All three ports uses the same service routine, RECEIVER.
                            ;
                            ; On entry to RECEIVER, DS:BX must point to the appropriate Port Control Block.
                            ;
                            ;-----------------------------------------------------------------
013E                        ARXDAT:                                ; Port A receive character available
013E  BB 0000 E                     MOV     BX,OFFSET PCCOM ; get offset of the COMM port control block
0141  EB 08                         JMP     SHORT RECEIVER  ; go to common service routine.




0143                        BRXDAT:                                ; Port B receive character available
0143  BB 0000 E                     MOV     BX,OFFSET PCPRT ; get offset of the printer port control block
0146  EB 03                         JMP     SHORT RECEIVER  ; go to common service routine




0148                        XBRXDAT:                               ; Port A receive character available
0148  BB 0000 E                     MOV     BX,OFFSET PCXCOM ; get the extended COMM port control block
                            ;       JMP     SHORT RECEIVER  ; go to common service routine

                            ; +++++++++ note this code falls thru ++++++++++++++++++ !!!!!!!! @@@@@ ####

```

```


                            ;-----------------------------------------------------------------
                            ;
                            ;                        R E C E I V E R
                            ;
                            ; This is the common receive character routine used to process the receive
                            ; character interrupt; for all 3 ports (port A , B on the 7201 on the mother
                            ; board, and port B on the extended comm port).
                            ;
                            ;
                            ;       ENTRY CONDITIONS:       DS:BX   points to the port control block
                            ;
                            ;       EXIT CONDITIONS:
                            ;
                            ; !! NOTE !!    REC3 is an embedded entry point from special receive characters
                            ;
                            ;-----------------------------------------------------------------
014B                        RECEIVER:
014B  8B 57 10                      MOV     DX,PC7201[BX]             ; get data register of the 7201 PORT
014E  32 E4                         XOR     AH,AH                     ; Assume no error in character
0150  EC                            IN      AL,DX                     ; read the character
0151  80 7F 2F 02                   CMP     BYTE PTR PCRXOF[BX],2     ; check if receiv auto XON/XOFF enabled
0155  75 1B                         JNE     REC3                      ; > if not, do not filter these chars
0157  8A D0                         MOV     DL,AL                     ; mask out hi order bit for xon/xoff
0159  80 E2 7F                      AND     DL,07FH                   ;        protocol charaacter !!!
015C  3A 57 2D                      CMP     DL,PCXON[BX]              ; check if we have XON character
015F  75 06                         JNZ     REC1                      ; If so clear flag for xoff received
0161  80 67 14 BF                   AND     BYTE PTR PCFLAG[BX],NOT RCVXOF  ;
0165  EB 7C                         JMP     SHORT RECX                ; > exit
0167  3A 57 2E              REC1:   CMP     DL,PCXOFF[BX]             ; check if we have XOFF
016A  75 06                         JNZ     REC3                      ; If not treat it like other characters
016C  80 4F 14 40                   OR      BYTE PTR PCFLAG[BX],RCVXOF ; else set flag, xoff received
0170  EB 71                         JMP     SHORT RECX                ;
                            ;
                            ; This is an embedded entry point for special receive characters !!
                            ; AX has character from the 7201 (not yet modified according to the mask)

0172                        REC3:
0172  8B 57 15                      MOV     DX,PCMASK[BX]             ; get the mask
0175  22 C6                         AND     AL,DH                     ; alter the incoming character
0177  0A C2                         OR      AL,DL                     ;
0179  C4 6F 33                      LES     BP,DWORD PTR PCRADR[BX] ; get address of the receive buffer
017C  50                            PUSH    AX                        ; save character to be written
                            ; %1%   CLI                               ; make sure interrupts are disabled here
017D  26: 8B 46 00                  MOV     AX,ES:RBCOUNT[BP]         ; get # characters in the buffer
0181  26: 3B 46 02                  CMP     AX,ES:RBMAX[BP]           ; check to see if we are full
0185  26: 8B 7E 04                  MOV     DI,ES:RBIN[BP]            ; ( first get address for next char in)
0189  75 15                         JNZ     REC6                      ; If not go store character in buffer
                            ;
                            ; buffer full, replace last character with ascii sub
                            ;
018B  58                            POP     AX                        ; clean up stack
```

```
018C  26: 3B 7E 0C                    CMP     DI,ES:RBHEAD[BP]          ; If next char in points to top of buffer
0190  75 06                           JNZ     REC4                      ;        .
0192  26: 8B 7E 0E                    MOV     DI,ES:RBTAIL[BP]          ;        set pointer to tail of buffer
0196  EB 02                           JMP     SHORT REC5                ;        go put ascii sub there
0198                          REC4:                                     ; Else,
0198  4F                              DEC     DI                        ;        get address of last input character
0199  4F                              DEC     DI                        ;        .
019A                          REC5:                                     ; (now place ascii sub into buffer)
019A  26: C6 05 1A                    MOV     BYTE PTR ES:[DI],ASCSUB ; put sub character there
019E  EB 43                           JMP     SHORT RECX                ; return to caller
01A0                          REC6:
01A0  40                              INC     AX                        ; count up # characters in the buffer
01A1  26: 89 46 00                    MOV     ES:RBCOUNT[BP],AX         ; save new value for count
01A5  58                              POP     AX                        ; restore character
01A6  26: 89 05                       MOV     ES:[DI],AX                ; save new character
01A9  26: 3B 7E 0E                    CMP     DI,ES:RBTAIL[BP]          ; check if at end of buffer space
01AD  75 06                           JNZ     REC7                      ; If so update pointer to the beginning
01AF  26: 8B 7E 0C                    MOV     DI,ES:RBHEAD[BP]          ;     ;        .
01B3  EB 02                           JMP     SHORT REC8                ;        .
01B5                          REC7:                                     ; Else, increment input pointer
01B5  47                              INC     DI                        ;        to next available buffer space
01B6  47                              INC     DI                        ;        .
01B7                          REC8:                                     ; (now DI points to next avail  space)
01B7  26: 89 7E 04                    MOV     ES:RBIN[BP],DI            ; update new address
01BB  80 7F 30 02                     CMP     BYTE PTR PCTXOF[BX],2     ; check if transmit auto xoff enabled
01BF  75 22                           JNE     RECX                      ; If not, go exit
01C1  26: 8B 46 00                    MOV     AX,ES:RBCOUNT[BP]         ; get count of chars in buffer again
01C5  26: 3B 46 02                    CMP     AX,ES:RBMAX[BP]           ; check if buffer now full
01C9  74 0C                           JZ      REC9                      ; > if full see if xoff should be sent
01CB  26: 3B 46 08                    CMP     AX,ES:RBXOFF[BP]          ; check if over hi water mark
01CF  72 12                           JB      RECX                      ; > if not all done, go exit
01D1  F6 47 14 04                     TEST    BYTE PTR PCFLAG[BX],XMTXOF ; IF over hi water mark, but auto
01D5  75 0C                           JNZ     RECX                      ;        >xoff already sent, then return
01D7                          REC9:                                     ; (attempt to send auto xoff comes here)
                              ; %1%   STI                               ; enable interrupts again
01D7  8A 67 2E                        MOV     AH,PCXOFF[BX]             ; go send an xoff
01DA  E8 01F9 R                       CALL    AUTOXMT                   ;        .
01DD  74 04                           JZ      RECX                      ; If not sucessful go exit
01DF  80 4F 14 04                     OR      BYTE PTR PCFLAG[BX],XMTXOF ;     mark xoff sent by driver
01E3                          RECX:
                              ; %1%   STI                               ; enable interrupts
01E3  80 7F 17 01                     CMP     BYTE PTR PCRCVF[BX],1     ; check if there is a routine to call
01E7  75 0F                           JNE     RCVX                      ; > exit if none (or one in progress)
01E9  FE 47 17                        INC     BYTE PTR PCRCVF[BX]       ; mark that a routine is being called
01EC  FF 5F 18                        CALL    DWORD PTR PCRCVA [BX]     ; call service routine
01EF  FE 4F 17                        DEC     BYTE PTR PCRCVF[BX]       ; mark that routine has returned
01F2  79 04                           JNS     RCVX                      ; check that its not cancelled
01F4  C6 47 17 00                     MOV     BYTE PTR PCRCVF[BX],0     ; If it has, reset value to zero
01F8  C3                      RCVX:   RET                               ; return to caller
```

```


                              ;-------------------------------------------------------------------
                              ;
                              ;                          A U T O X M T
                              ;
                              ; This routine is used by the drivers to send the auto xon or xoff
                              ; characters.
                              ;
                              ; This has to be a special routine because when a driver sends
                              ; these control characters, different rules applies.
                              ;
                              ; ENTRY CONDITIONS:                DS:BX   points to port control block
                              ;                                  AH      has the character to be transmitted
                              ;
                              ; EXIT CONDITIONS:                 Z       set if character cannot be
                              ;                                          transmitted
                              ;
                              ;                                  Z       cleared if it is sucessfully placed
                              ;                                          into the transmit buffer.
                              ;                                  DX, AX  destroyed.
                              ;
                              ;--------------------------------------------------------------------
01F9                          AUTOXMT:
01F9  8B 57 12                        MOV     DX,PCS7201[BX]            ; get address of the 7201 status reg
01FC  9C                              PUSHF                             ; make sure interrupts are off
                              ; %1%   CLI                               ;        .
01FD  EC                              IN      AL,DX                     ; check if transmit buffer empty
01FE  A8 04                           TEST    AL,BIT2                   ;        .
0200  74 0F                           JZ      L2_10$                    ; > go exit if buffer not empty(ZF set)
0202  8A C4                           MOV     AL,AH                     ; put the character in transmit buffer
0204  22 47 16                        AND     AL,PCMASK+1[BX]           ;        mask outgoing data
0207  0A 47 15                        OR      AL,PCMASK[BX]             ;        .
020A  4A                              DEC     DX                        ;        (get address of data register)
020B  4A                              DEC     DX                        ;        .(note Z is now cleared if)
020C  EE                              OUT     DX,AL                     ;        .(DX not = 2)
020D  9D                              POPF                              ; restore flags
020E  0C 80                           OR      AL,80H                    ; clear Z flag
0210  C3                              RET                               ; return to caller

0211  9D                      12_10$: POPF                              ; restore flags
0212  32 C0                           XOR     AL,AL                     ; return with z set
0214  C3                              RET                               ; return to caller
```

```


                            ;-----------------------------------------------------------------
                            ;
                            ;                           A T X E M T
                            ;
                            ; Transmit buffer empty interrupt service for channel A, on the mother
                            ; board.
                            ;
                            ;-----------------------------------------------------------------
0215                        ATXEMT:                                 ; Port A transmit buffer empty
0215  BB 0000 E                         MOV     BX,OFFSET PCCOM ; get offset of the COMM port control block
0218  EB 08                             JMP     SHORT TXBUFFER  ; go to common routine to complete service


                            ;-----------------------------------------------------------------
                            ;
                            ;                           X B T X E M T
                            ;
                            ; Transmit buffer empty interrupt service for channel A, on the mother
                            ; board.
                            ;
                            ;-----------------------------------------------------------------
021A                        XBTXEMT:                                ; Port B transmit buffer empty
021A  BB 0000 E                         MOV     BX,OFFSET PCXCOM ; get the extended COMM port control block
021D  EB 03                             JMP     SHORT TXBUFFER  ; go to common routine for tranmit buffer empty


                            ;---------------------------------------------------------------
                            ;
                            ;                           B T X E M T
                            ;
                            ; Transmit buffer empty interrupt on port B, in the 7201 on the mother
                            ; board.
                            ;
                            ;
                            ;---------------------------------------------------------------
021F                        BTXEMT:                                     ; Port B transmit buffer empty
021F  BB 0000 E                         MOV     BX,OFFSET PCPRT         ; get the printer port control block
0222                        TXBUFFER:                                   ; (Common code for all transmit buffer
                                                                        ;  empty service interrupt)
0222  80 7F 1C 01                       CMP     BYTE PTR PCXMTF[BX],1   ; check if there is a routine to call
0226  75 0F                             JNE     TXBUFX                  ; > exit if none (or one in progress)
0228  FE 47 1C                          INC     BYTE PTR PCXMTF[BX]     ; mark that a routine is being called
022B  FF 5F 1D                          CALL    DWORD PTR PCXMTA [BX]   ; call service routine
022E  FE 4F 1C                          DEC     BYTE PTR PCXMTF[BX]     ; mark that routine has returned
0231  79 04                             JNS     TXBUFX                  ; check that its not cancelled
0233  C6 47 1C 00                       MOV     BYTE PTR PCXMTF[BX],0   ; If it has, reset value to zero
0237  8B 57 12                  TXBUFX: MOV     DX,PCS7201[BX]          ; get control register address
023A  B0 28                             MOV     AL,ENDTXBE              ; issue end of transmit buffer empty
023C  EE                                OUT     DX,AL                   ;
023D  C3                                RET                             ; return to caller
```

```


                            ;-----------------------------------------------------------------
                            ;
                            ;                           S P R E C E I V E
                            ;
                            ; This is the common routine that is used to process the special receive
                            ; character interrupt condition , for all 3 ports (port A and B on the
                            ; 7201 on the mother board, and port B on the extended comm port).
                            ;
                            ;       ENTRY CONDITIONS:       DS:BX   points to the port control block
                            ;
                            ;       EXIT CONDITIONS:
                            ;
                            ;-----------------------------------------------------------------
023E                        BRXSPC:                                 ; Port B (mother board) special receive condition
023E  BB 0000 E                         MOV     BX,OFFSET PCPRT ; get offset of the printer port control block
0241  EB 10                             JMP     SHORT SPREC     ; go to common routine to handle it.

0243                        XBRXSPC:                                ; Port A (extended comm) special rcv condition
0243  BB 0000 E                         MOV     BX,OFFSET PCXCOM ; get the extended COMM port control block
0246  EB 0B                             JMP     SHORT SPREC     ; go to common routine to handle it.

0248                        ARXSPC:                                 ; Port A (mother board) special rcv condition
0248  BB 0000 E                         MOV     BX,OFFSET PCCOM ; get offset of the COMM port control block

024B                        SPRECEIVE:
024B  8B 57 12                          MOV     DX,PCS7201[BX]  ; get address of the 7201 status register
024E  EC                                IN      AL,DX           ; read it.
024F  A8 02                             TEST    AL,BIT1         ; see if it's a special receive condition or:
0251  74 15                             JZ      SPRECX          ; > go exit if its no interrupt pending.
                                                                ; note that above check only necessary for
                                                                ; port A on the 7201
0253                        SPREC:
0253  8B 57 12                          MOV     DX,PCS7201[BX]  ; get address of the 7201 status register
                            ; %1%       CLI                     ; make sure interrupts are off
0256  B0 01                             MOV     AL,SR1          ; set 7201 pointer to status register 1
0258  EE                                OUT     DX,AL           ;
0259  EC                                IN      AL,DX           ; read status register 1
                            ; %1%       STI                     ; now reenable them
025A  8A E0                             MOV     AH,AL           ; save status in AH
025C  B0 30                             MOV     AL,ERR7201      ; issue error reset to 7201
025E  EE                                OUT     DX,AL           ;
025F  80 E4 70                          AND     AH,PARERR+OVRERR+FRMERR ; keep parity, overrun, framing errors

0262                        SPREC1:                             ; (this entry point is shared with end of break
                                                                ;  detect error)
                                                                ; On entry DX points to status/control reg
                                                                ; AH has the error flags set for the character
0262  4A                                DEC     DX              ; now point to data register
0263  4A                                DEC     DX              ;
0264  EC                                IN      AL,DX           ; get the data from 7201
0265  E9 0172 R                         JMP     REC3            ; go process character (in the same way as any
                                                        ;       other character)
      0268  C3                      SPRECX: RET         ; return to caller.
```

```


                                    ;---------------------------------------------------------------
                                    ;
                                    ;                       W R I T E 7 2 0 1
                                    ;
                                    ; This is a common subroutine used to write a value into a control
                                    ; register in the 7201.
                                    ;
                                    ; This routine is created to save code, and take advantage of the
                                    ; common data structure of all 3 ports (PORT CONTROL BLOCK).
                                    ;
                                    ;       ENTRY CONDITIONS:       DS:BX   points to the port control block
                                    ;                               AH      has the control register number
                                    ;                               AL      value programmed into the register
                                    ;                               interrutps are disabled.
                                    ;
                                    ;       EXIT CONDITION:
                                    ;
                                    ;---------------------------------------------------------------

      0269                          WRITE7201:                  ; common code to write to CONTROL REGISTER 3
                                                                ; value to be written in AL
                                                                ; DS:BX points to the PORT CONTROL BLOCK

      0269  8B 57 12                        MOV     DX,PCS7201[BX]  ; get status/control reg address of the 7201
      026C  86 C4                           XCHG    AL,AH           ; set pointer to correct control register
      026E  9C                              PUSHF                   ; make sure interrupts off
      026F  FA                              CLI                     ;      .
      0270  EE                              OUT     DX,AL           ; now select the register number
      0271  86 C4                           XCHG    AL,AH           ; now write to the control register
      0273  EE                              OUT     DX,AL           ; do it!
      0274  9D                              POPF                    ; retore flags
      0275  C3                              RET                     ; return to caller
```

```



                             ;-------------------------------------------------------------------
                             ;
                             ;       none of the follwoing conditions should ever happen; port not used
                             ;       But we'll service the interrupts per chance it happens.
                             ;
                             ;-------------------------------------------------------------------
0276                         XATXEMT:                               ; Port A transmit buffer empty
0276  B0 28                          MOV     AL,ENDTXBE             ; send end TXE enterrupt
0278  EB 02                          JMP     SHORT XAEXIT           ; go to common exit routine

027A                         XASTAT:                                ; Port A external status change
027A  B0 10                          MOV     AL,sx7201              ; send status change acknowledged

027C                         XAEXIT:                        ; (common exit for 422 port )
027C  BA 002A                        MOV     DX,2AH                 ; address of control register
027F  EE                             OUT     DX,AL                  ;
0280  C3                             RET                            ; bye, and don't disturb again !

0281                         XARXSPC:                               ; Port A special receive condition
0281  B0 30                          MOV     AL,ERR7201             ; send special condition clear
0283  E8 027C R                      CALL    XAEXIT                 ;       (use exit code to write to cr0 only)

0286                         XARXDAT:                               ; Port A receive character available
0286  BA 0028                        MOV     DX,28H                 ; read the data
0289  EC                             IN      AL,DX                  ; drop it on the floor
028A  C3                             RET
```

```


                             ;-------------------------------------------------------------------
                             ;
                             ;                   B S T A T , A S T A T , X B S T A T
                             ;
                             ; External status change routines for all 3 ports. These share a common
                             ; routine XSTATCHG. These routines are included only to handle the case
                             ; where break was detected.
                             ;
                             ;-------------------------------------------------------------------
028B                         BSTAT:                                 ; Port B external status change, mother board
028B  BB 0000 E                      MOV     BX,OFFSET PCPRT ; point to port control block
028E  EB 08                          JMP     SHORT XSTATCHG  ; go process external status change

0290                         XBSTAT:                                ; Port B external status change, extended comm
0290  BB 0000 E                      MOV     BX,OFFSET PCXCOM ; point to port control block
0293  EB 03                          JMP     SHORT XSTATCHG  ; go process external status change

0295                         ASTAT:                                 ; Port A external status change, mother board
0295  BB 0000 E                      MOV     BX,OFFSET PCCOM ; point to port control block
0298                         XSTATCHG:                              ; go process external status change
0298  8B 57 12                       MOV     DX,PCS7201[BX]  ; get address of 7201 status register
029B  EC                             IN      AL,DX                  ; read status
029C  8A E0                          MOV     AH,AL                  ; save status
029E  B0 10                          MOV     AL,SX7201              ; reset external status change interrupt
02A0  EE                             OUT     DX,AL                  ;
02A1  F6 C4 80                       TEST    AH,BIT7                ; check if break is now on
02A4  75 16                          JNZ     XSTAT1                 ; > if it is, go set flag in port control table
                                                                    ; If not, then first check if it was on before
                                                                    ;       (if it was on and now off, then
                                                                    ;        it must be the tail end of break)
02A6  F6 47 14 01                    TEST    BYTE PTR PCFLAG[BX],RCVBRK ;    check in the port control table
02AA  74 14                          JZ      XSTATX                 ;       If it was never on, exit
02AC  80 67 14 FE                    AND     BYTE PTR PCFLAG[BX],NOT RCVBRK  ; else, clear receive break flag
02B0  B0 01                          MOV     AL,SR1                 ; read other error on the character
                             ; %1%   CLI                            ; make sure interrupts off
02B2  EE                             OUT     DX,AL                  ; first point to status register 1
02B3  EC                             IN      AL,DX                  ; read the status
                             ; %1%   STI                            ; reenable interrupts
02B4  24 70                          AND     AL,PARERR+OVRERR+FRMERR ; keep parity, overrun, framing errors
02B6  0C 80                          OR      AL,BRKERR              ; set break detect error
02B8  8A E0                          MOV     AH,AL                  ; save it in AH (for common exit routines.
02BA  EB A6                          JMP     SHORT SPREC1           ; go process character in the same way as
                                                                    ; special receive condition with the exception
                                                                    ; that this is a break character bit set
02BC                         XSTAT1:
02BC  80 4F 14 01                    OR      BYTE PTR PCFLAG[BX],RCVBRK ; If break detected for the first
                                                                    ; time, set the flag in the port control
                                                                    ; block, and wait till it goes away before
                                                                    ; forming the break character.

                             ; NOTE THAT INTERRUPTS ARE NOT ENABLED YET !
```

INTERRUPT SERVICE ROUTINES

```

02C0                           XSTATX:                                       ; Do the status change service call
02C0  80 7F 21 01                      CMP     BYTE PTR PCSTCF[BX],1         ; check if there is a routine to call
02C4  75 0F                            JNE     L3_10$                        ; > exit if none (or one in progress)
02C6  FE 47 21                         INC     BYTE PTR PCSTCF[BX]           ; mark that a routine is being called
02C9  FF 5F 22                         CALL    DWORD PTR PCSTCA [BX]         ; call service routine
02CC  FE 4F 21                         DEC     BYTE PTR PCSTCF[BX]           ; mark that routine has returned
02CF  79 04                            JNS     L3_10$                        ; check that its not cancelled
02D1  C6 47 21 00                      MOV     BYTE PTR PCSTCF[BX],0         ; If it has, reset value to zero
02D5  C3                       L3_10$: RET
```

INTERRUPT SERVICE ROUTINES

```


                                                  SUBTTL  BIOS EXTENTION SERVICE ROUTINES
                               ;-----------------------------------------------------------------------
                               ;
                               ;                          P R G 7 2 0 1
                               ;
                               ; This is entry point to the BIOS to reprogram the 7201. It is also the
                               ; interface to the initialize (soft and hard reset) the devices routines.
                               ;
                               ; ENTRY CONDITIONS:       DS      points to bios data area
                               ;                         CX      has the offset of the Comm Control Block
                               ;                         DX      has the segment of the Comm Control Block
                               ;
                               ; EXIT CONDITIONS:        AL = FF programming device sucessful
                               ;                            = 0  programming device unsucessful
                               ;
                               ; This routine will call many subroutines to perform each function in the
                               ; reprogramming of the 7201. The convention used will be as follows:
                               ;
                               ; On entry to each subroutine,  ES:BP points to CCB
                               ;
                               ;                               DS:BX points to Port control block.
                               ;
                               ;                               CH      device number (comm, printer etc)
                               ;                                       This is not to be altered by the
                               ;                                       calling routines.
                               ;
                               ;                               CL      set to FF if for some reason the
                               ;                                       request cannot be met. Also, the
                               ;                                       high order bit of the appropriate
                               ;                                       variable in the CCB is to be set.
                               ;
                               ;                               CL      to be left untouched if request is
                               ;                                       sucessfully programmed.
                               ;
                               ;                               Initially CL is set to 0 by PRG7201.
                               ;
                               ;-----------------------------------------------------------------------
02D6                           PRG7201:
02D6  8E C2                            MOV     ES,DX                   ; set ES to have CCB segment address
02D8  8B E9                            MOV     BP,CX                   ; point ES:BP to the CCB (comm control block)
02DA  26: 8A 6E 00                     MOV     CH,ES:[BP]              ; get port number
02DE  E8 0911 R                        CALL    GETPCB                  ; call to get port control block for device
02E1  72 35                            JC      PRGERR                  ; > exit if port does not exist

02E3  B1 00                            MOV     CL,0                    ; assume everything is ok
02E5  E8 031B R                        CALL    STOPBIT                 ; go program # of stopbits
02E8  E8 0342 R                        CALL    PARITY                  ; go program parity
02EB  E8 03B6 R                        CALL    MODE                    ; set mode, data leads or limited modem control
02EE  E8 03E7 R                        CALL    DATABIT                 ; go program # of data bits
02F1  E8 0450 R                        CALL    RCVBAUD                 ; set receiver baud
02F4  E8 04A1 R                        CALL    XMTBAUD                 ; set transmit baud rate
```

```
02F7   E8 04FD R                            CALL    XONOFF            ; set any new xon/xoff characters
02FA   26: 8A 46 09                         MOV     AL,ES:CBRXOF[BP] ; see if auto receive xon/xof is changed
02FE   22 C0                                AND     AL,AL             ;
0300   74 03                                JZ      PRGCM2            ; > if not go on to next thing
0302   88 47 2F                             MOV     PCRXOF[BX],AL     ; save new value (enabled/ or not)
0305   26: 8A 46 0A                PRGCM2:  MOV     AL,ES:CBTXOF[BP] ; check auto transmit xon/xoff
0309   22 C0                                AND     AL,AL             ; see if there is a change
030B   74 03                                JZ      PRGCM3            ; > If none, go on to next thing
030D   88 47 30                             MOV     PCTXOF[BX],AL     ; Else save new value
0310   E8 0514 R                   PRGCM3:  CALL    BUFFER            ; call routine to setup buffer usage
0313   8A C1                                MOV     AL,CL             ; pass return code in AL
0315   34 FF                                XOR     AL,OFFH           ; compliment return code (to correct value)
0317   C3                                   RET                       ; return to caller

0318   B0 00                       PRGERR:  MOV     AL,0              ; indicate error
031A   C3                                   RET                       ; return to caller
```

```


                                   ;-----------------------------------------------------------------------
                                   ;
                                   ;  ********** This has to be the first parameter to be programmed *******
                                   ;
                                   ;                         S T O P B I T
                                   ;
                                   ; This routine is called to set up the number of stop bits.
                                   ;
                                   ;        ENTRY CONDITIONS:          DS:BX   points to the PORT CONTROL BLOCK
                                   ;                                   ES:BP   points to  the Comm Control Block
                                   ;
                                   ;        EXIT CONDTIONS:            Port control block updated.
                                   ;
                                   ;-----------------------------------------------------------------------
031B                               STOPBIT:
031B   26: 8A 66 02                         MOV     AH,ES:CBSTPB[BP]          ; get parity from Comm Control Block
031F   22 E4                                AND     AH,AH                     ; check if there is a change
0321   74 1E                                JZ      L4_10$                    ; > exit if no change
0323   80 FC 03                             CMP     AH,STPBMAX                ; check to see if in range
0326   76 08                                JBE     L4_1$                     ; If not in range, then
0328   26: 80 4E 02 80                      OR      BYTE PTR ES:CBSTPB[BP],80H ;      .
032D   B1 FF                                MOV     CL,OFFH                   ;          indicate error
032F   C3                                   RET                               ;          return to caller
0330                               L4_1$:
0330   88 67 28                             MOV     PCSTPB[BX],AH             ; save the value in Port Control Block
0333   D0 E4                                SHL     AH,1                      ; move it to correct value (for 7201)
0335   D0 E4                                SHL     AH,1                      ;      .
0337   8A 47 0E                             MOV     AL,PCCR4[BX]              ; get value programmed into CR4
033A   24 F3                                AND     AL,OFFH-BIT3-BIT2         ; clear loweer 2 bits
033C   0A C4                                OR      AL,AH                     ; set new mask
033E   88 47 0E                             MOV     PCCR4[BX],AL              ; save new value for 7201 CR4

                                   ; Note that value that goes into control register 4 is not yet put in!
                                   ; It will be done after parity is determined

0341   C3                          L4_10$:  RET                               ; return to caller.
```

```
                                  ;--------------------------------------------------------------------
                                  ;
                                  ;                           P A R I T Y
                                  ;
                                  ; This routine is called to set up the parity.
                                  ;
                                  ;        ENTRY CONDITIONS:       DS:BX   points to the PORT CONTROL BLOCK
                                  ;                                ES:BP   points to  the Comm Control Block
                                  ;
                                  ;        EXIT CONDTIONS:         Port control block updated.
                                  ;                                CL set to FF if port control block has errors
                                  ;
                                  ;--------------------------------------------------------------------
0342                              PARITY:
0342   26: 80 7E 03 05                    CMP     BYTE PTR ES:CBDATB[BP],5 ; check to see if 7S or 7M asked for
0347   72 18                              JB      L5_1$                    ; > If not, program parity as requested
0349   B4 03                              MOV     AH,3                     ; Else, force no parity
034B   26: 38 66 04                       CMP     BYTE PTR ES:CBPRTY[BP],AH ; make sure no parity asked for
034F   74 32                              JZ      L5_2$                    ; > if so, go program the parity
0351   26: 80 7E 04 00                    CMP     BYTE PTR ES:CBPRTY[BP],0 ; or parity is not to be changed
0356   74 2B                              JZ      L5_2$                    ;
0358   26: 80 4E 04 80                    OR      BYTE PTR ES:CBPRTY[BP],80H ; Else mark parity request in error
035D   B1 FF                              MOV     CL,0FFH                  ; indicate error in CCB
035F   EB 22                              JMP     SHORT L5_2$              ; then go program the parity
0361                              L5_1$:
0361   26: 8A 66 04                       MOV     AH,ES:CBPRTY[BP]         ; get parity from Comm Control Block
0365   22 E4                              AND     AH,AH                    ; check if there is a change
0367   74 44                              JZ      L5_15$                   ; > exit if no change
0369   80 FC 03                           CMP     AH,3                     ; if new parity is "NONE" then
036C   74 15                              JZ      L5_2$                    ;         skip following validity check
036E   26: 80 7E 03 00                    CMP     BYTE PTR ES:CBDATB[BP],0 ; if parity is changed and,
0373   75 0E                              JNZ     L5_2$                    ;         data bit also changed, go on
0375   80 7F 29 05                        CMP     BYTE PTR PCDATB[BX],5    ; else, parity changed and data bit not
0379   72 E6                              JB      L5_1$                    ;         and old data bit is 7S or 7M
037B   26: 80 4E 04 80                    OR      BYTE PTR ES:CBPRTY[BP],80H ;       mark parity request in error
0380   B1 FF                              MOV     CL,0FFH                  ;         indicate error in CCB
0382   C3                                 RET                              ;         and return to caller

0383                              L5_2$:
0383   80 FC 03                           CMP     AH,PRTYMAX               ; see if its in range
0386   76 08                              JBE     L5_3$                    ; If not then,
0388   26: 80 4E 04 80                    OR      BYTE PTR ES:CBPRTY[BP],80H ;
038D   B1 FF                              MOV     CL,0FFH                  ;         set error return code
038F   C3                                 RET                              ;         return to caller
0390                              L5_3$:
0390   88 67 2A                           MOV     PCPRTY[BX],AH            ; save the value in Port Control Block
0393   32 C0                              XOR     AL,AL                    ; assume its no parity
0395   80 FC 03                           CMP     AH,3                     ; transform value from CCB as follows:
0398   74 08                              JZ      L5_13$                   ;     (CCB value, CR4 value)
039A   B0 03                              MOV     AL,3                     ; even (1,3); Odd (2,1); none (3,0)
039C   FE CC                              DEC     AH                       ;
```

```
039E   74 02                              JZ      L5_13$                   ;
03A0   B0 01                              MOV     AL,1                     ;
03A2                              L5_13$:                                  ; (now mask correct for 7201 cr4)
03A2   8A 67 0E                           MOV     AH,PCCR4[BX]             ; get value programmed into CR4
03A5   80 E4 FC                           AND     AH,NOT (BIT0+BIT1)       ; clear lower 2 bits
03A8   0A C4                              OR      AL,AH                    ; set new mask
03AA   88 47 0E                           MOV     PCCR4[BX],AL             ; save new value for 7201 CR4
03AD                              L5_15$:                                  ; (common exit path to set CR4, note
                                                                           ;  that this must always be done!)
03AD   8A 47 0E                           MOV     AL,PCCR4[BX]             ; get value to be programmed to CR4
03B0   B4 04                              MOV     AH,CR4                   ; go to common routine to programm 7201
03B2   E9 0269 R                          JMP     WRITE7201                ; (and return to caller when done)
03B5   C3                                 RET                              ; return to caller.
```

```



                                   ;-----------------------------------------------------------------
                                   ;
                                   ;                             M O D E
                                   ;
                                   ; This routine is called to set up the mode, data leads or limited
                                   ; modem control.
                                   ;
                                   ;        ENTRY CONDITIONS:           DS:BX   points to the PORT CONTROL BLOCK
                                   ;                                    ES:BP   points to  the Comm Control Block
                                   ;
                                   ;        EXIT CONDTIONS:             Port control block updated.
                                   ;
                                   ;
                                   ;-----------------------------------------------------------------
03B6                               MODE:
03B6  26: 8A 66 01                         MOV     AH,ES:CBMODE[BP]          ; get the mode from Comm Control Block
03BA  22 E4                                AND     AH,AH                     ; check if there is a change
03BC  74 28                                JZ      MODEX                     ; > exit if no change
03BE  80 FC 02                             CMP     AH,MODEMAX                ; check if its in range
03C1  76 08                                JBE     L6_1$                     ; if not then,
03C3  26: 80 4E 01 80                      OR      BYTE PTR ES:CBMODE[BP],80H ;       .
03C8  B1 FF                                MOV     CL,0FFH                   ;          set error return code
03CA  C3                                   RET                               ;          return to caller
03CB                               L6_1$:
03CB  88 67 27                             MOV     PCMODE[BX],AH             ; save the value in Port Control Block
03CE  32 C0                                XOR     AL,AL                     ; assume its data leads only
03D0  FE CC                                DEC     AH                        ; check if its data leads
03D2  74 02                                JZ      MODE1                     ; > if so go program 7201
03D4  B0 20                                MOV     AL,20H                    ; Else assume its limited modem control
03D6                               MODE1:
03D6  8A 67 0D                             MOV     AH,PCCR3[BX]              ; get old value in 7201 CR3
03D9  80 E4 DF                             AND     AH,0FFH-BIT5              ; clear the bit that programs this
03DC  0A C4                                OR      AL,AH                     ; set new value
03DE  88 47 0D                             MOV     PCCR3[BX],AL              ; save new value
03E1  B4 03                                MOV     AH,CR3                    ; go to routine to program 7201
03E3  E9 0269 R                            JMP     WRITE7201                 ; go do it.

