User's Guide

for

Corporation

Tapestar™ for DOS

Tape Utility Application

Document 500521 H

Qualstar Corporation
6709 Independence Avenue
Canoga Park, CA 91303

FAX: (818) 592-0116
Phone: (818) 592-0061

Copyright Notice

Trademarks

Qualstar Corporation
6709 Independence Avenue
Canoga Park, CA 91303

FAX: (818) 592-0116
Phone: (818) 592-0061

Document Revision Table

CHAPTER	TITLE	REVISION
	Title Page	H
1	Introduction to Tapestar	H
2	MCS-1 Installation	H
3	SCSI Installation	H
4	Installing BACK-IT 4 LAN	E
5	Using TAPETEST.EXE	H
6	Using TAPEMENU.EXE	H
7	Drive Selection	H
8	Using TAPE2DSK.EXE	H
9	Using DSK2TAPE.EXE	H
10	Using TLABEL.EXE	H
11	Using TCONVERT.EXE	H
12	Using TAPELIST.EXE	H
13	Using DEX.EXE	H
14	Using COPYTAPE.EXE	H
15	Using TCLONE.EXE	H
16	Using TAR.EXE	H
17	Using TABLEDIT.EXE	H
18	Error Messages	F
19	Appendix A	H

Limited Warranty

QUALSTAR SHALL NOT BE RESPONSIBLE OR LIABLE FOR ANY SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES OR LOSS ARISING FROM THE USE OF THIS PRODUCT.

THIS WARRANTY IS IN LIEU OF, AND BUYER WAIVES, ALL OTHER WARRANTIES, EXPRESSED OR IMPLIED, ARISING BY LAW OR OTHERWISE, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. THE REMEDIES STATED IN THIS WARRANTY ARE EXCLUSIVE.

Qualstar warrants the hardware to be free from defects in material and workmanship under normal use for a period of one year from the date of shipment from the factory. Qualstar warrants the media on which software is furnished to be free from defects in material and workmanship under normal use for a period of 30 days from the date of shipment from the factory. As such, the entire liability of Qualstar Corporation, and the buyer's exclusive remedy, shall be a refund of the purchase price or the replacement of any product not meeting the limited warranty previously stated. Some states do not allow the exclusion of implied warranties or the limitation of remedies, so the above exclusions and limitations may not apply to you. You may also have other rights which vary from state to state.

Qualstar makes no representations as to the suitability of software supplied for use in any application, and the Buyer agrees to accept all such software on an "as is" basis. Furthermore, Qualstar does not warrant software to be free from defects, and assumes no responsibility for damages of any kind, either actual or consequential, for such software failing to perform as documented or in any other manner. In no event will Qualstar be liable for damages, including any lost profits, lost savings or other incidental or consequential damages arising out of the use or inability to use such software, even if Qualstar has been advised of the possibility of such damages, or for any claim by any other party.

This warranty shall immediately be null and void if, in Qualstar's sole judgment, the product has been subject to misuse, abuse, negligence or accident, damaged in shipment, subjected to improper environmental conditions or had its serial numbers and/or other product markings altered, defaced or removed.

This warranty will remain in effect notwithstanding Buyer's shipment to third parties, but warranty remedies defined herein are applicable only to Buyer and are not transferable. Buyer shall disclose to third parties the terms of this warranty and shall indemnify Qualstar from any failure to make such disclosure and from any warranties made by the Buyer beyond those set forth herein.

Qualstar Return Policy

IMPORTANT — Read this before opening the package!

Qualstar Corporation has made every effort to provide quality merchandise. If the merchandise arrives damaged or has been mishandled, please notify the delivery company immediately. If the merchandise is defective in any other way, please follow these instructions.

If this merchandise was purchased from a dealer or other reseller, contact that dealer or reseller for return instructions.

If this merchandise was purchased directly from Qualstar, please do the following:

1. Treat the merchandise carefully to assure its suitability for return.
2. Retain all shipping and packing materials. Returned merchandise must be shipped to Qualstar in the original packing material and carton. If the original packing material or carton has been damaged, a new set may be requested.
3. Call Qualstar Customer Service at (818) 592-0061 to request a Return Merchandise Authorization (RMA) number. Returned merchandise will not be accepted without this number.
4. Clearly identify the outside of the carton and the packing list with the RMA number.
5. Return the merchandise, including cables, manuals, software and all related documents to:

Qualstar Corporation
6709 Independence Avenue
Canoga Park, CA 91303

FAX: (818) 592-0116
Phone: (818) 592-0061

Technical Assistance

The information contained in this guide is sufficient to resolve most problems which may occur during software installation and program operation. Technical assistance is available if problems are experienced which are not covered in this guide. Qualstar technical support is available between 8 AM and 4:00 PM, Pacific Standard Time. Be ready to provide the following information:

1. Model and serial numbers of the tape drive being used;
2. Computer system manufacturer, model, bus (i.e., PC, ISA, EISA, MCA), CPU used (i.e.., 286, 386D, 386SX, 486D2, etc.) and speed (16, 33, 50 MHz);
3. Tape controller or host adapter model and serial number, including revision level (read this from the board itself);
4. The current jumper and switch settings on the host adapter card;
5. Operating system and version in use (i.e.., DOS 5.0, OS2 2.0, etc.);
6. A list of devices installed in the computer (network controllers, modems, etc.);
7. A list of any device drivers loaded (found in CONFIG.SYS);
8. A list of memory-resident programs (e.g., SideKick) installed (found in AUTOEXEC.BAT);
9. The name and revision of the program in which the problem occurs;
10. The command syntax used to create the problem;
11. A description of the problem encountered, along with any accompanying error messages.

For technical assistance, please write or call

Qualstar Corporation
6709 Independence Avenue
Canoga Park, CA 91303

FAX: (818) 592-0116
Tel: (818) 5920-0061

Table of Contents

Chapter 8 - Using TAPE2DSK.EXE

Chapter 9 - Using DSK2TAPE.EXE

Chapter 10 - Using TLABEL.EXE

Chapter 11 - Using TCONVERT.EXE

Chapter 12 - Using TAPELIST

Chapter 13 - Using DEX.EXE

Chapter 14 - Using COPYTAPE.EXE

Chapter 15 - Using TCLONE.EXE

Chapter 16 - Using TAR.EXE

Chapter 17 - TABLEDIT and Translation Tables

Chapter 18 - Error Messages

Chapter 19 - Appendix A

1. Introduction to Tapestar for DOS

1.1 Purpose of This Guide

This guide explains how to use *Tapestar for DOS* magnetic tape utilities for DOS computers.

1.2 Tapestar Overview

Tapestar for DOS allows you to access data on magnetic tape using DOS. Three versions of *Tapestar for DOS* are available:

1. ***Tapestar-MCS*** - This version supports the MCS-1 tape controller, which in turn supports all Qualstar tape drives.
2. ***Tapestar-SCSI*** - In addition to the support provided by *Tapestar-MCS*, this version also supports various SCSI host adapters and can be used with all Qualstar SCSI tape drives, including quarter-inch cartridge (QIC) tape drives.
3. ***Tapestar-Plus*** - In addition to the support provided by *Tapestar-SCSI*, this version also supports the LMSI half-inch cartridge tape drive.

1.3 Tapestar Package Contents

The *Tapestar for DOS* package consists of the following:

1. This manual.
2. One set of the following diskettes:
   a. Tapestar-MCS for DOS, PN 738-0045-0;
   b. Tapestar-SCSI for DOS, PN 738-0053-4;
   c. Tapestar Plus for DOS, PN 738-0064-1;
3. One *Back-It 4* backup and restore software package.

NOTE

Back-It 4 is a backup and restore program that is bundled in with each Tapestar package. This package comes with its own diskette and manual, and is supported by Gazelle Systems. To install *Back-It 4*, follow the instructions in Chapter 4.

1.4 Typographic Conventions

This guide uses the following typographical conventions to help the reader and to eliminate confusion which sometimes arises when reading software documentation.

1. In the text, program names, drive letters, paths and file names are shown in UPPER CASE. Occasionally, switches are shown in upper case for visual clarity as in the TAPE2DSK and DSK2TAPE chapters. Unless specifically mentioned otherwise, you may use upper or lower case on the command line.
2. Parameters which you supply the text for are shown in *italics*. Italics may also be used for emphasis.

3. `Examples and screen text are shown in this font.` (Some examples of screen text may be too long to fit in the space allotted in this guide. In such cases, second and subsequent lines will be indented.)

4. `Command syntax is shown in this font.`

1.5 Syntax Conventions

The syntax conventions used in this manual follow standard DOS conventions. The following is an example of a syntax line:

Syntax: **`sample`** *`file-specifier`* `[`**`+r`**`|`**`-r`**`]` `[`*`options`*`]`

1. Command elements shown in **bold type** must be entered exactly as they appear in the text. In the previous example, the command "**sample**" and the "**+r/-r**" are shown in bold because you must type these elements exactly as given without abbreviations or mispellings.

2. A command may be followed by one or more parameters or switches. Optional parameters and switches may be entered in any order.

   a. Optional parameters or switches (sometimes called *options*) are those which are not required. They are shown enclosed in [square brackets]. You do not type the [ ] characters if you include an option on the command line. In the preceding example, *options* represents zero, one or more parameters. In the preceding example, you do not have to enter any *options*.

   b. If the switch or parameter is required, it is shown unenclosed. In the preceding example, *file-specifier* is a required parameter. Required elements should be entered in the order they appear in the syntax.

3. Each parameter and switch *must* be preceded by a space and a slash (/) character.

4. Whenever *filename* appears, it may include a disk letter and a path, if desired. If you omit a file name, the command may assume a default value as explained in this guide.

5. Unless otherwise specified, you can type commands and parameters in either uppercase or lower-case letters.

6. The word "type", as it is used in this guide, means to press a key, a sequence of keys, or a key combination and then press the ENTER key.

   The stile (|) represents the word "or". You do not type the (|). To use the preceding example, you would type-

   ```
   sample mydata.txt +r
          or
   sample mydata.txt -r
          or
   sample mydata.txt
   ```

   plus any options you choose.

7. An ellipsis (...) means the previous parameter or switch can be repeated several times. Do not type the ellipsis.

1.6 How This Manual Is Organized

This guide is organized according to Table 1-1.

CHAPTER	NAME	CONTENTS
1	Introduction	
2	MCS-1 Installation	Read this chapter if you are installing an MCS-1 tape controller.
3	SCSI Installation	Read this chapter if you are installing a SCSI host adapter.
4	*Back-It 4 LAN* Installation	Read this chapter before installing *Back-It 4 LAN* backup software.
5	Using TAPETEST.EXE	Read this chapter to learn how to test your tape system.
6	Using TAPEMENU.EXE	Running in the interactive mode.
7	Drive Selection	Read this chapter if you have more than one tape drive on the system.
8	Using TAPE2DSK.EXE	How to perform tape to disk transfers.
9	Using DSK2TAPE.EXE	How to perform disk to tape transfers.
10	Using TLABEL.EXE	Read this chapter if you are working with labeled tapes.
11	Using TCONVERT.EXE	Read this chapter if you are working with unlabeled tapes.
12	Using TAPELIST.EXE	Looking at tape data using TAPELIST.
13	Data Extraction (DEX)	Read this chapter if you want to extract data from tape files and write it to disk.
14	Using COPYTAPE.EXE	Read this chapter if you are going to copy tapes and you have more than one tape drive.
15	Using TCLONE.EXE	Read this chapter if you are going to copy tapes and you only have one tape drive.
16	Using TAR.EXE	Read this chapter if you are going to transfer data to and from UNIX-based systems via tape.
17	Using TABLEDIT.EXE	Read this chapter only if you want to install and use the translation table editing program.
18	Error Messages	Alphabetical listing of error messages by program.
19	Appendix	Miscellaneous information.

Table 1-1 Chapter Organization

2. MCS-1 Installation

2.1 General Information

This chapter tells you how to install your MCS-1 nine-track tape controller, device driver, Tapestar for DOS application programs, and tape drive. A diagnostic procedure at the end of this chapter can be used to verify that your nine-track tape subsystem is functioning properly.

The MCS-1 is a nine-track tape controller and can be installed in any IBM PC/XT/AT or 100% compatible. It can be connected to any Qualstar tape drive, allowing you to read and write nine-track tapes on your PC. If your drive is a SCSI model (i.e., 1054, 1260S, or 34XXS), you can bypass the SCSI Interface circuit board and connect the MCS-1 cable directly to the native Pertec interface.

This chapter covers the following subjects:

1. Installing the Hardware - Describes how to install MCS-1 controller and the nine-track tape drive on your system (Section 2.2).
2. Installing the MCS-1 Device Driver for DOS - Describes how to install the MCS-1 device driver (Section 2.3).
3. Installing the Application Software - Describes how to copy the application files from the distribution diskettes to your hard drive (Section 2.4).
4. Testing the Installation - Describes how to test the MCS-1 and the tape drive (Section 2.5).

2.2 Installing the Hardware

The MCS-1 is shipped configured for an I/O base address of 380H, and an interrupt channel of 15. If these settings conflict with other devices already installed in your system, you will have to either change the settings of the MCS-1 or change the settings of the device(s) causing the conflict. Qualstar recommends changing the settings for the MCS-1, because it leaves the rest of your computer undisturbed.

A listing of interrupt channels and base I/O addresses are given in Tables 2-2 and 2-3 at the end of this chapter. Channels 3 through 15 may be used on AT-type (sixteen-bit) computers, while only channels 3 through 7 may be used on XT- and PC-type (eight-bit) computers.

If you do not know which I/O addresses or interrupt channels are in use, use one of the several good system utility software packages on the market to evaluate current system usage, or use the trial-and-error method until you find settings which work.

1. Using Tables 2-2 and 2-3 at the end of this chapter, select an unused interrupt channel (between 3 and 7 if using a PC/XT or compatible), and an unused base I/O address.
2. Set the address switches on the MCS-1 for the selected base I/O address (see Figure 2-1 and Table 2-3).
3. The interrupt channel is set by software; however, there are some early versions of the MCS-1 board which have a horizontal row of pins and a jumper just above the edge connector. If you have one of these early boards, you select the interrupt channel by positioning the jumper across the desired set of pins. The factory default is 15. Later versions do not have (and do not need) this jumper.

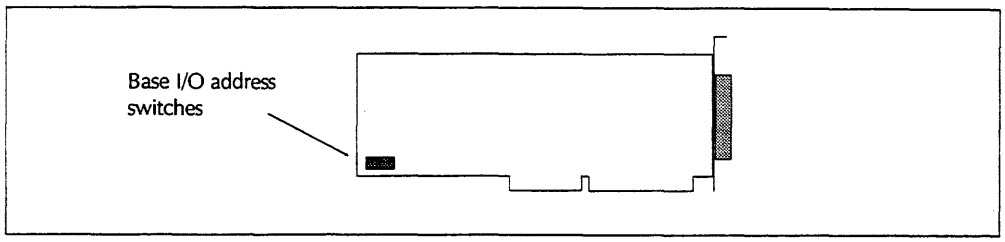


Figure 2-1 MCS-1 PCBA

4. Turn off the computer and remove the cover.
5. Find a vacant expansion slot on the motherboard. Use a 16-bit slot if one is available.
6. Remove the rear cover for the slot, and seat the MCS-1 squarely into the expansion slot connectors on the motherboard.
7. Secure the MCS-1 bracket to the rear of the computer case and replace the computer cover.
8. Connect the tape drive interface cable to the MCS-1 connector at the rear of the computer, tightening the connector jack screws ***finger-tight only***.
9. Verify that the tape drive is unplugged.
10. Connect P1 to J1 and P2 to J2 inside the tape drive. ***Orient the connectors correctly***. Pin 1 on the drive and cable connectors should be marked.

DANGER!

BEFORE PLUGGING IN THE TAPE DRIVE, VERIFY THAT THE TAPE DRIVE IS CONFIGURED FOR THE AVAILABLE AC LINE VOLTAGE. IF IT ISN'T, RECONFIGURE THE DRIVE FOR THE AVAILABLE AC LINE VOLTAGE BEFORE APPLYING POWER. REFER TO THE APPROPRIATE TAPE DRIVE USER'S GUIDE FOR INSTRUCTIONS.

CAUTION

The inside of your computer, and the host adapter board you are about to install are highly sensitive to electrostatic discharges (ESD). You should take normal ESD precautions whenever the inside of your computer is exposed and when handling the host adapter.

11. Turn on the computer, then plug in and turn on the tape drive.
12. Allow the computer to boot up and proceed with Section 2.3 to install the MCS-1 device driver for DOS.

2.3 Installing the MCS-1 Device Driver for DOS

Before you can use the MCS-1 tape controller under DOS, you must install a device driver. The name of the MCS-1 device driver for DOS is MCSTAPE.SYS. The device driver may be automatically installed using MCS1INST.EXE.

MCS1INST will do the following:

1. Asks you to enter the desired interrupt channel number and base I/O address of the MCS-1 board. The factory defaults are 15 and 380H, respectively, and may be changed to avoid conflicts.
2. Asks you to type in the letter of the disk drive your system boots from. The default is drive C.
3. Adds the following device driver line to the end of your CONFIG.SYS file:

   device= mcstape.sys i*xx* b*yyy*

   where *xx* is the interrupt channel and *yyy* is the base I/O address you have chosen.
4. Looks for the file MCSTAPE.SYS in the root directory of the specified boot device and if the file is not there, copies it from the installation diskette.

MCS1INST will not create directories or copy files other than MCSTAPE.SYS. It will not save your original CONFIG.SYS file, and it will not copy application software from the program diskettes.

2.3.1 Automatic Device Driver Installation Procedure

To let MCS1INST install the device driver, perform the following steps. If you wish to install the device driver manually, skip to Section 2.3.2.

1. Insert the diskette labeled *TAPESTAR-MCS FOR DOS* into the appropriate disk drive. In this explanation, assume that the drive is drive A.
2. At the DOS command prompt, type:

   ```
   a:\mcs1inst
   ```
3. When the installation program screen appears, choose item 1, *Installation.*
4. Answer the questions and follow the instructions on the screen.

NOTE

> Unless you have changed the address switches on the MCS-1 board or are going to use an interrupt other than 15, accept the default address of 380 and the default interrupt of 15.

5. After you accept the settings displayed on the screen, MCS1INST will modify your CONFIG.SYS file as previously described and will copy the device driver (MCSTAPE.SYS) to the root directory of the boot device. When finished, MCS1INST will display:

   ```
   Installation successfully completed!

   However, before the changes take effect, you must reboot your
   computer. Would you like to reboot now? <Y/N>
   ```

6. Type **Y**. The system will reboot. While the system is booting, look at the device driver messages displayed on the screen and verify that the following message is displayed:

```
Found 512 Kbytes of 16-bit memory, testing
- passed
Using interrupt 15, I/O address 380
```

   a. If the preceding message was displayed, the device driver was successfully installed. Go ahead and install the application software as described in Section 2.4 and then check the installation and the tape drive using the procedures described in Section 2.5.

   b. If the preceding message was not displayed, the following message (or one similar) will appear:

```
Nine track controller not found at specified base address
Driver not installed
```

   In this case, turn to Section 2.5.1 for help in resolving the problem.

2.3.2 Manual Device Driver Installation Procedure

If you feel comfortable working with DOS and would rather not have MCS1INST write on your hard disk or modify your CONFIG.SYS file, you can perform the installation manually as follows:

1. Insert the diskette labeled *TAPESTAR-MCS FOR DOS* into the appropriate disk drive.

2. Copy the file named MCSTAPE.SYS to the root directory of your boot device as in the following example:

```
copy a:\mcstape.sys c:
```

3. Add the following line to your CONFIG.SYS file:

device= mcstape.sys i*xx* **b***yyy*

   where *xx* is the interrupt channel you desire and *yyy* is the base I/O address you set in the switches on the MCS-1 board.

4. Reboot the system. While the system is booting, look at the device driver messages displayed on the screen and verify that the following message is displayed:

```
Found 512 Kbytes of 16-bit memory, testing
- passed
Using interrupt 15, I/O address 380
```

   a. If the preceding message was displayed, the device driver was successfully installed. Go ahead and install the application software as described in Section 2.4 and then check the installation and the tape drive using the procedures described in Section 2.5.

   b. If the preceding message was not displayed, the following message (or one similar) will appear:

```
Nine track controller not found at specified base address
Driver not installed
```

   In this case, turn to Section 2.5.1 for help in resolving the problem.

2.4 Installing the Application Software

TAPESTAR-MCS FOR DOS contains the files listed in Table 2-1.

```
POLYTAPE BIN        57,451 12-28-93   4:22a
BWCC     DLL       130,224 06-10-92   3:10a
TAPELIST DRV            72 09-29-93   2:03p
COPYTAPE EXE        43,700 01-10-94   4:22a
DEX      EXE        68,448 12-29-93   4:22a
DSK2TAPE EXE        38,978 12-29-93   4:22a
MCS1DIAG EXE        56,424 12-29-93   4:22a
MCS1INST EXE        46,786 02-04-93   4:22a
SEISMIC  EXE        26,457 11-09-92   1:04a
TABLEDIT EXE        89,677 09-30-92   4:22a
TAPE2DSK EXE        36,386 12-29-93   4:22a
TAPELIST EXE        30,182 12-29-93   4:22a
TAPEMENU EXE        41,062 12-29-93   4:22a
TAPETEST EXE        70,728 12-30-93   4:22a
TAR      EXE       124,250 01-10-94   4:22a
TCLONE   EXE        38,990 12-29-93   4:22a
TCONVERT EXE       178,168 12-29-93   4:22a
TLABEL   EXE        74,812 12-30-93   4:22a
MCSTAPE  SYS        15,563 12-29-93   4:22a
ATOE     TBL           256 09-24-93   1:00a
ETOA     TBL           256 12-03-92   1:00a
```

Table 2-1 TAPESTAR-MCS FOR DOS Files

To install *TAPESTAR-MCS FOR DOS* on your hard disk, simply copy all the files on the diskette to a directory or subdirectory of your choice on the hard disk. For example:

```
copy a:*.* c:\mcs
```

If the programs are to be used often, establish a way for DOS to find them by adding the drive and path name where they reside to the DOS Path command in your AUTOEXEC.BAT file. Refer to the DOS manual for instructions on using the DOS Path command.

2.5 Testing the Installation

This section describes how to test the MCS-1 controller, the installation of the device driver, and the tape drive and cable connections.

2.5.1 Testing the Device Driver Installation

To determine if the device driver is being loaded correctly, reboot the system and watch the display. When the MCS-1 device driver loads, the system displays an identifying message. The driver then looks for the MCS-1, and, if found, runs some diagnostic tests on it to determine its operational status. It is not necessary to have a tape drive online or even connected to the MCS-1 for the driver to be initialized properly. Following these diagnostics, the device driver will report whether or not it installed itself.

If everything went OK, the following message will appear:

```
Found 512 Kbytes of 16-bit memory, testing - passed
Using interrupt 15, I/O address 380
```

If errors were detected, the device driver will display a brief error message identifying the cause of the problem. The device driver will not be loaded and will you will not be able to use any of

the tape programs. The following paragraphs list some of the more common error messages, their causes, and corrective action you can take.

1. `Nine track controller not found at specified base address`
   `Driver not installed`

   CAUSE #1 - The base I/O address given on the device driver command line differs from the address actually set by the MCS-1 switches.

   FIX #1 - Either change the device driver command line in your CONFIG.SYS file, or change the address switch settings on the MCS-1. In either case, the two must agree if the device driver is to communicate with the controller.

   CAUSE #2 - The MCS-1 board is defective or not properly seated in the motherboard connectors.

   FIX #2 - Reset or replace the MCS-1 board.

2. `I/O conflict at selected address`
   `Driver not installed`

   CAUSE - The base I/O address set by jumpers on the MCS-1 is also being used by another device in the computer.

   FIX - Since no two devices in a PC can share the same base I/O addresses, the address used by one of the devices must be changed. If the MCS-1 address is changed, the corresponding device driver line in your CONFIG.SYS file must also be changed. Refer to Section 2.2 for details.

3. `Interrupt failure. Already in use?`
   `Driver not installed.`

   CAUSE - The interrupt channel use by the MCS-1 is being used by another device in the computer, or is not working correctly.

   FIX - Change the interrupt channel either of either the MCS-1 or the other device which is using the same interrupt. If the MCS-1 interrupt channel needs to be changed, the appropriate line in the device driver line in your CONFIG.SYS file must also be changed. Refer to Section 2.2 for details.

2.5.2 Using MCS1DIAG.EXE

MCS1DIAG.EXE is used to test both the MCS-1 controller and the tape drive. For this part of the test procedure, the tape drive does not need to be connected.

NOTE

Online help is available by choosing H from the main menu.

1. Go to the subdirectory in which the application programs reside and type:

   `mcs1diag`

2. If the device driver was correctly installed, an opening screen like the one shown in Figure 2-2 should appear. If so, skip ahead to Section to test the tape drive.

3. If the device driver was not correctly installed, a screen like the one shown in Figure 2-3 will appear instead, and the tape subsystem cannot be used. In this

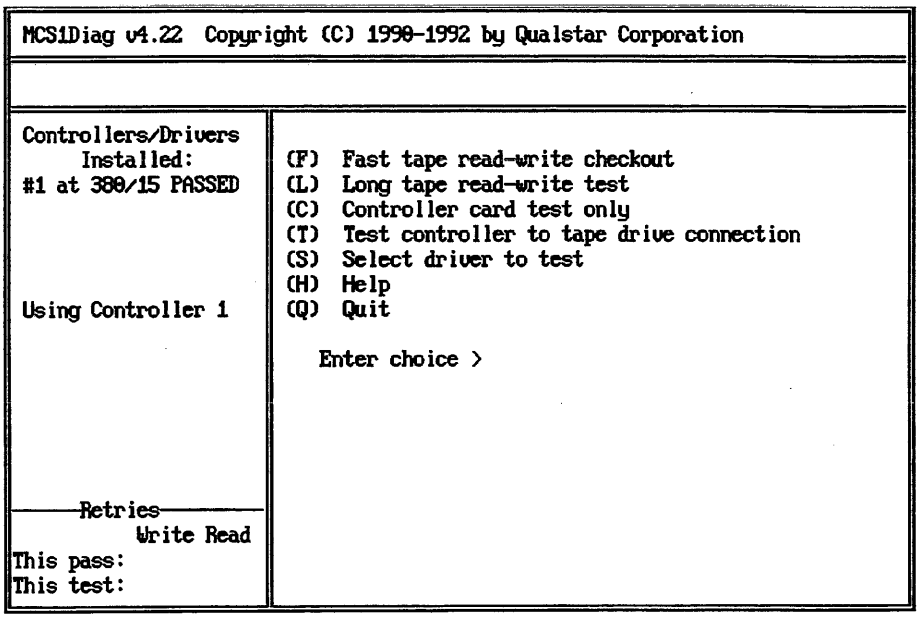


Figure 2-2 MCS1DIAG Main Screen (driver correctly installed)

case, the MCS-1 can still be tested (if it is installed) to verify that it is working. In order to test the MCS-1 at this point, the following information is required:

a. The base I/O address, set by the four switches at the lower left corner of the MCS-1 (default setting is 380_H);

b. The interrupt channel number (default is 15).

If either one of these settings has been changed, use the current settings. If the system has just been unpacked and the settings have not been changed, use the factory defaults.

4. Choosing C tests the MCS-1. At the end of the test, the following message will appear on the top line if the test was successful:

```
Controller test PASSED
```

2.5.3 Correcting MCS-1 Controller Problems

Some problems can be traced to the MCS-1, and MCS1DIAG will report any MCS-1 problems it finds. If you experience any difficulties, try the following:

1. Verify the MCS-1 board is properly installed.

   a. Remove the MCS-1 and clean the connector contacts at the bottom of the board with a pencil eraser so that they are bright.

   b. Place the MCS-1 on a flat surface and press down on all chips which are in sockets to ensure that they are well seated.

   c. Reinstall the MCS-1 in the computer, taking care that it seats fully in the expansion slot connector.

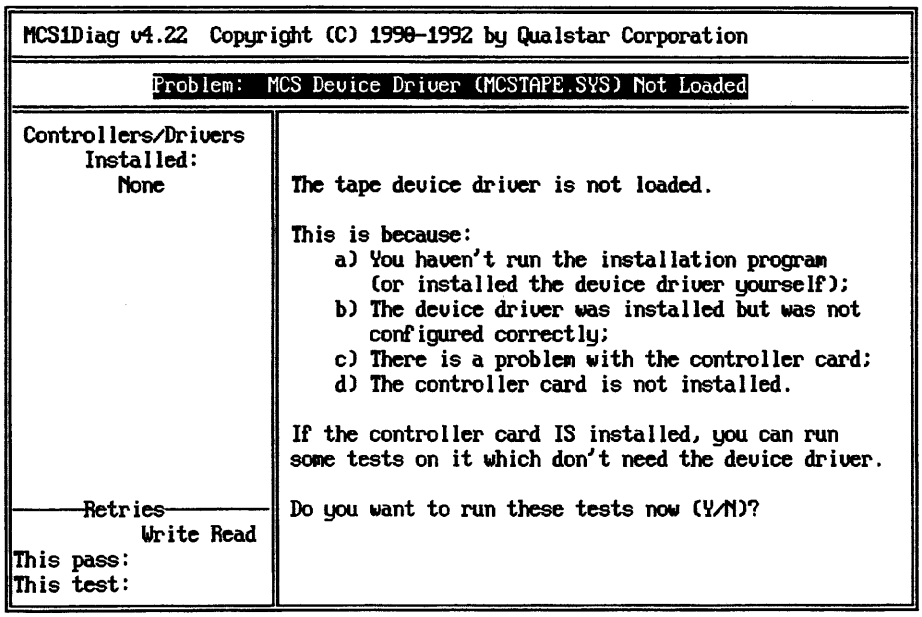


Figure 2-3 MCS1DIAG Main Screen (driver not correctly installed)

2. Verify the MCS-1 base I/O address does not conflict with another device. See Section 2.2 (page 2-1) for more information. If the test fails for any reason, MCS1DIAG will try to provide an explanation.

   a. If the message indicates there may not be a controller at the given address, double-check the base I/O address to see if the address actually set on the MCS-1 switches is the same one entered in MCS1DIAG. If it is not, run MCS1DIAG again and enter the correct address.

   b. For other remedies, turn to Section 2.5.3 (page 2-7).

2.5.4 Testing the Tape Drive

1. MCS1DIAG can also be used to test the tape drive. If you wish to do so, verify the following before running MCS1DIAG:

   a. The interface cable is connected to the MCS-1 and to the tape drive;

   b. A known good tape with a write-enable ring is mounted in the tape drive. A good tape is necessary to eliminate errors caused by defects in the media;

   c. The tape drive is online.

2. Go to the subdirectory in which MCS1DIAG.EXE resides and type:

   ```
   mcs1diag
   ```

3. The MCS1DIAG main menu will appear, showing several selections:

   a. Choosing **F** does a fast check-out. The fast check-out should be used to test the system for the first time to see if everything basically works.

b. Choosing **L** does a longer write-read test which you can customize. This is a more extensive test which should be used if you suspect problems with the tape system.

CAUTION

MCS1DIAG will overwrite and destroy any existing data on the tape; however, it will first ask permission.

2.5.4.1 Fast Check-out

With the tape loaded and ready to go (the tape must not be write-protected), answer the prompts by typing the requested keys. The fast check-out goes through several stages. At the end of each stage, the MCS1DIAG reports the results.

- If the drive passes the fast check-out test, you can be reasonably sure that everything is functional.
- If an error is detected, MCS1DIAG will stop, display an error message, and wait for your input. Turn to Section 2.5.5 for help in determining the cause.

2.5.4.2 Long Test

This option is intended to be used if the tape subsystem has problems. Sometimes a subsystem will pass the fast check-out, but will still have problems transferring data to the tape drive; Long Test can catch these problems, which may not occur until well into a reel of tape.

This test can also be used to certify reels of tape for writing by selecting the "customizing" option and telling MCS1DIAG to write all the way to the end of tape (EOT). If the drive is known to be functioning properly, this will evaluate the integrity of the media. MCS1DIAG not only reports hard (unrecoverable) tape errors, but also shows how many error retries occurred while writing or reading the tape. The lower these numbers, the better the tape (they will rarely be zero.)

2.5.5 Correcting Tape Drive Problems

Here are some of the more common tape subsystem problems:

- Dirty tape head - This is the number one source of all data-related tape problems. If tape problems are suspected, clean the tape head as described in the tape drive user's guide.
- Poor cable connections - Disconnect each interface cable connector, one at a time, and look at the cable connector pins. They should not be damaged or recessed. Reconnect and secure each connector by using the connector retaining mechanisms. Verify that P1 is connected to J1, P2 is connected to J2, and that pin 1 on each cable connector is lined up with pin 1 on each board connector on the tape drive.
- Bad tapes - Even a perfectly-operating tape subsystem cannot overcome poor quality tapes. Good tapes deteriorate after extensive use and/or storage, and the quality of a tape cannot be determined by looking at it (although it sometimes will be obvious when a tape is not good.)
- Defective cables - Sometimes reported errors can be misleading. Messages such as "possibly corrupted device driver", or "Failure with ganged drives" can also indicate a faulty interface cable.

IRQ	OTHER POSSIBLE DEVICES
3	2nd serial port (COM2)
4	1st serial port (COM1)
5	2nd parallel port (LPT2)
6	FD controller
7	1st parallel port (LPT1/PRN)
10	—
11	—
12	—
14	AT HD controller

Table 2-2 Interrupt Channel Settings

ADDRESS (hex)	SWITCH SETTINGS				OTHER POSSIBLE DEVICES
	A3	A4	A5	A6	
380	ON	ON	ON	ON	SDLC (MCS-1 Default)
388		ON	ON	ON	SDLC
390	ON		ON	ON	
398			ON	ON	
3A0	ON	ON		ON	Bisync
3A8		ON		ON	Bisync
3B0	ON			ON	Monochrome video (MGA)
3B8				ON	
3C0	ON	ON	ON		
3C8		ON	ON		
3D0	ON		ON		Color graphics (CGA)
3D8			ON		
3E0	ON	ON			
3E8		ON			3rd serial port (COM3)
3F0	ON				Diskette controller
3F8					1st serial port (COM1)
ON = Switch is ON; blank = Switch is off					

Table 2-3 Base I/O Address Settings

3. SCSI Installation

3.1 General Information

Follow the instructions in this chapter if you are installing a SCSI host adapter in your DOS system. If you are installing an MCS-1 coupler, follow the instructions in Chapter 2. This chapter tells you how to install the SCSI host adapter, the SCSI device driver, and how to attach the interface cable to a tape drive.

This chapter covers the following subjects:

1. Host Adapter Installation (UltraStor) - (Section 3.2).
2. Host Adapter Installation (Adaptec 1540B/1542B) - (Section 3.3).
3. Host Adapter Installation (Adaptec 1640) - (Section 3.4).
4. Installing *TAPESTAR-SCSI FOR DOS* - Describes how to copy the application files from the distribution diskette to the hard drive (Section 3.5).
5. Checking Out the SCSI Hardware and Software Installation - Describes how to test the SCSI tape drive and host adapter installation (Section 3.6).
6. SCSI Installation Problems - Describes problems which may occur during software or hardware installation (Section 3.7).

If you are going to be using a Qualstar 1054, 1260S, 34XXS, or TS/1000 (Tandberg 4100 QIC drive), you will need the distribution diskette titled, *TAPESTAR-SCSI FOR DOS*, (Qualstar PN 738-0053-4). If you are going to be using the LMSI 939X half-inch cartridge tape drive, use the diskette titled, *TAPESTAR-PLUS FOR DOS*, (Qualstar PN 738-0064-1).

DANGER!

BEFORE PLUGGING IN THE TAPE DRIVE, VERIFY THAT THE TAPE DRIVE IS CONFIGURED FOR THE AVAILABLE AC LINE VOLTAGE. IF IT ISN'T, RECONFIGURE THE DRIVE FOR THE AVAILABLE AC LINE VOLTAGE BEFORE APPLYING POWER. REFER TO THE APPROPRIATE TAPE DRIVE USER'S GUIDE FOR INSTRUCTIONS.

CAUTION

The inside of your computer, and the host adapter board you are about to install are highly sensitive to electrostatic discharges (ESD). You should take normal ESD precautions whenever the inside of your computer is exposed and when handling the host adapter.

3.2 Host Adapter Installation (UltraStor)

The UltraStor host adapter is designed for use in an IBM PC or 100% compatible utilizing an AT bus (ISA). Follow these instructions if your host adapter is an UltraStor 14F.

3.2.1 Package Contents

Your UltraStor package (Qualstar PN 730-0063-3) should contain:

1. One UltraStor 14F host adapter
2. One UltraStor 14F host adapter software diskette
3. One UltraStor User's Manual

The interface cable which connects the host adapter to the tape drive has a separate part number.

3.2.2 Installation Procedure

Use the following procedure to install the host adapter and device driver:

1. Unpack the host adapter and install the jumpers exactly as they are shown in Figure 3-1.

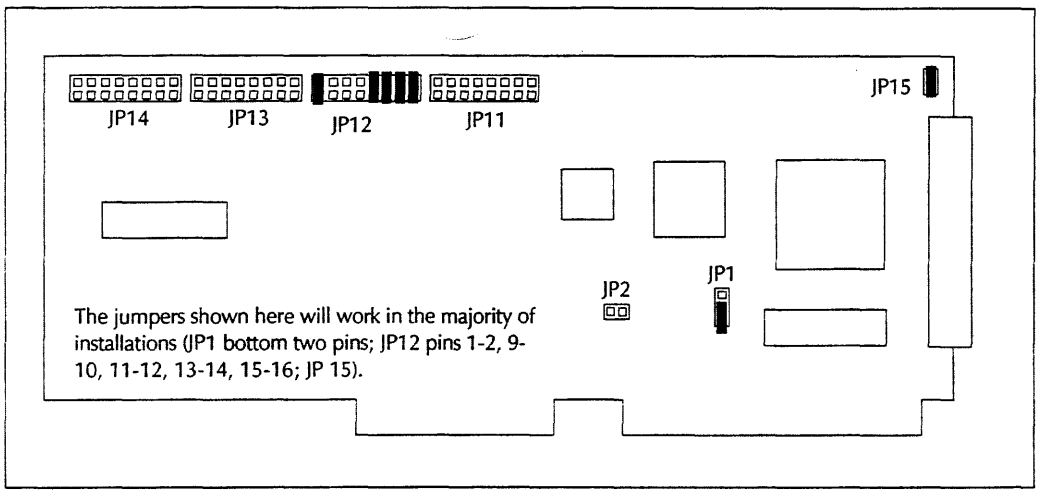


Figure 3-1 Host Adapter (UltraStor 14F) Jumper Installation

2. Power down the computer.
3. Install the host adapter in a vacant sixteen-bit slot.
4. Install the connecting cable from the host adapter to the tape drive and engage the locking tabs.
5. Verify that the tape drive SCSI ID is set to 5. Instructions on setting the SCSI ID are given in the appropriate User's Guide
6. Connect the tape drive to AC power.
7. Apply power to the system.
8. Insert the diskette labeled *ULTRA 14F ISA SCSI Host Adapter, Utilities & Device Drivers (DOS/Netware/OS2)* into drive A:.

9. Copy the file USPI14.SYS (located in the \DOS directory on the diskette) to the root directory of the start-up hard disk (C: assumed) by typing:

```
copy a:\dos\uspi14.sys c:
```

10. Add the following line to your CONFIG.SYS file:

```
device= c:\uspi14.sys
```

NOTE

> You can load this device driver high if you are using DOS 5.0. Refer to your DOS manual for more information.

11. Reboot the system to allow the updated CONFIG.SYS file to load the SCSI device driver into memory. If the host adapter and software were properly installed and are operational, the screen should present the following messages upon boot-up:

```
UltraStor ULTRA 14 SCSI Manager Device Driver Version 1.02

Scanning SCSI devices . . . done
```

NOTE

> If you remove the host adapter and do not disable the device driver line in your CONFIG.SYS file, your system will hang during boot-up.

12. If the preceding messages do not appear, or if the installation is not successful, you have a problem. Read Section 3.7 for suggestions and ideas on what the problem may be. There may be a conflict with the port address, interrupt level, or SCSI ID. The line "`No SCSI devices found!!`" will appear if the tape drive is not powered up.

13. Power up the tape drive and set the SCSI configurations parameters to their factory default values. The procedure for doing this varies according to the tape drive—refer to the appropriate user's guide for further information.

14. Install the Tapestar software as described in Section 3.5.

3.3 Host Adapter Installation (Adaptec 154X)

The Adaptec 1540B/1542B is designed for use in an IBM personal computer or 100% compatible utilizing an AT/ISA bus. Follow these instructions if using an Adaptec 154X host adapter.

3.3.1 Package Contents

Your host adapter package (Qualstar PN 738-0049-2) should contain:

1. One Adaptec 1542B host adapter
2. One Adaptec AHA-1540B/1542B Installation Guide
3. One Adaptec AHA-1540B/1542B User's Manual

The DOS ASPI driver Version 3.0 (Qualstar PN 738-0051-8) and the interface cable which connects the host adapter to the tape drive have separate part numbers.

3.3.2 Installation Procedure

Use the following procedure to install the host adapter and device driver:

1. Unpack the host adapter and install the jumpers as shown in Figure 3-2. These jumpers configure the host adapter for:

   a. SCSI ID 7;

   b. AT port address 330H;

   c. DMA channel 5;

   d. SCSI Parity enabled;

   e. Interrupt channel 11;

   f. On-board BIOS disabled;

   g. Floppy controller disabled.

   If your system requires different settings, refer to the Adaptec documentation for instructions.

NOTE

Unless a hard disk exists which relies on the host adapter on-board BIOS, Qualstar recommends that the on-board BIOS be disabled. In addition to disabling the BIOS, the floppy disk controller should also be disabled (1542 only).

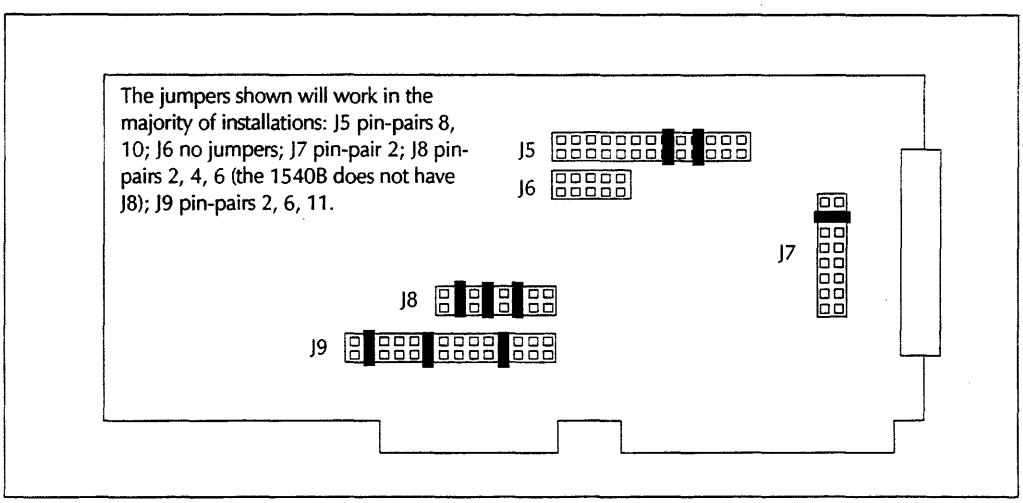


Figure 3-2 Adaptec 1542 Host Adapter

2. Power down the computer.

3. Install the host adapter in a vacant sixteen-bit slot.

4. Install the interface cable from the host adapter to the tape drive and engage the locking tabs.

5. Connect the tape drive to AC power.

6. Apply power to the computer.
7. Copy the file ASPI4DOS.SYS from the host adapter software diskette to the root directory of the start-up hard disk (C: assumed).
8. Add the following line to your CONFIG.SYS file:

```
device= c:\aspi4dos.sys /d
```

9. Reboot the system to allow the updated CONFIG.SYS file to load in the SCSI device driver. If the host adapter and software were properly installed and are operational, the screen should present the following messages upon boot-up:

```
AHA-154X/1640 ASPI Manager for DOS
Version 2.1
Copyright 1990 Adaptec, Inc.

    No SCSI devices found!!

    I/O Port Address:       330
    Interrupt Level:        11
    DMA Channel:    5
    Host Adapter SCSI ID:   7

Debug Status:   2

ASPI4DOS.SYS Installation Successful
```

10. If the preceding messages do not appear, or if the installation is not successful, you have a problem. Read Section 3.7 for suggestions and ideas on what the problem may be. There may be a conflict with the port address, interrupt level, or SCSI ID. The line "No SCSI devices found!!" will appear if the tape drive is not powered up.
11. Power up the tape drive and set the SCSI configurations parameters to their factory default values. The procedure for doing this varies according to the tape drive—refer to the appropriate user's guide for further information.
12. Install the Tapestar software as described in Section 3.5.

3.4 Host Adapter Installation (Adaptec 1640)

3.4.1 Package Contents

The Adaptec 1640 is designed for use in an IBM personal computer PS2 or 100% compatible utilizing an MCA bus. Follow these instructions if using an Adaptec 1640 host adapter.

Your host adapter package (Qualstar PN 738-0050-0) should contain:

1. One Adaptec 1640 host adapter
2. One Adaptec AHA-1640 Installation Guide
3. One Adaptec AHA-1640 Installation Diskette

The DOS ASPI driver Version 3.0 (Qualstar PN 738-0051-8) and the interface cable which connects the host adapter to the tape drive have separate part numbers.

3.4.2 Installation Procedure

Instead of option jumpers, the AHA-1640 is configured by software. Use the following procedure to install and configure the Adaptec 1640 host adapter and to install the Adaptec device driver:

1. Before starting, make a copy of your current System Reference Diskette using the DOS Diskcopy command. Use this copy for the remainder of this procedure.
2. Copy the file @0F1F.ADF from the Adaptec AHA-1640 Installation Diskette to the copy of the System Reference Diskette you just made. You may need to use drive C: as an intermediate step to effect the copy.
3. Power down the computer.
4. Install the host adapter in any vacant 16- or 32-bit slot.
5. Install the interface cable from the host adapter to the tape drive and engage the locking tabs at both ends of the cable.
6. Connect the tape drive to AC power.
7. Insert the copy of the System Reference Diskette in drive A:.
8. Switch on the computer and let it boot up.
9. When the IBM screen appears, press ENTER and follow the instructions on the screen.
10. When "Automatically configure the system? (Y/N)" appears, type N. A "main menu" will appear.
11. Select item 3, "`Set configuration`".
12. From the next menu, select item 2, "`Change configuration`".
13. The next screen will show all the system configuration information. You can navigate the items using the Up-Arrow and Down-Arrow keys and the Page Up and Page Down keys. Locate the entry for the Adaptec host adapter:

    ```
    slot# - AHA-1640 SCSI Host Adapter
    ```

14. The # sign will be the actual slot number where the host adapter is installed. Move down to the following item:

    ```
    Adapter BIOS
    ```

15. Using the F5 and/or F6 keys, change the choice for this item to the following:

    ```
    BIOS disabled
    ```

16. For all options for the Adaptec host adapter, we recommend you use the choices selected by the configuration program. If you need to change any settings, refer to the AHA-1640 Installation Guide.
17. Select Save by pressing F1.
18. Press ENTER to dismiss the Information box.
19. Exit the configuration program by pressing ESC until an Information box appears which says, "`Press Enter to restart computer.`"
20. Remove the copy of the System Reference Diskette and reboot the computer.

21. Copy the file ASPI4DOS.SYS from the host adapter software diskette to the root directory of the start-up hard disk (C: assumed).

22. Add the following line to your CONFIG.SYS file:

```
device= c:\aspi4dos.sys /d
```

23. Reboot the system to allow the updated CONFIG.SYS file to load in the SCSI device driver. If the host adapter and software were properly installed and are operational, the screen should present the following messages upon boot-up:

```
AHA-154X/1640 ASPI Manager for DOS
Version 2.1
Copyright 1990 Adaptec, Inc.

    No SCSI devices found!!

    I/O Port Address:       330
    Interrupt Level:        11
    DMA Channel:    5
    Host Adapter SCSI ID:   7

Debug Status:   2

ASPI4DOS.SYS Installation Successful
```

24. If the preceding messages do not appear, or if the installation is not successful, you have a problem. Read Section 3.7 for suggestions and ideas on what the problem may be. There may be a conflict with the port address, interrupt level, or SCSI ID. The line "No SCSI devices found!!" will appear if the tape drive is not powered up.

25. Power up the tape drive and set the SCSI configurations parameters to their factory default values. The procedure for doing this varies according to the tape drive—refer to the appropriate user's guide for further information.

26. Install the Tapestar software as described in Section 3.5.

3.5 Installing TAPESTAR-SCSI or TAPESTAR-PLUS FOR DOS

TAPESTAR-SCSI FOR DOS and *TAPESTAR-PLUS FOR DOS*, are both shipped on 5-1/4 inch and 3-1/2 inch diskettes, each containing the files listed in Table 3-1.

To install *TAPESTAR-SCSI* (or *TAPESTAR-PLUS*) *FOR DOS* on the hard disk, simply copy all the files on the diskette to a directory or subdirectory of your choice on the hard disk. For example:

```
copy a:*.* c:\tsscsi     or     copy a:*.* c:\tsplus
```

If the programs are to be used often, establish a way for DOS to find them by adding the drive and path name where they reside to the Path command in your AUTOEXEC.BAT file. Refer to the DOS manual for instructions on using the DOS Path command. This will allow the Tapestar software to be accessed from any location.

3.6 Testing the Tapestar System

The installation should be tested before using the Tapestar programs. Qualstar recommends the following procedure to test the installation:

1. Load a scratch tape onto the drive to be tested. (A scratch tape is a known good tape which can be written on without regard to preserving data which may be

```
POLYTAPE BIN        57,451 12-28-93   4:22a
BWCC     DLL       130,224 06-10-92   3:10a
TAPELIST DRV            72 09-29-93   2:03p
COPYTAPE EXE        43,700 01-10-94   4:22a
DEX      EXE        68,448 12-29-93   4:22a
DSK2TAPE EXE        38,978 12-29-93   4:22a
MCS1DIAG EXE        56,424 12-29-93   4:22a
MCS1INST EXE        46,786 02-04-93   4:22a
SEISMIC  EXE        26,457 11-09-92   1:04a
TABLEDIT EXE        89,677 09-30-92   4:22a
TAPE2DSK EXE        36,386 12-29-93   4:22a
TAPELIST EXE        30,182 12-29-93   4:22a
TAPEMENU EXE        41,062 12-29-93   4:22a
TAPETEST EXE        70,728 12-30-93   4:22a
TAR      EXE       124,250 01-10-94   4:22a
TCLONE   EXE        38,990 12-29-93   4:22a
TCONVERT EXE       178,168 12-29-93   4:22a
TLABEL   EXE        74,812 12-30-93   4:22a
MCSTAPE  SYS        15,563 12-29-93   4:22a
ATOE     TBL           256 09-24-93   1:00a
ETOA     TBL           256 12-03-92   1:00a
```

Table 3-1 TAPESTAR-SCSI FOR DOS Files

on it.) Preferably, a new, high-quality tape should be used. In that way, the tape drive rather than the tape will be tested. The reel or cartridge must be write-enabled.