03E6  C3                           MODEX:  RET                               ; return to caller.
```

```



                                   ;-----------------------------------------------------------------
                                   ;
                                   ;                          D A T A B I T
                                   ;
                                   ; This routine is called to set up the number of data bits
                                   ;
                                   ;        ENTRY CONDITIONS:           DS:BX   points to the PORT CONTROL BLOCK
                                   ;                                    ES:BP   points to  the Comm Control Block
                                   ;
                                   ;        EXIT CONDTIONS:             Port control block updated.
                                   ;
                                   ;-----------------------------------------------------------------
03E7                               DATABIT:
03E7  26: 8A 66 03                         MOV     AH,ES:CBDATB[BP]          ; # data bit from Comm Control Block
03EB  22 E4                                AND     AH,AH                     ; check if there is a change
03ED  74 60                                JZ      DATABX                    ; > exit if no change
03EF  80 FC 06                             CMP     AH,DATMAX                 ; check if its above maximum
03F2  76 08                                JBE     L7_10$                    ; If not then
03F4  26: 80 4E 03 80                      OR      BYTE PTR ES:CBDATB[BP],80H ;     indicate error in # of databits
03F9  B1 FF                                MOV     CL,0FFH                   ;          set error code
03FB  C3                                   RET                               ;          return to caller
03FC                               L7_10$:
03FC  88 67 29                             MOV     PCDATB[BX],AH             ; save the value in Port Control Block
03FF  32 C0                                XOR     AL,AL                     ; get register value for 5 data bits
0401  BA 1F00                              MOV     DX,1F00H                  ; get data mask for 5 data bit
0404  FE CC                                DEC     AH                        ; If so go to it (CCB value 1)
0406  74 20                                JZ      L7_1$                     ;          > go program 7201
0408  B0 40                                MOV     AL,40H                    ; next assume its 6 data bits
040A  B6 3F                                MOV     DH,3FH                    ;          (masks AND 3F, OR 00 )
040C  FE CC                                DEC     AH                        ; (CCB value 2)
040E  74 18                                JZ      L7_1$                     ; > If it is go program 7201
0410  B0 20                                MOV     AL,20H                    ; next assume its 7 data bits
0412  B6 7F                                MOV     DH,7FH                    ;          (mask AND 7F, OR 00)
0414  FE CC                                DEC     AH                        ;          .
0416  74 10                                JZ      L7_1$                     ; If it is then go program 7201
0418  B0 60                                MOV     AL,60H                    ; next assume its 8 bits
041A  B6 FF                                MOV     DH,0FFH                   ;          (mask AND FF, OR 00)
041C  FE CC                                DEC     AH                        ;          .
041E  74 08                                JZ      L7_1$                     ; > If so go program 7201

                                   ; 7M or 7S is done by programming to 8N then do mark or space in software

0420  B6 7F                                MOV     DH,07FH                   ;          (mask AND 7F)
0422  FE CC                                DEC     AH                        ; assume its 7 space
0424  74 02                                JZ      L7_1$                     ; > If so, go program 7201
0426  B2 80                                MOV     DL,80H                    ; Else it must be 7M ( mask OR 80)
0428                               L7_1$:
0428  50                                   PUSH    AX                        ; save value for # data bits in receive
0429  89 57 15                             MOV     PCMASK[BX],DX             ; save new mask
042C  8A 67 0F                             MOV     AH,PCCR5[BX]              ; get old value in control register 5
042F  80 E4 9F                             AND     AH,NOT (BIT6+BIT5)        ; clear bits for "data bits per byte"
```

```
0432  OA C4                     OR      AL,AH                   ; put in new value for # data bits
0434  88 47 OF                  MOV     PCCR5[BX],AL            ; save it in the port control block
0437  B4 05                     MOV     AH,CR5                  ; set to go to routine to program 7201
0439  E8 0289 R                 CALL    WRITE7201               ; go do it.
043C  58                        POP     AX                      ; get # stop bits for receive chars
043D  DO EO                     SHL     AL,1                    ; put it in correct place for 7201
043F  8A 67 OD                  MOV     AH,PCCR3[BX]            ; get old value in control register 5
0442  80 E4 3F                  AND     AH,NOT (BIT7+BIT6)      ; clear bits for "data bits per byte"
0445  OA C4                     OR      AL,AH                   ; put in new value for # data bits
0447  88 47 OD                  MOV     PCCR3[BX],AL            ; save it in the port control block
044A  B4 03                     MOV     AH,CR3                  ; set to go to routine to program 7201
044C  E9 0289 R                 JMP      WRITE7201              ; go do it.

044F  C3                DATABX: RET                             ; return to caller.
```

```


                        ;--------------------------------------------------------------
                        ;
                        ;                          R C V B A U D
                        ;
                        ; This routine is called to reprogram the receiver baud rate
                        ;
                        ;
                        ;       ENTRY CONDITIONS:       DS:BX   points to Port Control Block
                        ;                               AL      has value from Comm control block
                        ;                               CH      device number
                        ;
                        ;       EXIT CONDITIONS:        CL = FF device cannot be programmed as requested
                        ;                               HO bit in the CCB is set to 1 in the receive
                        ;                               baud rate variable.
                        ;                               otherwise CL is left untouched
                        ;
                        ;--------------------------------------------------------------
0450                    RCVBAUD:
0450  26: 8A 46 05              MOV     AL,ES:CBRCVB[BP] ; get new baud rate to be programmed
0454  22 CO                     AND     AL,AL           ; see if there is any change in rcv baud
0456  74 2A                     JZ      RCVBX           ; > exit if not
0458  E8 04E1 R                 CALL    GETBAUD         ; get the value to be written to generator
045B  3C FF                     CMP     AL,OFFH         ; check if the value is legal
045D  75 08                     JNZ     RCVB1           ; If not legal then
045F  B1 FF                     MOV     CL,OFFH         ;       indicate programming failed
0461  26: 80 4E 05 80           OR      BYTE PTR ES:CBRCVB[BP],80H ; mark the receive baud rate as not supported
0466  C3                        RET                     ;       and return to caller
0467                    RCVB1:
0467  80 FD 02                  CMP     CH,DEVPRT       ; if the device is the printer skip following
046A  74 17                     JZ      RCVBPRT         ;       treat it as special case
046C  26: 8A 66 05              MOV     AH,ES:CBRCVB[BP] ; get new baud rate to be programmed
0470  88 67 2B                  MOV     PCRCVB[BX],AH   ; save it in port control block
0473  8A 67 0A                  MOV     AH,PCRATE[BX]   ; get old value
0476  80 E4 FO                  AND     AH,OFOH         ; save transmit baud rate portion
0479  OA C4                     OR      AL,AH           ; get new baud rate value (both xmt & rcv)
047B                    RCVB2:
047B  88 47 0A                  MOV     PCRATE[BX],AL   ; save the new value
047E  8B 57 08                  MOV     DX,PCBAUD[BX]   ; get port address of baud rate generator
0481  EE                        OUT     DX,AL           ; do it!
0482                    RCVBX:
0482  C3                        RET

                        ; printer is a special case at this point we know that receive baud rate
                        ; is non zero and is valid

0483                    RCVBPRT:
0483  26: 8A 46 05              MOV     AL,ES:CBRCVB[BP]          ; get new baud rate to be programmed
0487  26: 80 7E 06 00           CMP     BYTE PTR ES:CBTXB[BP],0 ; if xmt baud not specified
048C  75 05                     JNZ     L8_5$                   ;       then
048E  26: 88 46 06              MOV     ES:CBTXB[BP],AL         ;       make transmit baud = rcv baud
0492  C3                        RET                             ;       return to caller
```

```

0493  26: 38 46 06           L8_5$:  CMP     ES:CBTXB[BP],AL         ; if xmt baud is the same as
0497  74 07                          JZ      L8_10$                  ;       rcv baud then return to caller

0499  26: 80 4E 05 80                OR      BYTE PTR ES:CBRCVB[BP],80H ; Else mark error in CCB
049E  B1 FF                          MOV     CL,0FFH                 ;       for receive baud

04A0  C3                     L8_10$: RET                             ; return to caller
```

```


                             ;-------------------------------------------------------------
                             ;
                             ;                      X M T B A U D
                             ;
                             ; This routine is called to reprogram the transmit baud rate. This routine
                             ; is never called for the printer transmit baud rate change. The next level
                             ; routine above this will determine if the printer has to be reprogrammed
                             ; and will call RCVBAUD; since transmit and receive baud rates must be the
                             ; same in the printer.
                             ;
                             ;
                             ;       ENTRY CONDITIONS:       DS:BX   points to Port Control Block
                             ;                               ES:BP   points to comm control block
                             ;                               CH      device number
                             ;
                             ;       EXIT CONDITIONS:        CL = FF device cannot be programmed as requested
                             ;                               HO bit in the CCB is set to 1 in the receive
                             ;                               baud rate variable.
                             ;                               otherwise CL is left untouched
                             ;
                             ;-------------------------------------------------------------
04A1                         XMTBAUD:
04A1  26: 8A 46 06                   MOV     AL,ES:CBTXB[BP] ; get new baud rate to be programmed
04A5  22 C0                          AND     AL,AL           ; check to see if there is a change
04A7  74 37                          JZ      XMTBX           ; > exit if there is no change
04A9  E8 04E1 R                      CALL    GETBAUD         ; get the value to be written to generator
04AC  3C FF                          CMP     AL,0FFH         ; If its bad value then,
04AE  75 08                          JNZ     XMTB0           ;       .
04B0  B1 FF                          MOV     CL,0FFH         ;       set return code
04B2  26: 80 4E 06 80                OR      BYTE PTR ES:CBTXB[BP],80H ;     indicate tranmit baud rate in error
04B7  C3                             RET                     ;       and return to caller
04B8                         XMTB0:
04B8  26: 8A 66 06                   MOV     AH,ES:CBTXB[BP] ; save new value from CCB to the Port control block
04BC  88 67 2C                       MOV     PCTXB[BX],AH    ;       .
04BF  80 FD 02                       CMP     CH,DEVPRT       ; If its printer then,
04C2  75 05                          JNZ     L9_1$           ;       .
04C4  88 67 2B                       MOV     PCRCVB[BX],AH   ;       mark printer rcv baud same as xmt baud
04C7  EB 10                          JMP     SHORT L9_4$     ; > skip rest (programming generator different)
04C9                         L9_1$:
04C9  8A 67 0A                       MOV     AH,PCRATE[BX]   ; get old value programmed into generator
04CC  80 E4 0F                       AND     AH,0FH          ; save receive baud rate portion
04CF  D0 E0                          SHL     AL,1            ; move value to ho nibble
04D1  D0 E0                          SHL     AL,1            ;       .(CL not to be touched)
04D3  D0 E0                          SHL     AL,1            ;       .
04D5  D0 E0                          SHL     AL,1            ;       .
04D7  0A C4                          OR      AL,AH           ; get the new baud rate value to be programmed

04D9  88 47 0A               L9_4$:  MOV     PCRATE[BX],AL   ; save new value into the port control block
04DC  8B 57 08                       MOV     DX,PCBAUD[BX]   ; get port address of baud rate generator
04DF  EE                             OUT     DX,AL           ; do it!
04E0                         XMTBX:
```

```
04E0  C3                                RET
```

```


                                ;-----------------------------------------------------------------
                                ;
                                ;                          G E T B A U D
                                ;
                                ; This routine is used to get the value to be that the generator is
                                ; to be programmed corresponding to the baud rate value specified in
                                ; the Comm control block.
                                ;
                                ;         ENTRY CONDITIONS:      DS:BX   points to port control block
                                ;                                        it is assumed that it resides
                                ;                                        in the same segment address as
                                ;                                        the translate tables.
                                ;                                CH      device number
                                ;                                AL      new baud rate value
                                ;
                                ;         EXIT CONDITIONS:       AL      has the value to be written
                                ;                                        the baud rate generator
                                ;                                AL = FF this baud rate no supported
                                ;
                                ;
                                ;-----------------------------------------------------------------
04E1                            GETBAUD:
04E1  3C 00                             CMP     AL,0                    ; check range
04E3  72 15                             JB      L34_10$                 ; > If out of range indicate bad baud
04E5  3C 11                             CMP     AL,BAUDMAX              ;       .
04E7  77 11                             JA      L34_10$                 ;       .

04E9  53                                PUSH    BX                      ; save BX, for translation
04EA  BB 0000 E                         MOV     BX,OFFSET XBAUD1        ; assume its printer port
04ED  80 FD 02                          CMP     CH,DEVPRT               ; see if its the printer
04F0  75 03                             JNZ     L34_1$                  ; > If so go do table lookup
04F2  BB 0000 E                         MOV     BX,OFFSET XBAUD2        ; Else point to table for Comm & Xcomm
04F5                            L34_1$:
04F5  FE C8                             DEC     AL                      ; table starts at 0 (1 in ccb)
04F7  D7                                XLAT    XBAUD1                  ; translate value
04F8  5B                                POP     BX                      ; restore register
04F9  C3                                RET                             ; return to caller.

04FA  B0 FF                     L34_10$: MOV    AL,0FFH                 ; out of range, indicate bad value
04FC  C3                                RET                             ; return to caller
```

```


                                  ;----------------------------------------------------------------
                                  ;
                                  ;                        X O N O F F
                                  ;
                                  ; This routine is used to setup the characters to be used in the
                                  ; xon/xoff protocol.
                                  ;
                                  ; Note that the protocol is the same, as a standard xon/xoff, and that
                                  ; the only change allowed is which character can be used to perform
                                  ; this.
                                  ;
                                  ;      ENTRY CONDITIONS:       DS:BX   points to port control block
                                  ;                                      it is assumed that it resides
                                  ;                                      in the same segment address as
                                  ;                                      the translate tables.
                                  ;                              CH      device number
                                  ;                              ES:BP   Comm control block
                                  ;
                                  ;      EXIT CONDITIONS:        AL      destroyed.
                                  ;
                                  ;----------------------------------------------------------------
04FD                              xonoff:
04FD  26: 8A 46 07                        MOV     AL,ES:CBXON[BP]         ; get new xon character
0501  22 C0                               AND     AL,AL                   ; check if there is a change
0503  74 03                               JZ      xonof1                  ; > if not, go check xoff
0505  88 47 2D                            MOV     PCXON[BX],AL            ; save new xon character
0508  26: 8A 46 08                XONOF1: MOV     AL,ES:CBXOFF[BP]        ; get new xoff character
050C  22 C0                               AND     AL,AL                   ; check if there is a change
050E  74 03                               JZ      XONOFX                  ; > go exit if no change
0510  88 47 2E                            MOV     PCXOFF[BX],AL           ; save new value
0513  C3                          XONOFX: RET                             ; return to caller
```

```


                                  ;------------------------------------------------------------------
                                  ;
                                  ;                        B U F F E R
                                  ;
                                  ; This routine is called to calculate the buffer control information.
                                  ;
                                  ; ENTRY CONDITIONS:      DS:BX    points to port control block
                                  ;                        ES:BP    points to Comm control block
                                  ;
                                  ; EXIT CONDITIONS:       CL = FF If unsucessful (buffer area too small)
                                  ;                        CL       left alone otherwise.
                                  ;
                                  ;------------------------------------------------------------------
0514                              BUFFER:
0514  26: 8B 46 0B                        MOV     AX,ES:CBSIZE[BP]        ; check new size of buffer
0518  23 C0                               AND     AX,AX                   ; If its zero, exit
051A  74 5A                               JZ      BUFFX                   ;     .
051C  3D 0050                             CMP     AX,RBLEN+2*(RBDFLT)     ; check if its > default size
051F  77 0A                               JA      BUFF1                   ; If not then,
0521  B1 FF                               MOV     CL,0FFH                 ;       set return code
0523  26: 81 4E 0B 0080                   OR      WORD PTR ES:CBSIZE[BP],80H ;    indicate offending variable
0529  EB 4B                               JMP     SHORT BUFFX             ;       go return to caller
052B                              BUFF1:
052B  24 FE                               AND     AL,0FEH                 ; round down to even number
052D  26: C4 76 0D                        LES     SI,DWORD PTR ES:CBRADR[BP] ; get address of the buffer
                                  ;
                                  ; now calculate and enter all the buffer control information
                                  ;
0531  9C                          BUFF10: PUSHF                           ; diable interrupts first
0532  FA                                  CLI                             ;     .
                                                                          ; This is an embedded entry point
                                                                          ; for INITCOM, used when the default
                                                                          ; buffer is set up
                                                                          ; ( ES:SI point to buffer area )
                                                                          ;   DS:BX points to port control block
                                                                          ;   AX    has the size of the buffer

                                  ; First clear all flags related to xon/xoff, since we now have a new buffer

0533  80 67 14 39                         AND     BYTE PTR PCFLAG[BX],NOT (XMTXON+XMTXOF+RCVXOF+APPXOF)
0537  89 47 31                            MOV     PCSIZE[BX],AX           ; save size of new buffer
053A  89 77 33                            MOV     PCRADR[BX],SI           ; save new offset to buffer
053D  8C 47 35                            MOV     PCRADR+2[BX],ES         ; save segment address of new buffer
0540  2D 0010                             SUB     AX,RBLEN                ; subtract buffer space from buffer
                                                                          ; header information.
0543  8B D6                               MOV     DX,SI                   ; calculate address of first character in
0545  83 C2 10                            ADD     DX,RBLEN                ; (address where first char go to)
0548  26: 89 54 04                        MOV     ES:RBIN[SI],DX          ; put it into buffer control info
054C  26: 89 54 06                        MOV     ES:RBOUT[SI],DX         ; use same offset for initial char out
0550  26: 89 54 0C                        MOV     ES:RBHEAD[SI],DX        ; use same offset for head of buffer pointer
0554  03 D0                               ADD     DX,AX                   ;
```

```
0556  4A                            DEC     DX                        ;       (want address of the last word
0557  4A                            DEC     DX                        ;        of the buffer)
0558  26: 89 54 0E                  MOV     ES:RBTAIL[SI],DX          ; put address into buffer control info

055C  D1 E8                         SHR     AX,1                      ; divide by 2 to get max number chars
055E  26: 89 44 02                  MOV     ES:RBMAX[SI],AX           ; save new value

0562  26: C7 04 0000                MOV     WORD PTR ES:RBCOUNT[SI],0 ; put indicate no characters in buffer
0567  D1 E8                         SHR     AX,1                      ; xoff threshold is buffer half full
0569  26: 89 44 08                  MOV     ES:RBXOFF[SI],AX          ; save calculated xoff threshold value
056D  D1 E8                         SHR     AX,1                      ; calcuate auto xon threshold (1/4 of
056F  D1 E8                         SHR     AX,1                      ;        xoff threshold)
0571  26: 89 44 0A                  MOV     ES:RBXON[SI],AX           ; save value.
0575  9D                            POPF                              ; restore user flags

0576  C3                    BUFFX:  RET
```

```


                            ;--------------------------------------------------------------------
                            ;
                            ;                         D F L T 7 2 0 1
                            ;
                            ; This routine is called to reprogram a particular device to the default
                            ; value.
                            ;
                            ;
                            ; ENTRY CONDITIONS:               CH      device number
                            ;                                 DS      points to bios data area
                            ;
                            ;
                            ;--------------------------------------------------------------------

0577                        DFLT7201:
0577  E8 0911 R                     CALL    GETPCB                    ; check if port is present
057A  72 3C                         JC      L10_10$                   ; > exit if not
057C  E8 05B9 R                     CALL    DFLTBUF                   ; call routine to setup default receive
                                                                      ; char buffer (and get address of pcb)

057F  E8 07DF R                     CALL    RCVCANCEL                 ; cancel rcv  interrupt service
0582  E8 0819 R                     CALL    XMTCANCEL                 ; cancel xmt buffer empty service
0585  E8 0000 E                     CALL    RDNVMCOM                  ; go read NVM for default comm values

0588  51                            PUSH    CX                        ; save device number
0589  8B 4F 04                      MOV     CX,PCDFLT[BX]             ; get address of default settings (CCB)
058C  80 4F 0E 40                   OR      BYTE PTR PCCR4[BX],CLK16X ; make sure its 16X for clock divisor
0590  8C DA                         MOV     DX,DS                     ; set segment of CCB to bios data area
0592  E8 02D6 R                     CALL    PRG7201                   ; go reprogram 7201 to default values
0595  59                            POP     CX                        ; restore device number

0596  C6 47 14 00                   MOV     BYTE PTR PCFLAG[BX],0     ; clear all flags        <-------------

059A  8A 47 0F                      MOV     AL,PCCR5[BX]              ; enable transmitter (note that at this
059D  0C 08                         OR      AL,08                     ;       point the port is guaranteed to
059F  88 47 0F                      MOV     PCCR5[BX],AL              ;       be present, so it is ok to
05A2  B4 05                         MOV     AH,CR5                    ;       write to the device
05A4  E8 0269 R                     CALL    WRITE7201                 ;
05A7  E8 05EA R                     CALL    RCVENA                    ; enable the receiver

05AA  B0 10                         MOV     AL,SX7201
05AC  8B 57 12                      MOV     DX,PCS7201[BX]
05AF  EE                            OUT     DX,AL

05B0  8A 47 0B                      MOV     AL,PCCR1[BX]      ; now set register 1 before anything else
05B3  B4 01                         MOV     AH,CR1            ;         .
05B5  E8 0269 R                     CALL    WRITE7201         ;         .

05B8  C3                    L10_10$:        RET
```

```
                                   ;-----------------------------------------------------------------------
                                   ;
                                   ;                              D F L T B U F
                                   ;
                                   ; This routine is called to reset receive character buffer area to the
                                   ; default buffer.
                                   ;
                                   ; ENTRY CONDITIONS:                 CH      has device number
                                   ;
                                   ; EXIT CONDITIONS:                  command ignored if port number invalid
                                   ;
                                   ;                                   DS:BX   points to port control block
                                   ;                                   buffer pointer reset.
                                   ;
                                   ;-----------------------------------------------------------------------
 05B9                              DFLTBUF:
 05B9   E8 0911 R                          CALL    GETPCB                  ; get address of port control block
 05BC   73 01                              JNC     L11_1$                  ; if port number invalid,
 05BE   C3                                 RET                             ;       return to caller

 05BF   1E                         L11_1$: PUSH    DS                      ; point ES to bios data area too
 05C0   07                                 POP     ES                      ;       .(where default ccb is)
 05C1   8D 77 37                           LEA     SI,PCLEN[BX]            ; get the address of default buffer
 05C4   B8 0050                            MOV     AX,RBLEN+2*RBDFLT       ; get size of default buffer
 05C7   E9 0531 R                          JMP     BUFF10                  ; go reset buffer control information
                                                                           ; then return to caller
```

```
                                   ;-----------------------------------------------------------------------
                                   ;
                                   ;                              R D S E T U P
                                   ;
                                   ; This routine is called to retreive the information that is programmed
                                   ; into the specified port.
                                   ;
                                   ;       ENTRY CONDITIONS:        CX,DX   contains offset,segment of CCB
                                   ;                                        first byte of CCB must have
                                   ;                                        port number
                                   ;
                                   ;       EXIT CONDITIONS:         AL = 0  port information loaded
                                   ;                                AL = FF invalid port number, or port
                                   ;                                        not connected.
                                   ;
                                   ;-----------------------------------------------------------------------
 05CA                              RDSETUP:
 05CA   8B E9                              MOV     BP,CX           ; point ES:BP to comm control block
 05CC   8E C2                              MOV     ES,DX           ;
 05CE   26: 8A 6E 00                       MOV     CH,ES:[BP]      ; get port number
 05D2   32 C0                              XOR     AL,AL           ; assume invalid port address
 05D4   E8 0911 R                          CALL    GETPCB          ; get address of port control block (DS:BX)
 05D7   72 10                              JC      L12_10$         ; > exit if bad port number or not connected
 05D9   83 C3 27                           ADD     BX,PCMODE       ; point to beginning of Port control block
                                                                   ; that matches comm control block
 05DC   8B F3                              MOV     SI,BX           ; get ready to transfer
 05DE   B9 0010                            MOV     CX,CBLEN-1      ; transfer the whole table minus port number
 05E1   8B FD                              MOV     DI,BP           ; get destination offset into DI
 05E3   47                                 INC     DI              ; point DI passed port number

 05E4   FC                                 CLD                     ; get the correct direction
 05E5   F3/ A4                             REP  MOVSB              ; move whole string
                                   ;       MOVSB                   ;
 05E7   B0 FF                              MOV     AL,0FFH         ; indicate information sucesfully passed
 05E9                              L12_10$:
 05E9   C3                                 RET                     ; return to caller
```

```


                              ;-----------------------------------------------------------------
                              ;
                              ;                       R C V E N A
                              ;
                              ; This routine is called to enable the receiver.
                              ;
                              ;          ENTRY CONDITIONS:       CH      device number
                              ;
                              ;          EXIT CONDITIONS:        Command ignored if port is invlaid
                              ;
                              ;-----------------------------------------------------------------
05EA                          RCVENA:
05EA  E8 0911 R                          CALL    GETPCB          ; get port control blcok address
05ED  73 01                              JNC     L13_1$          ; If port number is invlaid then,
05EF  C3                                 RET                     ;       return to caller
05F0                          L13_1$:
05F0  8A 47 0D                           MOV     AL,PCCR3[BX]    ; read last value put into control register 3
05F3  0C 01                              OR      AL,BIT0         ; enable receiver
05F5  80 67 14 DF                        AND     BYTE PTR PCFLAG[BX],0FFH-RCVOFF ; set flag that receiver is enabled
05F9  EB 0F                              JMP     SHORT RCVCOMMON ; go use common code for disable & enable



                              ;-----------------------------------------------------------------
                              ;
                              ;                       R C V D I S
                              ;
                              ; This routine is called to disable the receiver.
                              ;
                              ;          ENTRY CONDITIONS:       CH      device number
                              ;
                              ;          EXIT CONDITIONS:        Command ignored if port is invlaid
                              ;
                              ;-----------------------------------------------------------------
05FB                          RCVDIS:
05FB  E8 0911 R                          CALL    GETPCB          ; get port control blcok address
05FE  73 01                              JNC     L14_1$          ; if port number invalid then,
0600  C3                                 RET                     ; > return to caller
0601                          L14_1$:
0601  8A 47 0D                           MOV     AL,PCCR3[BX]    ; get last value put into control register 3
0604  24 FE                              AND     AL,0FFH-BIT0    ; clear last bit
0606  80 4F 14 20                        OR      BYTE PTR PCFLAG[BX],RCVOFF ; set flag that receiver is disabled