2. Place the drive online and type: `tapetest`
3. After initializing, a screen similar to the one in Figure 3-3 should appear.

```
Qualstar Corporation
Tapestar (C) Test Utility v 2.108

Type "help" for help.

Initializing ... done.

 #      drive          type     ID:LUN    controller     type    (#)
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 1:  Qualstar 3410    9-track     5:0        Adaptec       SCSI    (1)

==>
```

Figure 3-3 TAPETEST Startup Screen Example

4. If an error message is displayed, turn to Section 3.7 for help.
5. Type: `sel #`

   replacing the # sign with the number of the drive to be tested (use 1 if you have only one drive).
6. Type the following command:

   `mode format` *`tape-format`*

replacing *tape-format* with the highest supported density (i.e., 1600, 3200 or 6250) if the tape drive is a Qualstar 1054, 1260S or 34XXS, or QIC-24 if the tape drive is a Tandberg 4100.

a. If an invalid entry was made, TAPETEST will display the correct command syntax and valid choices.

b. If a correct entry was made, Tapestar will display:

```
set mode returned "OK"
```

7. Start the test by typing: `test`

   TAPETEST will write 1,000 blocks of data, each block consisting of a 12,000 byte test pattern. After the blocks are successfully written, TAPETEST rewinds the tape, reads the blocks and checks them for data errors.

   a. TAPETEST may be canceled by pressing ESC.

   b. If the read pass is successful, the drive is considered operational.

   c. If an error occurs, it is reported and the test stops; turn to Section 3.7 for assistance.

8. The default test is fairly short, writing about twelve megabytes of data to tape. To change the length of the test, type: `help test`

   at the TAPETEST prompt. Tapestar's interactive online help explains how.

9. To test another drive, go back to the sel # step, select the next drive to be tested, and continue as previously described.

10. To exit Tapestar and return to DOS, type: `quit`

3.7 Troubleshooting Tapestar Problems

This section does not cover problems caused by a broken host adapter. Refer to the documentation which came with the host adapter if problems are experienced. If an error message is displayed while running Tapestar, read this section for help on tracking down and solving the problem. Locate the specific error in the following sections and follow the advice given for that error. Information about technical assistance can be found at the front of this manual.

1. `UNIT ATTENTION` - This is not actually an error, but rather a SCSI *condition* indicating a tape was just loaded into a drive, or that the drive was just powered up or reset.

   FIX: Retry the test.

2. `TAPE WRITE-PROTECTED` - The tape used for the test does not have a write enable ring installed, or the tape drive is configured to a file-protect or read-only mode.

   FIX: Unload the tape and install a write enable ring, or configure the tape drive for writing. Retry the test.

3. `DRIVE NOT READY` - The tape drive is not online and ready, or it is busy completing another operation (e.g., rewinding).

   FIX: Make sure the drive is online, ready and not busy, then retry the test.

4. `BAD/INVALID TAPE MEDIUM` - This could be one of two problems:

a. The wrong density/format was selected for the tape being tested;

b. There is a problem with the tape drive or the tape.

FIX: Make sure the correct density/format has been selected using the Mode command in TAPETEST (chapter 5). If this does not correct the problem, see the following problem.

5. `WRITE ERROR` or `BAD/INVALID TAPE MEDIUM` - These could indicate a problem either with the tape drive or the tape. Verify the tape is known to be good—preferably a new brand name tape.

   FIX: Retry the test using another tape. Also, try cleaning the drive's read/write head with a cotton swab and 91% isopropyl alcohol or with a head cleaning cartridge (cartridge tape drives only).

6. `INTERNAL ERROR` - This is a catch-all error condition which can have many causes, most of them drive- or controller-related. This error may indicate low-level problems with the SCSI host adapter.

   FIX: Review the set-up of the controller being used for the test:

   a. Are the host adapter's switches and jumpers set correctly?

   b. Is the device driver used by the host adapter correctly installed, with the correct options?

   c. Is there a hardware or software conflict between the host adapter, or its device driver, and any other devices, drivers or software being used in the computer?

   d. Is the tape drive correctly connected to the computer?

   e. Is the tape drive properly terminated?

   f. Is the drive's SCSI device ID unique (not used by another SCSI device)?

7. `NO ADAPTERS FOUND` - This message means that Tapestar searched for adapters but did not find any. This can mean one of several things:

   a. No adapters known to Tapestar are installed in the computer.

   b. A host adapter is installed, but the device driver needed to communicate with that adapter is not.

   c. Both the host adapter and its device driver are installed, but the host adapter is not "alive", due to cabling, tape drive, or other problems.

   FIX: There are many possible causes for this problem. The best advice is to go back over the installation process for the adapter in question. Refer to the relevant installation guide for more detailed information.

8. `NO DRIVES FOUND` - This message indicates that the Tapestar program could not find any drives which it could identify as connected to any adapters. This can mean any of several things:

   a. No drives are actually connected to any host adapters.

   b. At least one drive is connected to a host adapter, but the drive is not powered up or online.

c. At least one drive is connected, powered up and online, but that drive cannot be found because of cabling, host adapter or other problems.

d. The tape drive is not being recognized by the program as a valid tape drive.

FIX: There are too many reasons for this problem to cover here. If the drive is correctly connected, powered up and online, the most likely cause is incorrect termination. Refer to the appropriate User's Guide for the particular tape drive in question.

4. Back-It 4 LAN Installation

This chapter will guide you through the installation of *Back-It 4 LAN* on your system. Use the instructions in this chapter in place of those in the *Back-It 4 LAN* manual from Gazelle.

4.1 Installing Back-It 4 LAN

Back-It 4 LAN has been designed so that installation is quick and simple. BKSETUP, included on your software diskette, is used to configure *Back-It 4 LAN* so that it will function properly with your system.

To install *Back-It 4 LAN* on your computer, follow these steps:

1. Insert the *Back-It 4 LAN* diskette into floppy drive A:. If installing from drive B:, substitute B: for A: in these instructions.
2. Type `a:` and press ENTER.
3. Type `bkinstal` and press ENTER.

NOTE

> You do not have to delete anything if you want to reinstall *Back-It 4 LAN* in the same subdirectory. BKINSTAL will overwrite the duplicate files and add the new files.

4. The opening screen of the install program describes the steps you will complete as you install *Back-It 4 LAN*. Read through this introductory information and then press any key to continue, or press ESC to abort.
5. Follow the instructions on each screen.
6. As the files are being copied to the hard disk, their names will be displayed.
7. After the files have been copied, the Color Configuration screen will appear. Press ENTER to accept the default colors for now. You can always change the colors later on.
8. Upon completion of the entire installation process, you must reboot the computer for the changes to take effect.

4.2 Defining a Tape Device

Before you can use *Back-It 4 LAN* with your tape drive, you must tell *Back-It 4 LAN* that the tape drive exists (i.e., create a definition). To do this, use the following procedure.

NOTE

> Use the cursor control keys to highlight an item. You may also highlight an item by typing its highlighted letter. To "back out" of the dialog boxes and menus, press ESC.

1. Change directories (CD) to the directory which contains the *Back-It 4 LAN* files.

2. Start *Back-It 4 LAN* by typing `bk41an`.
3. Highlight the Configure menu and press ENTER.
4. Highlight **Utilities for Tape** and press ENTER. **Select Definition** will be highlighted and a Select box will appear with **Microtech** highlighted.
5. Press ESC to dismiss the Select box.
6. Highlight **Create Definition**. A Select box appears with **MICROTEC.GSD** highlighted.
7. Press ENTER. A Configure box will appear, providing choices for creating a valid device definition. A description of the functions of each of these choices is given in the Menu Option Description box on the lower half of every screen.
8. Highlight **Device Driver ID Number?** and press ENTER. A Select box will appear.
9. Highlight the number of your tape drive and press ENTER.

NOTE

For SCSI tape drives, this is the SCSI ID number. The SCSI ID number is set to 5 on Qualstar SCSI tape drives before shipping. The ID number of other Qualstar tape drives is set to 0 before shipping.

10. Highlight **Controller Number?** and press ENTER.
11. Highlight **Controller #1** and press ENTER.
12. Highlight **LUN Number?** and press ENTER.
13. Highlight **LUN 0** and press ENTER.
14. Multitasking is set to OFF by default. If you want to turn it on, highlight **Multitasking ON** and press ENTER. The choices "Yes" and "No" appear. Highlight your choice and press ENTER.
15. Your definition is now complete. Save it by pressing F2. A Configure Name box will appear.
16. Type in a name which specifically identifies the tape drive you just defined.
17. Press ENTER. The device definition will be saved and the Configure Menu will open with **Select Definition** highlighted.
18. Press ENTER. **Microtech** will still be highlighted.
19. Make sure the tape drive is powered up and online.
20. Highlight the device name you just created and press ENTER.
21. An "Accessing drive" message will appear while the tape drive is initialized.
22. After the drive has been initialized, the Configuration menu will appear with **Utilities for Tape** highlighted. Highlight **Tape Select** and press ENTER.
23. A Tape Drives Supported box will appear, listing the supported devices. Highlight the device you just created and press ENTER.

24. Save your tape device selection by highlighting **Save Current Configuration** and pressing ENTER.

25. Press ESC to return to the main menu.

You are now ready to use *Back-It 4 LAN*.

5. Using TAPETEST.EXE

TAPETEST is a program which lets you test the tape subsystem. It can be used to help track down tape system problems while on the telephone with Qualstar support personnel. TAPETEST can also be used to execute commands to a tape drive from a DOS batch file.

5.1 Running TAPETEST Interactively

TAPETEST can be run in an interactive mode, during which the program prompts you to type in commands, one at a time, and then executes them as they are entered. To run TAPETEST interactively, type

```
tapetest
```

After the program begins, simply type in the desired command. You should begin by selecting a tape drive. See Section 5.3.25 for information about the Select command. After selecting the drive, type the desired command and any required or optional parameters. When you are finished with TAPETEST, to quit the program, type

```
q
```

5.2 Running TAPETEST in the Batch Mode

TAPETEST can also be run in a batch mode in which TAPETEST executes commands in the sequence you type them on the command line. You can also use a text editor to create a text-only file mixing TAPETEST commands with DOS commands.

Syntax: **TAPETEST** [command] [*;command*] [...]

Example:

```
tapetest sel 1;test;unload
```

The preceding command line selects the first available tape drive, runs the standard write/read test on the drive, then unloads the tape before returning to the DOS command prompt.

As in the interactive mode, the first command should be a Select command, even if there is only one tape drive on the system. After executing the last command on the command line, TAPETEST exits automatically—you do not need to include a Quit command.

If a command fails, TAPETEST will display an error message and will exit without executing any additional commands. TAPETEST sets the DOS ERRORLEVEL exit code to indicate whether the last command executed successfully. An ERRORLEVEL of 0 indicates success, and a value of 1 indicates failure. This exit code can be tested to allow branching within DOS files by using the DOS IF ERRORLEVEL... STATEMENT.

NOTE

TAPETEST will rewind the tape when exiting unless you tell it not to (by using the NOREWIND command).

5.3 TAPETEST Commands

The remainder of this chapter is an alphabetical list of all TAPETEST commands. Some commands take additional parameters (such as block size, etc.) which are shown after the command keyword. Parameters shown in [square brackets] are optional, the rest are required.

5.3.1 ASCII

Syntax: **`ascii`**

The ASCII command sets the display mode used by the Read and Show commands. Data will be displayed as ASCII text (see also EBCDIC).

5.3.2 Clear Return

Syntax: **`clrrtn`**

This command clears a tape drives internal error retry counters. These counters are maintained by the drive's own hardware and firmware, and keep cumulative counts of all read or write errors which have occurred since the current tape operation (reading, writing, or verifying) was started.

This command is only used for diagnostic or technical support purposes.

5.3.3 Copy

Syntax: **`copy`** *`source dest [#blocks|blocksize]`*

The Copy command copies data from a tape drive or disk file to another drive/file. It can be used in situations where the usual TAPESTAR file transfer programs cannot be used for one reason or another (such as reading damaged tapes). Data can be reblocked with this command.

1. The Copy command will transfer data until one of the following occurs:
   a. A filemark (tape mark) is read from the source tape drive.
   b. The source is a disk file and end-of-file is hit.
   c. EOM (end-of-medium) is hit on the source tape drive.
   d. EOD (end-of-data) is hit on the source tape drive.
   e. The block limit count for the destination drive has expired.
2. source specifies a tape drive or disk file in one of the following forms:
   a. Specify a tape drive by giving the drive's number from the drive list (use the List command to get the list).
   b. Specify the currently selected drive by using an asterisk (*).
   c. Specify a disk file by giving the file name—for example:

      ```
      c:\junk\recover.dat
      ```

3. destination specifies a tape drive or disk file in one of the following forms:
   a. Specify a tape drive by giving the drive's number from the drive list (use the List command to get the list).
   b. Specify the currently selected drive by using an asterisk (*).
   c. Specify a disk file by giving the file name—for example:

      ```
      c:\junk\recover.dat
      ```

   If the destination is a disk file, the file will be created, and any existing file by the same name will be completely overwritten. The only limitation is that you cannot use DOS filenames which consist only of numeric characters.
4. *#blocks\blocksize* - If the destination is a tape drive which is running in variable block mode, then you must give the destination block size as the third

parameter.

If the destination is a tape drive running in fixed block mode, then the third parameter becomes an optional block limit count. If specified, this will limit the transfer to the specified number of blocks written on the destination drive.

The Copy command will transfer data until one of the following occurs:

- A filemark is read from the source tape drive
- The source is a disk file and an EOF character is read
- EOM or EOD is detected on the source drive
- The block limit count for the destination drive has been reached.

Example:

```
copy 2 * 16384
```

The preceding Copy command copies from drive 2 to the currently selected drive. The destination drive is in the variable block mode, and the output block size will be 16K.

5.3.4 EBCDIC

Syntax: **`ebcdic`**

This command sets the display mode used by the Read and Show commands. Data will be interpreted as EBCDIC text (see also ASCII).

5.3.5 Exercise

Syntax: **`exe`**

This (undocumented) command is for internal Qualstar use only.

5.3.6 Filemark

Syntax: **`FMK`**

Writes a single file mark at the current tape position.

5.3.7 Help

Syntax: **`help`** [*`command`*]

This command, when given with no arguments, displays a list of all available commands. For help on a specific command (e.g., Test), specify the name of the command.

5.3.8 Information

Syntax: **`info`** [*`drive`*]

This command displays information about various drive capabilities for the specified drive. If no drive is specified, information about the currently selected drive is displayed. *drive* is the drive's number from the drive list (use the List command to get the list).

5.3.9 Inquiry

Syntax: **`inq`** *`controller-type drive-addr lun controller#`*

This issues a SCSI Inquiry command to a particular drive (notice that you don't need a selected a drive to use this command). It can be used to inquire any drives connected to any controller which is recognized by TAPESTAR. This command can be used to determine if a drive is

operating correctly, but is not a drive which TAPESTAR recognizes. (TAPESTAR uses the drive's Inquiry ID string to identify the drive.)

1. *controller-type* - Either:
   a. **ad** (Adaptec SCSI host adapter), or
   b. **mcs** (MCS-1 tape controller).
2. *drive-addr* - SCSI ID or the address of the desired tape drive.
3. *lun* - is the SCSI LUN of the desired tape drive.
4. *controller#* - Controller number, starting at 0, of that particular type of controller.

Example:

```
inq ad 5 0 0
```

The preceding command sends a SCSI Inquiry command to the drive at SCSI ID 5, LUN 0 on the first Adaptec SCSI controller in the system.

5.3.10 Last Error

Syntax: **`lasterr`**

The Last Error command displays internal data areas which store information about the last tape error which occurred.

5.3.11 List Drives

Syntax: **`list`**

The List command displays a list of all drives in the system currently known to exist to TAPESTAR as a result of the last drive scan (either at the start of the program, or after the Rescan command).

The currently selected drive (if any) is indicated by an asterisk (*) next to the drive entry.

5.3.12 Load Tape

Syntax: **`load`**

For drives with an attached autoloader mechanism (for example, 3480 drives), this command will load the tape at the current autoloader position. The autoloader can be positioned with the POS command.

In order to use this command, the drive must be capable of supporting an autoloader: Check by using the Info command.

5.3.13 Log

Syntax: **`log on|off`** *`[logfile]`*

Log On starts logging the results of TAPETEST commands to the specified log file.

logfile is the name of the logfile. The log file is an unformatted text file which can be easily viewed or printed. If the named file already exists, it will be overwritten. If it doesn't exist, TAPETEST will create it. If you do not specify a name, TAPETEST will create the file POLYFILE in the current directory.

Log Off closes the log file and can be specified at any time. The log file is automatically closed when you quit TAPETEST.

5.3.14 Mode

Syntax: **mode** [*modes-to-set* [*mode-parameter*]]

The Mode command either displays current mode information for the currently selected drive, or changes the current mode. To display the mode information, give the Mode command alone, without any parameters.

modes-to-set - Specifies which mode to set. Some modes let you specify an optional mode parameter.The following mode values are valid:

1. **fix** [*blocksize*] - Set fixed block mode. The blocksize parameter sets the size of the block on tape.
2. **var** - Set variable block mode.
3. **rbuf on|off** - Enable or disable read buffering.
4. **wbuf on|off** - Enable or disable write buffering.
5. **speed high|low** - Set tape drive speed.
6. **format** [*format*] - Set drive format/density, where *format* is one of:
   a. **800, 1600, 3200, 6250**
   b. **QIC-525, QIC-1000, QIC-2000**
   c. **DAT, DAT-DCLZ, DAT-LZS**

5.3.15 No Rewind

Syntax: **norewind**

The No Rewind command toggles the "no-rewind" mode for the currently selected tape drive. If this mode is enabled, then this drive will not be automatically rewound when the program ends. (This mode is disabled by default each time TAPETEST is run.)

5.3.16 Position

Syntax: **pos** *tape#*

For drives with an attached autoloader mechanism (for example, 3480 drives), this command will position the autoloader to the specified tape number. The tape will not be loaded; use the Load command for this.

To use this command, the drive must be capable of supporting an autoloader. Check by using the Info command.

5.3.17 Quit

Syntax: **quit** or **q**

Exits back to DOS (can also be entered as simply "**q**". This command is not needed in "batch" mode.

5.3.18 Read

Syntax: **`read`**

Reads the next block of data from the currently selected drive, shows the first 128 bytes of the block in a "hex dump" format. If any exceptions occur (FMK, EOD or EOT detected), a corresponding error message will appear.

After the Read command, data is kept in the internal read buffer, and can be viewed with the Show command, or verified with the Verify command.

NOTE

The largest block (variable block mode) or group of blocks (fixed block mode) which can be read is limited by the current I/O buffer size. See the Set Buffer command in Section 5.3.27 for more information.

5.3.19 Ready

Syntax: **`ready`**

This command issues a SCSI Test Unit Ready command to the currently selected tape drive to see if it is ready to accept a new command.

5.3.20 Rescan

Syntax: **`rescan`**

This command rescans all tape controllers and host adapters recognized by TAPESTAR for any recognized tape drives. After the rescan, the current drive list is displayed. Note that the new drive list can be different than the previous list if new drives are found during the rescan, or if previously found drives are no longer found.

5.3.21 Reset

Syntax: **`reset`**

This does a SCSI Reset on the currently selected drive.

5.3.22 Retension Tape

Syntax: **`reten`**

On drives which are capable of this operations, initiates a tape retension sequence (valid for QIC drives only). During a retension sequence, the tape is moved to the end and then rewound.

To use this command, the drive must be capable of performing a Retension command. This can be checked by using the Info command.

5.3.23 Rewind

Syntax: **`rew`**

Rewinds the tape on the currently selected tape drive.

5.3.24 Reverse Read

Syntax: **rread**

This command is identical to the Read command described in Section 5.3.18 except that the read is done in reverse. Only certain tape drives support this operation. Use the Information command described in Section 5.3.8 to get a list of the current tape drive's capabilities.

5.3.25 Select

Syntax: **sel** *drive#*

Selects the desired drive as the currently selected drive. Almost all other TAPETEST commands require that the drive be selected first. *drive#* is the drive's number from the drive list (use the List command described in Section 5.3.11 to get the list).

5.3.26 Select Partition

Syntax: **selpart** *partition#*

This command selects a tape partition on drives which support partitioning. Partitions are numbered starting with zero.When successfully executed, the specified partition is the current active partition and all dta read or written will be within this partition.

Use the Mode command described in Section 5.3.14 to show partition information (nunmber and size of partitions) for the current tape.

5.3.27 Set Buffer

Syntax: **setbuff** *buffersize* [**kb**|**k**]

TAPETEST sets up a memory buffer when transferring data to or from a tape drive. The size of this buffer is adjustable by both you and by TAPETEST. When you select a drive, TAPETEST automatically sizes the buffer to a reasonable value for that tape drive.

The memory buffer size determines the largest block (or set of fixed length blocks) which can be read from or written to a tape drive. Therefore, if you need to read or write a larger block than the current buffer size, or if you get a "size" error when reading or writing, use the Set Buffer command to change the buffer size.

buffersize - Specified in bytes, unless k or kb is specified, in which case the size is in kilobytes. 1k = 1kb = 1024 bytes.

5.3.28 Set Partition

Syntax: **setpart** *partition_size* [**kb**|**mb**]

The Set Partition command creates a new partition on a tape in the current drive (assuming the drive supports the operation). Tape partitions are subdivisions which can be read or written independently. The tape drive will alert you when you reach the end of a partition even though the tape is not at its physical end. Partitions are only supported on certain drive types (8mm and DAT).

5.3.29 Show

Syntax: **show** [**e**|**a**] [*start-offset/block#*]

This displays the data placed into the internal data buffer by the most recent Read command. Data is shown in screens, with a pause between each screen, at which point you can look at the next line, screen, or cancel the display. If the tape drive which was read from was operating in fixed block mode, then several blocks will be read and shown; otherwise, a single block will be read and shown.

The display is in standard "hex dump" format. Each line starts with the byte offset of the first byte in the line (in decimal, starting at zero), followed by sixteen hexadecimal bytes, followed by the character representation of the data in either ASCII or EBCDIC. The current interpretation of the character data can be set by use of the EBCDIC or ASCII commands. The default is ASCII.

The first option (**e** or **a**) selects the character set to be used for displaying data (on the right side of the hex dump display)—EBCDIC or ASCII. The second option selects a byte offset or block number to start the display from, depending on whether the tape was read in variable block or fixed block mode.

5.3.30 Space

Syntax: **`space files|blocks|eod`** `[amount-to-space]`

Positions the currently selected tape relative to the current tape position:

- **files** - Space by *amount-to-space* files
- **blocks** - Space by *amount-to-space* blocks
- **eod** - Space to the end-of-data on tape

For files and blocks, positive numbers will space towards EOT, and negative numbers will space towards BOT from the current position.

5.3.31 Special

Syntax: **`special`**

This command is for internal Qualstar use and is not documented here.

5.3.32 Status

Syntax: **`stat`**

For tape drives connected to an MCS-1 controller, this shows the immediate status indicators (RDY, ONL, etc.). This command does not operate with SCSI devices.

5.3.33 Summarize Tape

Syntax: **`summ`**

Summarizes an entire tape by showing block sizes and block counts, as well as filemarks and EOD markers on the tape.

5.3.34 Test Tape

Syntax: **`test`** `[#blocks] [blocksize]`

This command writes a test pattern to the currently selected tape drive, then rewinds the tape and reads the blocks back, comparing the data read to the original pattern. The drive will be rewound back to the load point. This can be used for a quick write-read test of any tape drive. This test will overwrite existing tape data.

Caution

This test will overwrite whatever data is on the tape.

By default, the test consists of 1,000 blocks of 12,000 bytes each (for drives operating in variable block mode,) or 1,000 blocks of 512 bytes each (for drives operating in fixed block mode.) Other block counts and block sizes can be substituted on the command line.

For example, to write 5,000 blocks of 16K bytes each, type:

```
test 5000 16384
```

5.3.35 Unload Tape

Syntax: **`unload`**

The Unload command rewinds and unloads the currently selected tape drive, if that drive supports an Unload function (returns an error otherwise.)

In order to use this command, the drive must be capable of performing an Unload command. Check for the capability of REW_UNLD by using the Information command described in Section 5.3.8.

5.3.36 Verify Tape

Syntax: **`verify`**

Verifies the next block of data on the currently selected tape drive against the data already in TAPETEST's read buffer. This buffer is filled by use of the Read command.

In order to use this command, the drive must be capable of performing a Verify command. Check for the capability of FF_VERIFY by using the Information command described in Section 5.3.8.

5.3.37 Write

Syntax: **`write`** `[#blocks] [blocksize]`

This command will write the specified number of blocks of the specified block size. The largest block which can be written is limited by the current I/O buffer size. See the Set Buffer command in Section 5.3.27.