060A                          RCVCOMMON:
060A  88 47 0D                           MOV     PCCR3[BX],AL    ; restore CR3 value in port control block
060D  B4 03                              MOV     AH,CR3          ; write it to control register 3
060F  E9 0269 R                          JMP     WRITE7201       ; go write it to the control register
```

```


                              ;-------------------------------------------------------------
                              ;
                              ;                       I N S T A T
                              ;
                              ;          This routine is called to get the status of the receive
                              ; character buffer.
                              ;
                              ; ENTRY CONDITIONS:        DS:     has segment address of bios data area
                              ;                          CH      has the device number
                              ;
                              ; EXIT CONDITIONS:         AX destroyed.
                              ;                          AL = 0  no character avialable, OR invlid port #
                              ;                          AL = FF character avialable
                              ;
                              ;-------------------------------------------------------------
0612                          INSTAT:
0612  32 C0                              XOR     AL,AL           ; assume invlaid port number
0614  E8 0911 R                          CALL    GETPCB          ; get address of the port control block
0617  72 0C                              JC      INSTX           ; > exit if port number invalid
0619  C4 5F 33                           LES     BX,DWORD PTR PCRADR[BX] ; get address of the receive character buffer
061C  26: 8B 07                          MOV     AX,ES:RBCOUNT[BX] ; get # characters available
061F  23 C0                              AND     AX,AX           ; If no character available then
0621  74 02                              JZ      INSTX           ;       go exit (return code all set)
0623  B0 FF                              MOV     AL,0FFH         ; else set proper return code
0625  C3                      INSTX:     RET
```

```


                            ;-----------------------------------------------------------------
                            ;
                            ;                              I N C H A R
                            ;
                            ;       This is the lowest level of the routines available to draw a
                            ; character out of the receive character buffer. This routine will return
                            ; to the caller if no character is available.
                            ;
                            ; ENTRY CONDITIONS:
                            ;                              CH       device number
                            ;
                            ; EXIT CONDITIONS:
                            ;                              AL = FF
                            ;                                       If character available BP, CX, AX is destroyed.
                            ;                                       CH = status of character
                            ;                                       CX = ascii character
                            ;
                            ;                              AL = 0
                            ;                                       If none available, or invalid port number,
                            ;                                       all registers except AX is preserved.
                            ;
                            ;---------------------------------------------------------------------
0626                        INCHAR:
0626  E8 0911 R                     CALL    GETPCB                  ; get address pf pcb in to DS:BX
0629  73 03                         JNC     L15_1$                  ; If invalid port number
062B  32 C0                         XOR     AL,AL                   ;       set error code
062D  C3                            RET                             ;       return to caller
062E                        L15_1$:
062E  C4 6F 33                      LES     BP,DWORD PTR PCRADR[BX] ; get receive character buffer to ES:BP
0631  E8 066A R                     CALL    TXXON                   ; go do auto transmit xon/xoff chores
0634  9C                            PUSHF                           ; disable interrutps
0635  FA                            CLI                             ;
0636  26: 8B 46 00                  MOV     AX,ES:RBCOUNT[BP]       ; get # characters in the buffer
063A  23 C0                         AND     AX,AX                   ; check to see if any characters
063C  74 20                         JZ      INCHAX                  ; > go exit if none
063E  48                            DEC     AX                      ; count down # characters in the buffer
063F  26: 89 46 00                  MOV     ES:RBCOUNT[BP],AX       ; update count
0643  26: 8B 76 06                  MOV     S!,ES:RBOUT[BP]         ; get pointer to next character drawn
0647  26: 8B 0C                     MOV     CX,ES:0[SI]             ; get the next character & status
054A  26: 3B 76 0E                  CMP     SI,ES:RBTAIL[BP]        ; check to see if at end of buffer
064E  75 06                         JNZ     INCHA1                  ; If we are then, set pointer to the
0650  26: 8B 76 0C                  MOV     SI,ES:RBHEAD[BP]        ;       beginning of the buffer
0654  EB 02                         JMP     SHORT INCHA2            ;       (go put it into buffer table)
0656  46                    INCHA1: INC     SI                      ; Else, increment pointer by 2
0657  46                            INC     SI                      ;       (point to next character)
0658  26: 89 76 06          INCHA2: MOV     ES:RBOUT[BP],SI         ; update pointer to the next character
065C  B0 FF                         MOV     AL,0ffh                 ; set return code = character available

065E  9D                    INCHAX: POPF                            ; restore caller's flags
065F  C3                            RET                             ; return to caller.
```

```


                            ;---------------------------------------------------------------------
                            ;
                            ;                              G E T C H A R
                            ;
                            ; This routine is called to read a character out of the receive buffer, and
                            ; to wait till a character is available
                            ;
                            ; ENTRY CONDITIONS:            same as INCHAR
                            ;
                            ; EXIT COMDITIONS:             same as INCHAR, but character will always be available
                            ;                              and the character will be in AX
                            ;
                            ;---------------------------------------------------------------------
0660                        GETCHAR:
0660  E8 0626 R                     CALL    INCHAR                  ; go to common routine to get character
0663  22 C0                         AND     AL,AL                   ; check if sucessful
0665  74 F9                         JZ      GETCHAR                 ; repeat till sucesful
0667  8B C1                         MOV     AX,CX                   ; put character in correct register
0669  C3                            RET                             ; return to caller
```

```



                            ;-------------------------------------------------------------------------
                            ;
                            ;                              T X X O N
                            ;
                            ; This routine is called to see if its time to send an XON thru the port.
                            ; It will exit immediately if xon, for any reason cannot be sent; and
                            ; a flag is set to send the auto xon as soon as possible.
                            ;
                            ; The criteria to send an XON are as follows:
                            ;
                            ;               1 - an auto XOFF was previously sent by the driver
                            ;               2 - receive character buffer reached a low mark
                            ;               3 - application did not send an XOFF
                            ;
                            ;       ENTRY CONDITIONS:         DS:BX    points to port control block
                            ;                                 ES:BP    points to receive buffer area
                            ;
                            ;
                            ;       EXIT CONDITIONS:          CX       preserved
                            ;                                 DS:BX    preserved
                            ;                                 ES:BP    preserved
                            ;
                            ; NOTE: If application sent an xoff, and auto xoff was also sent, then
                            ;       at the low water mark, auto xoff sent flag is cleared, but no
                            ;       auto xon is sent !!!
                            ;
                            ;-------------------------------------------------------------------------
 066A                       TXXON:
 066A  80 7F 30 02                  CMP     BYTE PTR PCTXOF[BX],2   ; check if transmit auto xon/xoff is
 066E  75 36                        JNZ     TXXONX                  ;        enabled, exit if not

 0670  F6 47 14 02                  TEST    BYTE PTR PCFLAG[BX],XMTXON ; If already flagged to send XON asap
 0674  75 1E                        JNZ     TXXON5                  ;        > go check if it can be sent

 0676  F6 47 14 04                  TEST    BYTE PTR PCFLAG[BX],XMTXOF ; check if auto xoff was sent
 067A  74 2A                        JZ      TXXONX                  ; > if not go return
 067C  26: 8B 46 00                 MOV     AX,ES:RBCOUNT[BP]       ; get # characters in the buffer
 0680  26: 3B 46 0A                 CMP     AX,ES:RBXON[BP]         ; If buffer not at the right threshold
 0684  75 20                        JNZ     TXXONX                  ; > go return to caller
 0686  80 67 14 FB                  AND     BYTE PTR PCFLAG[BX],NOT XMTXOF ; clear auto xoff sent flag
 068A  F6 47 14 80                  TEST    BYTE PTR PCFLAG[BX],APPXOF ; check if application sent xoff
 068E  75 16                        JNZ     TXXONX                  ; > if so return to caller
 0690  80 4F 14 02                  OR      BYTE PTR PCFLAG[BX],XMTXON ; set flag to transmit xon asap
 0694                       TXXON5:                                 ; now try to transmit the auto XON
 0694  F6 47 14 08                  TEST    BYTE PTR PCFLAG[BX],XMTBRK ; check to see if break being sent
 0698  75 0C                        JNZ     TXXONX                  ; > if so don't send the auto xon now
 069A  8A 67 2D                     MOV     AH,PCXON[BX]            ; try to send the xon
 069D  E8 01F9 R                    CALL    AUTOXMT                 ;
 06A0  74 04                        JZ      TXXONX                  ; >If not sucessful return to caller

                            ; clear transmit xon asap and xoff was sent flag
```

```

 06A2  80 67 14 F9                  AND     BYTE PTR PCFLAG[BX],NOT (XMTXON + XMTXOF)
 06A6  C3                   TXXONX: RET                             ; return to caller
```

```
                                        ;-------------------------------------------------------------------
                                        ;
                                        ;                       O U T S T A T
                                        ;
                                        ; This routine is called to get the status of the transmitter.
                                        ; The items checked are as follows:
                                        ;
                                        ;       1 - Transmit buffer empty
                                        ;       2 - Not in an xoff'd mode
                                        ;
                                        ; This is the routine called when the application program wishes to
                                        ; check the status. Note there are other items that may be usefull
                                        ; that can be added to this, namely:
                                        ;
                                        ;       3 - break being transmitted. If this condition arises, it is
                                        ;               up to the appication program to keep track of this.
                                        ;
                                        ;       4 - CTS not prresent (for limited modem control). Note that
                                        ;               in limited modem control, if CTS is not on, transmit
                                        ;               buffer will eventually get to a point where it won't
                                        ;               be ready.
                                        ;
                                        ; The following are the motivation to leave them out of this routine:
                                        ;
                                        ;       1 - most CP/M programs are written in such a way that they
                                        ;           tend to check the status often and not always needing
                                        ;           the remaining information. In those cases, efficiency
                                        ;           of execution time is the most important factor.
                                        ;
                                        ;       2 - The information on the other items can be obtained in
                                        ;           other bios function calls.
                                        ;
                                        ;
                                        ;       ENTRY CONDITIONS:       CH      device number
                                        ;                               DS      has BIOS segment address
                                        ;
                                        ;       EXIT CONDITIONS:        DX      destroyed
                                        ;                               AL = FF if character can be transmitted
                                        ;                                       that is transmit buffer empty,
                                        ;                                       not in an xoff'd mode.
                                        ;                               Carry flag is cleared.
                                        ;
                                        ;                               AL = 0 OTHERWISE, AND
                                        ;                               Carry flag set.
                                        ;                               AH  BIT6 set means XOFF received
                                        ;                               AH  BIT2 set means tranmit buffer not
                                        ;                                       empty
                                        ;
                                        ; Note: the carry flag set/not set is to be used only between
                                        ;       BIOS routines for quick execution time, and is not to
                                        ;       be used by the appication program, since we cannot predict
```

```
                                        ;       what the operating system will do with this flag.
                                        ;
                                        ;------------------------------------------------------------------
      06A7                              OUTSTAT:
      06A7  E8 0911 R                           CALL    GETPCB          ; get address pf pcb in to DS:BX
      06AA  72 15                               JC      L16_5$          ; > exit if port number invalid
      06AC  8A 67 14                            MOV     AH,PCFLAG[BX]   ; check if XOFF was received
      06AF  80 E4 40                            AND     AH,RCVXOF       ;
      06B2  8B 57 12                            MOV     DX,PCS7201[BX]  ; get address of 7201 status/control register
      06B5  EC                                  IN      AL,DX           ; read the status
      06B6  24 04                               AND     AL,BIT2         ; get the transmit buffer empty bit
      06B8  80 CC 04                            OR      AH,BIT2         ; Now set bit 2 in AH if transmit buffer
      06BB  32 E0                               XOR     AH,AL           ; is not empty  Note 7201 (value in AL)
                                                                        ; has bit 2 set if buffer IS empty !
      06BD  B0 FF                               MOV     AL,0FFH         ; assume ok to transmit (AH zero)
      06BF  74 03                               JZ      L16_10$         ; If its so go exit
      06C1  32 C0                       L16_5$: XOR     AL,AL           ; Else set flag to indicate otherwise.
      06C3  F9                                  STC                     ; set carry flag to indicate cannot transmit
      06C4  C3                          L16_10$:        RET                     ; return to caller
```

```
                                ;-----------------------------------------------------------------------
                                ;
                                ;                         O U T C H A R
                                ;
                                ; This routine is called to put a character into the transmit buffer.
                                ; If for any reasons the character cannot be immediately placed on to
                                ; the buffer, a return is made to the caller, with an indication that
                                ; transmission was unsucessful. No attempt is made here to find out
                                ; why the character cannot be transmitted (for fast execution time), if
                                ; the 7201 has buffer empty, and the port is not xoff'd then the character
                                ; will be placed into the buffer.
                                ;
                                ;       ENTRY CONDITIONS:         CH      device number
                                ;                                 DS      has BIOS data area segment address
                                ;                                 DL      character to be transmitted
                                ;
                                ;       EXIT CONDITIONS:          DL      is destroyed in all cases.
                                ;                                         (this is ok, since in the next
                                ;                                          call, the value is reloaded from
                                ;                                          the BIOS descriptor).
                                ;                                 CL = 0  cannot send character
                                ;                                 CL = FF character placed into the buffer
                                ;
                                ;       Note, that to implement auto xon/xoff properly, the transmited
                                ;               characters have to be examined to see if xoff or xon
                                ;               is sent by the application (or user).
                                ;               Further, it is faster and easier to always check for this
                                ;               instead of only when auto transmit xon/xoff is enabled.
                                ;
                                ;----------------------------------------------------------------------

06C5                            OUTCHAR:
06C5   8A CA                            MOV     CL,DL             ; save character to be transmitted
06C7   9C                               PUSHF                     ; make sure interrutps are off
06C8   FA                               CLI                       ;       .
06C9   E8 06A7 R                        CALL    OUTSTAT           ; go see if we can transmit, also get PCB
06CC   86 C8                            XCHG    CL,AL             ; transfer return codes
06CE   72 25                            JC      OUTCHX            ; > exit if we can't transmit.

06D0   3A 47 2D                         CMP     AL,PCXON[BX]      ; check if xon charcter is sent
06D3   75 06                            JNZ     L17_1$            ; If it is, make sure "application sent xoff"
06D5   80 67 14 7F                      AND     BYTE PTR PCFLAG[BX], NOT APPXOF ; flag is cleared.
06D9   EB 0D                            JMP     SHORT L17_2$      ; now go send character
06DB   3A 47 2E                 L17_1$: CMP     AL,PCXOFF[BX]     ; check if xoff being sent
06DE   75 08                            JNZ     L17_2$            ; > If not go send the character
                                ;
                                ; else, set flag that application program sending xoff, and clear flag
                                ; to send auto xon as soon as possible.
                                ;
06E0   80 4F 14 80                      OR      BYTE PTR PCFLAG[BX],APPXOF
06E4   80 67 14 FD                      AND     BYTE PTR PCFLAG[BX],NOT XMTXON
```

```
06E8                            L17_2$:                           ; now prepare data to be transmitted
06E8   8B 57 15                         MOV     DX,PCMASK[BX]     ; mask the outgoing data
06EB   22 C6                            AND     AL,DH             ;         .
06ED   0A C2                            OR      AL,DL             ;         .
06EF   8B 57 10                         MOV     DX,PC7201[BX]     ; get data register address of 7201
06F2   EE                               OUT     DX,AL             ; put out the character
06F3   B1 FF                            MOV     CL,0FFH           ; set return code that character transmitted

06F5   9D                       OUTCHX: POPF                      ; restore caller's flags
06F6   C3                               RET                       ; return to caller
```

```



                                    ;------------------------------------------------------------------
                                    ;
                                    ;                         P U T C H A R
                                    ;
                                    ; This routine is called to transmit a character, and to repeat until
                                    ; the character is sucessfully accepted by the 7201
                                    ;
                                    ; ENTRY CONDITIONS:              same as OUTCHAR
                                    ;
                                    ; EXIT CONDITIONS:               same as OUTCHAR
                                    ;                                charcater will be taken by 7201
                                    ;------------------------------------------------------------------
      06F7                          PUTCHAR:
      06F7  52                              PUSH    DX              ; save character to be transmitted
      06F8  E8 06C5 R                       CALL    OUTCHAR         ; call common routine to output char
      06FB  5A                              POP     DX              ; restore character
      06FC  22 C9                           AND     CL,CL           ; check if it was sucessful
      06FE  74 F7                           JZ      PUTCHAR         ; repeat til sucessful
      0700  C3                              RET
```

```


                                    ;------------------------------------------------------------------
                                    ;
                                    ;                         O U T N O W
                                    ;
                                    ; This routine is called to transmit the character regardless of the
                                    ; state of the modem signals or anything else.
                                    ;
                                    ; Note that the absense of CTS, in limited modem control mode made
                                    ; prevent the transmission of the character indefinitely. It is for
                                    ; this reason that the CTS signal is ignored for the transmission of
                                    ; this particular character.
                                    ;
                                    ; ENTRY CONDITIONS:     CH      has device number
                                    ;                       DS      points to BIOS data area
                                    ;                       DL      has character to be transmitted
                                    ;
                                    ; EXIT CONDITIONS:              command ignored if port number invalid
                                    ;
                                    ; NOTE THAT THIS WILL NOT PROCESS auto XON/XOFF
                                    ;
                                    ;------------------------------------------------------------------

      0701                          OUTNOW:
      0701  E8 0911 R                       CALL    GETPCB          ; get address pf pcb in to DS:BX
      0704  73 01                           JNC     L18_1$          ; If port number invalid
      0706  C3                              RET                     ;       > return to caller
      0707                          L18_1$:                         ; Else continue
      0707  8A CA                           MOV     CL,DL           ; save character to be transmitted
      0709  8B 57 12                        MOV     DX,PCS7201[BX]  ; get status/control reg address of 7201
      070C  F6 47 0D 20                     TEST    BYTE PTR PCCR3[BX],BIT5 ; temporary disable "auto enable" if used
      0710  74 1E                           JZ      OUTN2           ; > If not used go wait for buffer empty
      0712  B0 03                           MOV     AL,CR3          ; (Now disable it)
      0714  9C                              PUSHF                   ;       make sure interrupts disabled
      0715  FA                              CLI                     ;       .
      0716  EE                              OUT     DX,AL           ;       .
      0717  8A 47 0D                        MOV     AL,PCCR3[BX]    ; get value programmed into control reg 3
      071A  24 DF                           AND     AL,0FFH-BIT5    ; take out auto enable bit
      071C  EE                              OUT     DX,AL           ; do it!
      071D  9D                              POPF                    ; restore interrupt state
      071E  E8 0730 R                       CALL    OUTN2           ; go wait for buffer empty and then send char
      0721  42                              INC     DX              ; point Dx back to status register
      0722  42                              INC     DX              ;       .
      0723  EC                      L18_5$: IN      AL,DX           ; wait for buffer empty
      0724  A8 04                           TEST    AL,BIT2         ;       .
      0726  75 FB                           JNZ     L18_5$          ;       .
      0728  8A 47 0D                        MOV     AL,PCCR3[BX]    ; get original value programmed into control reg 3
      072B  B4 03                           MOV     AH,CR3          ; set up to use common routine to do this
      072D  E9 0269 R                       JMP     WRITE7201       ; go to it, and bye bye!

      0730  9C                      OUTN2:  PUSHF                   ; make sure interrutps are off
      0731  FA                              CLI                     ;       .
```

```
0732  EC                        IN      AL,DX           ; wait for buffer empty
0733  A8 04                     TEST    AL,BIT2         ;
0735  75 03                     JNZ     L19_1$          ;
0737  9D                        POPF                    ; restore state of interrupt mask
0738  EB F6                     JMP     SHORT OUTN2     ; loop, till buffer empty
073A                    L19_1$:
073A  8A C1                     MOV     AL,CL           ; restore character to be transmitted
073C  8B 4F 15                  MOV     CX,PCMASK[BX]   ; get mask for character
073F  22 C5                     AND     AL,CH           ;
0741  0A C1                     OR      AL,CL           ;
0743  4A                        DEC     DX              ; point to data register
0744  4A                        DEC     DX              ;
0745  EE                        OUT     DX,AL           ; send it!
0746  9D                        POPF                    ; restore caller's interrupt mask
0747  C3                        RET                     ; return to caller
```

```



                        ;-----------------------------------------------------------------
                        ;
                        ;                       R D M O D E M
                        ;
                        ; This routine is called to read the modem signals.
                        ;
                        ;       ENTRY CONDITIONS:       CH      device number
                        ;
                        ;       EXIT CONDITIONS:
                        ;                               AL bit0-bit6    has the modem signals
                        ;
                        ;                               AL bit7 = 1     if modem signals not available
                        ;
                        ;                               AH bit 0 = 1    transmitter empty
                        ;                                        = 0    transmitter not empty
                        ;
                        ;-----------------------------------------------------------------
0748                    RDMODEM:
0748  33 C0                     XOR     AX,AX           ; set error code
074A  E8 0911 R                 CALL    GETPCB          ; get port control address
074D  72 25                     JC      L35_10$         ; > exit if port is invalid
074F  8B 57 06                  MOV     DX,PCMODM[BX]   ; get address of the modem signal port
0752  81 FA FFFF                CMP     DX,OFFFFH       ; see if modem signals can be read
0756  74 07                     JZ      L35_1$          ; If modem signals can be read then,
0758  EC                        IN      AL,DX           ;       read the signals
0759  34 FF                     XOR     AL,OFFH         ;       make it bit set means signal asserted
075B  24 1F                     AND     AL,1FH          ;       take out unwanted signals
075D  EB 02                     JMP     SHORT L35_2$    ;
075F  B0 80             L35_1$: MOV     AL,80H          ; Else, mark modem signals not available
0761  8A 67 14          L35_2$: MOV     AH,PCFLAG[BX]   ; read the state of the port flags
0764  80 E4 E0                  AND     AH,OE0H         ; take only high order 3 bits
0767  50                        PUSH    AX              ; save the information
0768  8B 57 12                  MOV     DX,PCS7201[BX]  ; get control/status register for 7201
076B  EC                        IN      AL,DX           ; check if transmitter empty
076C  A8 04                     TEST    AL,BIT2         ;
076E  58                        POP     AX              ; restore outgoing info
076F  74 03                     JZ      L35_10$         ; > if not empty exit
0771  80 CC 01                  OR      AH,BIT0         ; Else set flag that transmitter is empty
0774  C3                L35_10$:        RET                     ; return to caller.
```

```


                            ;-------------------------------------------------------------------
                            ;
                            ;                       S E T M O D E M
                            ;
                            ; This routine is called to set the modem signals. Note that
                            ; the first 3 bits in the mask are set (to clear the diagnostic leds)
                            ;
                            ;       ENTRY CONDITONS:        CH      device number
                            ;                               DL      modem signal mask
                            ;
                            ;       EXIT CONDITIONS:        AL = FF if sucessful
                            ;                               AL = 0  if port number invalid or,
                            ;                                       modem signal cannot be set
                            ;
                            ;-------------------------------------------------------------------
 0775                       SETMODEM:
 0775  8A C2                        MOV     AL,DL           ; put the modem signals into correct reg
 0777  E8 0911 R                    CALL    GETPCB          ; get port control block
 077A  72 11                        JC      L20_10$         ; > go exit if invalid port number
 077C  8B 57 06                     MOV     DX,PCMODM[BX]   ; get address of the port to set modem signals
 077F  81 FA FFFF                   CMP     DX,0FFFFH       ; see if modem signals can be set
 0783  74 08                        JZ      L20_10$         ; > if not go exit with error condition set
 0785  24 0F                        AND     AL,0FH          ; clear first 4 bits
 0787  34 FF                        XOR     AL,0FFH         ; (zero bit value asserts the line)
 0789  EE                           OUT     DX,AL           ; do it.
 078A  B0 FF                        MOV     AL,0FFH         ; set command processed return code
 078C  C3                           RET                     ; return to caller

 078D  32 C0                L20_10$:        XOR     AL,AL           ; set error return code
 078F  C3                           RET                     ; return to caller
```

```


                            ;-------------------------------------------------------------------
                            ;
                            ;                       B R K O N
                            ;
                            ; This routine is called to initiate the sending of break.
                            ;
                            ;       ENTRY CONDITIONS:       CH      port number
                            ;
                            ;       EXIT CONDTIONS:         command ignored if port number invalid
                            ;
                            ;-------------------------------------------------------------------
 0790                       BRKON:
 0790  E8 0911 R                    CALL    GETPCB          ; get address of port control block
 0793  73 01                        JNC     L21_1$          ; If port number invalid,
 0795  C3                           RET                     ;       return to caller
 0796                       L21_1$:                         ; Else,
 0796  8A 4F 14                     MOV     CL,PCFLAG[BX]   ; set flag to indicate break being sent
 0799  80 C9 08                     OR      CL,XMTBRK       ;
 079C  8A 47 0F                     MOV     AL,PCCR5[BX]    ; set byte to be written to 7201
 079F  0C 10                        OR      AL,BIT4         ;
 07A1  EB 11                        JMP     SHORT BRKCOMMON ; go and write it to 7201


                            ;-------------------------------------------------------------------
                            ;
                            ;                       B R K O F F
                            ;
                            ; This routine is called to stop the sending of break.
                            ;
                            ;       ENTRY CONDITIONS:       CH      port number
                            ;
                            ;       EXIT CONDTIONS:         Command ignored if port is invalid
                            ;
                            ;-------------------------------------------------------------------
 07A3                       BRKOFF:
 07A3  E8 0911 R                    CALL    GETPCB          ; get address of port control block
 07A6  73 01                        JNC     L22_1$          ; If invalid port number
 07A8  C3                           RET                     ; > return to caller

 07A9  8A 4F 14             L22_1$: MOV     CL,PCFLAG[BX]   ; clear flag that show break being sent
 07AC  80 E1 F7                     AND     CL,NOT XMTBRK   ;
 07AF  8A 47 0F                     MOV     AL,PCCR5[BX]    ; get old value in control register 5
 07B2  24 EF                        AND     AL,NOT BIT4     ; clear send break bit in 7201
                            ;
                            ;       will have to consider sending the auto Xon.
                            ;       This is messy, but necessary for proper handling!
                            ;       If buffer empty, and auto xoff was sent, we must send it
                            ;       now, since it may be the last time we get control.
                            ;       host is Xoff'd, and missed Xon because of break being sent!
                            ;

```

```
        07B4                            BRKCOMMON:
        07B4  88 4F 14                          MOV     PCFLAG[BX],CL   ; restore flag
        07B7  B4 05                             MOV     AH,CR5          ; This is to be written to control reg 5
        07B9  E9 0269 R                         JMP     WRITE7201       ; Go to common subroutine to do it.
```

```


                                        ;-------------------------------------------------------------------
                                        ;
                                        ;                           R C V I N T
                                        ;
                                        ; This routine is called to set up the receive character interrupt
                                        ; service routine.
                                        ;
                                        ;       ENTRY CONDITIONS:       CX,DX   points to user information block
                                        ;
                                        ;
                                        ;       EXIT CONDITIONS:        AL = 0  invalid port number
                                        ;                               AL = FF function sucessfully performed.
                                        ;
                                        ;-------------------------------------------------------------------

        07BC                            RCVINT:
        07BC  8B E9                             MOV     BP,CX                   ; point ES:BP to user's control block
        07BE  8E C2                             MOV     ES,DX                   ;
        07C0  26: 8A 6E 00                      MOV     CH,ES:[BP]              ; get port number
        07C4  32 C0                             XOR     AL,AL                   ; assume invalid port number
        07C6  E8 0811 R                         CALL    GETPCB                  ; get address of port control block
        07C9  72 13                             JC      L23_10$                 ; > exit if port number invalid

        07CB  26: C4 46 01                      LES     AX,DWORD PTR ES:1[BP]   ; get address of routine to be called
        07CF  9C                                PUSHF                           ; disable interrupts to do rest
        07D0  FA                                CLI                             ;
        07D1  89 47 18                          MOV     PCRCVA[BX],AX           ; put it into the port control block
        07D4  8C 47 1A                          MOV     PCRCVA+2[BX],ES         ; put segment address into PC blcok
        07D7  C6 47 17 01                       MOV     BYTE PTR PCRCVF[BX],1   ; set flag to indicate its on
        07DB  9D                                POPF                            ; restore user's flag state
        07DC  B0 FF                             MOV     AL,0FFH                 ; set sucessful return code

        07DE  C3                        L23_10$:        RET                             ; return to caller


                                        ;-------------------------------------------------------------------
                                        ;
                                        ;                         R C V C A N C E L
                                        ;
                                        ; This routine is called to cancel the receive character interrupt
                                        ; service routine.
                                        ;
                                        ;       ENTRY CONDITIONS:       CH      port number
                                        ;
                                        ;       EXIT CONDITIONS:        command ignored if port number invalid
                                        ;
                                        ;-------------------------------------------------------------------
        07DF                            RCVCANCEL:
        07DF  E8 0811 R                         CALL    GETPCB                  ; get address of port control block
        07E2  72 04                             JC      L24_1$                  ; ignore command if invalid port #
        07E4  C6 47 17 00                       MOV     BYTE PTR PCRCVF[BX],0   ; clear flag to call service routine
```

```
07E8  C3                              L24_1$: RET                               ; return to caller
```

```



                                        ;----------------------------------------------------------------------
                                        ;
                                        ;                          X M T M T Y
                                        ;
                                        ; This routine is called to set up the transmit empty buffer interrupt
                                        ; service routine.
                                        ;
                                        ;       ENTRY CONDITIONS:       CX,DX   points to the BIOS descriptor
                                        ;
                                        ;       EXIT CONDITIONS:        AL = 0  if port number invalid or not
                                        ;                                       connected.
                                        ;
                                        ;                               AL = FF if command sucessfully implemented
                                        ;
                                        ;----------------------------------------------------------------------
07E9                                    XMTMTY:
07E9  8B E9                                     MOV     BP,CX                   ; point ES:BP to user's control block
07EB  8E C2                                     MOV     ES,DX                   ;       .
07ED  26: 8A 6E 00                              MOV     CH,ES:[BP]              ; get port number
07F1  32 C0                                     XOR     AL,AL                   ; assume invalid port number
07F3  E8 0911 R                                 CALL    GETPCB                  ; get address of port control block
07F6  72 20                                     JC      L25_1$                  ; > ignore command if invalid port
07F8  26: C4 46 01                              LES     AX,DWORD PTR ES:1[BP]   ; get address of routine to be called
07FC  9C                                        PUSHF                           ; disable interrupts to do rest
07FD  FA                                        CLI                             ;       .
07FE  89 47 1D                                  MOV     PCXMTA[BX],AX           ; put it into the port control block
0801  8C 47 1F                                  MOV     PCXMTA+2[BX],ES         ; put segment address into PC blcok
0804  C6 47 1C 01                               MOV     BYTE PTR PCXMTF[BX],1   ; set flag to indicate its on
0808  9D                                        POPF                            ; restore user's flag state
0809  8A 47 0B                                  MOV     AL,PCCR1[BX]            ; get image of control register 1
080C  0C 02                                     OR      AL,CR1TXBE              ;       .
080E  88 47 0B                                  MOV     PCCR1[BX],AL            ; save new value
0811  B4 01                                     MOV     AH,1                    ; go write it to 7201
0813  E8 0269 R                                 CALL    WRITE7201               ;       .
0816  B0 FF                                     MOV     AL,0FFH                 ; set return sucessful code
0818  C3                                L25_1$: RET                             ; return to caller


                                        ;----------------------------------------------------------------------
                                        ;
                                        ;                        X M T C A N C E L
                                        ;
                                        ; This routine is called to cancel the transmit empty buffer interrupt
                                        ; service routine.
                                        ;
                                        ;       ENTRY CONDITIONS:       CH      port number
                                        ;
                                        ;       EXIT CONDITIONS:
                                        ;
                                        ;----------------------------------------------------------------------
0819                                    XMTCANCEL:
```

```
0819  E8 0911 R                     CALL   GETPCB                  ; get address of port control block
081C  72 11                         JC     L26_1$                  ; ignore command if port number invalid
081E  C6 47 1C 00                   MOV    BYTE PTR PCXMTF[BX],0   ; clear flag to call service routine
0822  8A 47 0B                      MOV    AL,PCCR1[BX]            ; get image of control register 1
0825  24 FD                         AND    AL,NOT CR1TXBE          ;
0827  88 47 0B                      MOV    PCCR1[BX],AL            ; save new value
082A  B4 01                         MOV    AH,1                    ; go write it to 7201
082C  E8 0289 R                     CALL   WRITE7201               ;
082F  C3                    L26_1$: RET                            ; return to caller
```

```


                            ;----------------------------------------------------------------
                            ;
                            ;                          H A C K 7 2 0 1
                            ;
                            ; This routine is called to setup the interrupt vectors so that on
                            ; the following interrupt from the appropriate 7201, the application
                            ; service routine is called instead of the one in the BIOS.
                            ;
                            ;
                            ;       ENTRY CONDITION:        CX,DX   point to a 5 byte descriptor
                            ;                                       where the first byte is the
                            ;                                       device number, and the next 4
                            ;                                       are the offset, and segment of
                            ;                                       the user service routine.
                            ;
                            ;       EXIT CONDITIONS:        NONE, no check is done at all.
                            ;
                            ;
                            ; DEVICE NUMBER ARE AS FOLLOWS:
                            ;
                            ;                               1 = comm port (PORT A)
                            ;                               2 = printer port (PORT B)
                            ;                               3 = optional comm (RS423) PORT B
                            ;                               4 = optional comm (RS422) PORT A
                            ;
                            ;----------------------------------------------------------------
0830                        HACK7201:
0830  8B E9                         MOV    BP,CX             ; point ES:BP to descriptor area
0832  8E C2                         MOV    ES,DX             ;
0834  26: 8A 5E 00                  MOV    BL,ES:[BP]        ; get device number
0838  32 FF                         XOR    BH,BH             ; make it into offset to table
083A  4B                            DEC    BX                ;
083B  D1 E3                         SHL    BX,1              ;
083D  26: C4 6E 01                  LES    BP,DWORD PTR ES:1[BP]   ; point ES:BP to user service routine
0841  9C                            PUSHF                    ; make sure interrupts are off
0842  06                            PUSH   ES                ; save segment of service routine
0843  FA                            CLI                      ;
0844  8B 87 0865 R                  MOV    AX,HACKTAB[BX]    ; get address of the kludge to jump to user
0848  8B BF 086D R                  MOV    DI,FIXTAB[BX]     ; now modify the offset of service routines
084C  B9 0004                       MOV    CX,4              ;       (4 offsets to be fixed)
084F  1E                            PUSH   DS                ;       point ES to bios area
0850  07                            POP    ES                ;
0851  FC                            CLD                      ;       (make sure of directions correct)
0852  AB                    L27_1$: STOS   WORD PTR[DI]      ;
0853  E2 FD                         LOOP   L27_1$            ;       repeat for 4 offset values

0855  07                            POP    ES                ; restore segment of service routine
0856  D1 E3                         SHL    BX,1              ; calculate save area for vector (dev# -1)*4
0858  89 AF 0000 E                  MOV    HVECTOR[BX],BP    ; save user service routines into the table
085C  83 C3 02                      ADD    BX,2              ;
085F  8C 87 0000 E                  MOV    HVECTOR[BX],ES    ;
```

```
2157      0863  9D                          POPF                     ; reinstate interrupt status
2158      0864  C3                          RET                      ; return to caller
2159
2160      0865                      HACKTAB LABEL   WORD
2161      0865  012A R                      DW      HACKA
2162      0867  012F R                      DW      HACKB
2163      0869  0134 R                      DW      HACKXB
2164      086B  0139 R                      DW      HACKXA
2165
2166      086D                      FIXTAB  LABEL   WORD
2167      086D  00F2 R                      DW      INT7201+8
2168      086F  00EA R                      DW      INT7201
2169      0871  00FA R                      DW      INT7201+16
2170      0873  0102 R                      DW      INT7201+24
```

```
2171
2172
2173                                ;-----------------------------------------------------------------
2174                                ;
2175                                ;                          V E C T 7 2 0 1
2176                                ;
2177                                ; This routine is called to clean the vectors so that they will point
2178                                ; to the default service routines.
2179                                ;
2180                                ;
2181                                ;       ENTRY CONDITION:        CH      has device number
2182                                ;
2183                                ;       EXIT CONDITIONS:        NONE, no check is done at all.
2184                                ;
2185                                ; DEVICE NUMBER ARE AS FOLLOWS:
2186                                ;
2187                                ;                               1 = comm port (PORT A)
2188                                ;                               2 = printer port (PORT B)
2189                                ;                               3 = optional comm (RS423) PORT B
2190                                ;                               4 = optional comm (RS422) PORT A
2191                                ;
2192                                ;-----------------------------------------------------------------
2193      0875                      VECT7201:
2194      0875  8A DD                       MOV     BL,CH             ; calculate the offset from device number
2195      0877  32 FF                       XOR     BH,BH             ;
2196      0879  4B                          DEC     BX                ;
2197      087A  D1 E3                       SHL     BX,1              ;
2198      087C  8B BF 0892 R                MOV     DI,VECTAB [BX]    ; point to table to be fixed
2199      0880  8B B7 089A R                MOV     SI,OURTAB [BX]    ; point to default table
2200      0884  B9 0004                     MOV     CX,4              ; 4 offsets to be fixed
2201      0887  FC                          CLD                       ; make sure of direction for STOS, LODS
2202      0888  1E                          PUSH    DS                ; point ES to BIOS area
2203      0889  07                          POP     ES                ;
2204      088A  9C                          PUSHF                     ; make sure were not interrupted
2205      088B  FA                          CLI                       ;
2206      088C  AD                  L28_1$: LODS    WORD PTR [SI]     ; repeat  get default vector
2207      088D  AB                          STOS    WORD PTR [DI]     ;         put it into vector table
2208      088E  E2 FC                       LOOP    L28_1$            ; until all 4 vectors (per port) are fixed
2209      0890  9D                          POPF                      ; reinstate interrupts status
2210      0891  C3                          RET                       ; return to caller
2211
2212      0892                      VECTAB  LABEL   WORD
2213      0892  00F2 R                      DW      INT7201+8
2214      0894  00EA R                      DW      INT7201
2215      0896  00FA R                      DW      INT7201+16
2216      0898  0102 R                      DW      INT7201+24
2217
2218      089A                      OURTAB  LABEL   WORD
2219      089A  0112 R                      DW      PORTA
2220      089C  010A R                      DW      PORTB
2221      089E  011A R                      DW      PORTXB
2222      08A0  0122 R                      DW      PORTXA
```

```


                                ;-----------------------------------------------------------------
                                ;
                                ;                           S T A T C H G
                                ;
                                ; This routine is called to set up the status change interrupt
                                ; service routine.
                                ;
                                ;       ENTRY CONDITIONS:       CX,DX   points to user information block
                                ;
                                ;
                                ;       EXIT CONDITIONS:        AL = 0  invalid port number
                                ;                               AL = FF function sucessfully performed.
                                ;
                                ;-----------------------------------------------------------------
08A2                            STATCHG:
08A2  8B E9                             MOV     BP,CX                   ; point ES:BP to user's control block
08A4  8E C2                             MOV     ES,DX                   ;     .
08A6  26: 8A 6E 00                      MOV     CH,ES:[BP]              ; get port number
08AA  32 C0                             XOR     AL,AL                   ; assume invalid port number
08AC  E8 0911 R                         CALL    GETPCB                  ; get address of port control block
08AF  72 13                             JC      L29_10$                 ; > exit if port number invalid

08B1  26: C4 46 01                      LES     AX,DWORD PTR ES:1[BP]   ; get address of routine to be called
08B5  9C                                PUSHF                           ; disable interrupts to do rest
08B6  FA                                CLI                             ;     .
08B7  89 47 22                          MOV     PCSTCA[BX],AX           ; put it into the port control block
08BA  8C 47 24                          MOV     PCSTCA+2[BX],ES         ; put segment address into PC blcok
08BD  C6 47 21 01                       MOV     BYTE PTR PCSTCF[BX],1   ; set flag to indicate its on
08C1  9D                                POPF                            ; restore user's flag state
08C2  B0 FF                             MOV     AL,0FFH                 ; set sucessful return code

08C4  C3                        L29_10$:        RET                             ; return to caller

                                ;-----------------------------------------------------------------
                                ;
                                ;                           S T C A N C E L
                                ;
                                ; This routine is called to cancel the status change interrupt
                                ; service routine.
                                ;
                                ;       ENTRY CONDITIONS:       CH      port number
                                ;
                                ;       EXIT CONDITIONS:        command ignored if port number invalid
                                ;
                                ;-----------------------------------------------------------------
08C5                            STCANCEL:
08C5  E8 0911 R                         CALL    GETPCB                  ; get address of port control block
08C8  72 04                             JC      L30_1$                  ; ignore command if invalid port #
08CA  C6 47 21 00                       MOV     BYTE PTR PCSTCF[BX],0   ; clear flag to call service routine
08CE  C3                        L30_1$: RET                             ; return to caller
```

```


                                ;----------------------------------------------------------------------
                                ;
                                ;                          I N I T C O M
                                ;
                                ; This is the hard reset routine to reset all the variables and tables for
                                ; all three ports (comm, printer and extended comm if it isinstalled)
                                ;
                                ; ENTRY CONDITIONS:     DS              points to BIOS data area
                                ;                       DS:XOPTION      contains 2, if optional comm
                                ;                                       port present.
                                ;
                                ; EXIT CONDITIONS:
                                ;
                                ;----------------------------------------------------------------------
08CF                            INITCOM:
08CF  BA 0042                           MOV     DX,AS7201               ; reset 7201 on mother board
08D2  E8 0904 R                         CALL    RESET7201               ;     .

08D5  80 3E 0000 E 02                   CMP     XOPTION,2               ; check if optional comm board present
08DA  C6 06 0003 E 01                   MOV     PCXCOM+PCSTAT,1         ; mark that optional board not present
08DF  75 0B                             JNZ     L31_1$                  ; > if not skip optional board's 7201
08E1  C6 06 0003 E 00                   MOV     PCXCOM+PCSTAT,0         ; mark that optional board present
08E6  BA 002A                           MOV     DX,XAS7201              ; reset 7201 on the optional board
08E9  E8 0904 R                         CALL    RESET7201               ;     .

08EC                            L31_1$:                                 ; (now set default values to each port)
08EC  B5 03                             MOV     CH,DEVMAX               ; start with highest device number
08EE  E8 0911 R                 L31_2$: CALL    GETPCB                  ; get port control block address
08F1  72 0C                             JC      L31_3$                  ; skip this port if device don't exist
08F3  C6 47 0B 15                       MOV     BYTE PTR PCCR1[BX],CR17201 ; set CR1 so to be programmed later
08F7  E8 0577 R                         CALL    DFLT7201                ; to default values
08FA  32 D2                             XOR     DL,DL                   ; deassert modem signals, and for the
08FC  E8 0775 R                         CALL    SETMODEM                ; xcomm port, set internal clocking
08FF  FE CD                     L31_3$: DEC     CH                      ; repeat for all devices
0901  75 EB                             JNZ     L31_2$                  ;     .

                                ; Note that if AL is 0FFh, some part of the default was not programmed
                                ; this is set by PRG7201

0903  C3                                RET


0904                            RESET7201:
0904  9C                                PUSHF                           ; first disable interrupts
0905  FA                                CLI                             ;     .
0906  B0 18                             MOV     AL,RST7201              ; reset 7201
0908  EE                                OUT     DX,AL                   ; channel A
0909  B0 02                             MOV     AL,CR2                  ; point to control register 2A
090B  EE                                OUT     DX,AL                   ;     .
090C  B0 10                             MOV     AL,CR27201              ; put 7201 in status affect vector mode
090E  EE                                OUT     DX,AL                   ;     .
```

```
090F  9D                POPF                        ; all done, restore interrupt flags
0910  C3                RET
```

BIOS EXTENTION SERVICE ROUTINES

```
                                                        ENDIF
                                          ;----------------------------------------------------------------
                                          ;
                                          ;                               G E T P C B
                                          ;
                                          ; GETPCB is called to get the port control block for the appropriate
                                          ; device.
                                          ;
                                          ; ENTRY CONDITIONS:       CH = device number
                                          ;                         DS = address of bios data area
                                          ;
                                          ; EXIT CONDITION:         If port number valid & known to be connected then:
                                          ;                                 BX = address of the port control block.
                                          ;                                 DS = segment address of the bios data area
                                          ;                                 All other registers are untouched
                                          ;                                 CARRY FLAG CLEARED
                                          ;
                                          ;                         ELSE, Carry flag set.
                                          ;
                                          ;----------------------------------------------------------------

       0911                               GETPCB:
       0911  80 FD 03                             CMP     CH,DEVMAX                 ; check if port number valid
       0914  77 14                                JA      L33_1$                    ; > if not go set error flag & return
       0916  22 ED                                AND     CH,CH                     ;
       0918  74 10                                JZ      L33_1$                    ;
       091A  8A DD                                MOV     BL,CH                     ; set up table pointer
       091C  32 FF                                XOR     BH,BH                     ;
       091E  D1 E3                                SHL     BX,1                      ;
       0920  8B 9F 092A R                         MOV     BX,PCBTAB-2[BX]           ; do table lookup.
       0924  80 7F 03 00                          CMP     BYTE PTR PCSTAT[BX],0     ; check if port present
       0928  74 01                                JZ      L33_2$                    ; > go exit if present
       092A  F9                           L33_1$: STC                               ; set return code not to touch port
       092B  C3                           L33_2$: RET                               ; return to caller


       092C                               PCBTAB  LABEL   WORD
       092C  0000 E                               DW      OFFSET PCCOM    ; comm port control block
       092E  0000 E                               DW      OFFSET PCPRT    ; printer port control block
       0930  0000 E                               DW      OFFSET PCXCOM   ; extended comm port control blockM

       0932                               CODE    ENDS
                                                  END
```

```
Macros:

                N a m e                 Length

CALRET . . . . . . . . . . . . . .      0001
SEQ. . . . . . . . . . . . . . . .      0002
TABLE. . . . . . . . . . . . . . .      0002

Segments and groups:

                N a m e                 Size    align   combine class

CGROUP . . . . . . . . . . . . . .      GROUP
  CODE . . . . . . . . . . . . . . .    0932    BYTE    PUBLIC  'CODE'

Symbols:

                N a m e                 Type    Value   Attr

A7201. . . . . . . . . . . . . . .      Number  0040
ABAUD. . . . . . . . . . . . . . .      Number  000E
AMODEM . . . . . . . . . . . . . .      Number  0002
APPXOF . . . . . . . . . . . . . .      Alias   BIT7
ARXDAT . . . . . . . . . . . . . .      L NEAR  013E    CODE
ARXSPC . . . . . . . . . . . . . .      L NEAR  0248    CODE
AS7201 . . . . . . . . . . . . . .      Number  0042
ASCSUB . . . . . . . . . . . . . .      Number  001A
ASTAT. . . . . . . . . . . . . . .      L NEAR  0295    CODE
ATXEMT . . . . . . . . . . . . . .      L NEAR  0215    CODE
AUTOXMT. . . . . . . . . . . . . .      L NEAR  01F9    CODE
B7201. . . . . . . . . . . . . . .      Number  0041
BAKGRND. . . . . . . . . . . . . .      Number  24AF
BAUDMAX. . . . . . . . . . . . . .      Number  0011
BBAUD. . . . . . . . . . . . . . .      Number  0006
BIT0 . . . . . . . . . . . . . . .      Number  0001
BIT1 . . . . . . . . . . . . . . .      Number  0002
BIT2 . . . . . . . . . . . . . . .      Number  0004
BIT3 . . . . . . . . . . . . . . .      Number  0008
BIT4 . . . . . . . . . . . . . . .      Number  0010
BIT5 . . . . . . . . . . . . . . .      Number  0020
BIT6 . . . . . . . . . . . . . . .      Number  0040
BIT7 . . . . . . . . . . . . . . .      Number  0080
BMODEM . . . . . . . . . . . . . .      Number  00FF
BRKCOMMON. . . . . . . . . . . . .      L NEAR  07B4    CODE
BRKERR . . . . . . . . . . . . . .      Alias   BIT7
BRKOFF . . . . . . . . . . . . . .      L NEAR  07A3    CODE    Global
BRKON. . . . . . . . . . . . . . .      L NEAR  0790    CODE    Global
BRXDAT . . . . . . . . . . . . . .      L NEAR  0143    CODE
BRXSPC . . . . . . . . . . . . . .      L NEAR  023E    CODE
BS7201 . . . . . . . . . . . . . .      Number  0043
BSTAT. . . . . . . . . . . . . . .      L NEAR  028B    CODE
BTXEMT . . . . . . . . . . . . . .      L NEAR  021F    CODE
BUFF1. . . . . . . . . . . . . . .      L NEAR  052B    CODE
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE      Symbols-2
COMBIOS


BUFF10 . . . . . . . . . . . . . .          L NEAR   0531      CODE
BUFFER . . . . . . . . . . . . . .          L NEAR   0514      CODE
BUFFX. . . . . . . . . . . . . . .          L NEAR   0576      CODE
CBDATB . . . . . . . . . . . . . .          Number   0003
CBLEN. . . . . . . . . . . . . . .          Number   0011
CBMODE . . . . . . . . . . . . . .          Number   0001
CBPORT . . . . . . . . . . . . . .          Number   0000
CBPRTY . . . . . . . . . . . . . .          Number   0004
CBRADR . . . . . . . . . . . . . .          Number   000D
CBRCVB . . . . . . . . . . . . . .          Number   0005
CBRXOF . . . . . . . . . . . . . .          Number   0009
CBSIZE . . . . . . . . . . . . . .          Number   000B
CBSTART. . . . . . . . . . . . . .          L BYTE   0000      CODE
CBSTPB . . . . . . . . . . . . . .          Number   0002
CBTXB. . . . . . . . . . . . . . .          Number   0006
CBTXOF . . . . . . . . . . . . . .          Number   000A
CBXOFF . . . . . . . . . . . . . .          Number   0008
CBXON. . . . . . . . . . . . . . .          Number   0007
CLK16X . . . . . . . . . . . . . .          Number   0040
COMM_STACK . . . . . . . . . . . .          E NEAR   00EA      CODE
CR1. . . . . . . . . . . . . . . .          Number   0001
CR17201. . . . . . . . . . . . . .          Number   0015
CR1TXBE. . . . . . . . . . . . . .          Number   0002
CR2. . . . . . . . . . . . . . . .          Number   0002
CR27201. . . . . . . . . . . . . .          Number   0010
CR3. . . . . . . . . . . . . . . .          Number   0003
CR4. . . . . . . . . . . . . . . .          Number   0004
CR5. . . . . . . . . . . . . . . .          Number   0005
CR7201 . . . . . . . . . . . . . .          L NEAR   0083      CODE
CTS. . . . . . . . . . . . . . . .          Alias    BIT3
DATABIT. . . . . . . . . . . . . .          L NEAR   03E7      CODE
DATABX . . . . . . . . . . . . . .          L NEAR   044F      CODE
DATMAX . . . . . . . . . . . . . .          Number   0006
DC1. . . . . . . . . . . . . . . .          Number   0011
DC3. . . . . . . . . . . . . . . .          Number   0013
DEVCOM . . . . . . . . . . . . . .          Number   0001
DEVMAX . . . . . . . . . . . . . .          Number   0003
DEVPRT . . . . . . . . . . . . . .          Number   0002
DEVXCOM. . . . . . . . . . . . . .          Number   0003
DFLT7201 . . . . . . . . . . . . .          L NEAR   0577      CODE      Global
DFLTBUF. . . . . . . . . . . . . .          L NEAR   05B9      CODE      Global
DFLTCOM. . . . . . . . . . . . . .          V BYTE   0000      CODE      External
DFLTPRT. . . . . . . . . . . . . .          V BYTE   0000      CODE      External
DFLTXCOM . . . . . . . . . . . . .          V BYTE   0000      CODE      External
DSR. . . . . . . . . . . . . . . .          Alias    BIT2
DTR. . . . . . . . . . . . . . . .          Alias    BIT2
ENDTXBE. . . . . . . . . . . . . .          Number   0028
EOI7201. . . . . . . . . . . . . .          Number   0038
ERR7201. . . . . . . . . . . . . .          Number   0030
FIXTAB . . . . . . . . . . . . . .          L WORD   086D      CODE
FRMERR . . . . . . . . . . . . . .          Alias    BIT6
GETBAUD. . . . . . . . . . . . . .          L NEAR   04E1      CODE
GETCHAR. . . . . . . . . . . . . .          L NEAR   0660      CODE      Global
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE      Symbols-3
COMBIOS


GETPCB . . . . . . . . . . . . . .          L NEAR   0911      CODE
HACK7201 . . . . . . . . . . . . .          L NEAR   0830      CODE      Global
HACKA. . . . . . . . . . . . . . .          L NEAR   012A      CODE
HACKB. . . . . . . . . . . . . . .          L NEAR   012F      CODE
HACKTAB. . . . . . . . . . . . . .          L WORD   0865      CODE
HACKXA . . . . . . . . . . . . . .          L NEAR   0139      CODE
HACKXB . . . . . . . . . . . . . .          L NEAR   0134      CODE
HVECTOR. . . . . . . . . . . . . .          V WORD   0000      CODE      External
INCHA1 . . . . . . . . . . . . . .          L NEAR   0656      CODE
INCHA2 . . . . . . . . . . . . . .          L NEAR   0658      CODE
INCHAR . . . . . . . . . . . . . .          L NEAR   0626      CODE      Global
INCHAX . . . . . . . . . . . . . .          L NEAR   065E      CODE
INITCOM. . . . . . . . . . . . . .          L NEAR   08CF      CODE      Global
INSTAT . . . . . . . . . . . . . .          L NEAR   0612      CODE      Global
INSTX. . . . . . . . . . . . . . .          L NEAR   0625      CODE
INT7201. . . . . . . . . . . . . .          L WORD   00EA      CODE
INTEXIT. . . . . . . . . . . . . .          L NEAR   0071      CODE
L10_10$. . . . . . . . . . . . . .          L NEAR   05B8      CODE
L11_1$ . . . . . . . . . . . . . .          L NEAR   05BF      CODE
L12_10$. . . . . . . . . . . . . .          L NEAR   05E9      CODE
L13_1$ . . . . . . . . . . . . . .          L NEAR   05F0      CODE
L14_1$ . . . . . . . . . . . . . .          L NEAR   0601      CODE
L15_1$ . . . . . . . . . . . . . .          L NEAR   062E      CODE
L16_10$. . . . . . . . . . . . . .          L NEAR   06C4      CODE
L16_5$ . . . . . . . . . . . . . .          L NEAR   06C1      CODE
L17_1$ . . . . . . . . . . . . . .          L NEAR   06DB      CODE
L17_2$ . . . . . . . . . . . . . .          L NEAR   06E8      CODE
L18_1$ . . . . . . . . . . . . . .          L NEAR   0707      CODE
L18_5$ . . . . . . . . . . . . . .          L NEAR   0723      CODE
L19_1$ . . . . . . . . . . . . . .          L NEAR   073A      CODE
L1_1$. . . . . . . . . . . . . . .          L NEAR   0054      CODE
L20_10$. . . . . . . . . . . . . .          L NEAR   078D      CODE
L21_1$ . . . . . . . . . . . . . .          L NEAR   0796      CODE
L22_1$ . . . . . . . . . . . . . .          L NEAR   07A9      CODE
L23_10$. . . . . . . . . . . . . .          L NEAR   07DE      CODE
L24_1$ . . . . . . . . . . . . . .          L NEAR   07E8      CODE
L25_1$ . . . . . . . . . . . . . .          L NEAR   0818      CODE
L26_1$ . . . . . . . . . . . . . .          L NEAR   082F      CODE
L27_1$ . . . . . . . . . . . . . .          L NEAR   0852      CODE
L28_1$ . . . . . . . . . . . . . .          L NEAR   088C      CODE
L29_10$. . . . . . . . . . . . . .          L NEAR   08C4      CODE
L2_10$ . . . . . . . . . . . . . .          L NEAR   0211      CODE
L30_1$ . . . . . . . . . . . . . .          L NEAR   08CE      CODE
L31_1$ . . . . . . . . . . . . . .          L NEAR   08EC      CODE
L31_2$ . . . . . . . . . . . . . .          L NEAR   08EE      CODE
L31_3$ . . . . . . . . . . . . . .          L NEAR   08FF      CODE
L33_1$ . . . . . . . . . . . . . .          L NEAR   092A      CODE
L33_2$ . . . . . . . . . . . . . .          L NEAR   092B      CODE
L34_1$ . . . . . . . . . . . . . .          L NEAR   04F5      CODE
L34_10$. . . . . . . . . . . . . .          L NEAR   04FA      CODE
L35_1$ . . . . . . . . . . . . . .          L NEAR   075F      CODE
L35_10$. . . . . . . . . . . . . .          L NEAR   0774      CODE
L35_2$ . . . . . . . . . . . . . .          L NEAR   0761      CODE
```

```
L3_10$ . . . . . . . . . . . . . .          L NEAR   02D5    CODE
L4_1$. . . . . . . . . . . . . . .          L NEAR   0330    CODE
L4_10$ . . . . . . . . . . . . . .          L NEAR   0341    CODE
L5_1$. . . . . . . . . . . . . . .          L NEAR   0361    CODE
L5_13$ . . . . . . . . . . . . . .          L NEAR   03A2    CODE
L5_15$ . . . . . . . . . . . . . .          L NEAR   03AD    CODE
L5_2$. . . . . . . . . . . . . . .          L NEAR   0383    CODE
L5_3$. . . . . . . . . . . . . . .          L NEAR   0390    CODE
L6_1$. . . . . . . . . . . . . . .          L NEAR   03CB    CODE
L7_1$. . . . . . . . . . . . . . .          L NEAR   0428    CODE
L7_10$ . . . . . . . . . . . . . .          L NEAR   03FC    CODE
L8_10$ . . . . . . . . . . . . . .          L NEAR   04A0    CODE
L8_5$. . . . . . . . . . . . . . .          L NEAR   0493    CODE
L9_1$. . . . . . . . . . . . . . .          L NEAR   04C9    CODE
L9_4$. . . . . . . . . . . . . . .          L NEAR   04D9    CODE
MODE . . . . . . . . . . . . . . .          L NEAR   03B6    CODE
MODE1. . . . . . . . . . . . . . .          L NEAR   03D6    CODE
MODEMAX. . . . . . . . . . . . . .          Number   0002
MODEX. . . . . . . . . . . . . . .          L NEAR   03E6    CODE
NXTINT . . . . . . . . . . . . . .          L NEAR   002A    CODE
NXTINT1. . . . . . . . . . . . . .          L NEAR   0065    CODE
OURTAB . . . . . . . . . . . . . .          L WORD   089A    CODE
OUTCHAR. . . . . . . . . . . . . .          L NEAR   06C5    CODE     Global
OUTCHX . . . . . . . . . . . . . .          L NEAR   06F5    CODE
OUTN2. . . . . . . . . . . . . . .          L NEAR   0730    CODE
OUTNOW . . . . . . . . . . . . . .          L NEAR   0701    CODE     Global
OUTSTAT. . . . . . . . . . . . . .          L NEAR   06A7    CODE     Global
OVRERR . . . . . . . . . . . . . .          Alias    BIT5
PARERR . . . . . . . . . . . . . .          Alias    BIT4
PARITY . . . . . . . . . . . . . .          L NEAR   0342    CODE
PC7201 . . . . . . . . . . . . . .          Number   0010
PCBAUD . . . . . . . . . . . . . .          Number   0008
PCBTAB . . . . . . . . . . . . . .          L WORD   092C    CODE
PCCOM. . . . . . . . . . . . . . .          V BYTE   0000    CODE     External
PCCR1. . . . . . . . . . . . . . .          Number   000B
PCCR2. . . . . . . . . . . . . . .          Number   000C
PCCR3. . . . . . . . . . . . . . .          Number   000D
PCCR4. . . . . . . . . . . . . . .          Number   000E
PCCR5. . . . . . . . . . . . . . .          Number   000F
PCDATB . . . . . . . . . . . . . .          Number   0029
PCDFLT . . . . . . . . . . . . . .          Number   0004
PCFAIL . . . . . . . . . . . . . .          Number   0026
PCFLAG . . . . . . . . . . . . . .          Number   0014
PCID . . . . . . . . . . . . . . .          Number   0000
PCLEN. . . . . . . . . . . . . . .          Number   0037
PCMASK . . . . . . . . . . . . . .          Number   0015
PCMODE . . . . . . . . . . . . . .          Number   0027
PCMODM . . . . . . . . . . . . . .          Number   0006
PCPRT. . . . . . . . . . . . . . .          V BYTE   0000    CODE     External
PCPRTY . . . . . . . . . . . . . .          Number   002A
PCRADR . . . . . . . . . . . . . .          Number   0033
PCRATE . . . . . . . . . . . . . .          Number   000A
PCRCVA . . . . . . . . . . . . . .          Number   0018
```