If the drive is operating in the fixed length mode, blocks are written at the current fixed block size, and you must specify the number of block.

If the drive is operating in the variable length mode, you must specify the block size. You may specify the number of blocks (the default is one).

Each block contains the following:

```
block *****:pattern
```

***** is the block number. `pattern` is a simple ascending and repeating pattern of bytes (8-bit unsigned numbers), starting at the block number modulo 256. These pattern bytes increase from 0 to hex FF, then repeat from 0. Block numbers range from 0 to 65,535, then repeat from 0.

6. Using TAPEMENU.EXE

TAPEMENU allows you to read or write a tape in an interactive mode without having to remember any confusing command-line parameters and without having to constantly refer to the manual. It performs the most common tape operations using a question-and-answer approach.

TAPEMENU performs these most-often used tape operations:

- Transferring files from disk to tape
- Transferring files from tape to disk
- Displaying the contents of a tape (labeled or unlabeled)

TAPEMENU only allows you to use the first tape drive installed in the computer (not a problem if you only have one tape drive). To access other drives, you must use the other TAPESTAR programs in this package.

To run TAPEMENU, type

```
tapemenu
```

TAPEMENU does not require any command-line parameters or options. When TAPEMENU starts, it displays a screen like that shown in Figure 6-1.

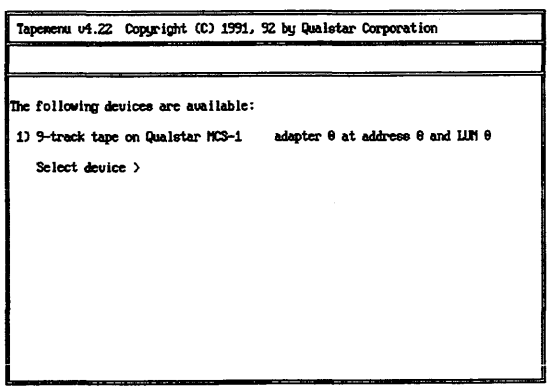


Figure 6-1 TAPEMENU Opening Screen

Select the drive you want to use by typing its number, then press ENTER. The main TAPEMENU screen will then appear like that shown in Figure 6-2.

The TAPEMENU screen contains the following parts:

1. **Current Active Keys** - Lists all valid keys at any given point (aside from the normal "typewriter" keys).
   a. **Previous** - Displays the previous question.
   b. **Cancel** - Cancels any partial entry and remains at the current question.
   c. **Restart** - Returns to the initial, "top level" menu.
2. **Current Selection List** - Lists all selections (answers to questions) which have been made so far. To go back to a previous question, look at this list to see how far back to go.

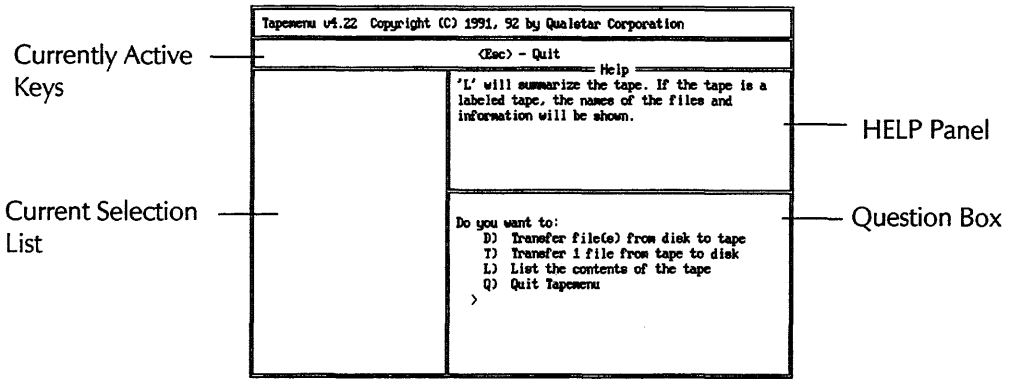


Figure 6-2 TAPEMENU Main Screen

3. **HELP Panel** - Contains text explaining the current question in more detail.
4. **Question Box** - This asks the "current question," and receives the user response.

6.1 How TAPEMENU Accepts Keystrokes

TAPEMENU accepts keystrokes in two different ways, depending on the response required:

a. If the only possible responses are single keystrokes (like "Y/N"), that keystroke is typed without pressing ENTER afterwards.

b. If the response can contain more than one keystroke, such as a file name or a number for record size, then the response must be followed by pressing ENTER.

NOTE

Any of the keys listed in the *Current Active Keys* panel can be used at any point during a response.

6.2 Troubleshooting TAPEMENU Problems

If an error occurs while using TAPEMENU, read this section for help on tracking down and solving the problem. Locate the specific error in the following sections and follow the advice given for that error.

1. **MESSAGE** - `Could not allocate buffer for screen.`

   CAUSE - Not enough free conventional DOS memory. TAPEMENU may be running from a "shell" within another program, or something else is using up conventional memory (too many device drivers, memory-resident software, system crash, etc.)

FIX -

a. Make sure sufficient conventional memory is free. TAPEMENU and the programs which it executes require about 256K free conventional memory.)

b. Do not run TAPEMENU from a "shell-to-DOS" function within another program;

c. Reboot the computer if a previous program has corrupted or not released memory.

2. **MESSAGE** - `Could not execute <program-name>.`

   CAUSE - The indicated program could not be loaded because of insufficient free memory.

   FIX - Check the previous problem for causes and solutions.

3. **MESSAGE** - `Program file <program-name.exe> not found.`

   CAUSE - The indicated program could not be located because:

   a. It may be on the disk but not in the right subdirectory;

   b. It may be on the disk and in the right subdirectory but without the correct PATH statement;

   c. It is not on the disk at all.

   FIX - Verify the correct files have been copied to the correct locations on the hard drive, and that the PATH statement is correctly set up (if needed).

4. **MESSAGE** - `Could not open batch file '<batch-file-name>' for writing.`

   CAUSES -

   a. An invalid path name was specified as part of the indicated batch file;

   b. "Illegal" characters appear in the indicated batch file.

   c. No free disk space on the drive where the batch file is to be written.

   FIX -

   d. Verify that the indicated file name contains the path name (if any), and that all characters in the file name are legal DOS filename characters.

   e. Verify you have enough space free on the disk where the batch file will be written.

5. **MESSAGE** - `Error writing to the indicated batch file.`

   CAUSE - Not enough free disk space on the drive where the batch file is to be written.

   FIX - Verify you have is enough free space on the disk where the batch file will be written.

6. **MESSAGE** - `<Program-name> failed <reason>.`

   This message shows that an error occurred in the program which TAPEMENU executed.

Reasons -

a. `Tape drive not ready` - The tape drive was not online and ready when the tape operation was started.

b. `Tape error` - An error occurred while accessing the tape drive.

c. `Disk error` - An error occurred while reading or writing a disk file. The most likely cause is a full disk when writing, or a floppy disk error.

d. `Memory error` - Not enough conventional memory for the program to allocate buffers or other work space.

e. `Tape device driver not loaded` - The device driver (MCS-TAPE.SYS or ASPI4DOS.SYS) is not properly installed, or there is a problem with the MCS-1 controller or SCSI host adapter.

f. `Canceled` - The tape program was canceled by pressing ESC.

g. `Too many file names` - (DSK2TAPE only) The limit of total number of disk filenames, or total length of filenames, was exceeded. (Since DSK2TAPE expands "wild-carded" filenames, this could happen if a wildcard specifier was given which matched a lot of filenames.)

h. `No matching files found` - (DSK2TAPE only) No files on disk matched any of the specified names (wild-carded or not).

i. `File exists` - (TAPE2DSK only) The specified file to be created already exists on the disk, and the choice to overwrite existing files was not selected.

7. Drive Selection

This chapter explains how to specify tape drive(s) for TAPE2DSK, DSK2TAPE, TAPELIST, COPYTAPE, TCLONE and TAR.

If only one tape drive is attached to the computer, all Tapestar programs will use that drive. If more than one tape drive is installed, Tapestar will automatically use the first tape drive found unless a drive has been specified on the command line.

NOTE

> The "first" tape drive in SCSI systems is the one with the lowest SCSI ID. Refer to the user's guide for your drive for instructions on hot to set the drive address or SCSI ID.

To select a tape drive other than the first tape drive if you have more than one, include the following option on the command line for all of the TAPESTAR programs listed above except COPYTAPE and TCLONE:

drive=*drive-specifier*

COPYTAPE and TCLONE use these variations:

source=*drive-specifier*, and **dest=***drive-specifier*

7.1 Drive Specifier

The *drive-specifier* parameter can take two forms. The first form is:

9t|qic|3480[**:***drive#*][**:ad|:mcs**][**:***controller#*]

There are no spaces between any parts of the drive specifier. This form specifies:

- A drive type;
- A drive nlumber (not address) (if you have more than one drive of that type);
- A controller type;
- A controller number (if you have more than one controller of that type).

The second form is:

9t|qic|3480[**::***drive-address*][**:ad|:mcs**][**:***controller#*]

This form is like the preceding except the *drive address* is preceded by *two* colons as shown. This form specifies a drive type, a drive address, a controller type and controller address. If you use a drive specifier, you ***must*** specify the drive-type. All other parts are optional.

1. **9t|qic|3480** - Specifies the type of drive to be used: 9-track, 1/4-inch cartridge drive, or 3480 1/2-inch cartridge drive
2. *:drive#* - Chooses a second, third, fourth, etc., drive, if more than one nine-track drive is installed. If a drive # is not specified, the first drive matching the specified drive type will be used.
3. **:ad|mcs** - Specifies which controller to use if Tapestar recognizes more than one controller in the computer. The choices are:

a. **ad** - Adaptec SCSI host adapter. This covers *all* SCSI host adapters which use the ASPI device driver.

b. **mcs** - MCS-1 nine-track tape controller

4. *:controller#* - Specifies which controller (of the type specified by *:controller*) to use, if you have more than one controller of that type installed (for example, two MCS-1 controllers, or two Adaptec 1640 SCSI host adapters).

7.2 Examples of Drive Specifiers

The following examples assume you are using TAPE2DSK.EXE, but can be used with any of the programs listed at the beginning of this chapter.

1. To select the first nine-track tape drive found, type

```
tape2dsk drive=9t
```

2. To select the second QIC drive found, type

```
tape2dsk drive=qic:2
```

3. To select a nine-track tape drive at SCSI target ID 5, type

```
tape2dsk drive=9t::5
```

4. To select the first nine-track tape drive connected to a host adapter, type

```
tape2dsk drive=9t:1:ad
```

8. Using TAPE2DSK.EXE

The TAPE2DSK program copies a single tape file to disk, starting at the current tape position. It can read labeled and unlabeled tapes, fixed and variable length blocks, and both ASCII and EBCDIC data.

8.1 Determining Tape Data Type

To obtain correct results using TAPE2DSK, one must first know what kind of records the source tape contains, the record length, and the character set. If the record format is unknown, use the following questions as a guide to determine what is on the tape and how to correctly use TAPE2DSK.

1. Is the tape labeled or unlabeled?
2. Is the tape block length fixed or variable?
3. Is the tape data written in ASCII or EBCDIC?
4. Does the tape file have fixed length records?
5. What should the resulting disk file be like?

NOTE

Tapestar includes programs which allow tapes to be examined to determine what kind of data they contain. These programs are described in Chapters 11 and 12. If uncertain of the tape format, these tools can help decipher "mystery" tapes.

8.1.1 Labeled and Unlabeled Tapes

Both types are common. Most tapes created on mainframes (large computer systems) are labeled tapes. One clue that a tape is labeled is an accompanying printed listing showing the names of datasets (files) on the tape. Because unlabeled tapes do not have dataset names recorded on the tape, an unlabeled tape listing usually lists each file by size (number of blocks or records) only.

When working with labeled tapes, refer to Chapter 10.

8.1.2 Fixed or Variable Length Blocks

This can be quickly determined if the tape has been identified as having fixed length records or if a record layout of the tape file is available. Otherwise, the tape can be examined as described in Chapters 11 and 12.

Fixed length blocks are the general rule with magnetic tape files; however, a few special formats use variable length blocks. TAPE2DSK can handle two kinds of variable length block formats:

1. If the tape contains variable length blocks, use the **/TB** transfer mode (see Section 8.2).
2. If the tape contains IBM variable length blocks, use the **/TI** transfer mode (see Section 8.2).

If the tape format is something else, or if the format is not known, use the **/TR** ("raw") transfer mode (see Section 8.2).

8.1.3 Tape File Format

Here, two questions need to be answered:

1. Is the tape written using ASCII or EBCDIC character sets?

   Sometimes this information will be written on an accompanying information sheet about the tape, or on the tape itself. If the character set is not indicated, it can sometimes be deduced from other aspects of the tape:

   a. IBM OS/VS or IBM DOS/VSE labeled tapes are usually written in EBCDIC.

   b. ANSI labeled tapes are usually written in ASCII.

   c. Tapes created on Unix-based computer systems are usually written in ASCII.

   d. Tapes created on an IBM system (mini or mainframe, not a PC) are usually written in EBCDIC.

   If the tape is EBCDIC, use TAPE2DSK's **/E** parameter to translate to ASCII. But beware! EBCDIC-to-ASCII translation only works if the tape file contains no packed or binary data. Refer to the following item to determine if this is true.

2. What types of fields do the tape records contain?

   There are three basic data types in tape files. If a record description for the file to be transferred is available, it may contain the answer to answer this question.

   a. **Text** - This includes any data which is in a readable format—names, addresses, etc. It also includes any numeric data which is readable as a series of digits (0-9), as opposed to numeric data in one of the following two forms:

   b. **Packed numeric data** - Sometimes called *packed decimal.* This space-saving format for numeric data looks like "garbage."

   c. **Binary numeric data** - This is the most efficient way to store numeric data, especially for large numbers. Like packed data, it is not readable.

Files containing packed or binary data can be transferred using the DEX (Data EXtractor) program. This utility converts data from files with fixed length records field-by-field. See Chapter 13 for information.

NOTE

Only text data can be translated. If a tape file contains either packed or binary data in addition to text data, then the data cannot be translated from EBCDIC to ASCII. If there is packed or binary data in a tape file which also includes ASCII text, then the file can be transferred with no translation. However, the resulting disk file may not be usable without further processing to convert the packed or binary data to a usable form.

8.1.4 Desired Disk File Structure

There are three choices of how the new disk file created with TAPE2DSK will look, depending on the requirements of the application which will be using the file. The disk file structure is determined by specifying a particular transfer mode on the command line.

1. **Formatted variable length records** - This means that the disk file will consist of records of (possibly) varying lengths, where the end of each record (line) is marked by a delimiter. This is the format for most plain-text files on the PC. All of TAPE2DSK's transfer modes except **/TR** will produce this type of disk file.

NOTE

Variable length indicates that the records have delimiters. They can actually all be the same length and still be considered variable length.

2. **Formatted fixed length records** - Fixed length means that all records are the same length and there are no delimiters at the end of each record (unlike variable length records). If the tape file has fixed length records, use either the **/TV** or **/TT** transfer modes (use the latter to reduce the size of the disk file by trimming trailing spaces and nulls).

3. **Unformatted data** - This is a "raw dump" of the tape data to a disk file. Use this format when reading a tape file with fixed length records into a database or other program on the PC which can handle files with fixed length records. To produce an unformatted disk file, use the **/TR** transfer mode.

If you are not sure how the tape data is formatted, transferring the data using the "raw" transfer mode will always capture all of the tape data to a disk file; however, the resulting file may not be in a form suitable for a particular application (word processor, spreadsheet, database, etc.)

8.2 TAPE2DSK Command Line Parameters

TAPE2DSK is a command line driven program and takes its orders from you on the DOS command line. The command line syntax for TAPE2DSK is:

```
tape2dsk /ttransfer-mode [filename] [options]
[drive=drive-specifier]
```

8.2.1 TAPE2DSK Transfer Mode

The *transfer-mode* parameter tells TAPE2DSK how to read and format the tape data, and is required for all modes of transfers. The transfer mode identifies the type of tape records being read and how they will be written to disk. The following are valid values for *transfer-mode*:

1. **B** - indicates that the tape contains variable length blocks. This means that the tape is written with variable length blocks, each tape block consisting of one record. Using this transfer mode, each tape block is written as one record to the file, with carriage return/line feed (CR/LF) delimiters added.

2. **I** - indicates that the tape contains IBM variable length records. Such records have special block and record prefixes, containing the block byte count and record byte count, respectively. This transfer mode will read such a tape file and produce a disk file with standard ASCII variable length, CR/LF delimited records.

3. **R** - means *raw transfer*. Data is simply copied from tape to disk with no processing. No delimiters are added or stripped, and no record padding is done.

   EBCDIC-to-ASCII translation may be specified with this mode of transfer (see the /E option). This transfer mode corresponds to the raw transfer mode of DSK2TAPE.

4. **T** - specifies reading fixed length tape records and writing them to disk with delimiters added. In addition, each record is trimmed of any trailing spaces or nulls (binary 0s) at the end of the record. As a result, the disk file will consist of true variable length ASCII records, each with CR/LF delimiters. The record length must be specified using the **/R***record-size* option.

5. **V** - specifies reading fixed length tape records and writing them to disk with delimiters added. Each record on disk will have a CR/LF delimiter at the end. The record length must be specified using the **/R***record-size* option. Other than adding delimiters (and optional EBCDIC-to-ASCII translation), no record processing is done with this mode of transfer.

When specifying that the tape has fixed length records (transfer mode **V** or **T**), then the actual tape block length must be an even multiple of the given record length, with no leftover bytes. If this is not the case, an error message will be displayed, and the transfer will be terminated.

8.2.2 TAPE2DSK Filename

filename specifies the name of the destination disk file. If this file already exists, TAPE2DSK will request confirmation before overwriting it, unless you use the **/O** option. If you do not specify a file name, the tape data will be written to "standard output," normally the screen of the PC. The drive and path names may be specified as part of the file name.

8.2.3 TAPE2DSK Options

options contains additional instructions for TAPE2DSK Each option *must* be preceded by a space and a forward slash. The following are valid values for *parameters*:

1. **A** - causes TAPE2DSK to add data to the end of an existing output file, rather than to overwrite it (create it from scratch).

2. **CR***#recs* - specifies the maximum number of tape records to transfer to disk. If specified, the transfer will end either when this many records have been read or when a filemark is encountered. If not specified, then all records up to the next filemark (or end-of-tape) will be transferred.

   This option can only be used if the transfer mode is not **/TR**).

3. **E** - specifies that EBCDIC-to-ASCII conversion will be performed on the tape data. This conversion is not checked for validity and should not be used on binary or packed-decimal data.

4. **N** - inhibits rewinding the tape after the transfer is done. It not specified, the tape will be rewound.

5. **O** - tells TAPE2DSK to overwrite any existing disk file named by *filename* on the command line without asking permission. Otherwise, confirmation will be requested and a chance to stop the transfer before overwriting the file.

   Overwriting the disk file without asking permission is useful for batch files which must run with no human interaction.

6. **R***record-size* - specifies the length of fixed length records, and this parameter must be specified when using the **/TV** or **/TT** transfer mode. It is not used for any other transfer mode.

7. **SF***#files* - specifies the number of tape files to skip before starting the transfer. For example, to transfer the third file on a tape when the tape is at BOT, use **/SF2** to skip the first two files.

8. **SB***#blocks* - specifies the number of tape blocks to skip before starting the transfer. Block skipping takes place after any file skipping. To transfer the fourth block in the file in the previous example, use **/SB3** in addition to the file skip option.

9. **XE***translation-file* - Loads a custom translation table to be used in place of the built-in EBCDIC-to-ASCII translation table. You will need to use this option if you are transferring data from a tape made on a machine which uses a nonstandard version of the EBCDIC character set, or if you have other special data translation needs. Information about creating custom translation tables is given in Chapter 17.

8.2.4 TAPE2DSK Drive Specifier

The **drive=***drive-specifier* parameter selects one of several tape drives if more than one is connected to the computer. It is not required if only one tape drive is connected. Refer to Chapter 7 for information on selecting drives for TAPESTAR programs.

8.3 Reading Files from Labeled Tape

Labeled tapes have special blocks (labels) which contain information used by the system reading the tape. These labels must be skipped in order to transfer data files correctly. The labels are separated from data files by filemarks, which make the labels themselves short tape files.

One of TAPE2DSK's options, **/SF***#files*, specifies a number of files to skip before starting the transfer, allowing any file on a labeled or unlabeled tape to be located.

Even though the three standard labeled tape formats are different, they can be treated identically when used to locate files. The following formula gives the number of tape files to skip in order to transfer data file **N** (**N** being 1 for the first file, 2 for the second, etc.):

FILES-TO-SKIP = (N - 1) X 3 + 1

To transfer the first data file, skip one file; to transfer the 2nd data file, skip four files, etc.

8.4 Handling Variable Length Blocks

Most tapes are written with fixed length blocks; however, two tape formats are currently used in which the block length is variable.

NOTE

The record length for variable length transfers cannot be specified. If the data is in EBCDIC, use the **/E** option to translate it to ASCII.

- **Variable length block tapes** - This format is not commonly-used. On such a tape, each block is a separate data record and may vary in length. To transfer a file from this kind of tape, use the **/TB** transfer mode on the TAPE2DSK command line. This will create a separate output record from each tape block read, with appended CR/LF delimiter.

- **IBM variable length tapes** - This more common variable length block format has variable length blocks containing control words at the beginning of each block, and at the beginning of each record within the block. These control words identify the length of the following block or record. When working with such a tape, specify the **/TI** transfer mode on the TAPE2DSK command line. This will read the tape and write each record

in the tape file as a single variable length record (with appended CR/LF delimiter) to the disk file.

NOTE

IBM variable length tapes can be spotted by looking at a "hex dump" of several tape blocks. If the blocks are different sizes (not a requirement), and if the value of the hex "word" in the first two bytes of the block equals the block length, then the tape is probably written in the IBM variable length format.

8.5 TAPE2DSK Examples

In the following examples, unless otherwise noted, the resulting disk file is always TAPE.DAT, the first (or only) tape drive will be used, and the tape is rewound after the transfer.

1. `tape2dsk tape.dat /tr`

   Transfers the first file on an unlabeled tape.

2. `tape2dsk tape.* /tr /n`

   Transfers all files on an unlabeled tape. The resulting disk files are named TAPE.001, TAPE.002, etc. Does not rewind the tape upon completion (**/n**).

3. `tape2dsk tape.dat /tr /sf1 /e`

   Transfers the second file on an unlabeled tape and translates from EBCDIC to ASCII (unformatted disk output).

4. `tape2dsk tape.dat /tt /r80`

   Transfers the first file on an unlabeled tape, converts fixed length 80-byte records to variable length records and trims spaces at the end.

5. `tape2dsk tape.dat /tt /r80 drive=9t:2`

   Same as the preceding example, but will read from the second nine-track tape drive.

6. `tape2dsk tape.dat /tt /sf1 /r250 /e`

   Transfers the first file from a labeled tape, translates to ASCII, converts fixed length 250-byte records to variable length records and trims spaces at the end.

7. `tape2dsk tape.dat /tb /sf3`

   Transfers the fourth file from an unlabeled tape with variable length blocks, making a disk file with variable length records.

8. `tape2dsk tape5.dat /tb /sf3 drive=qic:3`

   Transfers the fourth file from an unlabeled tape with variable length blocks, making a disk file with variable length records. The data will be read from the third QIC drive.

9. `tape2dsk tape5.dat /tv /sf13 /r120 /e`

   Transfers the fifth and sixth files from a labeled tape, translates to ASCII, converts fixed length 120-byte records to variable length records, and does not trim spaces at the end.

9. Using DSK2TAPE.EXE

DSK2TAPE copies one or more files from disk to unlabeled tape. It can read both variable length and fixed length file formats, and it always creates tapes with fixed length blocks.

NOTE

DSK2TAPE cannot be used to create labeled tapes. To write a labeled tape, use the TLABEL program described in Chapter 10.

9.1 Determining the Disk Record Format

To successfully transfer files using DSK2TAPE, you must first know the following:

1. Does the disk file contain fixed or variable length records?
2. In what character set and format is the disk file written?
3. What character set should be used to create the tape file?

9.1.1 Fixed and Variable Length Disk Records

Most files on a PC have variable length records. This means that not only can each record be a different length, but also that each record has *delimiters* at the end, to tell programs reading the file where each line ends. If a file displayed by using the DOS Type command is recognizable and readable, it probably contains variable length records. If the file looks garbled, with readable text scattered all over the screen, it may be a fixed length file without delimiters.

1. **Fixed Length Disk Files** - DSK2TAPE has one mode for transferring disk files with fixed length records to tape. Choose this mode by specifying the **/TR** transfer mode on the command line. This "raw" transfer mode simply copies the disk file to tape in fixed length blocks. When transferring fixed length disk files, you can use the **/E** option to translate ASCII data on disk to EBCDIC data on tape.
2. **Variable Length Disk Files** - DSK2TAPE has three transfer modes for transferring variable length disk files. While all produce the same results on tape (with one minor difference), they differ in how they handle records which exceed the specified tape record length. For all of these modes, you *must* specify the length of the tape record to write using the **/R***record-size* option. Each of the following parameters must be preceded by a space and a slash.
    a. **TS** - will *split* any disk records which exceed the specified tape record length, with the excess characters going into a new tape record. No data will be lost in the transfer if any records exceed the stated record length.
    b. **TT** - will *truncate* any disk records which exceed the specified tape record length. Any excess characters will be lost (this applies to the tape file only—the disk file is not altered.). Data will be lost in the transfer if any records exceed the stated record length.
    c. **TE** - will display an *error* message and stop the transfer if any disk records exceed the specified tape record length.