```
PCRCVB . . . . . . . . . . . . . .          Number   002B
PCRCVF . . . . . . . . . . . . . .          Number   0017
PCRXOF . . . . . . . . . . . . . .          Number   002F
PCS7201. . . . . . . . . . . . . .          Number   0012
PCSIZE . . . . . . . . . . . . . .          Number   0031
PCSTART. . . . . . . . . . . . . .          L BYTE   0000    CODE
PCSTAT . . . . . . . . . . . . . .          Number   0003
PCSTCA . . . . . . . . . . . . . .          Number   0022
PCSTCF . . . . . . . . . . . . . .          Number   0021
PCSTPB . . . . . . . . . . . . . .          Number   0028
PCTXB. . . . . . . . . . . . . . .          Number   002C
PCTXOF . . . . . . . . . . . . . .          Number   0030
PCXCOM . . . . . . . . . . . . . .          V BYTE   0000    CODE     External
PCXMTA . . . . . . . . . . . . . .          Number   001D
PCXMTF . . . . . . . . . . . . . .          Number   001C
PCXOFF . . . . . . . . . . . . . .          Number   002E
PCXON. . . . . . . . . . . . . . .          Number   002D
PORTA. . . . . . . . . . . . . . .          L WORD   0112    CODE
PORTB. . . . . . . . . . . . . . .          L WORD   010A    CODE
PORTXA . . . . . . . . . . . . . .          L WORD   0122    CODE
PORTXB . . . . . . . . . . . . . .          L WORD   011A    CODE
PRG7201. . . . . . . . . . . . . .          L NEAR   02D6    CODE     Global
PRGCM2 . . . . . . . . . . . . . .          L NEAR   0305    CODE
PRGCM3 . . . . . . . . . . . . . .          L NEAR   0310    CODE
PRGERR . . . . . . . . . . . . . .          L NEAR   0318    CODE
PRTYMAX. . . . . . . . . . . . . .          Number   0003
PUTCHAR. . . . . . . . . . . . . .          L NEAR   06F7    CODE     Global
RBCOUNT. . . . . . . . . . . . . .          Number   0000
RBDFLT . . . . . . . . . . . . . .          Number   0020
RBHEAD . . . . . . . . . . . . . .          Number   000C
RBIN . . . . . . . . . . . . . . .          Number   0004
RBLEN. . . . . . . . . . . . . . .          Number   0010
RBMAX. . . . . . . . . . . . . . .          Number   0002
RBOUT. . . . . . . . . . . . . . .          Number   0006
RBSTART. . . . . . . . . . . . . .          L BYTE   0000    CODE
RBTAIL . . . . . . . . . . . . . .          Number   000E
RBXOFF . . . . . . . . . . . . . .          Number   0008
RBXON. . . . . . . . . . . . . . .          Number   000A
RCVB1. . . . . . . . . . . . . . .          L NEAR   0467    CODE
RCVB2. . . . . . . . . . . . . . .          L NEAR   047B    CODE
RCVBAUD. . . . . . . . . . . . . .          L NEAR   0450    CODE
RCVBPRT. . . . . . . . . . . . . .          L NEAR   0483    CODE
RCVBRK . . . . . . . . . . . . . .          Alias    BIT0
RCVBX. . . . . . . . . . . . . . .          L NEAR   0482    CODE
RCVCANCEL. . . . . . . . . . . . .          L NEAR   07DF    CODE     Global
RCVCOMMON. . . . . . . . . . . . .          L NEAR   080A    CODE
RCVDIS . . . . . . . . . . . . . .          L NEAR   05FB    CODE     Global
RCVENA . . . . . . . . . . . . . .          L NEAR   05EA    CODE     Global
RCVINT . . . . . . . . . . . . . .          L NEAR   07BC    CODE     Global
RCVOFF . . . . . . . . . . . . . .          Alias    BIT5
RCVX . . . . . . . . . . . . . . .          L NEAR   01F8    CODE
RCVXOF . . . . . . . . . . . . . .          Alias    BIT6
RDMODEM. . . . . . . . . . . . . .          L NEAR   0748    CODE     Global
```

```
 The Microsoft MACRO Assembler                02-20-84      PAGE      Symbols-6
COMBIOS


RDNVMCOM . . . . . . . . . . . . . .        L NEAR   0000      CODE      External
RDSETUP. . . . . . . . . . . . . . .        L NEAR   05CA      CODE      Global
REC1 . . . . . . . . . . . . . . . .        L NEAR   0167      CODE
REC3 . . . . . . . . . . . . . . . .        L NEAR   0172      CODE
REC4 . . . . . . . . . . . . . . . .        L NEAR   0198      CODE
REC5 . . . . . . . . . . . . . . . .        L NEAR   019A      CODE
REC6 . . . . . . . . . . . . . . . .        L NEAR   01A0      CODE
REC7 . . . . . . . . . . . . . . . .        L NEAR   01B5      CODE
REC8 . . . . . . . . . . . . . . . .        L NEAR   01B7      CODE
REC9 . . . . . . . . . . . . . . . .        L NEAR   01D7      CODE
RECEIVER . . . . . . . . . . . . . .        L NEAR   014B      CODE
RECX . . . . . . . . . . . . . . . .        L NEAR   01E3      CODE
RESET7201. . . . . . . . . . . . . .        L NEAR   0904      CODE
RI . . . . . . . . . . . . . . . . .        Alias    BIT0
RLSD . . . . . . . . . . . . . . . .        Alias    BIT4
RST7201. . . . . . . . . . . . . . .        Number   0018
RTS. . . . . . . . . . . . . . . . .        Alias    BIT3
RUPT7201 . . . . . . . . . . . . . .        L NEAR   0000      CODE      Global
RUPTCOMMON . . . . . . . . . . . . .        L NEAR   0012      CODE
SAVE_SP. . . . . . . . . . . . . . .        L WORD   00A6      CODE
SAVE_SS. . . . . . . . . . . . . . .        L WORD   00A8      CODE
SETMODEM . . . . . . . . . . . . . .        L NEAR   0775      CODE      Global
SPDI . . . . . . . . . . . . . . . .        Alias    BIT1
SPREC. . . . . . . . . . . . . . . .        L NEAR   0253      CODE
SPREC1 . . . . . . . . . . . . . . .        L NEAR   0262      CODE
SPRECEIVE. . . . . . . . . . . . . .        L NEAR   024B      CODE
SPRECX . . . . . . . . . . . . . . .        L NEAR   0268      CODE
SPSEL. . . . . . . . . . . . . . . .        Alias    BIT0
SR1. . . . . . . . . . . . . . . . .        Number   0001
SR2. . . . . . . . . . . . . . . . .        Number   0002
SR3. . . . . . . . . . . . . . . . .        Number   0003
SRLSD. . . . . . . . . . . . . . . .        Alias    BIT1
SRTS . . . . . . . . . . . . . . . .        Alias    BIT1
STATCHG. . . . . . . . . . . . . . .        L NEAR   08A2      CODE      Global
STCANCEL . . . . . . . . . . . . . .        L NEAR   08C5      CODE      Global
STOPBIT. . . . . . . . . . . . . . .        L NEAR   031B      CODE
STPBMAX. . . . . . . . . . . . . . .        Number   0003
SX7201 . . . . . . . . . . . . . . .        Number   0010
TXBUFFER . . . . . . . . . . . . . .        L NEAR   0222      CODE
TXBUFX . . . . . . . . . . . . . . .        L NEAR   0237      CODE
TXXON. . . . . . . . . . . . . . . .        L NEAR   066A      CODE
TXXONS . . . . . . . . . . . . . . .        L NEAR   0694      CODE
TXXONX . . . . . . . . . . . . . . .        L NEAR   06A6      CODE
VECT7201 . . . . . . . . . . . . . .        L NEAR   0875      CODE      Global
VECTAB . . . . . . . . . . . . . . .        L WORD   0892      CODE
WRITE7201. . . . . . . . . . . . . .        L NEAR   0269      CODE
XA7201 . . . . . . . . . . . . . . .        Number   0028
XABAUD . . . . . . . . . . . . . . .        Number   0021
XAEXIT . . . . . . . . . . . . . . .        L NEAR   027C      CODE
XAMODEM. . . . . . . . . . . . . . .        Number   00FF
XARXDAT. . . . . . . . . . . . . . .        L NEAR   0286      CODE
XARXSPC. . . . . . . . . . . . . . .        L NEAR   0281      CODE
XAS7201. . . . . . . . . . . . . . .        Number   002A
```

```
 The Microsoft MACRO Assembler                02-20-84      PAGE      Symbols-7
COMBIOS


XASTAT . . . . . . . . . . . . . . .        L NEAR   027A      CODE
XATXEMT. . . . . . . . . . . . . . .        L NEAR   0276      CODE
XB7201 . . . . . . . . . . . . . . .        Number   0029
XBAUD1 . . . . . . . . . . . . . . .        V BYTE   0000      CODE      External
XBAUD2 . . . . . . . . . . . . . . .        V BYTE   0000      CODE      External
XBMODEM. . . . . . . . . . . . . . .        Number   0002
XBRXDAT. . . . . . . . . . . . . . .        L NEAR   0148      CODE
XBRXSPC. . . . . . . . . . . . . . .        L NEAR   0243      CODE
XBS7201. . . . . . . . . . . . . . .        Number   002B
XBSTAT . . . . . . . . . . . . . . .        L NEAR   0290      CODE
XBTXEMT. . . . . . . . . . . . . . .        L NEAR   021A      CODE
XINT7201 . . . . . . . . . . . . . .        L WORD   00FA      CODE
XMTBO. . . . . . . . . . . . . . . .        L NEAR   04B8      CODE
XMTBAUD. . . . . . . . . . . . . . .        L NEAR   04A1      CODE
XMTBRK . . . . . . . . . . . . . . .        Alias    BIT3
XMTBX. . . . . . . . . . . . . . . .        L NEAR   04E0      CODE
XMTCANCEL. . . . . . . . . . . . . .        L NEAR   0819      CODE      Global
XMTMTY . . . . . . . . . . . . . . .        L NEAR   07E9      CODE      Global
XMTXOF . . . . . . . . . . . . . . .        Alias    BIT2
XMTXON . . . . . . . . . . . . . . .        Alias    BIT1
XONOF1 . . . . . . . . . . . . . . .        L NEAR   0508      CODE
XONOFF . . . . . . . . . . . . . . .        L NEAR   04FD      CODE
XONOFX . . . . . . . . . . . . . . .        L NEAR   0513      CODE
XOPTION. . . . . . . . . . . . . . .        V BYTE   0000      CODE      External
XRUPT7201. . . . . . . . . . . . . .        L NEAR   000A      CODE      Global
XSTAT1 . . . . . . . . . . . . . . .        L NEAR   02BC      CODE
XSTATCHG . . . . . . . . . . . . . .        L NEAR   0298      CODE
XSTATX . . . . . . . . . . . . . . .        L NEAR   02C0      CODE
?N . . . . . . . . . . . . . . . . .        Number   0010

Warning Severe
Errors  Errors
0       0
```

```
COMBIOS

Symbol Cross Reference                  (# is definition)       Cref-1

?N . . . . . . . . . . . . . .   78#    78    78#    78    78#    78    78#    78    78#    78    78#    78    78#    78
                                 78#    78    78#    78    78#    78    78#    78    78#    78    78#    78    78#    78
                                 78#    78    78     78#   78     78#   78     78#   78     78#   78     78#   78     78#
                                 78     78#   78     78#   78     78#   78     78#   78     78#   78     78#   78     78#
                                 78     78#   78     78#   78     78#   78     78#   78     78#   78     78#   78     78#
                                 78     78#   78     78#   78     78#   78     78#   78     78#   78     78#   78     78#
                                 78     78#   78     78#   78     78#   78     78#   78     78#   78     78#   78     78#
                                 78     78#   78     78#   78#    78    78#    78    78#    78    78#    78    78#    78
                                 78#    78    78#    78    78#    78    78#    78    78#    78

A7201. . . . . . . . . . . . .   78#
ABAUD. . . . . . . . . . . . .   78#
AMODEM . . . . . . . . . . . .   78#
APPXOF . . . . . . . . . . . .   78#  1283  1619   1748   1756
ARXDAT . . . . . . . . . . . .  299    330   375#
ARXSPC . . . . . . . . . . . .  300    331   615#
AS7201 . . . . . . . . . . . .   78#   205  2292
ASCSUB . . . . . . . . . . . .   78#   456
ASTAT. . . . . . . . . . . . .  298    329   728#
ATXEMT . . . . . . . . . . . .  297    328   550#
AUTOXMT. . . . . . . . . . . .  484    520#  1626

B7201. . . . . . . . . . . . .   78#
BAKGRND. . . . . . . . . . . .   78#
BAUDMAX. . . . . . . . . . . .   78#  1193
BBAUD. . . . . . . . . . . . .   78#
BIT0 . . . . . . . . . . . . .   78#    78    78     78    936   1447   1470   1897
BIT1 . . . . . . . . . . . . .   78#    78    78     78     78    621    936
BIT2 . . . . . . . . . . . . .   78#    78    78     78    525    871   1696   1697   1835   1844   1894
BIT3 . . . . . . . . . . . . .   78#    78    78    871
BIT4 . . . . . . . . . . . . .   78#    78    78   1952   1975
BIT5 . . . . . . . . . . . . .   78#    78    78    978   1037   1821   1828
BIT6 . . . . . . . . . . . . .   78#    78    78   1037   1045
BIT7 . . . . . . . . . . . . .   78#    78   736   1045
BMODEM . . . . . . . . . . . .   78#
BRKCOMMON. . . . . . . . . . . 1953   1984#
BRKERR . . . . . . . . . . . .   78#   750
BRKOFF . . . . . . . . . . . .  117   1967#
BRKON. . . . . . . . . . . . .  116   1944#
BRXDAT . . . . . . . . . . . .  295    324   382#
BRXSPC . . . . . . . . . . . .  296    325   607#
BS7201 . . . . . . . . . . . .   78#   167   203
BSTAT. . . . . . . . . . . . .  294    323   720#
BTXEMT . . . . . . . . . . . .  293    322   577#
BUFF1. . . . . . . . . . . . . 1262   1266#
BUFF10 . . . . . . . . . . . . 1272#  1388
BUFFER . . . . . . . . . . . .  834   1257#
BUFFX. . . . . . . . . . . . . 1260   1265  1310#

CALRET . . . . . . . . . . . .  261
CBDATB . . . . . . . . . . . .   78#   895   911   1000   1005
CBLEN. . . . . . . . . . . . .   78#  1418
CBMODE . . . . . . . . . . . .   78#   962   967
CBPORT . . . . . . . . . . . .   78#
CBPRTY . . . . . . . . . . . .   78#   898   900    902    906    915    922
```

```
COMBIOS

Symbol Cross Reference                  (# is definition)       Cref-2

CBRADR . . . . . . . . . . . .   78#  1268
CBRCVB . . . . . . . . . . . .   78#  1072  1079   1084   1100   1109
CBRXOF . . . . . . . . . . . .   78#   826
CBSIZE . . . . . . . . . . . .   78#  1258  1264
CBSTART. . . . . . . . . . . .   78#
CBSTPB . . . . . . . . . . . .   78#   858   863
CBTXB. . . . . . . . . . . . .   78#  1101  1103   1106   1137   1144   1147
CBTXOF . . . . . . . . . . . .   78#   830
CBXOFF . . . . . . . . . . . .   78#  1237
CBXON. . . . . . . . . . . . .   78#  1233
CGROUP . . . . . . . . . . . .   73     75    75     75     75
CLK16X . . . . . . . . . . . .   78#  1339
CODE . . . . . . . . . . . . .   73     74#   74   2373
COMM_STACK . . . . . . . . . .  186    273#
CR1. . . . . . . . . . . . . .   78#  1358
CR17201. . . . . . . . . . . .   78#  2306
CR1TXBE. . . . . . . . . . . .   78#  2074  2099
CR2. . . . . . . . . . . . . .   78#  2324
CR27201. . . . . . . . . . . .   78#  2326
CR3. . . . . . . . . . . . . .   78#   981  1048   1475   1823   1838
CR4. . . . . . . . . . . . . .   78#   942
CR5. . . . . . . . . . . . . .   78#  1040  1349   1986
CR7201 . . . . . . . . . . . .  209    212   220    248#
CTS. . . . . . . . . . . . . .   78#

DATABIT. . . . . . . . . . . .  822    999#
DATABX . . . . . . . . . . . . 1002   1051#
DATMAX . . . . . . . . . . . .   78#  1003
DC1. . . . . . . . . . . . . .   78#
DC3. . . . . . . . . . . . . .   78#
DEVCOM . . . . . . . . . . . .   78#
DEVMAX . . . . . . . . . . . .   78#  2303  2354
DEVPRT . . . . . . . . . . . .   78#  1082  1149   1198
DEVXCOM. . . . . . . . . . . .   78#
DFLT7201 . . . . . . . . . . .  102   1327# 2307
DFLTBUF. . . . . . . . . . . .  103   1330  1379#
DFLTCOM. . . . . . . . . . . .   84#
DFLTPRT. . . . . . . . . . . .   86#
DFLTXCOM . . . . . . . . . . .   85#
DSR. . . . . . . . . . . . . .   78#
DTR. . . . . . . . . . . . . .   78#

ENDTXBE. . . . . . . . . . . .   78#   589   690
EOI7201. . . . . . . . . . . .   78#   231
ERR7201. . . . . . . . . . . .   78#   633   702

FIXTAB . . . . . . . . . . . . 2144   2166#
FRMERR . . . . . . . . . . . .   78#   635   749

GETBAUD. . . . . . . . . . . . 1075   1140  1190#
GETCHAR. . . . . . . . . . . .  109   1571# 1574
GETPCB . . . . . . . . . . . .  815   1328  1380   1413   1442   1465   1496   1529   1690   1815   1879   1918   1945   1968
                               2010   2038  2064   2095   2244   2271   2304   2353#

HACK7201 . . . . . . . . . . .  123   2132#
```

```
HACKA. . . . . . . . . . . . . . .  346#  2161
HACKB. . . . . . . . . . . . . . .  350#  2162
HACKTAB. . . . . . . . . . . . . . 2143   2180#
HACKXA . . . . . . . . . . . . . .  358#  2164
HACKXB . . . . . . . . . . . . . .  354#  2163
HVECTOR. . . . . . . . . . . . . .   94#   347    351    355    359   2154   2156

INCHA1 . . . . . . . . . . . . . . 1546   1549#
INCHA2 . . . . . . . . . . . . . . 1548   1551#
INCHAR . . . . . . . . . . . . . .  108   1528#  1572
INCHAX . . . . . . . . . . . . . . 1540   1554#
INITCOM. . . . . . . . . . . . . .   98   2291#
INSTAT . . . . . . . . . . . . . .  107   1494#
INSTX. . . . . . . . . . . . . . . 1497   1501   1503#
INT7201. . . . . . . . . . . . . .  166    292#  2167   2168   2169   2170   2213   2214   2215   2216
INTEXIT. . . . . . . . . . . . . .  213    221    234#

L10_10$. . . . . . . . . . . . . . 1329   1361#
L11_1$ . . . . . . . . . . . . . . 1381   1384#
L12_10$. . . . . . . . . . . . . . 1414   1428#
L13_1$ . . . . . . . . . . . . . . 1443   1445#
L14_1$ . . . . . . . . . . . . . . 1466   1468#
L15_1$ . . . . . . . . . . . . . . 1530   1533#
L16_10$. . . . . . . . . . . . . . 1701   1704#
L16_5$ . . . . . . . . . . . . . . 1691   1702#
L17_1$ . . . . . . . . . . . . . . 1747   1750#
L17_2$ . . . . . . . . . . . . . . 1749   1751   1759#
L18_1$ . . . . . . . . . . . . . . 1816   1818#
L18_5$ . . . . . . . . . . . . . . 1834#  1836
L19_1$ . . . . . . . . . . . . . . 1845   1848#
L1_1$. . . . . . . . . . . . . . .  204    215#
L20_10$. . . . . . . . . . . . . . 1919   1922   1929#
L21_1$ . . . . . . . . . . . . . . 1946   1948#
L22_1$ . . . . . . . . . . . . . . 1969   1972#
L23_10$. . . . . . . . . . . . . . 2011   2022#
L24_1$ . . . . . . . . . . . . . . 2039   2041#
L25_1$ . . . . . . . . . . . . . . 2065   2079#
L26_1$ . . . . . . . . . . . . . . 2096   2103#
L27_1$ . . . . . . . . . . . . . . 2149#  2150
L28_1$ . . . . . . . . . . . . . . 2206#  2208
L29_10$. . . . . . . . . . . . . . 2245   2258#
L2_10$ . . . . . . . . . . . . . .  526    537#
L30_1$ . . . . . . . . . . . . . . 2272   2274#
L31_1$ . . . . . . . . . . . . . . 2297   2302#
L31_2$ . . . . . . . . . . . . . . 2304#  2311
L31_3$ . . . . . . . . . . . . . . 2305   2310#
L33_1$ . . . . . . . . . . . . . . 2355   2357   2364#
L33_2$ . . . . . . . . . . . . . . 2363   2365#
L34_1$ . . . . . . . . . . . . . . 1199   1201#
L34_10$. . . . . . . . . . . . . . 1192   1194   1207#
L35_1$ . . . . . . . . . . . . . . 1883   1888#
L35_10$. . . . . . . . . . . . . . 1880   1896   1898#
L35_2$ . . . . . . . . . . . . . . 1887   1889#
L3_10$ . . . . . . . . . . . . . .  765    769    771#
L4_1$. . . . . . . . . . . . . . .  862    866#
```

```
L4_10$ . . . . . . . . . . . . . .  860    878#
L5_1$. . . . . . . . . . . . . . .  896    905#   914
L5_13$ . . . . . . . . . . . . . .  929    932    934#
L5_15$ . . . . . . . . . . . . . .  908    939#
L5_2$. . . . . . . . . . . . . . .  899    901    904    910    912    919#
L5_3$. . . . . . . . . . . . . . .  921    925#
L6_1$. . . . . . . . . . . . . . .  966    970#
L7_1$. . . . . . . . . . . . . . . 1013   1017   1021   1025   1031   1033#
L7_10$ . . . . . . . . . . . . . . 1004   1008#
L8_10$ . . . . . . . . . . . . . . 1107   1112#
L8_5$. . . . . . . . . . . . . . . 1102   1106#
L9_1$. . . . . . . . . . . . . . . 1150   1153#
L9_4$. . . . . . . . . . . . . . . 1152   1162#

MODE . . . . . . . . . . . . . . .  821    961#
MODE1. . . . . . . . . . . . . . .  974    976#
MODEMAX. . . . . . . . . . . . . .   78#   965
MODEX. . . . . . . . . . . . . . .  964    984#

NXTINT . . . . . . . . . . . . . .  193#
NXTINT1. . . . . . . . . . . . . .  201    223#

OURTAB . . . . . . . . . . . . . . 2199   2218#
OUTCHAR. . . . . . . . . . . . . .  111   1738#  1785
OUTCHX . . . . . . . . . . . . . . 1744   1767#
OUTN2. . . . . . . . . . . . . . . 1822   1831   1841#  1847
OUTNOW . . . . . . . . . . . . . .  113   1814#
OUTSTAT. . . . . . . . . . . . . .  110   1689#  1742
OVRERR . . . . . . . . . . . . . .   78#   635    749

PARERR . . . . . . . . . . . . . .   78#   635    749
PARITY . . . . . . . . . . . . . .  820    894#
PC7201 . . . . . . . . . . . . . .   78#   414   1763
PCBAUD . . . . . . . . . . . . . .   78#  1091   1163
PCBTAB . . . . . . . . . . . . . . 2361   2388#
PCCOM. . . . . . . . . . . . . . .   80#   207    376    551    616    729   2369
PCCR1. . . . . . . . . . . . . . .   78#   253   1357   2073   2075   2098   2100   2306
PCCR2. . . . . . . . . . . . . . .   78#
PCCR3. . . . . . . . . . . . . . .   78#   259    977    980   1044   1047   1446   1469   1474   1821   1827   1837
PCCR4. . . . . . . . . . . . . . .   78#   250    870    873    935    938    941   1339
PCCR5. . . . . . . . . . . . . . .   78#   256   1036   1039   1346   1348   1951   1974
PCDATB . . . . . . . . . . . . . .   78#   913   1009
PCDFLT . . . . . . . . . . . . . .   78#  1338
PCFAIL . . . . . . . . . . . . . .   78#   208    211    219
PCFLAG . . . . . . . . . . . . . .   78#   423    427    479    486    741    743    756   1283   1344   1448   1471   1610   1613
                                   1618   1619   1621   1623   1631   1692   1748   1756   1757   1889   1949   1972   1985

PCID . . . . . . . . . . . . . . .   78#
PCLEN. . . . . . . . . . . . . . .   78#  1386
PCMASK . . . . . . . . . . . . . .   78#   434    528    529   1035   1760   1850
PCMODE . . . . . . . . . . . . . .   78#   971   1415
PCMODM . . . . . . . . . . . . . .   78#  1881   1920
PCPRT. . . . . . . . . . . . . . .   82#   210    383    578    608    721   2370
PCPRTY . . . . . . . . . . . . . .   78#   926
PCRADR . . . . . . . . . . . . . .   78#   437   1285   1286   1498   1534
```

```
 COMBIOS

 Symbol Cross Reference                    (# is definition)       Cref-5

PCRATE . . . . . . . . . . . .   78#   1086    1090    1154    1162
PCRCVA . . . . . . . . . . . .   78#    492    2016    2017
PCRCVB . . . . . . . . . . . .   78#   1085    1151
PCRCVF . . . . . . . . . . . .   78#    489     491     493     495    2018    2040
PCRXOF . . . . . . . . . . . .   78#    417     829
PCS7201. . . . . . . . . . . .   78#    249     521     588     619     626     672     731    1354    1694    1820    1892
PCSIZE . . . . . . . . . . . .   78#   1284
PCSTART. . . . . . . . . . . .   78#
PCSTAT . . . . . . . . . . . .   78#   2296    2298    2362
PCSTCA . . . . . . . . . . . .   78#    767    2250    2251
PCSTCF . . . . . . . . . . . .   78#    764     766     768     770    2252    2273
PCSTPB . . . . . . . . . . . .   78#    867
PCTXB. . . . . . . . . . . . .   78#   1148
PCTXOF . . . . . . . . . . . .   78#    472     833    1607
PCXCOM . . . . . . . . . . . .   81#    218     390     564     612     725    2296    2298    2371
PCXMTA . . . . . . . . . . . .   78#    584    2069    2070
PCXMTF . . . . . . . . . . . .   78#    581     583     585     587    2071    2097
PCXOFF . . . . . . . . . . . .   78#    425     483    1240    1750
PCXON. . . . . . . . . . . . .   78#    421    1236    1625    1746
PORTA. . . . . . . . . . . . .  327#   2219
PORTB. . . . . . . . . . . . .  321#   2220
PORTXA . . . . . . . . . . . .  339#   2222
PORTXB . . . . . . . . . . . .  333#   2221
PRG7201. . . . . . . . . . . .  101     811#   1341
PRGCM2 . . . . . . . . . . . .  828     830#
PRGCM3 . . . . . . . . . . . .  832     834#
PRGERR . . . . . . . . . . . .  816     839#
PRTYMAX. . . . . . . . . . . .   78#    920
PUTCHAR. . . . . . . . . . . .  112    1783#   1788

RBCOUNT. . . . . . . . . . . .   78#    440     460     474    1302    1499    1538    1542    1615
RBDFLT . . . . . . . . . . . .   78#   1261    1387
RBHEAD . . . . . . . . . . . .   78#    448     465    1293    1547
RBIN . . . . . . . . . . . . .   78#    442     471    1291
RBLEN. . . . . . . . . . . . .   78#   1261    1287    1290    1387
RBMAX. . . . . . . . . . . . .   78#    441     475    1300
RBOUT. . . . . . . . . . . . .   78#   1292    1543    1551
RBSTART. . . . . . . . . . . .   78#
RBTAIL . . . . . . . . . . . .   78#    450     463    1297    1545
RBXOFF . . . . . . . . . . . .   78#    477    1304
RBXON. . . . . . . . . . . . .   78#   1307    1616
RCVB1. . . . . . . . . . . . . 1077    1081#
RCVB2. . . . . . . . . . . . . 1089#
RCVBAUD. . . . . . . . . . . .  823    1071#
RCVBPRT. . . . . . . . . . . . 1083    1099#
RCVBRK . . . . . . . . . . . .   78#    741     743     756
RCVBX. . . . . . . . . . . . . 1074    1093#
RCVCANCEL. . . . . . . . . . .  119    1333    2037#
RCVCOMMON. . . . . . . . . . . 1449    1473#
RCVDIS . . . . . . . . . . . .  106    1464#
RCVENA . . . . . . . . . . . .  105    1351    1441#
RCVINT . . . . . . . . . . . .  118    2005#
RCVOFF . . . . . . . . . . . .   78#   1448    1471
RCVX . . . . . . . . . . . . .  490     494     496#
RCVXOF . . . . . . . . . . . .   78#    423     427    1283    1693
```

```
 COMBIOS

 Symbol Cross Reference                    (# is definition)       Cref-6

RDMODEM. . . . . . . . . . . .  114    1877#
RDNVMCOM . . . . . . . . . . .   92#   1335
RDSETUP. . . . . . . . . . . .  104    1408#
REC1 . . . . . . . . . . . . .  422     425#
REC3 . . . . . . . . . . . . .  418     426     433#    644
REC4 . . . . . . . . . . . . .  449     452#
REC5 . . . . . . . . . . . . .  451     455#
REC6 . . . . . . . . . . . . .  443     458#
REC7 . . . . . . . . . . . . .  464     467#
REC8 . . . . . . . . . . . . .  466     470#
REC9 . . . . . . . . . . . . .  476     481#
RECEIVER . . . . . . . . . . .  377     384     413#
RECX . . . . . . . . . . . . .  424     428     457     473     478     480     485     487#
RESET7201. . . . . . . . . . .  206     217    2293    2300    2319#
RI . . . . . . . . . . . . . .   78#
RLSD . . . . . . . . . . . . .   78#
RST7201. . . . . . . . . . . .   78#   2322
RTS. . . . . . . . . . . . . .   78#
RUPT7201 . . . . . . . . . . .   96     163#
RUPTCOMMON . . . . . . . . . .  168     178#