When variable length records are written to tape, incoming records which are shorter than the specified record size will be padded out to the full size. The default pad character is a space (blank) in the character set being used, but you can change it with the **/Z** option. See Section 9.2 on page 9-3.

9.1.2 Disk File Format

The two character sets in use today are ASCII and EBCDIC. Most text files (those which consist of readable text) created on a PC are ASCII. In fact, you can generally be assume that any file created on a PC will be ASCII. Only files which come from some outside source (i.e., from mainframes or minicomputers) are likely to be EBCDIC.

There are three basic data types found in disk data files. If a record description for the file to be transferred exists, it may reveal the data type:

1. **Text** - This includes any data which is in a readable format (names, addresses, etc.). It also includes any numeric data which is readable as a series of digits (0-9), as opposed to numeric data.
2. **Packed numeric data** - Sometimes called *packed decimal.* This space-saving format for numeric data looks like "garbage" when printed or viewed on the screen.
3. **Binary numeric data** - This is the most efficient way to store numeric data, especially for large numbers. Like packed data, it is not readable.

NOTE

Only text data can be translated. If the disk file contains either packed or binary data in addition to text data, then the data cannot be translated from ASCII to EBCDIC. If there is packed or binary data in a disk file which also includes text in the proper character set (either ASCII or EBCDIC), you can transfer the file with no translation.

If you do not know the disk file type, you can transfer the data into a tape file by using the "raw" transfer mode (**/TR**).

9.1.3 Output Tape Character Set

Chances are that the file being transferred to tape is written in ASCII. Sometimes there will be explicit instructions concerning what character set should be used to create the tape file. In other cases, you must make an educated guess based upon what can be determined about how and on what computer system the tape will be used:

1. If the tape is to be used on a UNIX-based computer system, it should be written in ASCII.
2. If the tape is to be read on an IBM system (mini- or mainframe, not a PC), chances are the tape should be written in EBCDIC.
3. If the tape is to be written in EBCDIC, use DSK2TAPE's **/E** option to translate from ASCII to EBCDIC.

NOTE

ASCII-to-EBCDIC translation only works if the tape file does not contain any packed or binary data.

9.2 The DSK2TAPE Command Line

DSK2TAPE is a command line driven program and takes all of its orders from you on the DOS command line.

Syntax: **dsk2tape /t***transfer-mode files* **/b***blocksize* [*options*] [**drive=***drive-specifier*]

9.2.1 Transfer Mode Parameter

The *transfer-mode* parameter tells DSK2TAPE how to read and format the disk data, and is *required* for all DSK2TAPE transfers. Valid values for *transfer-mode* are:

1. **R** - Does a raw transfer from disk to fixed length tape records. No reformatting is done, except to merge records together into tape blocks (known as *blocking*). Data may be translated into EBCDIC using the **/E** option. This transfer mode corresponds to the raw transfer mode of TAPE2DSK. Use this transfer mode to transfer fixed length disk records to fixed length tape records.

2. The following three transfer modes can be used to transfer variable length disk records to fixed length tape records. In each of these modes, you must specify **/B***blocksize* and the **/R***record-size* option.

   a. **S** - specifies to read variable length records from the disk file, write fixed length tape records, and split any disk records which exceed the specified record length.

   b. **T** - specifies to read variable length records from the disk file, write fixed length tape records, and truncate (throw away excess characters) any disk records which exceed the specified record length.

   c. **E** - specifies to read variable length records from the disk file, write fixed length tape records, and to end the transfer and display an error message if any disk records exceed the specified record length.

9.2.2 Files Parameter

The *files* parameter specifies the name of one or more files to transfer to tape. *You must specify at least one name.* You can enter several names on the command line, including wildcard file names and names of subdirectories. You must separate the names by spaces, and names may not contain embedded spaces. If you specify a subdirectory name, then all files in that subdirectory will be copied to tape (but not files in subdirectories of that subdirectory).

If a file matches more than one wildcard specifier or name, that file will only be copied once. For example, if JUNE.* and *.DAT are specified, the file JUNE.DAT will only be copied once to tape.

9.2.3 Blocksize Parameter

The **/B***blocksize* parameter specifies the length of the tape blocks to be written (as a number of bytes per block). You must choose the block size because DSK2TAPE has no way of doing so. If you are reading variable-length records from disk and writing them as fixed-length records on tape (all transfer modes except **R**), *the block size must be an exact multiple of the specified record length.*

For all transfer modes, any records remaining at the end of a transfer will be written as a short block at the end of the tape file (standard practice). You may want to check with whoever is going to read the tape to see if they require a certain block size. In general, choose the largest block size possible, within reason. The generally accepted upper limit on block size is 32,760 bytes.

9.2.4 DSK2TAPE Options

The *options* parameter provides additional information about the transfer. Each option must be preceded by a space and a slash character. Valid values for *options* are:

1. **E** - Specifies that ASCII-to-EBCDIC conversion be performed on the tape data. As conversion is not checked for validity, it should not be used on binary or packed-decimal data.

2. **N** - Specifies not to rewind the tape after the transfer is done. The tape will be left positioned just after the last filemark written. The default action is to rewind the tape after the file transfer is done.

3. **R***record-size* - The logical record size must be given for all transfer modes (see above) except **R**. the specified tape block length must be an even multiple of this number.

4. **XA***translation_table* - Loads a custom translation table to be used in place of the built-in ASCII-to-EBCDIC table. You must use this option if you are transferring data to a tape which will be read on a machine which uses a nonstandard version of the EBCDIC character set, or if you have other special data translation needs.

5. **Z***pad-char* - For transfer modes of **/TS**, **/TT** and **/TE**, this specifies what character will be used to pad short input records. *pad-char* is given as the decimal value of the ASCII character to be used; the default is the space character (ASCII 32).

6. **1** - (Number one) If specified, only a single filemark will be written to tape after the last disk file is transferred. The default action is to write two filemarks, indicating the logical end-of-tape. Use this option, along with the **/N** option, if more files are going to be appended to the tape after the transfer.

7. **format=***tape-format/density* - Specifies a tape format or density other than the default or current one.

   a. For QIC tape drives, valid entries for *tape-format/density* are **QIC-11**, **QIC-24**, **QIC-120**, **QIC-150**, **QIC-320**, **QIC-525**, and **QIC-1000.**

   b. For nine-track tape drives, valid entries for *tape-format-/density* are **800** (NRZI), **1600** (PE), **3200** (DPE), and **6250** (GCR).

   c. Other types of tape drives (i.e. 3480) only have one possible format/density which can be used.

NOTE

Unlike the other options listed here, the **format=** option does *not* take a slash before it, and there must be no spaces on either side of the equal sign.

9.2.5 DSK2TAPE Drive Specifier

The **drive=***drive-specifier* parameter is explained in Chapter 7.

9.3 DSK2TAPE Examples

1. `dsk2tape my.ego /b16384 /tt /n`

   This example copies the file MY.EGO containing fixed length records to tape in 16K blocks. The tape will not be rewound after the transfer.

2. `dsk2tape *.dat *.log /b5120 /ts /r128`

   This example copies all files ending with the extensions .DAT and .LOG in the current directory to tape, with a tape block length of 5,120 bytes. The files are ASCII text (variable length records). Any records which may be longer than this will be split (**/TS**).

3. `dsk2tape year1.dat /b4000 /ts r80 /z33`

   This example copies the text file YEAR1.DAT tape as 80-byte fixed length records, 50 records to a block. Short records will be padded with exclamation point characters (" ! ", whose decimal ASCII value is 33).

10. Using TLABEL.EXE

The TLABEL program is used to create labeled tapes and to copy disk files to those tapes. It creates the proper tape labels in any of the three most widely used tape formats—IBM OS/VS, IBM DOS/VSE, and ANSI. You can run TLABEL two ways:

1. **Interactively** - your choices are made by filling in fields in a menu screen.
2. **Batch mode operation** - all choices are passed to the program via the DOS command line. In this mode, the program copies the disk file(s) to tape, then ends and exits to the DOS prompt. To use TLABEL in this mode, a settings file must first be created by running TLABEL interactively.)

NOTE

> TLABEL does not allow the selection of a particular tape drive. The program will automatically select the first tape drive found which supports variable length blocks. This limitation is irrelevant if only one tape drive is available.

10.1 Running TLABEL Interactively

Syntax: **`tlabel`** *`[settings-file]`*

If the settings file you specify exists, TLABEL will configure itself according to the options in that file.

When TLABEL's main screen appears (similar to the one in Figure 10-1), fill in all the needed items on this screen. Another screen of supplemental tape header label options is available by pressing Function key F5, and is similar to the one in Figure 10-2 on page 10-4. To transfer the disk file to tape, press F10.

```
TLABEL version 4.02 Copyright (C) 1989-1991 by Qualstar Corporation

F1 = General Help      F2 = Field-Specific Help   F3 = Save Settings
F5 = Change Screens    F10 = Write Tape           ESC = Quit

Disk Filename [                                                    ]

          Options                          Volume Header Information

Label Type              [IBM DOS/VSE]   Volume ID                [000001]
Translate Data to EBCDIC      [YES]     Owner ID                 [       ]
Remove CRLF Delimiters        [YES]
Will another file follow      [NO ]
Block Length                  [     ]          Header 1 Information
Record Length                 [     ]
                                        Dataset ID          [           ]
                                        Creation Date            [91255]
                                        Expiration Date          [91255]

STRING FIELD: type normally to enter data.
```

Figure 10-1 TLABEL Main Screen

NOTE

Some of the fields in the main screen *must* be filled in. If any of these required fields are blank, TLABEL will stop and wait until they are filled in before proceeding.

10.2 TLABEL Keyboard Usage

TLABEL uses the following keys to navigate the screen and enter choices:

1. **UP and DOWNARROW** - The cursor up and down keys move one field backward and forward in the screen respectively.
2. **ENTER** - Same as cursor-down, moving to the next screen field.
3. **F1** - Displays general help.
4. **F2** - Brings up a help panel specific to the current screen location.
5. **F3** - Saves TLABEL's settings to a file for later use.
6. **F5** - Toggles between the main screen and a second screen of tape label options. The fields in this second screen are described in Section 10.4 on page 10-3.
7. **F10** -Initiates the tape transfer.
8. **ESC** - In a type-in entry field (for example, the file name field), this key cancels any partial entry and restores the previous contents; otherwise, ESC ends the program.

10.3 Main Screen Choices

Before writing files to tape, certain fields on the TLABEL screen must be filled in. Other fields may also be needed, depending on what the reader of the tape expects to find. When the TLABEL main screen first appears, all required fields will be blank.

10.3.1 TLABEL Main Screen Fields

1. `Label type` - selects what type of labeled tape will be written, and determines the content and sequence of tape labels. There are three choices, covering the most common labeled tape formats:

   a. IBM OS/VS (EBCDIC)

   b. DOS/VSE (EBCDIC)

   c. ANSI (used by many systems which handle data in ASCII).

   Choose the format which the reader of the tape expects to see. If ASCII data is expected, ANSI is probably the correct format.

2. `Translate Data To EBCDIC?` - This choice is automatically set depending on the selected label type, but it can be overridden. If set to **YES**, all data written to the labeled tape file will be translated from ASCII to EBCDIC (with no checking done on validity). Otherwise, data is written to tape "as is."

NOTE

Do not use translation if the file to be transferred contains binary or packed-decimal data.

3. `Remove CR/LF Delimiters?` - Set this to **YES** when copying a text file in which the lines are delimited by a CR/LF pair. Most text files on a PC fall into this category. If set to **YES**, then-

   a. All CR/LF characters in the tape file will be removed;

   b. The record length must be long enough to hold the longest record in the disk file, not including the CR/LF pair. If not, any characters in the disk record longer than the specified record length will be not be copied;

   c. Any disk records shorter than the specified record length will be padded to the record length with spaces in the chosen character set (ASCII or EBCDIC).

4. `Will another file follow?` - If only transferring one file, select **NO**. To transfer more than one file, select **YES**. After each file is transferred, the name of the next file to be transferred can be entered until all files have been transferred.

5. `Block length:` - Fill this field with the desired tape block length in bytes. If working with fixed length records, the block length must be an even multiple of the record length.

6. `Record length:` - If working with fixed length records, enter the record size in bytes here. If transferring unformatted data with no defined record size, make this number equal to the block length.

7. `Volume ID:` - This field identifies the tape volume and can consist of any six characters. To assure readability by the target system, this field should consist only of the characters A to Z, 0 to 9, and the hyphen (-).

8. `Owner ID:` - This can be any name desired to identify the owner of the tape volume.

9. `Dataset ID:` - This 17-character field can consist of any characters and names the dataset (file) currently being written to tape. When writing multiple files to a tape volume, type the identifier for each new file in this field before starting the file transfer.

10. `Creation Date:` - This specifies the date when the data set was created, in the form **yyddd**, where **yy** is the last two digits of the year and **ddd** is the day of the year (1-366).

11. `Expiration Date:` - This specifies the date when the data set can be overwritten, in the same format as the creation date.

10.4 Supplemental Tape Label Fields

Tape labels contain many supplemental fields, some of which are used by mainframe computer operating systems, others which are only used for documentation purposes. TLABEL normally puts "default" data in those supplemental fields which are required, and leaves them blank if they

are not required. To place different values in some of these supplemental fields, bring up the Supplemental Fields screen while the main screen is displayed by pressing F5. The Supplemental Fields screen is similar to that shown in Figure 10-2.

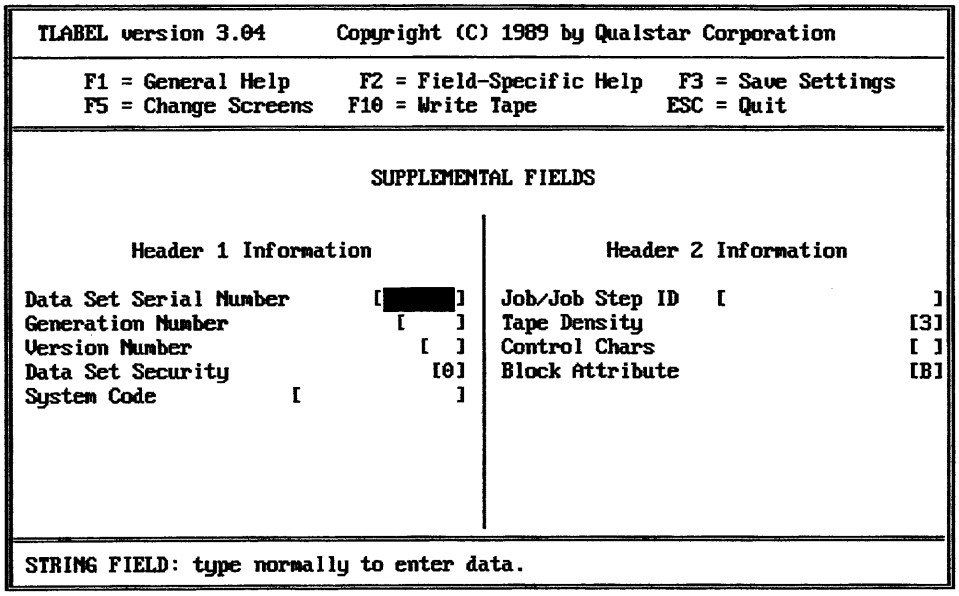


Figure 10-2 TLABEL Supplemental Fields Screen

The supplemental fields are divided into two groups: those which go into the Header 1 label, and those which go into the Header 2 label. DOS/VSE labeled tapes have no Header 2 label.

10.4.1 Header 1 Label

1. `Data Set Serial Number:` - The serial number of the tape volume which contains the data set. This field can consist of any characters, up to six characters long. It is recommended that only alphanumeric characters be used in this field.

2. `Generation Number:` - If the data set (file) being transferred is part of a generation data group, this field can contain a number from 1 to 9999 to indicate the absolute generation number.

3. `Version Number:` - If the data set (file) being transferred is part of a generation data group, a generation version number from 0 to 99 may be entered here.

4. `Data Set Security:` - This selects one of three possible security conditions for the data set. The default is **0**.

   a. **0** - No password protection.

   b. **1** - Password required for reading, writing or deleting this data set.

   c. **2** - Password required for writing or deleting only.

10.4.2 Header 2 Label

1. `Job Step ID:` - This can contain the name of the job and the job step which (would have) created the data set. In this 17-character field, the first eight

characters specify the job name; the ninth character should be a slash (/); the last eight characters are the name of the job step.

2. `Tape Density:` - This single character is a code for the density at which the tape was written. The default is **3**; possible values are:

   a. **2** - 800 cpi (NRZI)

   b. **3** - 1600 cpi (PE)

   c. **4** - 6250 cpi (GCR)

3. `Control Characters:` - This field indicates whether control characters were used to create the data set, and if so, what type they were:

   a. **A** - ASCII control characters

   b. **M** - machine control characters

   c. (blank) - no control characters

4. `Block Attribute:` - specifies what type of blocking applies to the data records in the data set with this field. The default is **B**.

   a. **B** - blocked records

   b. **S** - spanned records

   c. **R** - blocked and spanned records

   d. (blank) - records neither blocked nor spanned.

10.5 Running TLABEL in the Batch Mode

When writing many labeled tapes using the same settings, the names of the files to be transferred can be entered on a command line, and the program menus can be bypassed entirely. However, TLABEL must first be run in the interactive mode to create a settings file which the command line can use. This section describes how to do this.

1. Run TLABEL interactively as described in the preceding sections.

2. Once all the required settings have been specified, save them into a settings file using F3.

3. Name the file appropriately so you will remember which tape or type of tapes it goes with.

4. Exit TLABEL.

5. When it comes time to do the actual transfer from disk to tape using this settings file:

   a. Run TLABEL by using the following syntax at the DOS prompt:

      `tlabel` *`settings-file`* **`-b`** `[file1] [...] [options]`

   b. Substitute the name of an existing settings file for the *settings-file* parameter. The *settings-file* must be the first filename on the command line.

   c. Specify the names of all the files to be transferred to tape after the **B** indicator, and precede each filename with a space.

When using TLABEL in the batch mode, the dataset ID field is automatically filled in with the file name of the file being transferred, including the root and extension of that name, but excluding any drive or path specifiers. For example: If the file D:\HOLMES\MORIARTY.DAT were specified, only MORIARTY.DAT would be automatically placed in the dataset ID field.

10.6 TLABEL Command Line Options

TLABEL supports the following options (which are explained in preceding chapters):

- **drive=***drive-specifier*
- **format=***tape-format/density*
- **XA***translation-table-file*

10.6.1 Changing Record and Block Lengths from the Command Line

Starting with version 4.08, you can change the record and block lengths on a file-by-file basis. This option is invoked by using parentheses after the file name to indicate the new lengths. In this case, you type in the parentheses on the command line, and you do not precede the left parenthesis with a space.

The first number in the parentheses is the new record length and the second number is a blocking factor representing the number of records within a block. Once these numbers are used in the command line, they apply to all subsequent files until new numbers are entered. For example:

```
tlabel my.set 0b cars.txt(200,10) boats.txt planes.txt(40,20)
```

The preceding command uses a settings file named MY.SET. It will write CARS.TXT at the current record length, BOATS.TXT using record lengths of 200 at 10 records per block, and PLANES.TXT. using record lengths of 40 with 20 records per block.

11. Using TCONVERT.EXE

TCONVERT is a full-screen, menu-driven utility program which copies files between the PC and unlabeled tapes. TCONVERT also performs other useful functions, such as viewing tape data, reading files from labeled tapes, and positioning the tape.

Syntax: `tconvert`

A TCONVERT main screen similar to that shown in Figure 11-1 will appear. The upper half of the screen displays the current source and destination settings for transfer operations. Any file transfer operation will use the currently displayed settings. These settings can be individually changed by selecting the appropriate command from the `COMMANDS:` list and entering the new choice for the setting.

```
Data Conversion Utility, Copyright 1992, Qualstar Corporation  v4.22

                          SOURCE              DESTINATION
Filename      :             test.src            test.dst
Tape/Disk     :             Disk                Disk
ASCII/EBCDIC  :             ASCII               ASCII
Fixed Length  :             No                  No
Labeled       :             No                  No
Record Size   :             <N/A>               <N/A>
Block Size    :             <N/A>               <N/A>

Select a menu command.  Press H for help with menus.

                    COMMANDS:
                      C) Convert
                      V) View Source
                      S) Change Source...
                      D) Change Destination...
                      L) Write Label...
                      M) Tape Motion...
                      T) Change Settings...
                      H) Help with menus
                      Q) Quit
```

Figure 11-1 TCONVERT Main Screen

To select a command, either type the capitalized letter of the choice, or highlight your choice using the up- and down-arrow keys and then press ENTER. For example, typing **M** (or highlighting `Tape Motion...` and pressing ENTER) activates the Tape Motion command and brings up a list of tape motion commands similar to that shown in Figure 11-2.

When the tape motion command list appears, either type the capitalized letter of the choice, or highlight your choice using the up- and down-arrow keys and then press ENTER. For example, highlighting `Write Tape Mark` and pressing ENTER causes the tape drive to write a filemark.

In some cases, an additional selection box will appear with a list of choices from which to select. This would be the case had you chosen the `Change Destination...` command and then the `Device` entry in the `Change` column. This is shown in Figure 11-3.

To copy a file from disk to tape or vice versa:

1. Choose the proper source and destination settings for the transfer displayed in the upper half of the screen.

2. Make sure that the tape is loaded in the drive, the drive is online, the tape is not write-protected (if copying to tape), and the tape is at the correct position. Section 11.2 on page 11-4 explains how to position the tape.

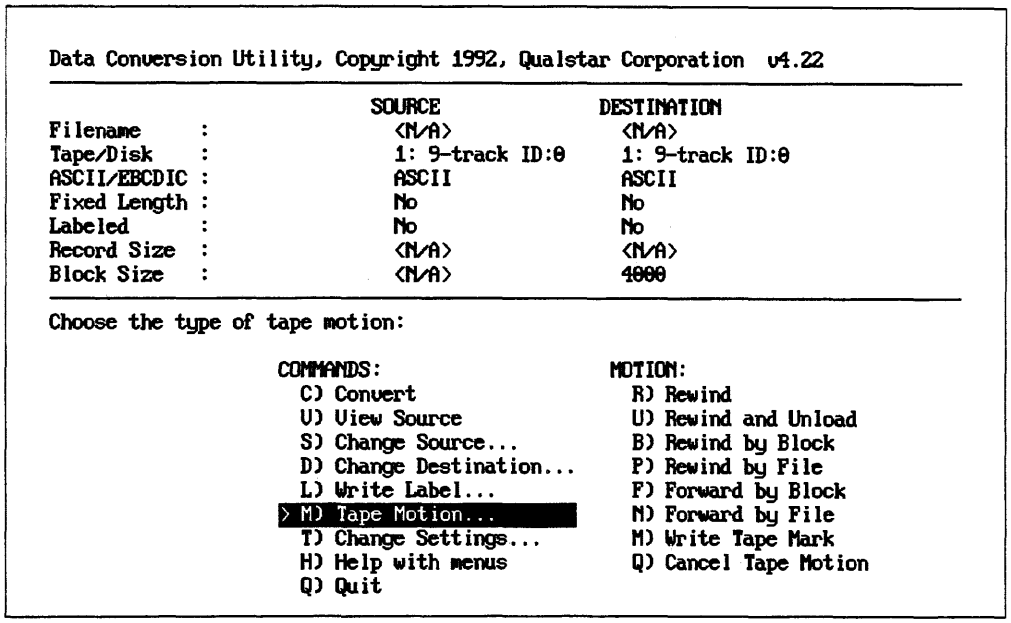


Figure 11-2 TCONVERT Tape Motion Commands

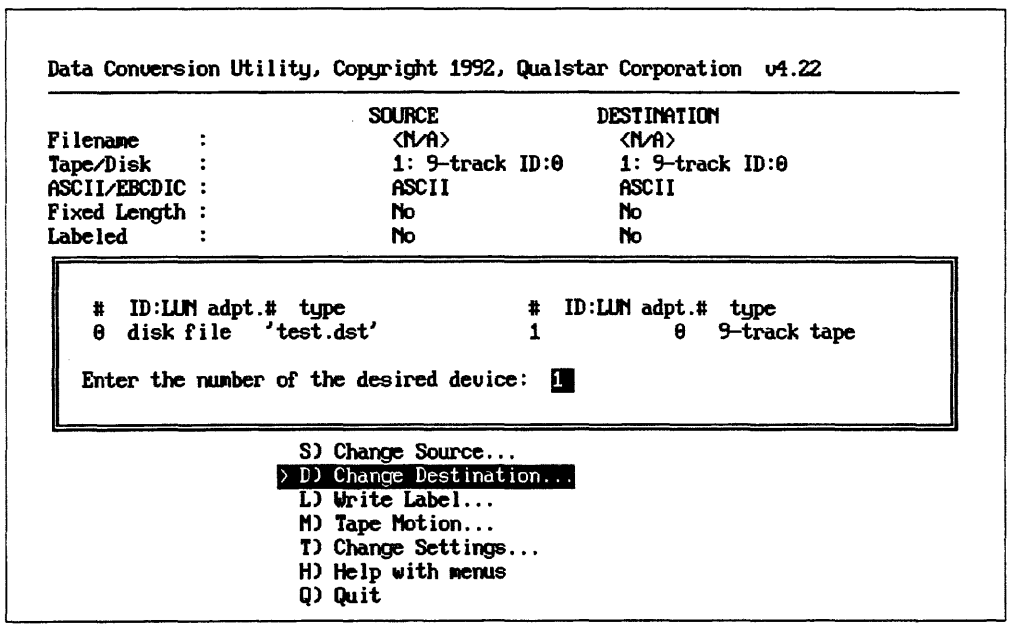


Figure 11-3 TCONVERT Drive Selection Pop-Up

3. Type **C** to start the transfer operation.

11.1 Setting Source and Destination Controls

To change either source or destination settings, choose the corresponding menu item. When **S** or **D** is selected from the menu, a sub-menu will pop up with a list of settings from which to pick. The next sections explain these settings for both disk and tape.

11.1.1 Settings for Disk

1. **Filename:** - This is the name of the DOS disk file to be read or written.

2. **Device:** - If the desired entry (source or destination) is not currently set to use a disk file (as shown by `Disk` on the Tape/Disk line), select this setting. The device selection pop-up box will appear. In this box, select device **0**, which is a disk file. The file name can be changed using the filename option.

3. **Fixed Length:** - **YES** here specifies that the disk file has fixed length records, while **NO** specifies variable length records. Text files on disk on a PC usually have variable length records. If the file has fixed length records, the record size setting must be specified. If it has variable length records, check the Conversion Settings item.

4. **Record Size:** - This is only meaningful if the file has fixed length records. The number here is the number of bytes (characters) in one file record, either source or destination.

5. **Block Size:** - For disk files, block size has no meaning.

6. **ASCII/EBCDIC:** - specifies the character set of the file's data. Text files on the PC are usually written in ASCII.

7. **Conversion settings:** - This item does not appear at the top of the screen, but is selected from the Source and Destination menu items.) For variable length records, this item specifies to TCONVERT what character(s) mark the end of a data record. By default, this item is preset to the ASCII carriage-return/line-feed pair as the record terminator; these characters are used in the majority of ASCII text files. For fixed length data, this item is ignored.

For disk files, no other settings are meaningful.

11.1.2 Settings for Tape

1. **Device:** - This specifies which device to use for the desired entry (source or destination). The device can either be a disk file (as shown by `Disk` on the Tape/Disk line), or a tape drive. To specify a device, choose this item. The device selection pop-up box will appear. Select the number (at the left of the selection entry) of the desired device.

2. **Fixed Length:** - **YES** here indicates that the tape file has fixed length records, while **NO** indicates variable length records.

   a. If the file has fixed length records, the record size for either source or destination tape files must be specified.

   b. If a tape with fixed length records is the destination, then the tape block size must also be specified.

   c. If the tape has variable length records, check the Conversion Settings item.

3. **Record Size:** - This is only meaningful if the file has fixed length records. This number is the number of bytes (characters) in one file record, either source or destination. On tape, usually several records will be written together in each tape block.

4. **Block Size:** - This choice is only needed when a tape with fixed length records is being written as the destination. The block size must be an even multiple of record size. When selecting the block size, keep in mind that some installations impose limits on the sizes of blocks which can be read from tape.

5. **ASCII/EBCDIC:** - This selects the character set of the file's data. If the tape file is the destination, it is necessary to know what the reader of the tape expects to find.

6. **Conversion Settings:** - This item does not appear at the top of the screen, but is selected from the Source and Destination menu items. For variable length records, this item specifies to TCONVERT what character(s) mark the end of a data record. By default, this item is preset to the ASCII CR/LF pair as the record terminator; these characters are used in the majority of ASCII text files. For fixed length data, this item is ignored.

7. **Labeled:** - For unlabeled transfers, this item should be set to unlabeled.

NOTE

TCONVERT does not write labeled tapes. To write a labeled tape, refer to Section 10. To use TCONVERT to read labeled tapes, refer to Section 11.4 on page 11-5.

11.2 Positioning the Tape

The Tape Motion menu item allows the tape to be moved to a particular position for either reading or writing. The following positioning operations are available:

1. **Rewind** - Returns the tape to BOT (beginning of tape);

2. **Rewind and Unload** - Returns the tape to BOT and then unloads the tape;

3. **Reverse by Block** - Spaces the tape one or more blocks in the reverse direction; the tape will be positioned at the start of the ***n***th block ahead of the current position;

4. **Reverse by File** - Spaces the tape one or more files in the reverse direction and counts filemarks. If the current position is between two filemarks (i.e., in the middle of a tape file), spacing back by one file will position the tape at the beginning of that file;

5. **Forward by Block** - Spaces the tape one or more blocks forward from the current position;

6. **Forward by File** - Spaces the tape one or more files forward from the current tape position and counts filemarks. If the current position is between two filemarks (i.e., in the middle of a tape file), spacing forward by one file will position the tape at the beginning of the next file (or the end of tape if this is the last file);

7. **Write Tape Mark** - Writes a filemark at the current tape position. This operation should be used with caution, since TCONVERT already writes a filemark at the end of each file written to tape. Use this command to write a double filemark at the logical end of tape.

11.3 Change Settings Menu

The Change Settings menu item contains the following miscellaneous TCONVERT options, most of which control tape behavior:

1. **Append/Overwrite** - When writing to a disk file, this option controls whether the destination file will be overwritten (the old contents of the file replaced completely), or appended to (the new data added at the end of an existing file);
2. **Convert File/Block** - When reading from tape, this option controls whether the entire file or only the first block of the file will be copied;
3. **Write One/Two Tape Marks** - Selects how many filemarks will be written after a file has been copied to tape. Two consecutive filemarks indicate the logical end-of-tape (logical EOT);
4. **Tape Density** - Selects the density on the tape drive through the controller. This feature only works on drives which support remote density selection;
5. **Save Settings File** - All of the options selected within TCONVERT can be saved in a special file which can then be used to recall those same settings. By default, TCONVERT creates a file called TCONVERT.SET each time the program is exited; TCONVERT also reads this file each time the program is started and sets the settings according to the information in the file. The Save Settings option can save into an alternate file whose name you specify;
6. **Load Settings File** - This selection allows all of TCONVERT's option settings to be set to the contents of a created using the Save Settings option.

11.4 Copying Files from Labeled Tape to Disk

To copy files from labeled tape to disk, use the TCONVERT program as previously explained, with these additional instructions:

1. Set the source to be **TAPE** with the appropriate label type:
   a. IBM OS/VS labeled (data in EBCDIC)
   b. IBM DOS/VSE labeled (data in EBCDIC)
   c. ANSI labeled (data in ASCII)
2. Set the other source parameters for the tape as explained in Section 11.1.2 on page 11-3;
3. Set the destination to be a disk file with the appropriate settings. A different file name must be specified for each file to be transferred;
4. Position the tape to the beginning of the file to be copied.
   a. If it is the first file on the tape, position the tape to BOT.
   b. On a tape with multiple files, the tape will be left positioned after the last file transferred.
   c. To skip files or go back by files, use TCONVERT's positioning operations as previously explained.

11.5 Resolving File Transfer Problems

This section describes a few of the problems which may occur during data transfers between disk and tape.

1. **PROBLEM** - While reading a labeled tape file to disk using TCONVERT, an error message appears stating "`No HDR2 block found`" and then the transfer stops.

   CAUSE - The wrong label format may have been specified. OS/VSE labeled tapes have both HDR1 and HDR2 header blocks, while DOS/VSE only has HDR1 blocks.

   FIX - Specify the correct label type for the source tape.

2. **PROBLEM** - When transferring variable length records to fixed length records, data at the end of some of the records was lost.

   CAUSE - Too small an output record length was specified, and input records which exceeded that length were truncated.

   FIX - Ensure that the output record length is at least as long as the longest record in the input file.

3. **PROBLEM** - When transferring fixed length records to variable length records, data ended up in the wrong place in each record; each record was offset by an increasing number of characters.

   CAUSE - The wrong input record length was specified.

   FIX - Specify the correct input record length. If unsure of the record length, look at the source data to determine where each record starts and ends. Look at the screen shots at the end of this chapter.

11.6 Using TCONVERT to View Files

TCONVERT lets you look at the data inside both tape and disk files. It has several display modes to show data in files which contain either fixed length or variable length records; TCONVERT can also interpret the label records found on labeled tapes.

To use TCONVERT to view file contents, type **TCONVERT** at the DOS prompt.

1. To view a file, first choose that file's source—either disk or tape.
2. Set all the needed choices for the source (fixed/variable length records, ASCII/EBCDIC, etc.)
3. Select the **View** menu item.
   a. If the source is a disk file, the first part of the file will be displayed.
   b. If the source is tape, then the next block, starting at the current tape position, will be displayed.

The following commands and keys control the view:

1. Viewing tape files:
   a. **N** - Moves to the next tape block

b. **P** - Moves to the previous block;

c. **HOME** - Moves to the top of the current block;

d. **END** - Moves to the end of the block.

2. Viewing disk files:

   a. **HOME** - Moves to the beginning of the disk file;

   b. **END** - Moves to the end of the file.

3. For either disk or tape, the following letter keys change the way data is displayed in the window:

   a. **L** - switches between line display mode, where each file record is displayed on a separate line, and raw display mode, where data is simply displayed without any line formatting.

   b. **T** - Switches between showing file data as characters in ASCII or EBCDIC, or as hexadecimal values. The hex display shows 16 bytes of data per line, with hex on the left and characters on the right.

   c. **A** - Switches between showing characters as ASCII or EBCDIC.

4. For both disk and tape, the up- and down-arrow keys scroll the display window by one line, while PAGE UP and PAGE DOWN scroll by windows.

The following screen shots illustrate how TCONVERT's display modes work. Table 11-1 lists the various mode combinations. The first four screen shots show the same block of data at the same offset (starting place in the block) in a file containing fixed length 275-byte records; the next two screen shots show the same block of data in a file containing variable length records with CR/LF feed delimiters; screen shot #7 illustrates what happens when the wrong record length is chosen on TCONVERT's "settings" menus.

SCREEN #	RECORD TYPE	LINE/RAW	CHAR/HEX
1	Fixed	Line	Characters
2	Fixed	Raw	Characters
3	Fixed	Line	Hexadecimal
4	Fixed	Raw	Hexadecimal
5	Variable	Line	Characters
6	Variable	Raw	Characters
7	Incorrect record length set		

Table 11-1 Screen Shot Summary

Notice how all the data "lines up" in this example. Since the data records are longer than the screen can display, they "extend" invisibly off the right side of the screen. The screen is actually a small window through which the file, (which is wider and taller) is viewed. The file can be moved around within the window by using the arrow keys.

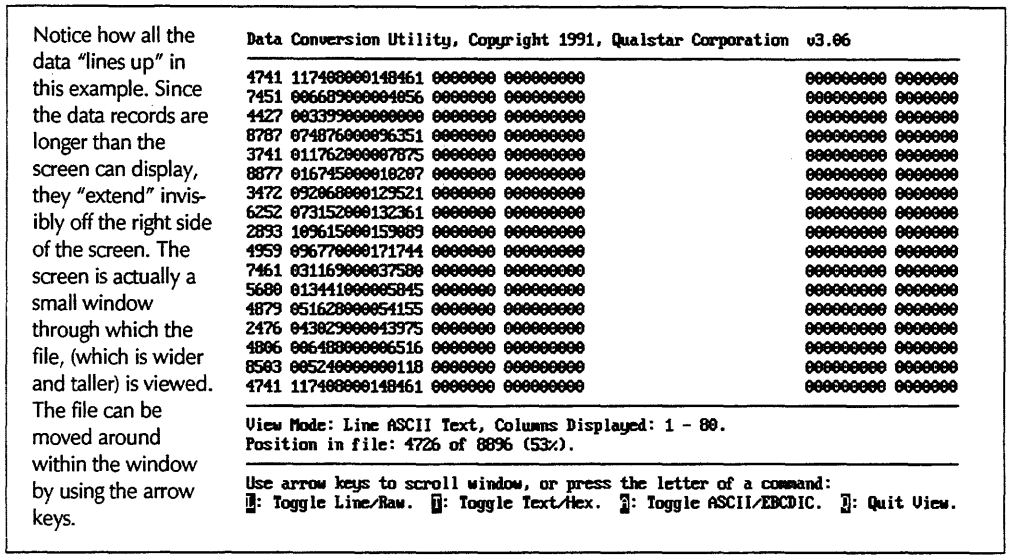


Figure 11-4 Screen #1 - Fixed-length records, line/character modes

The data does not line up because the display is in the "raw" view mode, causing the characters to wrap around from one line to the next. Notice that fewer records are shown here than in Screen #1. If the record length were shorter than the screen line length (80 characters), the raw view mode would display more records than the line view mode.)

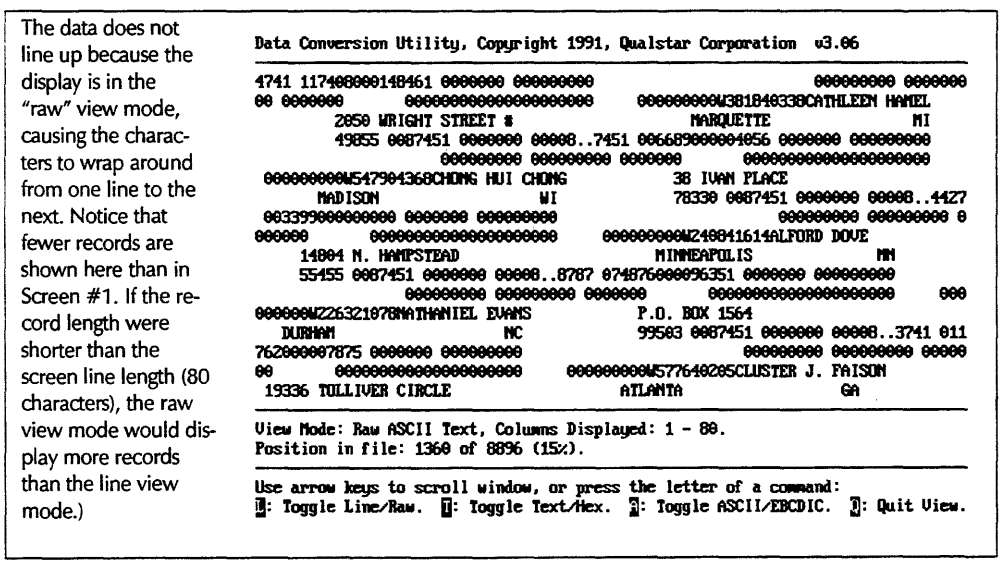


Figure 11-5 Screen #2, Fixed-length records, raw/character modes

This is the same view mode as that in Screen #1 except that the data is shown in hexadecimal and as characters (the characters are shown at the right side). The data lines up because the view mode is "line".

```
Data Conversion Utility, Copyright 1991, Qualstar Corporation  v3.06

     0: 34 37 34 31  20 31 31 37  34 30 38 30  30 30 31 34  4741 11740800014
   278: 37 34 35 31  20 30 30 36  36 38 39 30  30 30 30 30  7451 00668900000
   556: 34 34 32 37  20 30 30 33  33 39 39 30  30 30 30 30  4427 00339900000
   834: 38 37 38 37  20 30 37 34  38 37 36 30  30 30 30 39  8787 07487600009
  1112: 33 37 34 31  20 30 31 31  37 36 32 30  30 30 30 30  3741 01176200000
  1390: 38 38 37 37  20 30 31 36  37 34 35 30  30 30 30 31  8877 01674500001
  1668: 33 34 37 32  20 30 39 32  30 36 38 30  30 30 31 32  3472 09206800012
  1946: 36 32 35 32  20 30 37 33  31 35 32 30  30 30 31 33  6252 07315200013
  2224: 32 38 39 33  20 31 30 39  36 31 35 30  30 30 31 35  2893 10961500015
  2502: 34 39 35 39  20 30 39 36  37 37 30 30  30 30 31 37  4959 09677000017
  2780: 37 34 36 31  20 30 33 31  31 36 39 30  30 30 30 33  7461 03116900003
  3058: 35 36 38 30  20 30 31 33  34 34 31 30  30 30 30 30  5680 01344100000
  3336: 34 38 37 39  20 30 35 31  36 32 38 30  30 30 30 35  4879 05162800005
  3614: 32 34 37 36  20 30 34 33  30 32 39 30  30 30 30 34  2476 04302900004
  3892: 34 38 30 36  20 30 30 36  34 38 38 30  30 30 30 30  4806 00648800000
  4170: 38 35 30 33  20 30 30 35  32 34 30 30  30 30 30 30  8503 00524000000
  4448: 34 37 34 31  20 31 31 37  34 30 38 30  30 30 31 34  4741 11740800014

View Mode: Line ASCII Hex, Columns Displayed: 1 - 16.
Position in file: 4726 of 8896 (53%).

Use arrow keys to scroll window, or press the letter of a command:
1: Toggle Line/Raw.  2: Toggle Text/Hex.  3: Toggle ASCII/EBCDIC.  Q: Quit View.
```

Figure 11-6 Screen #3, Fixed-length records, line/hex modes

This is the same view mode as that in Screen #2, except that the display is hexadecimal.

```
Data Conversion Utility, Copyright 1991, Qualstar Corporation  v3.06

     0: 34 37 34 31  20 31 31 37  34 30 38 30  30 30 31 34  4741 11740800014
    16: 38 34 36 31  20 30 30 30  30 30 30 30  20 30 30 30  8461 0000000 000
    32: 30 30 30 30  30 30 20 20  20 20 20 20  20 20 20 20  000000
    48: 20 20 20 20  20 20 20 20  20 20 20 20  20 20 20 30                 0
    64: 30 30 30 30  30 30 30 30  20 30 30 30  30 30 30 30  00000000 0000000
    80: 30 30 20 30  30 30 30 30  30 30 20 20  20 20 20 20  00 0000000
    96: 20 30 30 30  30 30 30 30  30 30 30 30  30 30 30 30   0000000000000000
   112: 30 30 30 30  30 30 20 20  20 20 20 30  30 30 30 30  000000     00000
   128: 30 30 30 30  57 33 38 31  38 34 30 33  33 38 43 41  0000W381840338CA
   144: 54 48 4C 45  45 4E 20 48  41 4D 45 4C  20 20 20 20  THLEEN HAMEL
   160: 20 20 20 20  20 20 20 20  20 32 30 35  30 20 57 52           2050 WR
   176: 49 47 48 54  20 53 54 52  45 45 54 20  23 20 20 20  IGHT STREET #
   192: 20 20 20 20  20 20 20 20  20 20 20 20  20 20 20 20
   208: 20 4D 41 52  51 55 45 54  54 45 20 20  20 20 20 20   MARQUETTE
   224: 20 20 20 20  20 20 20 20  20 20 4D 49  20 20 20 20            MI
   240: 20 20 20 20  20 20 20 20  20 34 39 38  35 35 20 30           49855 0
   256: 30 38 37 34  35 31 20 30  30 30 30 30  30 30 20 30  087451 0000000 0

View Mode: Raw ASCII Hex, Columns Displayed: 1 - 16.
Position in file: 272 of 8896 (3%).

Use arrow keys to scroll window, or press the letter of a command:
1: Toggle Line/Raw.  2: Toggle Text/Hex.  3: Toggle ASCII/EBCDIC.  Q: Quit View.
```

Figure 11-7 Screen #4, Fixed-length records, raw/hex modes

This screen shot shows a variable length text file with carriage-return/ line-feed delimiters at the end of each line. Since the "source" control is set to variable, when viewed in "line" mode each line ends at the actual line ending (like using the DOS TYPE command to view at a disk file).

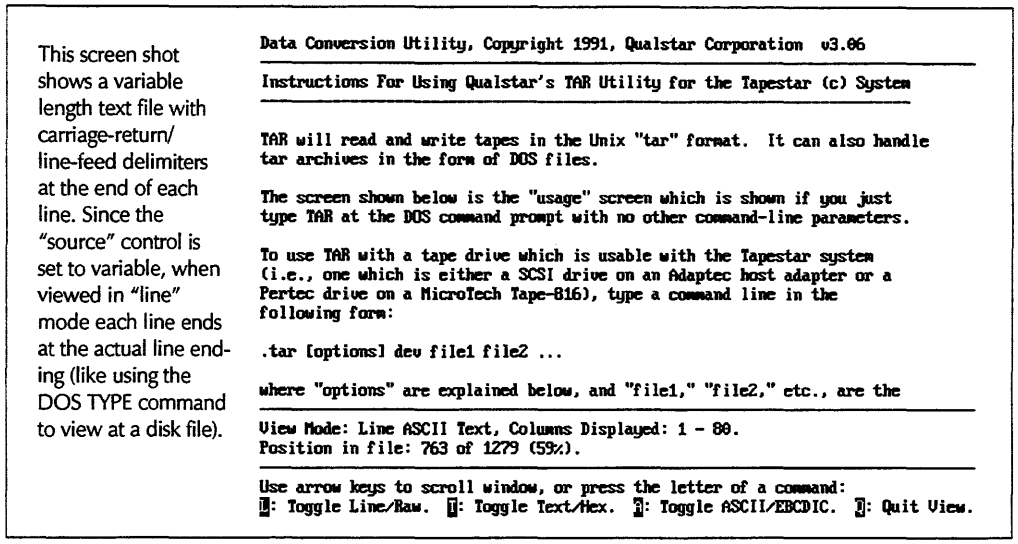


Figure 11-8 Screen #5, Variable-length records, line/character modes

In the "raw" view mode, the data is displayed without line breaks. The pairs of characters (shown as periods) between each line are the line delimiters (carriage-return/line-feed) which mark the line endings in variable length data.

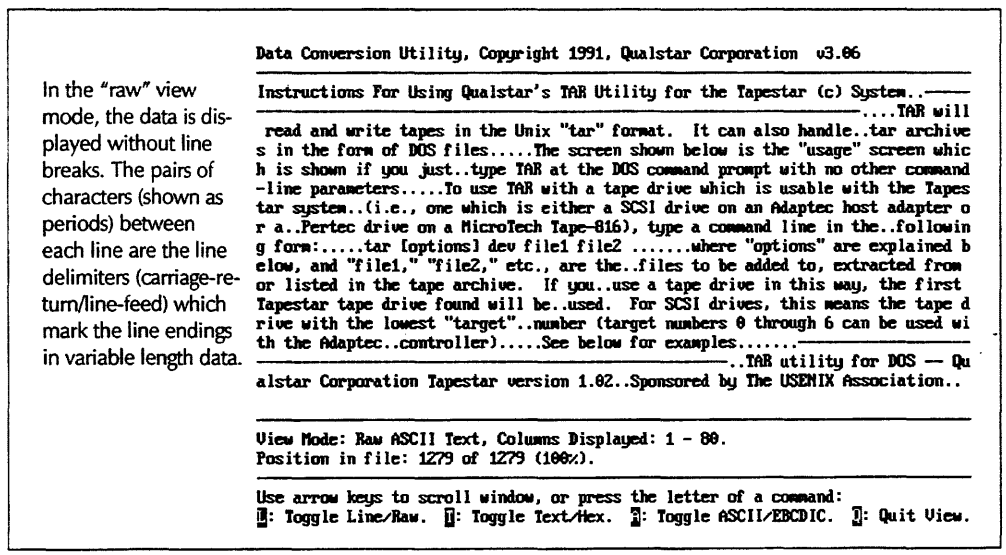


Figure 11-9 Screen #6, Variable-length records, raw/character modes

This display results when an incorrect record length is specified prior to viewing a fixed length file. In this case, the record length is off by one character. If the selected record length is too short, the display will slant to the left.

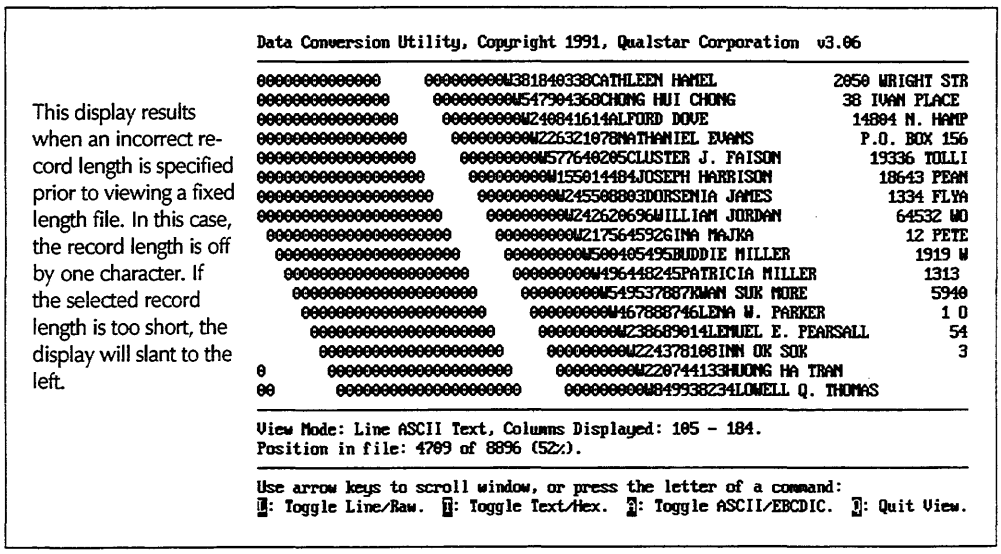


Figure 11-10 Screen #7, Incorrect record length set

12. Using TAPELIST

The TAPELIST program reads an entire tape and displays a summary report of that tape on the screen, providing information about all the files on the tape.