SAVE_SP. . . . . . . . . . . .  183     241     266#
SAVE_SS. . . . . . . . . . . .  184     240     267#
SEQ. . . . . . . . . . . . . .   78      78      78      78      78      78      78      78      78      78      78      78      78      78
                                 78      78      78      78      78      78      78      78      78      78      78      78      78      78
                                 78      78      78      78      78      78      78      78      78      78      78      78      78      78
                                 78      78      78      78      78      78      78      78      78      78      78      78      78      78
                                 78      78

SETMODEM . . . . . . . . . . .  115    1916#   2309
SPDI . . . . . . . . . . . . .   78#
SPREC. . . . . . . . . . . . .  609     613     625#
SPREC1 . . . . . . . . . . . .  637#    752
SPRECEIVE. . . . . . . . . . .  618#
SPRECX . . . . . . . . . . . .  622     646#
SPSEL. . . . . . . . . . . . .   78#
SR1. . . . . . . . . . . . . .   78#    628     744
SR2. . . . . . . . . . . . . .   78#
SR3. . . . . . . . . . . . . .   78#
SRLSD. . . . . . . . . . . . .   78#
SRTS . . . . . . . . . . . . .   78#
STATCHG. . . . . . . . . . . .  126    2239#
STCANCEL . . . . . . . . . . .  127    2270#
STOPBIT. . . . . . . . . . . .  819     857#
STPBMAX. . . . . . . . . . . .   78#    861
SX7201 . . . . . . . . . . . .   78#    694     734    1353

TABLE. . . . . . . . . . . . .   78      78      78
TXBUFFER . . . . . . . . . . .  552     565     579#
TXBUFX . . . . . . . . . . . .  582     586     588#
TXXON. . . . . . . . . . . . . 1535    1606#
TXXON5 . . . . . . . . . . . . 1611    1622#
TXXONX . . . . . . . . . . . . 1608    1614    1617    1620    1624    1627    1632#

VECT7201 . . . . . . . . . . .  124    2193#
```

```
 COMBIOS

 Symbol Cross Reference                  (# is definition)       Cref-7

VECTAB   . . . . . . . . . . . . . .   2198    2212#

WRITE7201. . . . . . . . . . . . . .    252     255     258     262     668#    943     982    1041    1049    1350    1359    1476    1839    1987
                                       2077    2102

XA7201 . . . . . . . . . . . . . . .     78#
XABAUD . . . . . . . . . . . . . . .     78#
XAEXIT . . . . . . . . . . . . . . .    691     696#    703
XAMODEM. . . . . . . . . . . . . . .     78#
XARXDAT. . . . . . . . . . . . . . .    309     342     705#
XARXSPC. . . . . . . . . . . . . . .    310     343     701#
XAS7201. . . . . . . . . . . . . . .     78#    216    2299
XASTAT . . . . . . . . . . . . . . .    308     341     693#
XATXEMT. . . . . . . . . . . . . . .    307     340     689#
XB7201 . . . . . . . . . . . . . . .     78#
XBAUD1 . . . . . . . . . . . . . . .     88#   1197    1203
XBAUD2 . . . . . . . . . . . . . . .     89#   1200
XBMODEM. . . . . . . . . . . . . . .     78#
XBRXDAT. . . . . . . . . . . . . . .    305     336     389#
XBRXSPC. . . . . . . . . . . . . . .    306     337     611#
XBS7201. . . . . . . . . . . . . . .     78#    174
XBSTAT . . . . . . . . . . . . . . .    304     335     724#
XBTXEMT. . . . . . . . . . . . . . .    303     334     563#
XINT7201 . . . . . . . . . . . . . .    173     302#
XMTBO. . . . . . . . . . . . . . . .   1142    1146#
XMTBAUD. . . . . . . . . . . . . . .    824    1136#
XMTBRK . . . . . . . . . . . . . . .     78#   1623    1950    1973
XMTBX. . . . . . . . . . . . . . . .   1139    1165#
XMTCANCEL. . . . . . . . . . . . . .    121    1334    2094#
XMTMTY . . . . . . . . . . . . . . .    120    2059#
XMTXOF . . . . . . . . . . . . . . .     78#    479     486    1283    1613    1618    1631
XMTXON . . . . . . . . . . . . . . .     78#   1283    1610    1621    1631    1757
XONOF1 . . . . . . . . . . . . . . .   1235    1237#
XONOFF . . . . . . . . . . . . . . .    825    1232#
XONOFX . . . . . . . . . . . . . . .   1239    1241#
XOPTION. . . . . . . . . . . . . . .     91#   2295
XRUPT7201. . . . . . . . . . . . . .     97     170#
XSTAT1 . . . . . . . . . . . . . . .    737     755#
XSTATCHG . . . . . . . . . . . . . .    722     726     730#
XSTATX . . . . . . . . . . . . . . .    742     763#
```

```
 The Microsoft MACRO Assembler               02-20-84     PAGE    1-1
End of BIOS

                                             PAGE    60,132
                                             TITLE   End of BIOS
                                             NAME    END

                                       ;
                                       ;     COMPANY CONFIDENTIAL
                                       ;     Copyright (C) 1983 Digital Equipment Corporation
                                       ;     All rights reserved.
                                       ;

                                       cgroup group code
       0000                            code segment byte public 'code'
                                       assume cs:cgroup

                                       public  endbios
                                       public  packet_adr,xfrpkt,i88pkt
                                       public  buffer,ttrack,tformat
                                       public  packet,fnccod,status,drvno,secno,trackn,nsect,dmaadr,exstat
                                       public  packet_base
                                       ;
                                       ;Merely defines the end of the BIOS
                                       ;

       = 0400                          packet_base equu this byte + 400h
       = 0400                          ttrack  equ     this byte + 400h
       0000     04 [                           db 4 dup (0ffh)         ;Z80 Track position
                         FF
                             ]

       = 0404                          tformat equ     this byte + 400h
       0004     04 [                           db 4 dup (0ffh)         ;Z80 disk type bytes
                         FF
                             ]


       0008     04 [                           db 4 dup (0cch)         ;Filler
                         CC
                             ]


       = 040C                          packet_adr equ this byte + 400h         ;Base address of pointers/buffers

       = 040C                          xfrpkt  equ     this word + 400h        ;pointer to "packet"
       000C  0000                              dw      0
       = 040E                          i88pkt  equ     this word + 400h        ;returned pointer from Z80
       000E  0000                              dw      0

                                       ;The packet for the z80.

       = 0410                          packet  equ     this byte + 400h        ;Z80 command packet
       = 0410                          fnccod  equ     this byte + 400h
       0010  00                                db      0                       ;Function code
       = 0411                          status  equ     this byte + 400h
       0011  00                                db      0                       ;retnd status,
       = 0412                          drvno   equ     this byte + 400h
 The Microsoft MACRO Assembler              02-20-84      PAGE     1-2
End of BIOS

        0012  00                                     db      0                      ;drive and side
        = 0413                              secno    equ     this byte + 400h
        0013  00                                     db      0
        = 0414                              trackn   equ     this byte + 400h
        0014  00                                     db      0                      ;Track #
        = 0415                              nsect    equ     this byte + 400h
        0015  00                                     db      0                      ;# of sectors to process
        = 0416                              dmaadr   equ     this word + 400h
        0016  0000                                   dw      0                      ;DMA address (abs)
        = 0418                              exstat   equ     this byte + 400h
        0018  00                                     db      0                      ;extended status

        0019  0200 [                        buffer   db 200h dup (0CCh)      ;Z80 buffer area
                         CC
                            ]


        0219                                endbios label byte

        0219                                code ends

                                                     end
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE     Symbols-1
End of BIOS

Segments and groups:

                N a m e                 Size    align   combine class

CGROUP . . . . . . . . . . . . . .      GROUP
  CODE . . . . . . . . . . . . . . .    0219    BYTE    PUBLIC  'CODE'

Symbols:

                N a m e                 Type    Value   Attr

BUFFER . . . . . . . . . . . . . .      L BYTE  0019    CODE    Global  Length =0200
DMAADR . . . . . . . . . . . . . .      E WORD  0416    CODE    Global
DRVNO. . . . . . . . . . . . . . .      E BYTE  0412    CODE    Global
ENDBIOS. . . . . . . . . . . . . .      L BYTE  0219    CODE    Global
EXSTAT . . . . . . . . . . . . . .      E BYTE  0418    CODE    Global
FNCCOD . . . . . . . . . . . . . .      E BYTE  0410    CODE    Global
I88PKT . . . . . . . . . . . . . .      E WORD  040E    CODE    Global
NSECT. . . . . . . . . . . . . . .      E BYTE  0415    CODE    Global
PACKET . . . . . . . . . . . . . .      E BYTE  0410    CODE    Global
PACKET_ADR . . . . . . . . . . . .      E BYTE  040C    CODE    Global
PACKET_BASE. . . . . . . . . . . .      E BYTE  0400    CODE    Global
SECNO. . . . . . . . . . . . . . .      E BYTE  0413    CODE    Global
STATUS . . . . . . . . . . . . . .      E BYTE  0411    CODE    Global
TFORMAT. . . . . . . . . . . . . .      E BYTE  0404    CODE    Global
TRACKN . . . . . . . . . . . . . .      E BYTE  0414    CODE    Global
TTRACK . . . . . . . . . . . . . .      E BYTE  0400    CODE    Global
XFRPKT . . . . . . . . . . . . . .      E WORD  040C    CODE    Global

Warning Severe
Errors  Errors
0       0
```

```
 End of BIOS

 Symbol Cross Reference                       (# is definition)       Cref-1

BUFFER . . . . . . . . . . . . . .      17      68#

CGROUP . . . . . . . . . . . . . .      11      13
CODE . . . . . . . . . . . . . . .      11      12#     12      75

DMAADR . . . . . . . . . . . . . .      18      63#
DRVNO. . . . . . . . . . . . . . .      18      55#

ENDBIOS. . . . . . . . . . . . . .      15      73#
EXSTAT . . . . . . . . . . . . . .      18      65#

FNCCOD . . . . . . . . . . . . . .      18      51#

I88PKT . . . . . . . . . . . . . .      16      45#

NSECT. . . . . . . . . . . . . . .      18      61#

PACKET . . . . . . . . . . . . . .      18      50#
PACKET_ADR . . . . . . . . . . . .      16      41#
PACKET_BASE. . . . . . . . . . . .      19      24#

SECNO. . . . . . . . . . . . . . .      18      57#
STATUS . . . . . . . . . . . . . .      18      53#

TFORMAT. . . . . . . . . . . . . .      17      30#
TRACKN . . . . . . . . . . . . . .      18      59#
TTRACK . . . . . . . . . . . . . .      17      25#

XFRPKT . . . . . . . . . . . . . .      16      43#
```

```
 The Microsoft MACRO Assembler               02-20-84      PAGE     1-1
MSDOS 2.00 BIOS and System initialization

                                                   PAGE    60,132
                                                   TITLE   MSDOS 2.00 BIOS and System initialization
                                                   NAME    BIOSINIT

                                          ;
                                          ;        COMPANY CONFIDENTIAL
                                          ;        Copyright (C) 1983 Digital Equipment Corporation
                                          ;        All rights reserved.
                                          ;

                                          cgroup group code
                                          ;
                                          ;This module sets the proper packet ptr, initializes
                                          ;the bios IO devices, and sets up SYSINIT. After the DOS takes
                                          ;control, this module is overwritten.

          = 0040                          biosseg equ     40h      ;where we load BIOS,
          = 4000                          biossiz equ     16*1024 ;generous BIOS size,

                                          public  biosinit

                                          extrn   devlst:byte
                                          extrn   ioinit:near
                                          extrn   endbios:near
                                          extrn   hd_init:near             ;Read home, etc
                                          extrn   pstr:near
                                          extrn   packet_base:near         ;location of bios data
                                          extrn   packet_adr:near          ;location of new packet ptr
                                          extrn   dos_pointer:word         ;in MSBIOS
                                          extrn   dskdev:word, hdskdev:near, aux2dev:near, xoption:byte
                                          ;
                                          ;SYSINIT externals:
                                          ;
                                          extrn sysinit:far                ;entry point,

                                          extrn current_dos_location:word ;where the DOS
                                                                           ;is loaded,
                                          extrn final_dos_location:word   ;where it will
                                                                           ;reside,
                                          extrn device_list:dword          ;top of device
                                                                           ;chain,
                                          extrn memory_size:word           ;size, paras,

                                          extrn default_drive:byte

                                          extrn buffers:byte

                                          extrn files:byte
```

```
                                      page
0000                                  code segment byte public 'code'
                                      assume cs:cgroup,ds:cgroup
                                      ;
                                      ;
                                      ;       USEFUL EQUATES
                                      ;
                                   C  include defs.ash
=-0001                             C  TRUE    EQU     -1
= 0000                             C  FALSE   EQU     NOT TRUE
                                   C
                                   C
                                   C     ENDIF
                                   C
                                   C
                                   C  ;Version number info
                                   C
= 0002                             C  VER_NO          EQU     2        ; version_number
= 0005                             C  REV_NO          EQU     5        ; rev_number
= 0018                             C  MOD_NO          EQU     24       ; mod_number %% update this equate
                                   C
= 000D                             C  CR              EQU     0Dh      ;carriage return
= 000A                             C  LF              EQU     0Ah      ;line feed
= 0007                             C  BELL            EQU     07H      ;Bell character
= 001B                             C  ESC             EQU     1Bh      ;EScape
= 0013                             C  CTRL_S          EQU     13h
= 0011                             C  CTRL_Q          EQU     11h
= 001A                             C  CTRL_Z          EQU     1Ah
                                   C
= 01E0                             C  DOSBASE         EQU     01E0h    ;MSDOS.SYS base segment
                                   C
=                                  C  IVRELOC         EQU     TRUE     ; vector relocate
                                      ;
= FEF0                                ldrdata equ     0FEF0h
= FF00                                ldrpkt  equ     ldrdata+10h      ;location of command packet
= FEFC                                xfrpkt  equ     0FEFCh           ;where Z80 thinks packet ptr is
= 0010                                datlen  equ     010h             ;length of disk data and packet
= 0025                                pktfcn  equ     025h             ;move packet function code
= 0000                                z80int  equ     00h              ;z80 interrupt
= 0001                                wprsnt  equ     01               ;Winnie present
= 0002                                xcprsnt equ     02               ;Xcomm present
                                      ;
                                      ;
                                      ;Initialize the BIOS, then set the stuff for
                                      ;SYSINIT, and jump to it. All of this code gets
                                      ;overwritten.
                                      ;
                                      ;The booted disk drive is in CX.
                                      ;
0000                                  biosinit:
0000  8C C8                                   mov     ax,cs
0002  8E D0                                   mov     ss,ax
0004  BC 0210 R                               mov     sp,offset stack ;temp stack,
0007  8E D8                                   mov     ds,ax
0009  8E C0                                   mov     es,ax
```

```
000B  51                                      push    cx              ;save booted drive

                                      ;Initialize all the hardware, and the interrupt vectors.

000C  E8 0000 E                               call    ioinit          ;setup interrupt vectors (IO.ASM)

000F  FB                                      sti                     ;Need ints on
0010  33 C0                                   xor     ax,ax
0012  8E C0                                   mov     es,ax           ;set ES: to 0
0014  1E                                      push    ds              ;save data segment
0015  8E D8                                   mov     ds,ax           ;and set it to 0 too
0017  BE FEF0                                 mov     si,offset ldrdata       ;move it from default pkt
001A  BF 0000 E                               mov     di,offset packet_base   ;to bios area
001D  B9 0010                                 mov     cx,offset datlen        ;mov all the data
0020  FC                                      cld                     ;make sure we mov it in the right way
0021  F3/ A4                                  rep     movsb           ;quickmove
0023  1F                                      pop     ds              ;restore mysterious ds:

0024  BB FF00                                 mov     bx,offset ldrpkt        ;build packet for move
0027  26: C6 07 25                            mov     byte ptr es:[bx],pktfcn ;command to move packet
002B  26: C7 47 02 0000 E                     mov     es:word ptr 2[bx],offset packet_adr
0031  26: C7 06 FEFC FF00                     mov     es:xfrpkt,offset ldrpkt

0038  E6 00                                   out     z80int,al       ;call the z80
003A  26: 83 3E FEFC 00               watack: cmp     es:word ptr xfrpkt,0    ;see if any return yet
0040  75 F8                                   jnz     watack          ;wait until he does

                                      ;Print the system signon message

0042  BB 00E1 R                               mov     bx,offset hello         ;print the hello message
0045  E8 0000 E                               call    pstr

                                      ;Do some preliminary hardware configuration (adjust device table, etc.)

0048  80 3E 0000 E 01                         cmp     xoption,wprsnt          ;have winnie?
004D  75 13                                   jne     xopti1
004F  E8 0000 E                               call    hd_init                 ;yes - call hard disk init
0052  0A C0                                   or      al,al                   ;any units
0054  74 0C                                   je      xopti1                  ;jump if none
0056  BB 0000 E                               mov     bx,offset dskdev        ;point at floppy
0059  C7 07 0000 E                            mov     word ptr 0[bx],offset hdskdev
005D  C7 47 02 0040                           mov     word ptr 2[bx],offset biosseg
0062  80 3E 0000 E 02                 xopti1: cmp     xoption,xcprsnt         ;eXtended comm present?
0067  75 0C                                   jne     xopti2                  ;jump if none
0069  BB 0000 E                               mov     bx,offset dskdev        ;point at floppy
006C  C7 07 0000 E                            mov     word ptr 0[bx],offset aux2dev
0070  C7 47 02 0040                           mov     word ptr 2[bx],offset biosseg
0075  B8 ---- E                       xopti2: mov     ax,seg sysinit          ;Setup for SYSINIT
0078  8E D8                                   mov     ds,ax

                                      assume ds:seg sysinit

007A  59                                      pop     cx              ;get booted
007B  FE C1                                   inc     cl              ;drive,
007D  88 0E 0000 E                            mov     default_drive,cl
```

```

0081  8C C8                             mov     ax,cs
0083  05 0400                           add     ax,(biossiz/16)
0086  A3 0000 E                         mov     current_dos_location,ax

0089  B8 0000 E                         mov     ax,offset endbios        ;loc of bios end (above Z80 region)
008C  B1 04                             mov     cl,4
008E  D3 E8                             shr     ax,cl                    ;convert to paragraph
0090  05 0041                           add     ax,41h                   ; + base segment + 1
0093  A3 0000 E                         mov     final_dos_location,ax
0096  2E: A3 0000 E                     mov     cs:dos_pointer,ax        ;FORMAT uses this to find MSDOS

                                ; Determine memory size (faster than SYSINIT)

009A  33 DB                             xor     bx,bx
009C  BA 1000                           mov     dx,1000h                 ;Start at 64K
009F                            tst_mem:
009F  8E C2                             mov     es,dx
00A1  B1 0E                             mov     cl,14                    ;Max # of sections (896K)
00A3  26: 8A 07                         mov     al,es:[bx]               ;get byte
00A6  F6 D0                             not     al                       ;compliment byte
00A8  26: 88 07                         mov     es:[bx],al               ;write it back
00AB  26: 3A 07                         cmp     al,es:[bx]               ;same?
00AE  F6 D0                             not     al                       ;invert it back to original value
00B0  26: 88 07                         mov     es:[bx],al
00B3  75 08                             jne     done_mem                 ;done if no match
00B5  81 C2 1000                        add     dx,1000h                 ;step to next 64K
00B9  FE C9                             dec     cl
00BB  75 E2                             jnz     tst_mem                  ;loop till done
00BD                            done_mem:
00BD  89 16 0000 E                      mov     memory_size,dx   ; DX := # of paragraphs in system
00C1  C6 06 0000 E 04                   mov     buffers,4        ;;A good number (maybe made larger by user)
00C6  C6 06 0000 E 08                   mov     files,8          ;; Edit CONFIG.SYS to change
                                ;
                                ;Setup the pointer to our chain of IO devices,
                                ;and leave default drive and buffers.
                                ;
00CB  8C 0E 0002 E                      mov word ptr device_list+2,cs
00CF  C7 06 0000 E 0000 E               mov word ptr device_list,offset devlst
00D5  BA 010C                           mov     dx,10Ch          ;;; *** Disable MHFU for SYSINIT
00D8  B0 00                             mov     al,0             ;;; MHFU port
00DA  FA                                cli                      ;;; Ints off first
00DB  EE                                out     dx,al            ;;; Now disable
00DC  EA 0000 ---- E                    jmp     sysinit          ;Invoke SYSINIT (ints restore in REINIT)

00E1                            hello   label   byte
                                   rept 0
                                        db      'Version '
                                        db      VER_NO+'0'
                                        db      '.'
                                        db      (REV_NO/10)+'0'
                                        db      (REV_NO MOD 10)+'0'
                                        db      '.'
                                        db      (MOD_NO/100)+'0'
                                        db      (MOD_NO MOD 100)/10+'0'
```

```
                                        db      (MOD_NO MOD 10)+'0'
                                        db      cr,lf
                                   endm
00E1  43 6F 70 79 72 69                 db      'Copyright 1983 Digital Equipment Corporation'
      67 68 74 20 31 39
      38 33 20 44 69 67
      69 74 61 6C 20 45
      71 75 69 70 6D 65
      6E 74 20 43 6F 72
      70 6F 72 61 74 69
      6F 6E
010D  0D 0A 00                          db      cr,lf,0

                                ;
                                ;Stack for booting.
                                ;
0110     80 [                           dw 128 dup (?)
                ????
                      ]

= 0210                          stack equ $

0210                            code ends

                                        end
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE     Symbols-1
MSDOS 2.00 BIOS and System initialization


Segments and groups:

                N a m e                     Size    align    combine class

CGROUP . . . . . . . . . . . . . .          GROUP
  CODE . . . . . . . . . . . . . . .        0210    BYTE     PUBLIC  'CODE'

Symbols:

                N a m e                     Type    Value    Attr

AUX2DEV. . . . . . . . . . . . . .          L NEAR  0000             External
BELL . . . . . . . . . . . . . . .          Number  0007
BIOSINIT . . . . . . . . . . . . .          L NEAR  0000     CODE    Global
BIOSSEG. . . . . . . . . . . . . .          Number  0040
BIOSSIZ. . . . . . . . . . . . . .          Number  4000
BUFFERS. . . . . . . . . . . . . .          V BYTE  0000             External
CR . . . . . . . . . . . . . . . .          Number  000D
CTRL_Q . . . . . . . . . . . . . .          Number  0011
CTRL_S . . . . . . . . . . . . . .          Number  0013
CTRL_Z . . . . . . . . . . . . . .          Number  001A
CURRENT_DOS_LOCATION . . . . . . .          V WORD  0000             External
DATLEN . . . . . . . . . . . . . .          Number  0010
DEFAULT_DRIVE. . . . . . . . . . .          V BYTE  0000             External
DEVICE_LIST. . . . . . . . . . . .          V DWORD 0000             External
DEVLST . . . . . . . . . . . . . .          V BYTE  0000             External
DONE_MEM . . . . . . . . . . . . .          L NEAR  00BD     CODE
DOSBASE. . . . . . . . . . . . . .          Number  01E0
DOS_POINTER. . . . . . . . . . . .          V WORD  0000             External
DSKDEV . . . . . . . . . . . . . .          V WORD  0000             External
ENDBIOS. . . . . . . . . . . . . .          L NEAR  0000             External
ESC. . . . . . . . . . . . . . . .          Number  001B
FALSE. . . . . . . . . . . . . . .          Number  0000
FILES. . . . . . . . . . . . . . .          V BYTE  0000             External
FINAL_DOS_LOCATION . . . . . . . .          V WORD  0000             External
HDSKDEV. . . . . . . . . . . . . .          L NEAR  0000             External
HD_INIT. . . . . . . . . . . . . .          L NEAR  0000             External
HELLO. . . . . . . . . . . . . . .          L BYTE  00E1     CODE
IOINIT . . . . . . . . . . . . . .          L NEAR  0000             External
IVRELOC. . . . . . . . . . . . . .          Alias   TRUE
LDRDATA. . . . . . . . . . . . . .          Number  FEF0
LDRPKT . . . . . . . . . . . . . .          Number  FF00
LF . . . . . . . . . . . . . . . .          Number  000A
MEMORY_SIZE. . . . . . . . . . . .          V WORD  0000             External
MOD_NO . . . . . . . . . . . . . .          Number  0018
PACKET_ADR . . . . . . . . . . . .          L NEAR  0000             External
PACKET_BASE. . . . . . . . . . . .          L NEAR  0000             External
PKTFCN . . . . . . . . . . . . . .          Number  0025
PSTR . . . . . . . . . . . . . . .          L NEAR  0000             External
REV_NO . . . . . . . . . . . . . .          Number  0005
STACK. . . . . . . . . . . . . . .          E NEAR  0210     CODE
SYSINIT. . . . . . . . . . . . . .          L FAR   0000             External
TRUE . . . . . . . . . . . . . . .          Number  - 0001
TST_MEM. . . . . . . . . . . . . .          L NEAR  009F     CODE
VER_NO . . . . . . . . . . . . . .          Number  0002
```

```
 The Microsoft MACRO Assembler              02-20-84      PAGE     Symbols-2
MSDOS 2.00 BIOS and System initialization


WATACK . . . . . . . . . . . . . .          L NEAR  003A     CODE
WPRSNT . . . . . . . . . . . . . .          Number  0001
XCPRSNT. . . . . . . . . . . . . .          Number  0002
XFRPKT . . . . . . . . . . . . . .          Number  FEFC
XOPTI1 . . . . . . . . . . . . . .          L NEAR  0062     CODE
XOPTI2 . . . . . . . . . . . . . .          L NEAR  0075     CODE
XOPTION. . . . . . . . . . . . . .          V BYTE  0000             External
Z80INT . . . . . . . . . . . . . .          Number  0000

Warning Severe
Errors  Errors
0       0
```

```
 MSDOS 2.00 BIOS and System initialization

 Symbol Cross Reference                         (# is definition)       Cref-1

AUX2DEV. . . . . . . . . . . . . .        30#    149

BELL . . . . . . . . . . . . . . .        72#
BIOSINIT . . . . . . . . . . . . .        20      98#
BIOSSEG. . . . . . . . . . . . . .        17#    145     150
BIOSSIZ. . . . . . . . . . . . . .        18#    161
BUFFERS. . . . . . . . . . . . . .        46#    190

CGROUP . . . . . . . . . . . . . .        11      51      51
CODE . . . . . . . . . . . . . . .        11      50#     50     236
CR . . . . . . . . . . . . . . . .        70#    225
CTRL_Q . . . . . . . . . . . . . .        75#
CTRL_S . . . . . . . . . . . . . .        74#
CTRL_Z . . . . . . . . . . . . . .        76#
CURRENT_DOS_LOCATION . . . . . . .        36#    162

DATLEN . . . . . . . . . . . . . .        85#    117
DEFAULT_DRIVE. . . . . . . . . . .        44#    158
DEVICE_LIST. . . . . . . . . . . .        40#    196     197
DEVLST . . . . . . . . . . . . . .        22#    197
DONE_MEM . . . . . . . . . . . . .       184     188#
DOSBASE. . . . . . . . . . . . . .        78#
DOS_POINTER. . . . . . . . . . . .        29#    169
DSKDEV . . . . . . . . . . . . . .        30#    143     148

ENDBIOS. . . . . . . . . . . . . .        24#    164
ESC. . . . . . . . . . . . . . . .        73#

FALSE. . . . . . . . . . . . . . .        58#     61
FILES. . . . . . . . . . . . . . .        48#    191
FINAL_DOS_LOCATION . . . . . . . .        38#    168

HDSKDEV. . . . . . . . . . . . . .        30#    144
HD_INIT. . . . . . . . . . . . . .        25#    140
HELLO. . . . . . . . . . . . . . .       133     204#

IOINIT . . . . . . . . . . . . . .        23#    108
IVRELOC. . . . . . . . . . . . . .        80#

LDRDATA. . . . . . . . . . . . . .        82#     83     115
LDRPKT . . . . . . . . . . . . . .        83#    122     125
LF . . . . . . . . . . . . . . . .        71#    225

MEMORY_SIZE. . . . . . . . . . . .        42#    189
MOD_NO . . . . . . . . . . . . . .        68#

PACKET_ADR . . . . . . . . . . . .        28#    124
PACKET_BASE. . . . . . . . . . . .        27#    116
PKTFCN . . . . . . . . . . . . . .        86#    123
PSTR . . . . . . . . . . . . . . .        26#    134

REV_NO . . . . . . . . . . . . . .        67#

STACK. . . . . . . . . . . . . . .       101     234#
SYSINIT. . . . . . . . . . . . . .        34#    151     154     202

TRUE . . . . . . . . . . . . . . .        57#     58      80
```

```
 MSDOS 2.00 BIOS and System initialization

 Symbol Cross Reference                         (# is definition)       Cref-2

TST_MEM. . . . . . . . . . . . . .       175#    187

VER_NO . . . . . . . . . . . . . .        66#

WATACK . . . . . . . . . . . . . .       128#    129
WPRSNT . . . . . . . . . . . . . .        88#    138

XCPRSNT. . . . . . . . . . . . . .        89#    146
XFRPKT . . . . . . . . . . . . . .        84#    125     128
XOPTI1 . . . . . . . . . . . . . .       139     142     146#
XOPTI2 . . . . . . . . . . . . . .       147     151#
XOPTION. . . . . . . . . . . . . .        30#    138     146

Z80INT . . . . . . . . . . . . . .        87#    127
```

```
                                                  PAGE    60,132
                                                  TITLE   - MSDOS driver initialization
                                                  NAME    IO

                                          ;
                                          ;       COMPANY CONFIDENTIAL
                                          ;       Copyright (C) 1983 Digital Equipment Corporation
                                          ;       All rights reserved.
                                          ;
                                          ;       09/23/83

                                          cgroup group code

        0000                              code segment byte public 'code'
                                          assume cs:cgroup,ds:cgroup

                                          public  ioinit
                                          extrn   r100:byte,xoption:byte
                                          extrn   fastcon:near,clkisr:near,z80isr:near,intret:near
                                          extrn   physdisk:near,inthdl:near
                                          extrn   rupt7201:near, xrupt7201:near
                                          extrn   clk_60_50:byte, clk_16_20:word, clk_adj:byte, ticker:byte

                                       C  include iodef.ash
                                       C  ;Rainbow Interrupt numbers.
                                       C  ;1-Apr-83 sgs added dskio int vector
                                       C  ;19-Mar-83 sgs added profile int vector [user clock]
                                       C  ;the int profile is called by the RTC interrupt service for each tick.
                                       C  ;The ax and ds register don't need to be saved [done in clkisr].
                                       C  ;
        = 0064                         C  profile equ      64h      ;100. user interface to clock interrupt
        = 002C                         C  clk_int equ      2ch      ;60Hz int,
        = 0065                         C  dskio   equ      65h      ;direct disk io for format
                                       C  ;
                                       C  ;17-Mar-83 sgs changed to include int 20-26
                                       C  ;interrupt 22-24h are duplicated at 42-44h for consistency.
                                       C  ;These are the relocated Rainbow interrupts.
                                       C  ;interrupts, 20h-26h moved to 40h-46h
                                       C  ;ATTENTION Modules IOINIT and REINIT may have to be
                                       C  ;changed if int vectors 40h to 46h are changed
                                       C  ;
        = 0040                         C  vert    equ      40h      ;new vert. freq.
        = 0041                         C  spare41 equ      41h
        = 0042                         C  spare42 equ      42h
        = 0043                         C  comdma  equ      43h      ;DMA ctrl optional comm. board
        = 0044                         C  aux_prn equ      44h      ;7201 comm./printer
        = 0045                         C  excom   equ      45h      ;extented comm. option
        = 0046                         C  uart    equ      46h      ;new UART vector,
        = 0047                         C  z80     equ      47h      ;Z80 interrupt,
        = 0018                         C  rom     equ      18h      ;new ROM access,
                                       C  ;
                                       C  ;DEC Rainbow IO port stuff.
                                       C  ;
        = 0010                         C  kdp     equ      10h      ;8251 data port,
        = 0011                         C  ksp     equ      11h      ;8251 status,
```

```
        = 0040                         C  auxdp   equ      40h      ;7201 data,
        = 0041                         C  prndp   equ      41h
        = 0042                         C  auxp    equ      42h      ;7201 command,
        = 0043                         C  prnp    equ      43h
                                       C  ;
                                       C  ; Values in XOPTION
                                       C  ;
        = 0001                         C  wprsnt  equ      01       ;Winnie preset
        = 0002                         C  xcprsnt equ      02       ;XCOMM present

        = EE00                            sysram  equ      0EE00h           ;segment of system RAM
        = 1FFE                            syspar  equ      01FFEh           ;offset to SYSPAR word
        = 0100                            bit8    equ      0100h            ;Majik bit in SYSPAR
```

```
                                        page
                                        ;
                                        ; First re-init the default vectors by invoking ROM service
                                        ;
        0000                            ioinit:
        0000  BF 000C                           mov     di,0Ch          ;Setup ROM vectors
        0003  CD 28                             int     40
                                        ;
                                        ;get clock rate from rom
                                        ;
        0005  0E                                push    cs
        0006  1F                                pop     ds              ;ds:=bios seg
        0007  BF 000E                           mov     di,0Eh          ;return clock rate
        000A  CD 28                             int     40

        000C  B3 3C                             mov     bl,60           ; assume 60 Hz
        000E  B7 FD                             mov     bh,-3           ; adjust value
        0010  BA 01AA                           mov     dx,426          ; 0.01 sec add value
        0013  22 C0                             and     al,al           ; 0 = 60 Hz
        0015  74 07                             jz      clk60
        0017  B3 32                             mov     bl,50           ; it's 50Hz
        0019  B7 03                             mov     bh,+3
        001B  BA 0200                           mov     dx,512
        001E                            clk60:
        001E  88 1E 0000 E                      mov     clk_60_50,bl    ; set 60/50 Hz
        0022  89 16 0000 E                      mov     clk_16_20,dx    ; set 0.01 sec add value
        0026  88 3E 0000 E                      mov     clk_adj,bh      ; set adjust value
        002A  88 1E 0000 E                      mov     ticker,bl       ; init ticker
        002E                            clkx1:
        002E  FC                                cld

                                        ;For the Rainbow, we must play horrible games
                                        ;with interrupt vectors. Copy the problem 2x
                                        ;vectors elsewhere.