1. If the tape is unlabeled, the summary will display:
   a. The number of files on the tape;
   b. The number of blocks in each file;
   c. The lengths of those blocks;
   d. The location of filemarks and EOD (end-of-data) on the tape.
2. If the tape is a standard labeled tape (IBM OS/VS, IBM DOS/VSE or ANSI), additional information contained in each tape file's header labels will be shown, including:
   a. The name, creation date and expiration date of the dataset (file);
   b. The generation and version numbers;
   c. The name of the creating system;
   d. The record and block lengths and formats;
   e. The number of blocks in the file.

12.1 The TAPELIST Command Line

Syntax: **tapelist** [**drive=***drive-specifier*] [**/c**] [**/p**] [**/s**]

1. **drive=***drive-specifier* - Selects the tape drive to use if you have is more than one tape drive in the system. (This parameter is not required if the system has only one tape drive.) Refer to Chapter 7 for more information.
2. **/C** - Tells TAPELIST to display its information continuously without stopping after each screen. Use this option when redirecting the TAPELIST report to a file or to a printer. Otherwise, TAPELIST will pause after each screen.
3. **/P** - Shows the name of partitioned dataset (PDS) names found on a tape. This option is only meaningful when reading PDS "library" tapes. Partitioned datasets are used by IBM and other compatible large computer systems.
4. **/S** - Limits TAPELIST to a short-form display of information. The short-form display only shows the names of files on a labeled tape, which is useful for tapes containing lots of files. This option has no effect when listing unlabeled tapes.

12.2 Sending TAPELIST Listing to a File or Printer

To send the output of TAPELIST directly to a DOS file or to a printer, place the following at the end of the command line: *>filename* where filename is the DOS file which will be created (or overwritten if it already exists), or PRN or LPT1. Be sure to include the **/C** option. To append to an existing file, use two redirection symbols (>>) instead of one.

12.3 Sample Run of TAPELIST

The following is a sample of TAPELIST's output.

```
TAPELIST  version 1.05   (1991) by Qualstar Corporation
Conversion Systems

Initializing . . . done.

Using drive: Qualstar 3410S    (ID:LUN  5:0)

Tape is OS/VS labeled.

Vol. serial #:  'MFICHE'
Owner ID:       '          '
Tape has 2 HDR labels.

File 1:            'GMF.M156         '
Creation date:      91073   expiration date:  00001
Generation #:      '0001'   version #:       '00'
Creating system:   'IBMOS400     '
Record length:     156       record format:   'F'
Block size:        12480     block attribute: 'B'
Block count in EOF1 label: 4        actual count:  4

File 2:            'GMF.BCAS         '
Creation date:      91073    expiration date:  00001
Generation #:      '0001'    version #:       '00'
Creating system:   'IBMOS400     '
Record length:     128       record format:   'F'
Block size:        10240     block attribute: 'B'
Block count in EOF1 label: 1        actual count:  1

File 3:           'GMF.STM1          '
Creation date:     91073     expiration date:  00001
Generation #:     '0001'     version #:       '00'
Creating system:  'IBMOS400     '
Record length:    128        record format:   'F'
Block size:       10240      block attribute: 'B'
Block count in EOF1 label: 1        actual count:  1

*** End Of Tape ***
3 files on tape.
## Total bytes of data on tape.
```

13. Using DEX.EXE

13.1 Introduction

DEX reads data from tape files and writes it to disk. Unlike other Tapestar transfer programs, DEX can extract data *field-by-field* from tape files with fixed length records. Data is correctly converted from fields of various types (packed decimal, binary, numeric, etc.) to plain ASCII text which can be used by databases, text editors, word processors, spreadsheets, etc.

This chapter discusses the following subjects:

1. Using DEX (Section 13.2)
2. DEX Examples (Section 13.5)
3. DEX Reference - This section explains all DEX control file header statements and field specifiers detail (Section 13.3)
4. Packed and Binary Files (Section 13.4)

13.1.1 DEX Capabilities

DEX allows you to select which fields to transfer to the output file. In addition, DEX can *selectively* extract data records. Using simple selection criteria, records can be selected by examining specific fields for desired values in each record.

Tapes most likely to be used by DEX typically come from mainframe computers, where data is often (but not always) stored in fixed length records. These files are fine for mainframe computers, but can be difficult or impossible to use with databases and other programs found in the PC environment.

13.1.2 DEX Limitations

DEX was designed as a simple tool to import tape data onto a PC's disk. DEX was not designed to generate reports or to manipulate data. Such tasks should be handled by other applications such as a database program. Providing DEX with the power and flexibility of a database or report generator makes it difficult and cumbersome to define a record layout for even a simple extraction task.

13.1.3 DEX Operation

The following paragraphs describe each item in Figure 13-1.

1. **Tape** - This can be any file with fixed length records on a tape in a Qualstar tape drive.
2. **Control File** - This is an ASCII text file which the user creates using a word processor or text editor. It tells DEX what kind of fields are in the tape file, their locations and lengths, and what to do with the data.
3. **Report File** - DEX will generate this file if requested.This text file contains a formatted report showing the type, location, length and other information about each field defined in the control file. The report can be printed or viewed while setting up the control file to check work.
4. **Extracted Output File** - This is where the extracted data goes. When selecting data by fields or by records, only the selected data goes into this file.

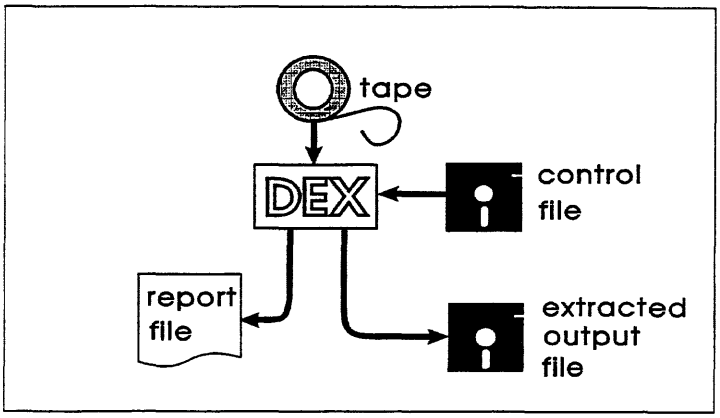


Figure 13-1 DEX and Its Files

13.1.4 Some Practical Uses for DEX

13.1.4.1 Extracting Mailing List Information

DEX can read a tape of mailing list entries, transferring only the required fields from each record. It can also select records between a certain range of zipcodes, using REJECT statements.

13.1.4.2 Transferring Tape Data to a PC Database

DEX can transfer a tape file which was created on a mainframe or minicomputer to a file which can be read into a PC database. You can select which fields to transfer, which fields to ignore, and can reject records from the transfer based on the value of certain fields.

13.2 Using DEX

The following steps are required to successfully use DEX:

1. Obtain a record layout.
2. Interprete the record layout.
3. Create the DEX control file.
4. Check the control file.
5. DEXtract the data.

13.2.1 Step 1: Obtaining a Record Layout

In order to use DEX, you must have a record layout, which is a kind of road map of the data file to be copied from the tape. Without knowing the record layout, using DEX is difficult or impossible. The layout may be in the form of COBOL file description source code, a site-specific layout form, or some other form, either on paper or in computer storage. A printed copy of the layout is advantageous.

If a record layout is not available, you will have to create one by looking at the tape using TCONVERT or TAPELIST. Keep these tips in mind while paging through the screens:

1. To determine the record length, look for recurring data patterns and for data which lines up vertically on the display.

2. Usually, each tape block will be divided evenly into data records, with no leftover bytes at the end; therefore, the block length will usually be an even multiple of the record length.

3. Any text in the file (names, addresses, etc.), should be readable on the screen. If the display is unrecognizable, try switching it from ASCII to EBCDIC or vice versa.

4. If the file might contain packed-decimal or binary numeric fields, read Section 13.4 on page 13-16.

When constructing a record layout, you must know the location and type of each field to be extracted. To do this:

1. Make a list (on paper) of all the fields in the record which can be identified. Each entry should have the field type and length in bytes (if there are any packed fields, use the length of the field in digits).

2. Use the following letters to designate the types. These also happen to be DEX's field-type specifiers.

LETTER	TYPE
A	ASCII text (alphanumeric)
E	EBCDIC text (alphanumeric)
Z	Numeric (one digit per byte)
B	Binary
R	"Raw" transfer—no processing
P	Packed-decimal
X	"Skip" field (don't transfer data)

Example:

If it is clear that each record has a name field in the first 25 bytes, which is in EBCDIC, then write down **E 25** at the top of the list as the first field entry.

1. If there's any data which cannot be identified, use one of the following options:

   a. If that part of the record is not needed or wanted, treat it as a *skip* field; that part of each record will not be transferred.

   b. If the data in question is to be retained, define it as a *raw* transfer field; that part of each record will be transferred to the output file with no translation or other processing.

NOTE

This option only works if that part of the record contains no packed or binary data. If there is, the output data will probably contain *control* characters which cannot be digested by most applications.

13.2.2 Step 2: Interpreting the Record Layout

This section assumes you have some knowledge about data processing and file formats, and is written to help clear up some common points of confusion on this subject.

Record layouts come in several different but generally similar forms. They can exist as typewritten or handwritten sheets which simply list fields by type, length, location and (sometimes) name, or they can be a printed listing of the part of a COBOL program where the file record is defined. This step consists of deciphering the record layout and listing (on paper) each field in DEX *shorthand* form. From this list, the DEX control file can then be created very easily.

In order to use DEX, you must determine the following information about each field:

1. Field location
2. Field data type
3. Field length

13.2.2.1 Field Location

Unlike some other programs, DEX does not require you to spell out, with exactly the right number of leading zeros, where each field begins. In fact, you do not explicitly specify the exact field location at all. DEX determines the field locations from the cumulative lengths of previous fields and from the order of the fields. DEX must only be told how long each field is; DEX calculates the starting and ending locations of each field.

13.2.2.2 Field Data Type

If a record layout has been manually created as previously described, this information is already available. When reading an existing record layout, reading between the lines may be necessary to figure out what kind of data the field contains.

Tables 13-1 and 13-3 show several field-type designations commonly used in record layouts. The last column has the corresponding DEX field-type specifier to be used when creating the DEX control file.

FIELD TYPE FROM RECORD LAYOUT	FIELD TYPE	DEX FIELD DESCRIPTOR
A/N Text Characters	Alphanumeric	A or E*
Numeric/9/N/X	Numeric: Zoned	Z
	Numeric: Binary	B
	Numeric: Packed Decimal	P
Packed	Packed Decimal	P

Table 13-1 Common Field-Type Designators

# OF DIGITS IN FIELD	FIELD LENGTH IN BYTES
1-4	2
5-9	4

Table 13-2 Binary Fields

13.2.2.3 Field Length

The last piece of information required for DEX is the length of the field. This is a point of potential confusion, since there are two units of measure used for field length:

1. The length of packed fields is specified as the number of *numeric digits* in the field;
2. The length of all other field types is specified as the number of *bytes* (characters) in the field.
3. For all field types except packed and binary, the length of the field should be exactly the same as the number of bytes/characters shown in the record layout.
4. For binary fields, Table 13-2 shows how to figure the length of the field (in bytes) if that length is unknown.

PICture	FIELD TYPE	DEX FIELD DESCRIPTOR
PIC X	Alphanumeric	A or E*
PIC 9	Zoned Numeric	Z
PIC 9 COMP	Binary Numeric	B
PIC 9 COMP-3	Packed Decimal	P

Table 13-3 Cobol Field-Type Designators

5. For packed-decimal fields, the number of digits in the field is usually shown on the record layout. If so, use this number for the DEX field specifier. If the field length is shown in bytes, use the following formula to convert it to the number of digits:

   # of digits in field = (# of bytes in field * 2) - 1

6. To skip a packed field (not transfer it to the output file), it is necessary to know the length of packed fields in bytes. Use the following formula to calculate the length in bytes if only the number of digits is known:

   # of bytes in field = (# of digits in field / 2) + 1

13.2.3 Step 3. Creating the DEX Control File

The control file is the file which contains instructions for DEX. The control file must be a standard ASCII text file, containing no formating codes (such as font style, bolding, underscoring, etc). Such files can be created by writing the file using a word processor and saving it in an ASCII format.

NOTE

Do not use a control file which has been saved in the word processor's internal format—DEX will not be able to cope with such a file.

The control file has two parts:

1. A field specification section where the fields in the record are defined;
2. An (optional) header section containing statements which control DEX's overall behavior when extracting from a file.

13.2.3.1 Creating the Field Specification

NOTE

> This section does not describe all of the options for field specifiers. For more information, see Section 13.3 beginning on page 13-11.

Each field is specified on a separate line. Each specifier consists of a single letter which gives the field type, followed by a number which gives the length of the field in bytes or digits. No space is allowed between the letter and number, and a # sign can be used to signify a comment (DEX ignores any text in the control file after a # character). In the preceding example, the Name field consisting of 25 EBCDIC alphanumeric characters would be specified as:

```
E25 # name
```

13.2.3.2 Skipping Data in the Tape Record

Any part of the tape record may be skipped by specifying the part to be skipped as a special DEX *skip* field. This is done by specifying field type **X**, which tells DEX to skip a specified number of bytes (characters) in the record. Since the length of skip fields is always given in bytes, determine the length of packed and binary fields using the formulas given previously.

Example:
To skip seven bytes of data between a name field and a date field, part of the control file might look like this:

```
E25 #name
X7  #skip some stuff here
P6  #date
```

13.2.3.3 Creating the Header Section

NOTE

> This section does not cover all of the header statements. For more information, see Section 13.3 beginning on page 13-11.

Statements in the control file's header section control DEX's overall behavior during the extraction process. Each statement goes on a separate line, and all header statements must come before any field specifiers in the control file. Even though the header section is optional, a FIELDS statement must precede any field specifiers in the control file. Several statements are available; this section describes only the three most commonly used.

1. **dbase output** - Tells DEX to create a comma-delimited output file. Each field is separated from the next by a comma, and text fields (those defined as types **A** or **E**) will be surrounded by double quotation marks. (This is not a Dbase file per se, but Dbase (and other databases) can easily import these files.)

2. **trim spaces** - Tells DEX to trim away trailing spaces from any text fields (types **A** or **E**). Without this statement, all trailing spaces will be kept in the output file.

3. **trim zeros** - Tells DEX to trim away leading zeros from any numeric fields (zoned, binary or packed). If this option is not specified, a field defined as a five-byte zoned field containing the value 7 would be converted to `00007`. Specifying this option would save space by removing the four leading zeros.

13.2.4 Step 4. Checking the Control File

If the control file contains only a few defined fields, this step may be skipped. If the control file contains more than about a half-dozen fields, it should be checked by generating a field report before actually doing the data extraction. This report can be compared to the record layout to make sure they agree.

Perform the following instructions to generate the field report:

1. Run DEX using the following command syntax at the DOS prompt:

   dex /cf*control-file* **/rf***report-file*

   replacing *control-file* with the name of the control file and replacing *report-file* with the name of the report file to be created (any previously-existing file of this name will be overwritten by the new report). For example, typing

   ```
   dex /cfytd.ctl /rfytd.rpt
   ```

   will create a field report in the file YTD.RPT from the control file YTD.CTL. A sample field report is shown on page 13-18.

2. Compare the report to the record layout and confirm that the locations and types of fields match. If there are any differences, edit the control file and re-run DEX to generate a new report. When the report checks out OK, continue with the next step.

13.2.5 Step 5. DEXtracting the Data

Syntax: **dex /cf***control-file* **/of***output-file* [*options*]

Replace *control-file* with the name of the control file, and replace *output-file* with the name of the output data file to be created. Any previously existing file of this name will be overwritten by the new data file.

Example:

```
dex /cfytd.ctl /ofytd.dat
```

will create a data file, YTD.DAT containing the data extracted from tape, using the control file YTD.CTL.

13.2.6 An Example Using DEX

An example of a record layout is shown in Table 13-4. While the exact terminology may differ, most layouts use terms to describe field types which will be close enough to those shown here to be easily identified.

For this example, assume that a report showing credit limits for customers is to be generated showing everything *except* the Curr.Bal. field. The report is to include only those customers

RECORD POSITION	FIELD NAME	LENGTH	FORMAT	DATA REPRESENTATION
001-004	Cust.No.	4	Packed	999999
005-010	Curr.Bal.	6	Packed	$$$$$$.cccc
011-016	Credit.Lim.	6	Packed	$$$$$$.cccc
017-041	Name	25	Alpha	XXX...XXX
042-066	Address	25	Alpha	XXX...XXX
067-086	City	20	Alpha	XXX...XXX
087-088	State	2	Alpha	XX
089-093	ZIP	5	Num	99999
094-103	Phone	10	Num	(999)999-9999
NOTE: Data is in EBCDIC.				

Table 13-4 Sample Record Layout

within a certain range of ZIP codes. The data is going to be imported into a database, so it needs to be in comma-delimited ASCII format.

1. Interpret the record layout. Make a list of fields (in DEX shorthand notation) on paper. That list should look something like that in Figure 13-5.

```
p 6    - cust #
x 6
p 6.4  - credit limit
e 25   - name
e 25   - address
e 20   - city
e 2    - state
z 6    - ZIP
z 10   - phone #
```

Table 13-5 DEX Field List

2. Create the DEX control file. The field specification part of the file is simple the preceding list but with no spaces between the field-specifier letters and the length numbers. If inserting comments, be sure to precede all comments with a # character.

3. Use DEX's REJECT statement to select individual records.

   a. First, find the field to be compared. Since this example calls for only a certain range of zipcodes, the "ZIP" field is the field to be compared.

   b. Specify the field by number in the REJECT statement. In this case, the field number is seven, even though it is the eighth entry in the list. This is because skipped fields do not count when numbering fields for REJECT.

      For example, assume the desired range of zipcodes is 80000 to (but not including) 90000. Use the following statements:

```
reject 7 lt 80000

reject 7 ge 90000
```

4. Finally, to put the data into the proper format for the database, include this statement:

```
dbase output
```

5. The complete DEX control file should read:

```
dbase output
reject 7 lt 80000
reject 7 ge 90000
fields
P6                  # cust #
X6                  # don't want this field
P6.4                # credit limit
E25                 # name
E25                 # address
E20                 # city
E2                  # state
Z6                  # ZIP
Z10                 # phone #
```

6. To verify everything is correct, generate a DEX field report. Assuming that this control file is called EXAMPLE.CTL, type the following command line:

```
dex /cfexample.ctl /rfexample.rpt
```

13.2.6.1 Record Layout Explanation

1. **P6** - Since this is a packed field, specify the length to DEX as the number of digits in the field. (Notice that the record layout shows the length of this field in bytes.)

2. **X6** - Tells DEX to skip the next six bytes from the tape record. This is one of the points of confusion having to do with packed fields: While the length of packed fields in DEX is specified as a number of digits, in order to skip a packed field, its length must be specified in bytes. The length given on the record layout (6) is the length in bytes, and this entry tells DEX to skip this field.

NOTE

> Refer to page 13-5 for formulas which convert between number of digits and length in bytes for packed fields.

3. **P6.4** - This is a packed field with an embedded edit character. The record layout shows **$$$$$$.cccc** for this field, indicating that data is interpreted as having six digits to the left of the decimal point, a decimal point, and four digits to the right of the decimal point. The punctuation (decimal point) does not actually appear in the data and is sometimes referred to as an *implied* decimal point. By placing a period in the DEX field specifier, a decimal point will appear in the extracted output, with six digits to its left and four to its right. Also notice that the record layout gives the length of this field as six. That is the length in *bytes*, but DEX needs the length in number of *digits*, which in this case is 10 (6 + 4).

4. **E25, E25, E20, E2, Z6,** - These are all text (alphanumeric) fields. Since the record layout identifies the tape data format as EBCDIC, use the **e** field

specifier. If the data is ASCII, use the **a** field specifier. If uncertain as to the character set, view the tape file as described elsewhere in this guide.

5. **Z10** - This is a zoned field, with one numeric digit per byte. Notice that the record layout shows this field as (999)999-9999, but the field list only shows `Z10`. The parentheses and dash show the *interpretation* of the data in the field. The actual data contains only ten digits (counting the number of 9's), so that's what must be specified to DEX.

13.2.7 DEX Command-Line Options

As explained previously, DEX takes its instructions from statements in the DEX control file which you created. In addition, the way DEX works can be modified by specifying options on the command line (the command actually given to run DEX). These options are different from those in the control file and can go anywhere on the command line. Except for the **drive=***drive-specifier* option, each option must be preceded by a forward slash (/) and separated by at least one space.

IMPORTANT NOTE

The following options only apply when DEX is used for data extraction, not when it is used to generate a field report.

1. **AP** - This option tells DEX to append to the output file if it already exists. Normally, an existing file will be started from scratch each time DEX is run.

2. **drive=***drive-specifier* - This selects the tape drive to use if more than one tape drive is installed in the system. See Chapter 7 for a complete description of this option.

3. **NR** - This tells DEX not to rewind the tape after the extraction is finished. The tape will be left at the start of the next file (if any). Use this option to extract multiple files from a tape.

4. **OV** - This option tells DEX to overwrite the output file without asking permission if it already exists. If this option is not specified and a named output file exists, DEX will ask permission to overwrite the file, and will allow you to terminate the program at that point without overwriting it.

5. **SF#** - If given, tells DEX how many tape files (#) to skip before starting to extract from the tape. This option can be used to extract from labeled tapes, or to select which file to extract from a tape.

6. **SR#** - If given, tells DEX how many records (#) to skip before starting to extract from the tape. If file skipping is specified (see following), then tape files are skipped before any records are skipped.

7. **TR# -** If given, tells DEX how many records (#) to extract from the tape. Normally, all records in a file from the starting position will be transferred.

8. **/XE***translation-table* - This tells DEX to use the named translation table file to translate text data from EBCDIC to ASCII instead of the built-in table. You may need to do this if you are reading tapes made on a computer system which uses a nonstandard version of the EBCDIC character set. Remember that this only affects text fields in the input data file (those fields which are defined using the **E** field specifier).

13.3 The DEX Control File

This section explains all DEX control file header statements and field specifiers detail.

13.3.1 DEX Control File Header Statements

All of the following statements are optional except the **fields** statement, which must be the last statement in header section. Each statement must be given on a separate line.

The following statements are available:

1. **`binary = bigend`**
2. **`binary = littlend`**
3. **`trim spaces`**
4. **`trim zeros`**
5. **`dbase output`**
6. **`xlate 'x' = 'y'`**
7. **`reject`** *`field# condition-value`*
8. **`fields`**

These statements are explained in the following sections.

13.3.1.1 binary= Statements

These statements tell DEX how to interpret any binary fields in the file (defined using the **b** specifier). There are two types of binary fields, referred to here as *bigendian* and *littlendian*. Bigendian fields are arranged with the *most*-significant bits left-most in the field, while littlendian fields have the *least*-significant digits left-most.

- Data produced on most mainframes is bigendian.
- Data written by IBM PCs and compatibles, and DEC VAX and PDP machines, is littlendian.
- Data written on 68000-based systems (e.g. Macintosh, NeXT) is bigendian.

There are exceptions to the above rules (e.g., some UNIX systems offer a choice of which binary type to use).

13.3.1.2 trim spaces Statement

trim spaces tells DEX to trim trailing spaces from *text* fields. A text field is any field defined as ASCII (A) or EBCDIC (E) such as names, addresses, etc. Without this statement, all trailing spaces will be kept in the output file. If a text field is only one character long and the field contains a space, then no trimming is done and the single space is written to the output file.

13.3.1.3 trim zeros Statement

trim zeros tells DEX to remove all leading zeros in all numeric fields. This applies to binary (B) and zoned decimal (Z) fields, and to packed (P) fields if no "edit" characters are inserted in the field definition. if there are any edit characters in the packed field, no leading-zero trimming takes place for that field.

As with space-trimming, if the value of any field is zero, then a single zero will be written to the output file.

13.3.1.4 dbase output Statement

dbase output tells DEX to produce an output file which can be read by dBASE or similar application. DEX inserts commas between each field in the output file, and encloses all ASCII and EBCDIC text fields with double quotes (" "). Such files are called *comma-delimited ASCII* files.

This statement does not produce database (.DBF) files. It produces text files which can be imported by database or similar products. These files are readable by programs written in BASIC, C, and other high-level languages on the PC.

13.3.1.5 xlate *'x'* = *'y'* and xlate *'\nnn'* = *'\mmm'* Statements

These translation statements tell DEX to substitute the character on the right side of the statement each time the character on the left side occurs in the extracted data. The first form specifies the characters as literal text characters; the second form specifies the characters as ASCII values in decimal. You can specify either form in either side.

All translations are done *after* any field translation (such as EBCDIC to ASCII), so the translation characters are always specified as ASCII characters. Only one translation is done on any one character. The first translation statement which matches a character (the left side) will be used.

A control file may contain up to 200 translation statements.

EXAMPLES:

1. `xlate ' '='*'` - "Change all space characters to asterisk characters."
2. `xlate '\13'='!'` - "Change all carriage-return characters to exclamation points."

13.3.1.6 reject *field# condition value* Statement

reject sets up a conditional selection which allows one to select which records go into the output file. The selection logic is very simple: If the *condition* specifier is true, the entire record will be rejected and not written to the output file.