        002F  1E                                push    ds
        0030  06                                push    es
        0031  33 C0                             xor     ax,ax
        0033  8E D8                             mov     ds,ax

                                        ;Setup Winnie/Xcomm vectors here (also set XOPTION)

        0035  B8 EE00                           mov     ax,offset sysram
        0038  8E C0                             mov     es,ax
        003A  26: F7 06 1FFE 0100               test    es:word ptr syspar,bit8
        0041  75 11                             jnz     optnop

        0043  B3 02                             mov     bl,xcprsnt      ;assume XCOMM
        0045  E4 68                             in      al,68h          ;check winnie present
        0047  24 E0                             and     al,0E0h
        0049  3C A0                             cmp     al,0A0h
        004B  75 02                             jnz     xoptini
        004D  B3 01                             mov     bl,wprsnt
        004F                            xoptini:
        004F  2E: 88 1E 0000 E                  mov     cs:xoption,bl   ;set option byte
```

```

        0054  FA                        optnop: cli                     ;Ints off here
        0055  33 C0                             xor     ax,ax
        0057  8E C0                             mov     es,ax           ;ES: base 0
        0059  BB 0090                           mov     bx,24h*4        ;base of INT 24

        005C  C7 07 0000 E                      mov     word ptr 0[bx],offset RUPT7201  ;comm port handler
        0060  8C 4F 02                          mov     word ptr 2[bx],cs

        0063  2E: 80 3E 0000 E 01               cmp     cs:xoption,wprsnt               ;Winnie?
        0069  75 08                             jne     notwini
        006B  C7 47 04 0000 E                   mov     word ptr 4[bx],offset inthdl    ;Winnie handler
        0070  8C 4F 06                          mov     word ptr 6[bx],cs
        0073                            notwini:
        0073  2E: 80 3E 0000 E 02               cmp     cs:xoption,xcprsnt              ;XCOMM
        0079  75 08                             jne     notxcomm
        007B  C7 47 04 0000 E                   mov     word ptr 4[bx],offset XRUPT7201 ;XCOMM handler
        0080  8C 4F 06                          mov     word ptr 6[bx],cs
        0083                            notxcomm:
        0083  C7 47 0C 0000 E                   mov     word ptr 12[bx],offset z80isr
        0088  8C 4F 0E                          mov     word ptr 14[bx],cs

        008B  B9 0010                           mov     cx,8*2          ;8 vectors
        008E  BE 0080                           mov     si,20h*4        ;copy vert.[20h to 27h]
        0091  BF 0100                           mov     di,vert*4       ;to int 40h-47h
        0094  F3/ A5                            rep movsw

        0096  3E 00A0                           mov     si,28h*4        ;move INT 40
        0099  BF 0060                           mov     di,rom*4
        009C  A5                                movsw
        009D  A5                                movsw

        009E  BF 00A4                           mov     di,29h*4        ;set FASTCON
        00A1  B8 0000 E                         mov     ax,offset fastcon
        00A4  AB                                stosw
        00A5  8C C8                             mov     ax,cs
        00A7  AB                                stosw

        00A8  BF 0080                           mov     di,clk_int*4    ;set RTC,
        00AB  B8 0000 E                         mov     ax,offset clkisr
        00AE  AB                                stosw
        00AF  8C C8                             mov     ax,cs
        00B1  AB                                stosw

        00B2  BF 0190                           mov     di,profile*4    ;set RTC user vector
        00B5  B8 0000 E                         mov     ax,offset intret ;set to iret
        00B8  AB                                stosw
        00B9  8C C8                             mov     ax,cs
        00BB  AB                                stosw

        00BC  BF 0194                           mov     di,dskio*4      ;disk io,
        00BF  B8 0000 E                         mov     ax,offset physdisk
        00C2  AB                                stosw
        00C3  8C C8                             mov     ax,cs
        00C5  AB                                stosw
        00C6  FB                                sti                     ;re-enable ints

```

```
                                        page

                                        ;Interrupt vector relocation checks...
                                        ;checks to see if r100 A or r100 B -
                                        ;if r100 b then if ROM version 5.02+, then
                                        ;relocate vectors

        00C7  55                                push    bp              ;make sure frame ptr is preserved
        00C8  BF 001A                           mov     di,1ah          ;Rom version # check command
        00CB  BA 0109 R                         mov     dx,offset temp  ;check buffer
        00CE  8C CD                             mov     bp,cs           ;set frame ptr to roms
        00D0  CD 28                             int     40              ;invoke firmware
        00D2  5D                                pop     bp              ;and restore the frame ptr

        00D3  0E                                push    cs              ;set ds to cs for convenience
        00D4  1F                                pop     ds
        00D5  C6 06 0000 E 00                   mov     byte ptr R100,0 ;assume 100a
        00DA  80 3E 0109 R 00                   cmp     byte ptr temp,0 ;100A doesn't return anything
        00DF  74 25                             je      ioi5            ;and don't go any further

        00E1  A1 0109 R                         mov     ax,word ptr temp ;fetch digits
        00E4  86 C4                             xchg    al,ah           ;swap bytes?
        00E6  3D 3035                           cmp     ax,'05'         ;check first rom #
        00E9  72 1B                             jb      ioi5            ;just bail out
        00EB  A1 010C R                         mov     ax,word ptr temp+3 ;get 2nd ROM #
        00EE  86 C4                             xchg    al,ah           ;swap bytes
        00F0  3D 3032                           cmp     ax,'02'         ;check other ROM #
        00F3  72 11                             jb      ioi5

                                        ;We now have a 100B with current ROMs; relocate the hardware vectors

        00F5  C6 06 0000 E 01           shvvec: mov     byte ptr r100,1 ;tell 'em that type we are
        00FA  B0 20                             mov     al,20h
        00FC  B4 A0                             mov     ah,0A0h         ;
        00FE  BF 001C                           mov     di,1Ch          ;set up for rom vector relocate
        0101  B9 0016                           mov     cx,16h          ;
        0104  CD 28                             int     40              ;and tell the rom
        0106                            ioi5:
        0106  07                                pop     es
        0107  1F                                pop     ds
        0108  C3                                ret

        0109  00 00 00 00 00 00         temp    db      0,0,0,0,0,0,0,0,0,0,0,0,0,0
              00 00 00 00 00 00
              00 00

        0117                            code ends

                                                end
```

```
Segments and groups:

                N a m e                     Size     align    combine class

CGROUP . . . . . . . . . . . . . .          GROUP
  CODE . . . . . . . . . . . . . . .        0117     BYTE     PUBLIC  'CODE'

Symbols:

                N a m e                     Type     Value    Attr

AUXDP. . . . . . . . . . . . . . .          Number   0040
AUXP . . . . . . . . . . . . . . .          Number   0042
AUX_PRN. . . . . . . . . . . . . .          Number   0044
BIT8 . . . . . . . . . . . . . . .          Number   0100
CLK60. . . . . . . . . . . . . . .          L NEAR   001E     CODE
CLKISR . . . . . . . . . . . . . .          L NEAR   0000     CODE    External
CLKX1. . . . . . . . . . . . . . .          L NEAR   002E     CODE
CLK_18_20. . . . . . . . . . . . .          V WORD   0000     CODE    External
CLK_60_50. . . . . . . . . . . . .          V BYTE   0000     CODE    External
CLK_ADJ. . . . . . . . . . . . . .          V BYTE   0000     CODE    External
CLK_INT. . . . . . . . . . . . . .          Number   002C
COMDMA . . . . . . . . . . . . . .          Number   0043
DSKIO. . . . . . . . . . . . . . .          Number   0065
EXCOM. . . . . . . . . . . . . . .          Number   0045
FASTCON. . . . . . . . . . . . . .          L NEAR   0000     CODE    External
INTHDL . . . . . . . . . . . . . .          L NEAR   0000     CODE    External
INTRET . . . . . . . . . . . . . .          L NEAR   0000     CODE    External
IOI5 . . . . . . . . . . . . . . .          L NEAR   0106     CODE
IOINIT . . . . . . . . . . . . . .          L NEAR   0000     CODE    Global
KDP. . . . . . . . . . . . . . . .          Number   0010
KSP. . . . . . . . . . . . . . . .          Number   0011
NOTWINI. . . . . . . . . . . . . .          L NEAR   0073     CODE
NOTXCOMM . . . . . . . . . . . . .          L NEAR   0083     CODE
OPTNOP . . . . . . . . . . . . . .          L NEAR   0054     CODE
PHYSDISK . . . . . . . . . . . . .          L NEAR   0000     CODE    External
PRNDP. . . . . . . . . . . . . . .          Number   0041
PRNP . . . . . . . . . . . . . . .          Number   0043
PROFILE. . . . . . . . . . . . . .          Number   0064
R100 . . . . . . . . . . . . . . .          V BYTE   0000     CODE    External
ROM. . . . . . . . . . . . . . . .          Number   0018
RUPT7201 . . . . . . . . . . . . .          L NEAR   0000     CODE    External
SHVVEC . . . . . . . . . . . . . .          L NEAR   00F5     CODE
SPARE41. . . . . . . . . . . . . .          Number   0041
SPARE42. . . . . . . . . . . . . .          Number   0042
SYSPAR . . . . . . . . . . . . . .          Number   1FFE
SYSRAM . . . . . . . . . . . . . .          Number   EE00
TEMP . . . . . . . . . . . . . . .          L BYTE   0109     CODE
TICKER . . . . . . . . . . . . . .          V BYTE   0000     CODE    External
UART . . . . . . . . . . . . . . .          Number   0046
VERT . . . . . . . . . . . . . . .          Number   0040
WPRSNT . . . . . . . . . . . . . .          Number   0001
XCPRSNT. . . . . . . . . . . . . .          Number   0002
XOPTINI. . . . . . . . . . . . . .          L NEAR   004F     CODE
XOPTION. . . . . . . . . . . . . .          V BYTE   0000     CODE    External
```

```
XRUPT7201. . . . . . . . . . . . .          L NEAR   0000     CODE    External
Z80. . . . . . . . . . . . . . . .          Number   0047
Z80ISR . . . . . . . . . . . . . .          L NEAR   0000     CODE    External

Warning Severe
Errors  Errors
0       0
```

```
 - MSDOS driver initialization

 Symbol Cross Reference                   (# is definition)       Cref-1

AUXDP. . . . . . . . . . . . . . . .      56#
AUXP . . . . . . . . . . . . . . . .      58#
AUX_PRN. . . . . . . . . . . . . . .      46#

BIT8 . . . . . . . . . . . . . . . .      68#    113

CGROUP . . . . . . . . . . . . . . .      12      15      15
CLK60. . . . . . . . . . . . . . . .      88      92#
CLKISR . . . . . . . . . . . . . . .      19#    163
CLKX1. . . . . . . . . . . . . . . .      97#
CLK_16_20. . . . . . . . . . . . . .      22#     94
CLK_60_50. . . . . . . . . . . . . .      22#     93
CLK_ADJ. . . . . . . . . . . . . . .      22#     95
CLK_INT. . . . . . . . . . . . . . .      32#    162
CODE . . . . . . . . . . . . . . . .      12      14#     14      227
COMDMA . . . . . . . . . . . . . . .      45#

DSKIO. . . . . . . . . . . . . . . .      33#    174

EXCOM. . . . . . . . . . . . . . . .      47#

FASTCON. . . . . . . . . . . . . . .      19#    157

INTHDL . . . . . . . . . . . . . . .      20#    135
INTRET . . . . . . . . . . . . . . .      19#    169
IOI5 . . . . . . . . . . . . . . . .     199     204     208      218#
IOINIT . . . . . . . . . . . . . . .      17      73#

KDP. . . . . . . . . . . . . . . . .      54#
KSP. . . . . . . . . . . . . . . . .      55#

NOTWINI. . . . . . . . . . . . . . .     134     137#
NOTXCOMM . . . . . . . . . . . . . .     139     142#

OPTNOP . . . . . . . . . . . . . . .     114     125#

PHYSDISK . . . . . . . . . . . . . .      20#    175
PRNDP. . . . . . . . . . . . . . . .      57#
PRNP . . . . . . . . . . . . . . . .      59#
PROFILE. . . . . . . . . . . . . . .      31#    168

R100 . . . . . . . . . . . . . . . .      18#    197     212
ROM. . . . . . . . . . . . . . . . .      50#    152
RUPT7201 . . . . . . . . . . . . . .      21#    130

SHVVEC . . . . . . . . . . . . . . .     212#
SPARE41. . . . . . . . . . . . . . .      43#
SPARE42. . . . . . . . . . . . . . .      44#
SYSPAR . . . . . . . . . . . . . . .      67#    113
SYSRAM . . . . . . . . . . . . . . .      66#    111

TEMP . . . . . . . . . . . . . . . .     190     198     201      205      223#
TICKER . . . . . . . . . . . . . . .      22#     96

UART . . . . . . . . . . . . . . . .      48#

VERT . . . . . . . . . . . . . . . .      42#    148
```

```
 - MSDOS driver initialization

 Symbol Cross Reference                   (# is definition)       Cref-2


WPRSNT . . . . . . . . . . . . . . .      63#    121     133

XCPRSNT. . . . . . . . . . . . . . .      64#    116     138
XOPTINI. . . . . . . . . . . . . . .     120     122#
XOPTION. . . . . . . . . . . . . . .      18#    123     133      138
XRUPT7201. . . . . . . . . . . . . .      21#    140

Z80. . . . . . . . . . . . . . . . .      49#
Z80ISR . . . . . . . . . . . . . . .      19#    143
```

Rainbow™
MS™–DOS V2.05
BIOS Listings
QV068–GZ

## READER'S COMMENTS

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

________________________________________

________________________________________

________________________________________

________________________________________

________________________________________

Did you find errors in this manual? If so, specify the error and the page number.

________________________________________

________________________________________

________________________________________

________________________________________

________________________________________

Please indicate the type of reader that you most nearly represent.

□ First-time computer user
□ Experienced computer user
□ Application package user
□ Programmer
□ Other (please specify)____________________________

Name____________________________________

Date____________________________________

Organization________________________________

Street____________________________________

City____________________________________

State____________________________________

Zip Code
or Country________________________________

------Do Not Tear - Fold Here and Tape------------------------------------------------

digital

No Postage<br>Necessary<br>if Mailed in the<br>United States

**BUSINESS REPLY MAIL**
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

**SOFTWARE PUBLICATIONS**
200 FOREST STREET MRO1–2/L12
MARLBOROUGH, MA 01752

------ Do Not Tear - Fold Here and Tape ------------------------------------------------

**Cut Along Dotted Line**

Rainbow™

# Guidelines for Producing Translatable Products

digital equipment corporation

**First Printing, November 1984**

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

Printed in U.S.A.

# CHAPTER 1

# INTRODUCTION

## 1.1 WHO SHOULD READ THIS DOCUMENT?

If you are a software developer or documentation writer, read this document before starting work on your product. This will ensure that international requirements are built in, rather than added after the development of a version for one country.

## 1.2 WHAT THIS DOCUMENT WILL TELL YOU

The first chapter of this document gives an overview of topics that can affect the final quality of a translated product. These topics are then presented in greater detail in the remainder of the document, with particular emphasis on developing software for translation and writing text for translation.

## 1.3 WHY BUILD TRANSLATABILITY INTO THE DESIGN?

Sales of DIGITAL's products are increasing to users whose native language, customs, and conventions are different from those of the developers of the product. These users are frequently unfamiliar with computers.

If international requirements are not taken into consideration, translation is likely to be difficult and expensive. The key is to design a core product that does not have to be altered to be translated.

Products to be translated must be developed in three stages:

1. Identification of all international requirements in the following categories:

- Software
- Documentation
- Packaging and distribution

2. Development of the base system to meet the requirements of one country/market without putting obstacles in the way of international requirements.

3. Adaptation of the base system to meet the requirements of other countries/markets.

## 1.4 SOFTWARE

Your product may appear in one country in more than one language. Therefore it may have to include the ability to communicate in more than one language, and to switch from one language to another easily.

Also, functions of many applications differ from country to country. This means that different versions of the software will be required. For example, an accountancy application must conform to the accountancy practices of the target country.

## 1.5 DOCUMENTATION

The documentation that DIGITAL provides in countries outside the U.S. depends on the product and the country. Sometimes only user documentation is translated; sometimes nothing is translated. The factors that influence translation include:

- Legal requirements
- National standards
- General knowledge of the English language in the target country
- Availability of English-speaking product support and service personnel
- Nature of the product

## 1.6 PACKAGING AND DISTRIBUTION

Translated products may be packaged as:

- Distinct versions of a base product. One kit is then developed for each language by adapting the base product. Each language's version of the product is then maintained and supported separately.

- Translated updates to a base product. This leads to logistics problems of support and maintenance once there is more than one version of the product in the field.

# CHAPTER 2

# DEVELOPING SOFTWARE FOR TRANSLATION

This chapter describes how to design software so that it can be translated with a minimum of change to the original product.

When developing software that is to be translated, follow these guidelines:

- Allow for the full DIGTAL Multinational Character Set.
- Isolate user text in modules separate from the code.
- Allow for expansion of the text.

Specific guidelines are set out in the following sections.

## 2.1 CONSTRUCTING TRANSLATABLE SOFTWARE

The following topics are discussed:

- Construction of messages and commands
- How to handle errors
- Designing Help
- Representing characters of countries outside the U.S.

### 2.1.1 Screen Text

To make translation easier, isolate all text used to communicate with the user. Observe the following guidelines:

- Hold all messages as variable-length strings in separate modules or files. Do not embed messages in code.

- Do not modify messages by overwriting or by inserting English text.

- Try to store messages as one unit. If this is not possible, bear in mind the following points:

  - An English message which can be split neatly into, say, three parts (see the example below) may not divide neatly into the same number of distinct parts after translation.

  - The order in which the different parts can be displayed must be completely flexible, to take into account a different sentence structure in another language.

    For example, suppose you stored an error message in three parts, as follows:

    ```
    error n
    input exceeds the quantity allowed
    at line n
    ```

    and the parts were translated into German. If you then wanted to display them in the order:

    ```
    Error n, input exceeds the quantity allowed at line n.
    ```

    the resulting message in German would be ungrammatical.

  - Variables for which different values will be inserted at run time must be allowed to occur at any point within any of the parts of the message.

  - Do not use the same piece of text in different contexts. Although the text may be the same in English, it may be different in another language. For example, consider the two messages:

    ```
    Printer has been disconnected
    Device  has been disconnected
    ```

    If the text "has been disconnected" was stored as one unit, and the words "Printer" or "Device" were inserted as appropriate, there would be no problem in English, but in French the participle "disconnected" would have to agree with the gender of the noun; and in this case the nouns have different genders.

- Follow the GUIDELINES FOR WRITING TEXT in Chapter 3.

Because translated text is frequently longer than the original (it may occupy 25% to 50% more space), it is important to take the following points into consideration:

- If you want to display text at a particular position on the screen, do not store the position in the code. The length of the text will almost certainly change when it is translated; therefore the system may need to display the translated text at a different position on the screen.

  You could either:

  - store the position in the same file as the text. This can be changed when the text is translated.

  - use the length of the text as a parameter for calculating its display position.

  An example of a routine which must be able to handle differing lengths of text is one which centres text:

  English version:

  ```
         RAINBOW 100
  Diskette Copy Program V1.0
      Press EXIT to quit
  ```

  French version:

  ```
              RAINBOW 100
  Programme de copie de disquette V1.0
   Appuyez sur <SORTIE> pour terminer
  ```

- A routine that displays text must be capable of handling more than one full screen of text at a time. In addition, you must design the routine to allow the user to display the text one page at a time.

- Tables that are very tightly packed with information are difficult to format on the screen when translated. In order to accommodate the table on the screen, the translator may be obliged to use obscure abbreviations.

  One solution is to split the table into several different displays on separate screens. Another is to arrange the table in rows as opposed to columns.

For example, use:

```
Surname       !  n         n         n          n
              !
Age           !  n         n         n          n
              !
Nationality   !  n         n         n          n
              !
Date of Birth !  n         n         n          n
              !
Occupation    !  n         n         n          n
              !
Salary        !  n         n         n          n
```

rather than:

```
Surname  Age  Nationality  Date of Birth  Occupation  Salary
------------------------------------------------------------
   n      n        n             n             n         n
   n      n        n             n             n         n
   n      n        n             n             n         n
   n      n        n             n             n         n
   n      n        n             n             n         n
```

### 2.1.2 Commands

Making commands easy for the user to understand also helps translation. Employ the user interface that the system supports for commands, and observe the following guidelines:

- All command names, qualifier names, and qualifier values must be table-driven, so that they can be easily replaced with translated text.

- Checking of user responses must be table-driven.

- Consider the implications of translation before using English sentences as input. For example, can the software be modified easily to recognize that "oui" is the same response as "yes"?

Use one or more of the following methods to select commands from menus:

- Positioning the cursor on the command.
- Selecting a number.
- Selecting the one or more letters of the command. In this case check the letter or letters against a table of valid commands. Do not hard-code them. Allow also for the selected letter(s) to occur at any position within the command, since after translation, several commands in a menu might start with the same word. Remember that you may need to use more letters to make the translated command unique.

### 2.1.3 Handling Errors

Use the system-supported user interface for reporting errors.

Ensure that the error handler can use parameters in any order.

### 2.1.4 Designing Help

Follow the GUIDELINES FOR WRITING TEXT in Chapter 3.

### 2.1.5 Representing Characters Of Countries Outside The U.S.

There are two main ways in which products running on DIGITAL hardware represent alphabetic characters not present in ASCII.

1. Using the DIGITAL Multinational Character Set. This is the way that new software products represent alphabetics not present in ASCII. The DIGITAL Multinational Character Set is represented using 8 bits.
2. Using the 7-bit codes standardized for the relevant countries (just as ASCII is the U.S. standard). These codes are indistinguishable from ASCII. Up to 11 characters that appear in ASCII are replaced by other characters depending on the language being represented. These character sets are called National Replacement Character Sets. They are used extensively in Europe. The Rainbow 100+ supports the NRC sets for applications doing CONSOLE IN/CONSOLE OUT (level 2) character Input/Output.

For character-coded data in particular, your software should:

- Not mask the 8th (most significant) bit of a byte, or use it for special purposes, if the byte may contain character data.

- Permit at least all graphic characters in text literals and comments.

- Allow collating sequences to be table-driven. To be completely accurate, collation needs to use a two-character lookup, because, for example, German o-umlaut is regareded as "oe" and u-umlaut is regarded as "ue".

Appendix B lists the characters that you should use when characters in addition to the 94 graphic characters of ASCII are needed.

## 2.2 LANGUAGE ISSUES RELATING TO FORMATS

The format of certain types of information varies from language to language. This section indicates how to take some of these different formats into consideration, including:

- Numbers

- Dates

- Time

- Currency

- Addresses and telephone numbers

- Proper names and titles

### 2.2.1 Numbers

Numbers may be displayed in several formats. Ensure that the user can vary the format, and that any default can be changed to suit the target language.

The format of a number may include any of the following:

- + or blank indicating a positive number, or - indicating a negative number.

- The + and - may be either prefix or suffix - Prefix signs are always directly beside the first significant figure.

- Parentheses denoting a negative number. For example, (2356) equals -2356. This convention is chiefly used in financial applications.

Numbers may be separated in two ways:

1. Thousands - period, comma, null.

   For example:

   2.345

   2,345

   2 345

2. Decimal point - period, comma.

   For example:

   | | |
   |---|---|
   | 1.34 | (English) |
   | 1,34 | (Continental European) |

If your product allows numbers to be entered in a tabular form, do not separate individual table entries with a period or comma. Instead, use a space or a semicolon. For example:

```
123.456 876.886 956.907

892.654 174.987 217.767

563.982 786.653 543.921
```

### 2.2.2 Dates

Where your product displays the name of a day or month as letters, you must allow sufficient storage and display space to accommodate these names in other languages.

For example, this table shows the maximum number of characters you must reserve for storage and display for French, German, and Dutch. Note that month names are not always abbreviated.

| | French | German | Dutch |
|---|---|---|---|
| Longest day name | 8 | 10 | 9 |
| Longest month name | 9 | 9 | 9 |

The position of each component in a displayed date may be varied or it may be omitted completely. The components required are described below.

- YEAR - 2 or 4 digits (two digits are frequently used)
- MONTH NUMBER - number in range 1 to 12
- MONTH NAME - Allow enough space for the name of the month not to be abbreviated, because, for example in French, a three-letter abbreviation of month names results in confusion between "juin" and "juillet"
- ORDINAL-DAY NUMBER - day number as an ordinal. For example:

| | | |
|---|---|---|
| 1st | 2nd | (English) |
| 1er | 2 | (French) |
| 1. | 2. | (German) |

- ORDINAL-DAY - day number as an ordinal in words. For example:

  First, Second (English)
  premier, deux (French)
  ersten, zweiten (German)

- ARTICLE - for example: the, le, den
- DAY NAME - Sunday through Saturday (English)
  dimanche through samedi (French)
  Sonntag through Sonnabend (or Samstag) (German)
- DAY NUMBER - 1 through 31

- Components may be separated by various editing characters including (at least) hyphen, comma, period, space, and slash.

For example:

| | |
|---|---|
| 26-Feb-84 | (DIGITAL standard format for dates) |
| jeudi le premier mars 1984 | |
| 14/12 84 | (European) |
| 84.11.17 | (ISO Standard) |
| 6/27/84 | (USA) |
| 1/5 | (means 1st May or 5th January) |
| March 1984 | |
| Saturday 3rd March | |
| Donnerstag | (German, Thursday) |

Because some formats lead to confusion between the day and the month, it is best to use a format that includes the month name.

### 2.2.3 Time

Time may be displayed in either a 12-hour or 24-hour format. A time comprises a number of components:

- HOURS - One or two digits. Give the option to have a blank or a leading zero.
- MINUTES - Two digits with leading zero displayed. If the HOURS field is absent, a leading zero may be suppressed.
- SECONDS - Two digits with leading zero displayed. If the MINUTES field is absent, a leading zero may be suppressed.
- 1/100 SECOND - In the absence of other components, these are displayed in the form 00.nn. In conjunction with other components, the 1/100 seconds follow the second component, separated by a period. For example, four minutes, three and one-half seconds past two is displayed as 02:04:03.50 in 24-hour clock format.

  The HOURS, MINUTES and SECONDS components must always appear in the order shown and no gaps are allowed. That is, HOURS and SECONDS is not allowed but HOURS and MINUTES is.

- AM/PM - AM indicates morning. PM indicates afternoon. These are used in 12-hour systems only. They are always shown as capital letters. In some Scandinavian countries, M is used to indicate noon.

- ZONE - allow up to four characters to indicate the time zone. For example: EST, GMT, MST, CET.

- Separating characters may be placed between the components. These include period, colon, comma, space, and null.

For example:

11 P.M.

13:56:08.40

14.06 hours, GMT

A combination of date and time can also be shown in ISO (International Standards Organization) format. The standard ISO format is a 14-character numeric string, which can be stored as:

YYYYMMDDHHMMSS

This shows year, month, day, hour, minute, and second with no separating characters, although leading zeros may be inserted in the string where an entry is less than 10. The time is specified on a 24-hour clock, according to the standard specified by the local time zone. Do not assume a time-zone differs from your time-zone by an integral number of hours: for example, Newfoundland, Canada is normally 1 hour 30 minutes later than Eastern Standard Time.

### 2.2.4 Currency

A number may be displayed as an amount of money using the following components:

- A text string of up to four characters - for example, $, Fr, or pts. The text string may occur before or after the numbers.
- Sign placement - prefix floated or suffix
- Decimal point character
- Thousands character
- Number of fractional decimal places
- Scaling factor (normally 1)

The decimal point and the thousands convention for currencies may differ from those used for numbers. For example, French usage is 1.746,23 for numbers but can be Fr1746.23 for money.

For any currency, there may be suffixes after integer amounts which indicate that the preceding number will be scaled before use in arithmetic. For example, thirty U.S. cents needs to be divided by 100 to give .3 in dollars ($0.30). This is indicated by a scaling factor of 0.01. This method could also be used to handle, say, millions of dollars ($M) by using a scaling factor of 1,000,000.

Note also that the characters and digits representing some currencies expand on conversion, and therefore require more display space. For example: $1 becomes 1380 lires. You may also need to distinguish between the currencies of countries that use similar names. For example, US dollars and Canada dollars; French francs and Swiss francs.

### 2.2.5 Addresses And Telephone Numbers

Allow sufficient space for different layouts of addresses and telephone numbers.

- The order of the components of an address differs from country to country.

  For example,

  England

  | | |
  |---|---|
  | Mr S. B. Turner, | (title, initials, surname) |
  | 55, High Street, | (number, street) |
  | Grantham, | (town) |
  | Lincolnshire, | (county) |
  | GR1 OBT | (post code) |
  | England | (country) |

  Germany

  | | |
  |---|---|
  | Herr Dipl. Ing. Schmidt | (title, professional title, surname) |
  | Stolbergerstrasse 90 | (street, number) |
  | D-2000 Hamburg 55 | (post code, town, district code) |
  | FDR | (country) |

  France

  | | |
  |---|---|
  | Madame Dupont Claudette | (title, surname, first name) |
  | 17, Rue Louis Guerin | (number, street) |
  | F-69626 Lyon-Villuerbanne | (post code, town, suburb) |
  | France | (country) |

  NOTE

  Not all post codes are numeric. For example:

  RG2 OSU

- Not all telephone numbers are the same length or have the same format. Telephone numbers often include special characters to separate different components. Commonly used separators are square brackets, parentheses, hyphens and spaces. For example:

  [1]-(603)-884-1234

(0734) 868711

Note also that the same number could be represented in different ways depending on whether it was for national or international use. For example:

| | |
|---|---|
| National | (089) 9591-2323 |
| International | +49 89 9591 2323 |

The "+" means that you should dial your own international prefix. This varies from country to country.

### 2.2.6 Proper Names And Titles

Allow for non-alphabetic characters in proper name and title fields (for example, accented letters, apostrophes, and hyphens), to take into account users with names such as de la Bassetière, D'Agostino, Torres-Ferrer. Allow for different formats in title fields, to take into account such titles as Herr Dipl. Ing.

## 2.3 LANGUAGE ISSUES RELATING TO CHARACTER SETS

This section covers three types of collating sequence, and refers to the DIGITAL Multinational Character Set used by the Rainbow personal computer.

- Character string comparisons
- Collating the DIGITAL Multinational Character Set
- Collating sequences for European countries

### 2.3.1 Character String Comparisons

Where collated output is not required, character strings are collated according to the numerical ordering of the DIGITAL Multinational Character Set. This can occur, for example, in programming languages, indexed file key processing, sort/merge and other utilities.

### 2.3.2 Collating The DIGITAL Multinational Character Set

These rules are used where the language of the text is not known:

- The letters AE diphthong, O with slash and A with ring, appear in that order, after Z.
- The letter N with a tilde accent appears immediately after N. It is treated as a separate letter falling between N and O.
- German small sharp s and OE ligature are treated as 2-character sequences, "ss" and "OE" respectively. If storage or efficiency is a problem, then German small sharp s is treated as a separate letter between s and t, and OE ligature is treated as a separate letter between the N with a tilde accent and O.

Apart from the letters listed above, the letters in the Multinational Character Set are treated as groups of letters with the same basic collating value. Within a group, small letters are ordered according to their numeric value, and capital letters immediately follow the corresponding small letter. For example, the group for the letter A is:

Small a
Capital A
Small a with grave
Capital A with grave
Small a with acute
Capital A with acute
Small a with circumflex
Capital A with circumflex
Small a with tilde
Capital A with tilde
Small a with diaeresis/umlaut
Capital A with diaeresis/umlaut

The whole group is viewed as being equivalent to A.

### 2.3.3 Collating Sequences For European Languages

Unless the rules for a particular language state otherwise, the following rules apply:

- AE diphthong and OE ligature are treated as 2-character strings, "AE" and "OE" respectively.
- German small sharp s is treated as the 2-character string "ss".
- N with tilde, O with slash and A with ring are treated as if they were separate characters with diacritical marks.
- All letters with diacritical marks are treated as for collating the Multinational Character Set, as described in Section 2.3.2.

Dutch, English, French, Italian, and Portuguese have no additional rules.

Although the above rules apply generally, there may be some exceptions in certain countries. For example, some countries have special rules for collating proper names. For instance, when collating proper names in German, A with diaeresis/umlaut, O with diaeresis/umlaut and U with diaeresis/umlaut are treated as 2-character strings, "AE", "OE" and "UE" respectively.

Danish and Norwegian:

- U with diaeresis/umlaut is treated as if it were Y.
- A with diaeresis/umlaut is treated as if it were AE diphthong.
- O with diaeresis/umlaut is treated as if it were O with slash.
- AE diphthong, O with slash and A with ring appear as separate letters, in that order, after Z.

Finnish and Swedish:

- U with diaeresis/umlaut is treated as if it were Y.
- O with slash is treated as if it were O with diaeresis/umlaut.
- A with ring, A with diaeresis/umlaut and O with diaeresis/umlaut appear as separate letters, in that order, after Z.

Spanish:

- All letters with diacritical marks, except N with tilde, are treated as though the diacritical mark is not present.
- The string CH is treated as a single letter between C and D.
- N with tilde is treated as a single letter after N.
- The string LL is treated as a single letter between L and M.

## 2.4 DIACRITICAL INSENSITIVITY IN SEARCHES

Some text editors use a technique called "diacritical insensitivity" in searches. Just as case-insensitive searching allows small "a" to match capital "A", diacritical-insensitive searching allows "a" without a diacritical mark to match "a" with a diacritical mark.

If your product uses the Multinational Character Set, diacritical insensitivity could be included as a function in string-matching algorithms. However, the user must be given the option of disabling this function.

If the editor is in case-insensitive mode, the tables for upper- and lower-case are effectively combined. Thus, a search for "a" will match every capital and small "a", with and without a diacritical mark.

The correspondences between characters for diacritical-insensitive searching are listed in Appendix A. Note that although a search for upper-case "A" will match upper-case "A" with a diacritical, the converse is not true. A search for upper-case "A" with a diacritical will NOT match upper-case "A".

# CHAPTER 3

# GUIDELINES FOR WRITING TEXT

A clear writing style is of utmost importance in any technical text, especially if the text is to be translated.

Errors produced during translation can sometimes be attributed to inconsistent vocabulary and poor sentence construction in the original text, combined with a lack of technical knowledge on the part of the translator.

This chapter describes how to design screen text for translation, and how to improve the quality of printed text. It also outlines some areas which require special consideration when writing for translation.

The following DIGITAL publications discuss more fully some of the topics covered in this chapter:

- Writing for the Reader
- Personal Computer Documentor's Guide
- Software Publications Style Guide

## 3.1 DESIGN OF SCREEN TEXT

Well-designed screen displays make any software product simpler, and more pleasant to use. They also make translation easier.

### 3.1.1 Text Of Restricted Length

Screen text is often restricted in length. For example, an error message may be restricted to one line of the screen. Allow for the translated version being at least 30% longer than the original.

For single sentences or phrases, try to reserve at least 50% extra space; otherwise you impose restrictions on the translation that could result in an ambiguous, or even meaningless, translated message.

Do not, however, restrict the length of the original English, in order to end up with a short translated message. Always use as many words as are necessary to make the original message unambiguous.

If you do need to abbreviate an English message, supply the translator with the full text, and tell him what the restrictions on the length of the message are.

Other ways of shortening text may cause problems in another language. For example:

- Acronyms

  Do not use acronyms, because they cannot be translated. If an English acronym is retained in a foreign language, there is a risk that it might be similar to a word with negative associations.

- Abbreviations

  In general, avoid abbreviations.

  In particular, avoid abbreviating words that will remain in English. A translator could easily misinterpret such abbreviations. It is acceptable to use internationally recognized (SI) unit symbols for units of measurement.

### 3.1.2 Short Words

Short, common words are easier to read and understand. Do not use computer jargon. It can confuse and intimidate users.

| Use | Instead of |
|---|---|
| assist, help | facilitate |
| change | modification |
| start | activate, initiate |
| end | terminate |
| improves | enhances |
| show | demonstrate |
| stop | discontinue |
| use | employ, utilize |
| wrong | erroneous |

Use a short word in place of a long one only when you are sure that the two words convey exactly the same meaning. In some cases, a longer word may provide a clearer contrast with some other term you have used, or may provide a more precise meaning. In these cases, it is better to use the long word.

### 3.1.3 Short Sentences

Brief, simple sentences are generally easier to understand than those with several clauses.

When designing menu items and error messages, beware of omitting words for the sake of brevity. For instance, do not omit "the" or "a". This telegraphic style may make it impossible for the translator to tell whether a word is being used as a verb, an adjective or a noun.

For example, does "set to lock at next level" mean "set to the lock which is at the next level" or "set this so that it locks at the next level"?

This style can also obscure who is to perform a certain operation, or the time at which it is performed. For example: "scratch files deleted? (Yes or No)" might mean "have you deleted the scratch files?" or "do you want the scratch files to be deleted?".

### 3.1.4 Affirmative Sentences

Affirmative statements are generally simpler to understand than negative statements. Using this form clarifies the meaning and helps the translator to make an accurate translation.

| Use | Instead of |
|---|---|
| Complete your entry before returning to the menu. | Do not return to menu before completing entry. |

### 3.1.5 Active Voice

The active voice is usually easier to understand than the passive voice. Chapter 7 of the DIGITAL publication Writing for the Reader discusses the use of voice in detail.

| Use | Instead of |
|---|---|
| The EXAM switch verifies the contents of memory. | The contents of memory are verified by the EXAM switch. |

### 3.1.6 Strings Of Nouns

Do not string nouns together as if they were adjectives, to produce long titles or names. Often, it is not clear which words qualify which. Such titles and names are hard to read and often almost impossible to translate.

| Use | Instead of |
|---|---|
| A form for sorting the table | Table Sort Description Form |

### 3.1.7 Stacking Words

Do not stack words. Stacked words require more eye movements and make text harder to read.

| Use | Instead of |
|---|---|
| Employee's Name: | Employee's<br>Name: |

In addition, stacked words can lead to misinterpretation on the part of the translator, who may not realize that the words are to be translated as one unit. For example, if a part of a diagram were labelled:

Possible Load
Exclusion Records

the translator could interpret this as two distinct items. This would lead to a mistranslation.

### 3.1.8 Displaying Columns And Alphanumeric Character Strings

Three to five spaces between columns make data, especially numbers, easier to read.

| Use | | | Instead of |
|---|---|---|---|
| 1066 | 1840 | 1918 | 1066 1840 1918 |
| 1492 | 1877 | 1928 | 1492 1877 1928 |
| 1563 | 1901 | 1929 | 1563 1901 1929 |
| 1701 | 1907 | 1963 | 1701 1907 1963 |

Where appropriate, put strings of alphanumeric characters in groups to make them more legible.

| Use | Instead of |
|---|---|
| ABCD 123 456 789 | ABCD123456789 |
| BCDE 234 567 890 | BCDE234567890 |
| CDEF 345 678 901 | CDEF345678901 |

### 3.1.9 Standardize Input Of Data

Standardize input of data for consistency and minimum confusion. Try to use an unambiguous format for dates. Use the month names, but make sure that the product can accept the translated names.

| Use | | Instead of | |
|---|---|---|---|
| Date of Birth: | 24/July/55 | Date of Birth: | 24/07/55 |
| Date of Hire: | 01/February/79 | Date of Hire: | February 1, 1979 |

### 3.1.10 Aligning Lists And Displays

Generally, left-aligned, vertical lists increase legibility.

| Use | Instead of |
|---|---|
| Monday | Monday, Tuesday, Wednesday, |
| Tuesday | Thursday, Friday |
| Wednesday | |
| Thursday | |
| Friday | |

Data-entry fields often have two parts: a name, and a part that the user can modify. If both parts are left-aligned, the text is usually more legible.

```
Use                                  Instead of

Account Name:   The Widget Company   Account Name: The Widget Company
Street Address: 123 Maple Street     Street Address:  123 Maple Street
City, State:    Boston, MA           City, State: Boston, MA
Post Code:      02100                Post Code: 02100
```

### 3.1.11 Use Of Capital Letters

Use a mixture of capital letters and small letters for running text.

| Use | Instead of |
|---|---|
| Text containing capital letters and small letters is easier to read than text containing all capital letters. Use capital letters sparingly to HIGHLIGHT special terms. | ALL CAPITAL LETTERS ARE MUCH HARDER TO READ THAN A MIXTURE OF CAPITAL LETTERS AND SMALL LETTERS. THE WORDS BLEND TOGETHER AND LOSE IMPACT. |

## 3.2 IMPROVING THE QUALITY OF PRINTED TEXT

This section presents guidelines for improving the quality of the English text of technical manuals. These guidelines apply at all times, not only when writing text for translation. Any increase in writing quality will help the translator.

### 3.2.1 Logical Sequence

Describe events in the order in which they occur. A logical sequence helps to keep the meaning clear and makes the translator's job easier.

| Use | Instead of |
|---|---|
| Select the line number and press DO. | Press DO after selecting the line number. |

### 3.2.2 Consistency

Keep to one idea within each paragraph or section. Try to keep to one tense and person within each book. Use the passive voice selectively.

Try to identify the typical user of the manual and choose the most suitable tense and person for that user. Generally, keep to this throughout the manual. Follow these guidelines:

- Use only one term to refer to the same concept.
- Do not use slang or jargon.
- Use terms consistently:
  - within a manual
  - across manuals

Remember that although the translator's command of English should be high, he or she encounters the text in circumstances very different from those of the English-speaking user. For instance, the translator will not always have the opportunity to experiment with the product in the way a user does.

Maintain your own glossary of terms with special meaning, and give a copy of this to the translator. Include titles of applications, routines, and specialized terms. Where possible, check that your use of terms is consistent with their use in other manuals. It is possible that the translator may refer to, or be familiar with, other related technical manuals.

### 3.2.3 Coherence

Always try to link the sentences in a paragraph, using words such as, "also", "but", "however", "similarly", and so on. Chapter 4 of Writing for the Reader discusses in detail how to achieve coherence in your writing.

Since it is possible that the translator will not have access to a computer, or to the product you describe, the translator cannot experiment with the system and may not be able to form a mental picture of some of the things you describe. If there are two possible meanings to what you are saying, the translator will have to render one of the two meanings into the foreign language. For example, consider the following use of the word "may":

> Strings must have matching quotation marks or no

> quotation marks at all, and numbers MAY have quotation marks.

This is open to the following interpretations:

> ... and numbers must not have quotation marks.
>
> or
>
> ... and numbers do not necessarily have to have quotation marks.

Clearly, if the translator chooses the wrong interpretation, the meaning of the sentence is changed dramatically. Even if your text appears clear to an English-speaking reader, always try to look for any possible ambiguity, which could lead to misinterpretation by the translator.

### 3.2.4 Sentence Construction

Use short introductory clauses. This places little strain on the reader's short-term memory.

Place the subject and verb of a sentence as close together as possible and use short sentences of 25 words or less. If you have to use a long sentence, use commas to clarify your meaning.

See Chapter 6 of Writing for the Reader for a discussion about how to avoid writing complex sentences.

### 3.2.5 Clarity

When writing English text, try not to use two words together that have more than four syllables.

Do not use two or more negative words in the same sentence. Use words with a concrete meaning, rather than abstract words.

Do not use abbreviations and acronyms.

Do not use Latin abbreviations, terms or phrases. Instead, use the following equivalents:

| | |
|---|---|
| c, ca, circa | about |
| cf. | compare |
| e.g. | for example; for instance |
| etc. | and so on; and so forth |

| | |
|---|---|
| i.e. | that is |
| via | by means of; by using |
| viz. | namely |

## 3.3 SPECIAL CONSIDERATIONS FOR TEXT TRANSLATION

This section covers:

- Culturally specific examples
- Text concerned with handling language
- Use of tables, diagrams, and figures

### 3.3.1 Culturally Specific Examples

The methods described in Section 3.2 were concerned with clarifying meaning for the translator. But there are also ways in which text, when translated literally, may be less suitable for a reader in another country than for an English-speaking reader. For example:

To clarify a new concept, you give an example from everyday life, which uses terms of reference with which readers in your own country are familiar. For example, you might show how a particular accounting package can be used to calculate mortgage repayments. A literal translation of the example will only confuse readers in countries where a mortgage system is not in use.

If there is no alternative but to give specific examples that are unique to their country of origin, ask the translator to substitute as much as possible of the example with local language equivalents.

Draw up a list of words or phrases that need adaptation, and include a specific note if anything in the original must be retained. For example, you might want measurements changed as a general rule, but the word 'inches' kept to refer specifically to the number of characters per inch put out by a printer.

Avoid using puns, metaphors, and similes. It is always difficult, and sometimes impossible, for the translator to find a suitable equivalent for these in another language.

### 3.3.2 Text Concerned With Handling Language

If you describe any operation that involves the order of words in a phrase or the characters in a word, any example you give will almost certainly have to be adapted or rewritten for translation. If you gave as an example, a list of items rearranged in alphabetic order, a translation of the MEANING of those items would not result in words beginning with the same initial letters. The example would therefore be meaningless.

If your list began with the words: apples, bananas..... a literal translation into French would result in: pommes, bananes.....

One possible way round this problem might be to use proper names, for example girls' names (Anna, Beatrice, Cleopatra...) which would retain the same initial letters after adaptation into the target language. The attention of the translator must be drawn to special examples like this.

Here are further examples of topics that give rise to the same kind of problem:

- Changes made to a word using the editor.

  English example:

  > In the phrase 'a letter form Graham', to change the word 'FORM' into 'FROM', place the cursor over the 'O', type 'R', move the cursor one place to the right, press the "delete character" key.

  Literal translation:

  > In the phrase '(a) formulaire a lettres Jacques', to change the word 'FORMULAIRE' into 'DE', place the cursor over the 'O', type 'R', move the cursor one place to the right, press the "delete character" key.

  The translator will obviously be aware that following these instructions will not produce the word DE. However, there is not sufficient information available for the translator to be able to adapt the example. In cases such as this, highlight the example, and give the translator enough information to be able to supply an equivalent example in the target language.

- Use of "wildcards" to search for words without specifying certain character positions.

  English example:

  > If you specify 'search for ????BORO', WESTBORO, EASTBORO will be found, NORTHBORO will not.

  Literal translation:

  > (If you specify) '(Search for) ????STADT', WESTSTADT, OSTSTADT (will be found,) NORDSTADT (will not.)

  Highlight examples like this for the translator, and ask that they adapt them to make sense in the language of the applicable country.

- File names made unique by judicious choice of the first word of the name, so that only a few keystrokes are necessary to identify the file.

  English example:

  > If you call your reports REPORT FOR MAY, REPORT FOR JUNE, and so on you will have to type the whole name to identify each document. Using MAY REPORT, JUNE REPORT, and so on, you need only type the first word.

  Literal translation:

  > (If you call your reports) RAPPORT MENSUEL DE MAI, RAPPORT MENSUEL DE JUIN, (and so on you will have to type the whole name to identify each document. Using) RAPPORT MENSUEL DE MAI, RAPPORT MENSUEL DE JUIN, (and so on, you need only type the first word.)

  Note that the French language does not have the possibility of translating these document names with the month name first. Even if the translator has understood the principle in English, it cannot be reproduced with this same example.

  If you know that an example cannot possibly be adapted for a particular language, you should try to find an alternative example in English, wherever possible.

- The ways in which a subfield of a date, identified by its POSITION, may be updated.

  American example:

  > In the date '5/24/82', the second field will change at 12 p.m.

  Literal translation:

  > (In the date) '24/5/82', (the second field will change at 12 p.m.)

  For examples like this, you should simply highlight the problem for the translator, and ask him or her to deal with it.

- The method of sorting a list of addresses, with the assumption that the house number will appear BEFORE the street name.

  English example:

  > To sort a list of addresses so that all the house numbers in the same street are in ascending order, select a NUMERIC FOLLOWED BY ALPHA sort. The result will look like this:
  >
  > 1 High Street<br>
  > 34 High St.<br>
  > 76 High St.<br>
  > 3 Station Rd.

  Literal translation:

  > (To sort a list of addresses so that all the house numbers in the same street are in ascending order, select a NUMERIC FOLLOWED BY ALPHA sort. The result will look like this:)
  >
  > Hochstr. 1<br>
  > Hochstr. 34<br>
  > Hochstr. 76<br>
  > Bahnhofstr. 3

  Note that even though the translator can see that "Hochstr. 1" is not numbers followed by letters, not enough information is available to supply the correct instructions to perform this task.

  You should therefore supply supplementary information so that the translator is in a position to adapt the example.

In general, try to avoid basing examples of field manipulation, searches, sorting, and so on, on any of the "language/country specific" categories listed in this section.

If the example could lose its value when certain features of the wording or spelling are not reproduced in translation, make the text as simple and explicit as possible about the principle being illustrated. Insert an extra phrase such as "Consider this example of numeric characters followed by alphabetic characters". In addition, help the translator and the editor in the target country, by supplying them with a list of the sections of your text which are likely to require adaptation of this kind. Add details cf the criteria you used in creating the original of these sections. Such notes would include:

- The purpose of the example.

- The restrictions that had been observed in the choice of the original example (for instance, "a file name not exceeding 6 characters").

- Indication of any text that was required ONLY for users operating in English (for instance, if an English example was included to show that the words "through" and "thru" were interchangeable in a certain command).

### 3.3.3 Use Of Tables, Diagrams And Figures

When writing for translation, do not include humorous cartoons or pictures that indicate words directly. For example, a cartoon familiar to American readers might not translate into anything remotely similar in another language.

- Label all parts of flowcharts and diagrams clearly and include a key to explain any symbols used.

- Specifically avoid national symbols, such as mailboxes, flags, baseball, and so on.

- Avoid putting text in boxes within diagrams. This causes problems if the translated text is much longer than the original.

With regard to illustration, number all the parts which require explanation, and then supply a separate key in the text to explain the numbers. This keeps any text to be translated separate from the illustration itself.

Alternatively, consider producing the diagram on one transparency, and the labelling on a second transparency. In this manner, the translated labelling can be overlaid on the original diagram.

# APPENDIX A

## DIACRITICAL CHARACTER MATCHING

This appendix lists those characters in the DIGITAL Multinational Character Set which have diacritical marks. Characters within the same group are treated as variants of the same character. Note that these rules apply when the language is unspecified. More complete rules are given in Section 2.3

| Letter | Diacritical Character Group | |
|---|---|---|
| A | À | Capital A with grave accent |
| | à | Small a with grave accent |
| | Á | Capital A with acute accent |
| | á | Small a with acute accent |
| | Â | Capital A with circumflex |
| | â | Small a with circumflex |
| | Ã | Capital A with tilde |
| | ã | Small a with tilde |
| | Ä | Capital A with diaeresis/umlaut |
| | ä | Small a with diaeresis/umlaut |
| C | Ç | Capital C with cedilla |
| | ç | Small c with cedilla |
| E | È | Capital E with grave accent |
| | è | Small e with grave accent |
| | É | Capital E with acute accent |
| | é | Small e with acute accent |
| | Ê | Capital E with circumflex |
| | ê | Small e with circumflex |
| | Ë | Capital E with diaeresis/umlaut |
| | ë | Small e with diaeresis/umlaut |
| I | Ì | Capital I with grave accent |
| | ì | Small i with grave accent |
| | Í | Capital I with acute accent |
| | í | Small i with acute accent |
| | Î | Capital I with circumflex |

DIACRITICAL CHARACTER MATCHING

| | | |
|---|---|---|
| | î | Small i with circumflex |
| | Ï | Capital I with diaeresis/umlaut |
| | ï | Small i with diaeresis/umlaut |
| O | Ò | Capital O with grave accent |
| | ò | Small o with grave accent |
| | Ó | Capital O with acute accent |
| | ó | Small o with acute accent |
| | Ô | Capital O with circumflex |
| | ô | Small o with circumflex |
| | Õ | Capital O with tilde |
| | õ | Small o with tilde |
| | Ö | Capital O with diaeresis/umlaut |
| | ö | Small o with diaeresis/umlaut |
| U | Ù | Capital U with grave accent |
| | ù | Small u with grave accent |
| | Ú | Capital U with acute accent |
| | ú | Small u with acute accent |
| | Û | Capital U with circumflex |
| | û | Small u with circumflex |
| | Ü | Capital U with diaeresis/umlaut |
| | ü | Small u with diaeresis/umlaut |
| Y | Ÿ | Capital Y with diaeresis/umlaut |
| | ÿ | Small y with diaeresis/umlaut |

# APPENDIX B

## DIGITAL MULTINATIONAL CHARACTER SET

This table describes the graphic characters included in the DIGITAL Multinational Character Set. The first 128 character positions (octal 0 to 177) are exactly the same as the ASCII Character Set. Positions 200 to 237 are addional control characters not described here.

| Octal | Graphic character | |
|---|---|---|
| 240 | | reserved |
| 241 | ¡ | inverted exclamation mark |
| 242 | ¢ | cent sign |
| 243 | £ | pound sign |
| 244 | | reserved |
| 245 | ¥ | yen sign |
| 246 | | reserved |
| 247 | § | section sign |
| 250 | ¤ | general currency sign |
| 251 | © | copyright sign |
| 252 | ª | feminine ordinal indicator |
| 253 | « | angle quotation mark left |
| 254 | | reserved |
| 255 | | reserved |
| 256 | | reserved |
| 257 | | reserved |
| 260 | ° | degree sign |
| 261 | ± | plus/minus sign |
| 262 | ² | superscript 2 |
| 263 | ³ | superscript 3 |
| 264 | | reserved |
| 265 | µ | micro sign |
| 266 | ¶ | paragraph sign |
| 267 | · | middle dot |

# DIGITAL MULTINATIONAL CHARACTER SET

| Octal | Graphic character | |
|---|---|---|
| 270 | | reserved |
| 271 | ¹ | superscript 1 |
| 272 | º | masculine ordinal indicator |
| 273 | » | angle quotation mark right |
| 274 | ¼ | fraction one quarter |
| 275 | ½ | fraction one half |
| 276 | | reserved |
| 277 | ¿ | inverted question mark |
| 300 | À | capital A with grave accent |
| 301 | Á | capital A with acute accent |
| 302 | Â | capital A with circumflex |
| 303 | Ã | capital A with tilde |
| 304 | Ä | capital A with diaeresis/umlaut |
| 305 | Å | capital A with ring |
| 306 | Æ | capital AE diphthong |
| 307 | Ç | capital C with cedilla |
| 310 | È | capital E with grave accent |
| 311 | É | capital E with acute accent |
| 312 | Ê | capital E with circumflex |
| 313 | Ë | capital E with diaeresis/umlaut |
| 314 | Ì | capital I with grave accent |
| 315 | Í | capital I with acute accent |
| 316 | Î | capital I with circumflex |
| 317 | Ï | capital I with diaeresis/umlaut |
| 320 | | reserved |
| 321 | Ñ | capital N with tilde |
| 322 | Ò | capital O with grave accent |
| 323 | Ó | capital O with acute accent |
| 324 | Ô | capital O with circumflex |
| 325 | Õ | capital O with tilde |
| 326 | Ö | capital O with diaeresis/umlaut |
| 327 | Œ | capital OE ligature |
| 330 | Ø | capital O with slash |
| 331 | Ù | capital U with grave accent |
| 332 | Ú | capital U with acute accent |
| 333 | Û | capital U with circumflex |
| 334 | Ü | capital U with diaeresis/umlaut |
| 335 | Ÿ | capital Y with diaeresis/umlaut |
| 336 | | reserved |
| 337 | ß | German small sharp s |

| Octal | Graphic character | |
|---|---|---|
| 340 | à | small a with grave accent |
| 341 | á | small a with acute accent |
| 342 | â | small a with circumflex |
| 343 | ã | small a with tilde |
| 344 | ä | small a with diaeresis/umlaut |
| 345 | å | small a with ring |
| 346 | æ | small ae diphthong |
| 347 | ç | small c with cedilla |
| 350 | è | small e with grave accent |
| 351 | é | small e with acute accent |
| 352 | ê | small e with circumflex |
| 353 | ë | small e with diaeresis/umlaut |
| 354 | ì | small i with grave accent |
| 355 | í | small i with acute accent |
| 356 | î | small i with circumflex |
| 357 | ï | small i with diaeresis/umlaut |
| 360 | | reserved |
| 361 | ñ | small n with tilde |
| 362 | ò | small o with grave accent |
| 363 | ó | small o with acute accent |
| 364 | ô | small o with circumflex |
| 365 | õ | small o with tilde |
| 366 | ö | small o with diaeresis/umlaut |
| 367 | œ | small oe ligature |
| 370 | ø | small o with slash |
| 371 | ù | small u with grave accent |
| 372 | ú | small u with acute accent |
| 373 | û | small u with circumflex |
| 374 | ü | small u with diaeresis/umlaut |
| 375 | ÿ | small y with diaeresis/umlaut |
| 376 | | reserved |
| 377 | | reserved |

Mon 19-Mar-1984 17:28 GMT