1. *field#* - Tells DEX the number (from 1 to n) of the field to be tested, based on the order of the fields as defined in the control file.

NOTE

> For more information about how DEX numbers fields, see Section 13.3.1.8.

2. *condition* - Tells DEX how to determine whether the record lives or dies in the output file, and is one of the following:
   a. **EQ** - equal to
   b. **NE** - not equal to
   c. **LT** - less than
   d. **GT** - greater than
   e. **LE** - less than or equal to
   f. **GE** - greater than or equal to
3. *value* - Tells DEX what to compare the value of the field to. The value specifier for numeric fields must be a number, but the value specifier for text fields can

be any character(s) including letters, numbers, and punctuation. If the value specifier contains spaces (blanks), the string must be enclosed in quotation marks (" "); otherwise, the string may be entered as-is.

Examples:

a. `reject 5 eq 2001` - "Reject Field 5 if its value is 2001."

b. `reject 8 ne "new york"` - "Reject Field 8 unless it equals *new york*. The quotes are needed because of the included space.

A control file may contain up to 100 Reject statements.

13.3.1.7 fields Statement

fields is required, even if there are no other statements in the control file header. It tells DEX that the following statements are field definition statements. It must be the last statement in the control file header.

13.3.1.8 Field Numbering for REJECT Statements

DEX fields are numbered from 1 to the number of fields defined. For the purposes of the **reject** statement, only those fields which actually transfer data from input to output are numbered. (This makes sense, since only these fields can have **reject** statements attached to them.) This includes the following field types: Z, A, R, B, E, P. *Literal* and *skip* fields are not counted. This numbering scheme is used when DEX generates a field report.

13.3.2 DEX Field Definitions

After the FIELDS statement appears in the control file, fields can be defined in the record. Each specifier must be placed on a separate line. DEX ignores spaces unless they occur within a quoted literal field.

The defined fields must match exactly in both type and length to the tape file which will be read. The field definitions can also add characters to the output record which are not in the input file. Such characters are referred to as *literals*, and can occur anywhere in the record definition. *Edit* characters can also be embedded within packed-decimal field specifiers. An input field can be prevented from being transferred to the output file by using the "skip" field type (see **X**, under ZARBEX Fields, Section 13.3.3.2 beginning on page 13-14), to skip any number of bytes of the input record.

The control file may contain up to 1000 fields, and each field may contain up to 2000 bytes.

13.3.3 DEX Field Specifiers

13.3.3.1 Packed-Decimal Fields

Syntax: `[+]p [d] [edit-chars] [d]`

where *d* indicates a decimal digit from 0 through 9.

Translation:

1. The optional first character is **+**, which will put the sign (plus or minus) of the field's value in that position in the output;
2. The next character *must* be **P**.
3. The **P** can be followed by a decimal digit indicating the number of packed digits.
4. The decimal digit can be followed by edit characters, with no embedded spaces.
5. The edit characters can be followed by another numeric digit.

The total length of the packed field is the sum of all the numeric digits given, which is the number of packed-decimal digits in the packed field. Any edit characters are inserted where specified in the output.

Examples:

1. `P6` - six packed digits;
2. `P4.2` - six digits, four before the period, two after;
3. `P2/2/2` - a date with slashes as edit characters;
4. `+P\P1` - the sign of the number, one digit, the letter **P** (Escape Operator), then one more digit.

13.3.3.2 ZARBEX Fields

So named for their letters, these are the remaining data types available in DEX. They are given in the following form:

`[+]type [d...]`

type is replaced by an ***upper case*** letter identifier for the field, and *d* indicates one or more numeric digits. Valid values for *type* are:

1. **Z** - zoned numeric digits (EBCDIC or ASCII)
2. **A** - ASCII text
3. **R** - raw (no translation)
4. **B** - binary
5. **E** - EBCDIC text (translate to ASCII)
6. **X** - skip (do not transfer) one byte of data from the input record

NOTE

Binary fields can only be defined as being 1, 2 or 4 bytes long, corresponding to the most common binary field sizes. Only fixed point (integer) binary values can be converted; DEX does not support floating point numbers.

The sign character (**+**) is only valid with the numeric types **Z** and **B**. As with packed data, placing the sign character before the field causes the sign (plus or minus) of the value of the field to be placed before the digits in the output file.

Examples:

1. **B4** - Defines a 4-byte (32-bit) binary field in the input record.
2. **X8** - Tells DEX to skip eight bytes of data from the input record.
3. **R12** - Defines a "raw" (untranslated) 12-byte field in the input record.

If you have zoned-numeric fields in which you want to embed "editing" characters (as in packed fields), you can split the zoned field into as many pieces as you need and simply insert the editing characters where desired.

For example, if you have a seven digit zoned field which is a telephone number, you can have this field output in standard telephone number format by splitting it as follows:

```
Z3-Z4
```

The - is a literal character which is transferred directly to the output file.

A full phone number with area code could be handled thus:

```
(Z3)\ Z3-Z4
```

The backslash followed by a space inserts a space after the right parenthesis of the area code.

13.3.3.3 Literal Characters

Any non-whitespace characters in the control file which are not part of the following list are inserted in the output stream literally:

```
P p, Z z, A a, R r, B b, E e, X x, +, \, /, #, "
```

To insert whitespace characters, use either the *escape* or *quote* operators.

Example:

`P3.2 *** P3.2 *** B4` - This defines three input fields, with three asterisks placed between each field in the output record.

13.3.3.4 Escape Operator (\ )

Placing a backslash (\ ) before any character forces that character to the output record, even if that character is a DEX control character (unless it's inside a quoted string, in which case the backslash and the character are both placed in the output record). The only place the escape operator is invalid is when nothing follows it except the end of a line.

Example:

`\X\X\X` - places three 'X's in the output record.

13.3.3.5 Quote Operator

Enclosing any text with pairs of double quote characters (" ") inserts them into the output stream literally. To embed the quote character itself inside a quote, use two of them in a row with no intervening space.

Quoted text, unlike other DEX field specifiers, can span line breaks (i.e., line endings are ignored).

Example:

```
"name: " a25 " address:" a25 " city: " a12
```

This defines three ASCII text fields in the input record. Literal text between the vertical bars, including spaces, will be inserted into the output record.

13.3.3.6 Comment Delimiter

DEX ignores any text which appears after the comment delimiter (#) and keeps it out of the output stream. This works only for field definition statements in the control file, and not for header statements. The comment delimiter is only good for a single line.

Example:

```
e25 #field 12: customer name
```

This identifies the field to live human beings reading the control file.

13.3.3.7 Whitespace Characters

Whitespace consists of space and tab characters. DEX ignores both of these and keeps them out of the output stream, unless they are preceded by a backslash \ (escape character).

Example:

```
\TAB-KEY
```

Tells DEX to insert an actual tab character into the output stream.

13.4 Packed and Binary Data Types

These two data types cause most of the confusion associated with data extraction. Some tape files do not contain either one of these types. In general, if the entire data record consists of text which can be read on-screen when viewing a tape, or during a "dump" of tape data, then that record contains no packed or binary fields.

Packed and binary fields may or may not be clearly identified if present in record layouts. Sometimes the layout will actually describe a field as "packed" (but rarely will anything as obvious as "binary" be seen).

13.4.1 Packed Data

The term *packed data* refers to two digits of data placed into the space normally required for one. (A digit is a numeric character—the number 59,071 has five digits). Unpacked data simply puts each *digit* in a separate byte. Since each data byte is eight bits, and since numeric digits (0-9) only require four bits, one could save space by putting two four-bit digits into an eight-bit byte.

Examples:

The unpacked number shown here 3,571,209 (EBCDIC zoned-decimal characters) takes up seven bytes:

F3	F5	F7	F2	F1	F0	F9

The same number only takes up four bytes when packed:

35	72	10	9F

The last digit is the sign of the number, positive or negative. The sign *nibble* (half a byte) can only have one of the following values:

SIGN NIBBLE	MEANING
F, C	Positive
D	Negative

When looking at a hexadecimal display of tape data, here's how to tell if a field contains packed data:

a. Each nibble except the last (right-most) nibble can only contain the digits 0 through 9;

b. The right-most nibble can only contain the values **C**, **D** or **F**.

The following are valid packed numbers: **00**, **19**, **87**, **02**, **4F**, **20**, **0D**.

The following are not valid packed numbers: **01**, **23**, **4A**, **0B**, **23**, **5F**.

13.4.2 Binary Data

Binary data is problematic, because binary fields can contain anything.

In order to correctly process files which contain binary data, the record layout must identify the binary fields. As Table 13-3 on page 13-5 shows, certain COBOL field PICtures will identify binary (COMP) fields. However, a COBOL record layout is usually not available.

One way to identify binary fields is when it is the only possible way to pack numbers of a certain size into a given length field. As previously explained, packed fields are so named because more data is packed into fewer bytes. Binary fields can be thought of as "super-packed" fields, since even more digits will fit in a smaller space. The following table shows this packing:

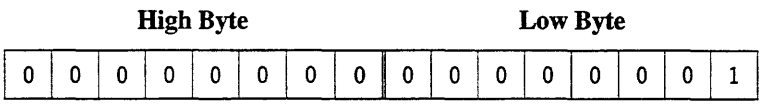
# OF DIGITS IN BINARYFIELD	BINARY FIELD LENGTH IN BYTES
1-4	2
5-9	4

Example:

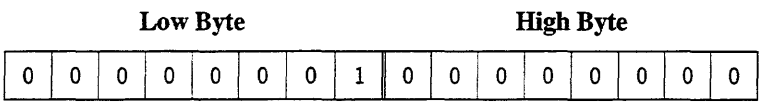
The record layout shows a field containing four numeric digits, but it only occupies two bytes in the data record. It is probably a binary field (a four-digit packed field would take three bytes, not two.) These are the technical terms. The vernacular is *MSB-first* (most-significant byte) and *LSB-first* (least-significant byte).

Binary data is interpreted as either *bigendian* or *littlendian*. The following diagrams show the same binary number (the number 1, for the sake of simplicity) as a sixteen-bit, two-byte field:

BIGENDIAN:



LITTLENDIAN:



Unfortunately, unless the value of a particular field is known, there is no way to tell which interpretation is correct by looking at binary data. The source of the data must be determined. As it turns out, most binary data on tape is bigendian.

The following are some rules of thumb:

1. Data produced on most mainframes is bigendian.
2. Data written by IBM PCs and compatibles, and by DEC VAX and PDP machines, is littlendian.
3. Data written on 68000-based systems (e.g. Macintosh) is bigendian.

There are exceptions to the above rules (e.g., some UNIX systems offer a choice of which binary type to use).

NOTE

The interpretation of binary data in DEX is controlled by the **binary=** control file header statement. See Section 13.4.2 on page 13-16.

13.5 DEX Examples

13.5.1 Generating a Field Report

The following command line will generate a field report:

```
dex /cfcensus.ctl /rfcensus.rpt
```

DEX will read control file CENSUS.CTL and create report file CENSUS.RPT, showing the format of the file according to the field definitions. This file is an ASCII text file which can be displayed or printed using DOS commands.

13.5.2 Transferring from Unlabeled Tape

These two runs of DEX will read 12,000 records from each of two tape files into the disk files MAIL1.DAT and MAIL2.DAT, using the same control file, MAIL.CTL:

a. `dex /cfmail.ctl /ofmail1.dat /tr12000 /nr`

b. `dex /cfmail.ctl /ofmail2.dat /tr12000`

The first example leaves the tape at the end of the file since the **/NR** (no rewind) option was used; the second example starts at the next file.

13.5.3 Transferring from Labeled Tape

These three DEX runs read three files from a labeled tape.

a. `dex /cfcensus.ctl /ofcensus1.dat /sf1 /nr`

b. `dex /cfcensus.ctl /ofcensus2.dat /sf2 /nr`

c. `dex /cfcensus.ctl /ofcensus3.dat /sf2`

The first example reads the first data set on the tape into the disk file CENSUS1.DAT; the **/SF1** skips the volume and header labels at the beginning of the tape. The second and third examples read the next two files on the tape, skipping the header labels between data sets each time.

For more information about labeled tapes, see Section 19.3.

13.6 Sample DEX Field Report

Figure 13-2 illustrates a sample DEX field report.

```
          DEX options selected:
                              * binary type:  BIGEND (IBM/Motorola)
                              * Comma-delimited output
                              * Don't trim zeros
                              * Don't trim spaces

          - Reject Conditions -
                              * Reject if field # 7 is less than 80000
                              * Reject if field # 7 is greater/equal to 90000
```

FIELD #	TYPE	INPUT LOC.	INPUT LEN.	OUTPUT LOC.	OUTPUT LEN.	# DIGITS	OUTPUT FORMAT
1	PACKED	1	4	1	6	6	dddddd
-	(skip)	5	6	8	-	6	
2	PACKED	11	6	9	11	10	dddddd.dddd
3	EBCDIC	17	25	21	25	25	ccccccccccccccccccccccccc
4	EBCDIC	42	25	49	25	25	ccccccccccccccccccccccccc
5	EBCDIC	67	20	77	20	20	cccccccccccccccccc
6	EBCDIC	87	2	100	2	2	cc
7	ZONED	89	6	105	6	6	dddddd
8	ZONED	95	10	112	10	10	ddddddddddd

```
          Input record length: 104 bytes

          Maximum output record length: 123 bytes
```

Figure 13-2 Sample DEX Field Report

14. Using COPYTAPE.EXE

COPYTAPE.EXE lets you make copies of entire tapes. COPYTAPE has the following capabilities which TCLONE (see Chapter 15) lacks:

1. It can handle *multivolume* tapes, both as source and destinations (a multivolume tape set is a single set of files which spans two or more reels or cartridges.) Both labeled and unlabeled multivolume tapes are handled correctly.

2. It can *reblock* data while copying. Large blocks on the source tape can be broken into smaller blocks, or small blocks can be merged into larger blocks.

COPYTAPE can only be used to make a single copy of a tape. If you need to make multiple copies of a single tape, use TCLONE (described in Chapter 15).

14.1 Requirements for Using COPYTAPE

COPYTAPE will use more than two drives if they are available (see Section 14.3). Because COPYTAPE copies files directly from one tape to another without creating an intermediate disk file, you must have at least two tape drives installed in your system to use it.

Syntax: **copytape** [*options*]

COPYTAPE options are explained in Section 14.5.

After displaying the program's ID information, you should see a display like that in Figure 14-1.

```
#  Drive    Type         ID:LUN     Controller     (#)  Use
-------------------------------------------------------------

1: Qualstar 34xx   9-track   4:0    Adaptec SCSI   (2)
2: DEC TZ30        TK-50     6:0    Adaptec SCSI   (2)

Select the number (1st column) of the SOURCE drive:
```

Figure 14-1 COPYTAPE Screen 1

1. Type the number (at the left) of the *source* drive—that is, the drive which has the tape from which you want to read. (For this example, the TK-50 drive will be the source.) Press ENTER to make your choice.

2. Next, you'll see basically the same display again, as shown in Figure 14-2.

```
#  Drive    Type         ID:LUN     Controller     (#)  Use
-------------------------------------------------------------

1: Qualstar 34xx   9-track   4:0    Adaptec SCSI   (2)
2: DEC TZ30        TK-50     6:0    Adaptec SCSI   (2)  Src

Select the number (1st column) of the DESTINATION drive:
```

Figure 14-2 COPYTAPE Screen 2

3. This time, type the number of the *destination* drive, which is the drive which has the tape to which you want to write. Notice that the source drive has been identified as such in the right-most column. Press ENTER to make your choice.

NOTE

Make sure that both tapes are loaded and ready to go *before* you select the destination, because that's when the copy process will start.

4. The progress of the copy operation will be reported on screen. What you see depends on whether the tape you're copying is *labeled* or *unlabeled*. For unlabeled tapes, you'll see something similar to this:

```
Source tape is unlabeled

Transferring file 1:

101 source blocks -> 101 destination blocks.

End of last tape: copy operation finished.
```

5. Since labeled tapes contain volume and file information at known locations in header labels, you'll see more information, similar to this:

```
Source tape is labeled  (IBM OS/VS or DOS/VSE)
Volume serial number:   000001
Owner ID:               "MICROTECH "

Transferring file 1  ('LITTLE.FILE.1    '):

5 source blocks -> 5 destination blocks.

End of last tape: copy operation finished.
```

6. In either case, COPYTAPE will report the number of blocks it copied with the following differences:

   a. Unlabeled tapes - the block count is the total number of blocks on the tape.

   b. Labeled tapes - the block count is the total number of blocks in each data file; this does not include the volume, header and trailer labels.

14.2 Copying and Creating Multivolume Tapes Using COPYTAPE

COPYTAPE allows both the *input* tape (the tape you're copying) and the *output* tape (the tape you're creating) to span multiple volumes (reels or cartridges). Such tapes are known as *multivolume* tapes.

Typically, multivolume tapes are *labeled* tapes, where the set of tapes and each file on the tape are identified by special, standard header and trailer labels. COPYTAPE can also make *unlabeled* multivolume tapes. However, this is an area where few, if any, standards exist. You really need to know what you're doing in order to create a usable tape. More information on this aspect of copying tapes is given in Section 14.2.2.

14.2.1 Copying a Labeled Tape

If you're copying a labeled tape, you don't need to do anything special to read or create multivolume tapes. COPYTAPE automatically senses when it has reached the end of an input or output tape, and takes the appropriate action:

1. End-of-tape on input: When the end of a reel or cartridge in a multi-volume set is reached, COPYTAPE reads the trailer labels of the tape to determine if this is the end of the set, or if another tape follows. If another tape follows, COPYTAPE will stop copying. It will then take one of two actions, depending on whether you have more than one source drive:

   a. If you have only one source drive, you'll see the following:

   ```
   End of source tape reached. Rewinding & unloading tape:

   Mount next source tape and hit any key
   to continue
       (<Esc> to cancel copy operation):
   ```

   b. At this point, you can continue copying by putting the next tape in the drive and pressing ENTER when the tape is ready. To cancel the copy operation instead, press ESC at this prompt.

   c. If you have more than one source drive, COPYTAPE will automatically switch to the next source drive in sequence and start reading the tape in that drive. It will *not* prompt you to put the next tape in. See Section 14.3 (*Using Multiple Source Or Destination Drives With COPYTAPE*) for a more detailed explanation.

2. End-of-tape on output: When the end of a tape being written is reached, the same action just described will be taken, except that the tape prompted for or switched to will be the next destination tape.

14.2.2 Copying Multivolume Unlabeled Tapes

Handling multivolume *unlabeled* tapes poses some problems. The biggest problem is that no real standards exist which define how multivolume unlabeled tapes should be written. Many *ad hoc* standards exist. One scheme might use two consecutive filemarks to indicate the end of a continued volume, and three consecutive filemarks at the end of the last tape. Often, multivolume unlabeled tapes are simply "normal" data tapes with one filemark between files and two consecutive filemarks at the logical end-of-tape. In this case, the person reading the tapes must tell the computer system when the last tape has been read.

Since there are no standards, COPYTAPE was designed to write multivolume unlabeled tapes, but not to read them. This means that any tape, labeled or not, can be split across multiple output volumes. If the tape is unlabeled, then the continuation tape will simply start with the next data in sequence, and no header or trailer records are written by COPYTAPE. This capability can be used to split a high capacity tape (a DAT tape, for example) across several smaller output tapes (nine-track tapes, for example).

Table 14-1 summarizes the various aspects of COPYTAPE's multivolume tape handling for both labeled and unlabeled tapes.

14.3 Using Multiple Source or Destination Drives with COPYTAPE

COPYTAPE can use multiple tape drives to both read and write tapes. Using more than one source or destination drive is useful when you know you're going to be reading or creating a multivolume tape set. When using multiple drives, COPYTAPE automatically switches to the next tape drive

ASPECT	LABELED TAPE	UNLABELED TAPE
Ability to read and write multivolume tapes	Can read and write multivolume tapes	Can write multivolume tapes, but cannot recognize multivolume tapes on input
Header & trailer records	On input, recognizes special labels which indicate that another tape follows. On output, writes these same labels when a new destination tape is started. Transforms labels so that the correct type and sequence of labels and data blocks is generated; copy tapes always contain exactly the same data in each tape file.	No header or trailer labels are recognized or generated. If the source tape contains any header or trailer records, these are simply treated as data and copied without special treatment. No special data is generated if a new output tape is started; tape data simply continues across the break.
Tape termination	The correct tape termination is generated for continuation tapes (when another volume follows) and the last (or only) tape in a set. Continuation tapes will terminate in one or no filemarks (for nine-track or cartridge tapes, respectively); last tapes will likewise terminate in two or one filemark(s).	All tapes terminate the same, regardless of whether they are continuation or last tapes. All nine-track tapes will end in two filemarks, while all other cartridge tapes (with the exception of TK-50 and 3480 cartridges) will end in one filemark.

Table 14-1 COPYTAPE Aspects

in sequence without prompting. If you have enough tape drives for all the tapes you expect to read or write, you can set up all of the drives and then let the copy operation proceed unattended.

Multiple tape drives can be selected in two ways:

1. By picking drives from the lists of available drives displayed using COPYTAPE's normal interactive mode.
2. By specifying all drives (source and destination) to be used on the DOS command line when running COPYTAPE. If this is done, then interactive drive selection, as described above, is not possible.

You could specify the source drives on the command line and pick the destination drives interactively, but it is generally easier to pick both the source and destination drives, one at a time interactively from the displayed list.

IMPORTANT NOTE

Multiple drives are used in the order in which they were selected.

As drives are selected, COPYTAPE builds two internal lists of drives to use for source and destination. Drives are added to this list in the order in which they were selected. When the end of a tape is reached, either on input or output, the next drive in the appropriate list is the next one used. These lists are *circular*, meaning that when the last drive in the list is reached, the next drive to be used will be the first drive again.

Both source and destination can be either single or multiple drives. You can have one drive for the source and multiple drives for the destination, or vice versa.

14.4 Reblocking Tape Data Using COPYTAPE

COPYTAPE is uniquely capable of *reblocking* data from tape to tape. This simply means that the output tape will have a different block size than the input tape.

14.4.1 About Fixed and Variable Length Blocks

All tape data is written in blocks, each being a specific size. They may all be the same size (fixed length blocks), or they may be different sizes (variable length blocks). Both fixed length and variable length block tapes can be reblocked. When a tape is reblocked, the copy (output tape) will have variable length blocks, all of the specified size. This sounds paradoxical, but is a common situation with magnetic tape. Before proceeding, some explanation of fixed versus variable block tape drives is in order.

Some types of magnetic tape media (mainly nine-track reels) have always been written with variable length blocks. Other media (mainly QIC) have historically been written with only fixed length blocks of 512 bytes each. The reasons for this have to do with physical limitations of the tape drives themselves. The point is that two different types of drives have evolved. Even today, some QIC drives can only write fixed length 512-byte blocks, while many nine-track drives can write blocks of any length.

Because a variable length block tape may in fact have blocks which are all the same size, the terms *fixed* and *variable* should be though of as having to do with the physical capabilities of the tape drive being used. For instance, a nine-track tape could be written, on a drive which is capable of writing variable length blocks, with blocks which are all 17,000 bytes long. If such a tape were copied to a QIC cartridge with blocks of the same size, though, it would not be readable on some fixed length QIC drives—even though it is (logically) a fixed length tape! The reason is that fixed length QIC drives generally expect (and require) blocks to be exactly 512 bytes long.

Of all the commonly used Qualstar tape drives, only the Tandberg TDC-36XX series tape drive is limited to fixed length, 512-byte blocks. (These drives are sometimes identified as *QIC-24* drives, since this is the highest density of cartridge which can be written on the TDC-3620. The TDC-3640 and TDC-3660 drives can write higher densities, but these are also limited to fixed length blocks.)

Further confusion stems from the wide variety of tape drives supported by Tapestar. Most of these drives are true variable length block drives which can read and write blocks of any size, but some are like the old QIC drives and can only handle fixed 512-byte blocks. In order to use COPYTAPE's reblocking option, the output drive must be capable of writing variable length blocks.

14.4.2 Reblocking Data with Fixed Length Drives

To reblock tape data where the destination drive is only capable of writing fixed length 512-byte blocks (like the Tandberg TDC-3620 QIC-24 drive), use the **/DB#** option (described in Section 14.5). When using this option, remember that this will create a tape with true fixed length blocks. The last block will never be a short block, and if the incoming data does not divide evenly into the fixed block size, the last block will contain "garbage" at the end.

14.4.3 Reblocking Data with Variable Length Drives

To reblock tape data when using variable block drives (such as nine-track tape), use the **/OB#** option (described in Section 14.5). This option will create blocks of "#" bytes on the output tape as follows:

1. Labeled Tapes - Only the data in tape files will be reblocked. Headers and trailers are never reblocked, and will always be written correctly.

2. Labeled Tapes - The selected block size will be written into the tape's HDR2 labels, if any are present (IBM DOS/VSE labeled tapes don't have HDR2 labels). The total block count for each file will be written into the file's EOF1 labels.

3. No tape data will be lost, on either labeled or unlabeled tapes. Data is simply blocked differently.

4. COPYTAPE has no idea what the actual record length of tape data is, and so it cannot check whether data is blocked correctly. You are responsible for selecting the new block size so that data records are not incorrectly "chopped." Choosing the wrong block size may cause records to incorrectly span blocks.

5. All tape file data will be written at the specified block size, except for a possible short block at the end to make up the correct length. The last block will not be padded out to the specified block size. This is standard, accepted practice with magnetic tape data.

14.5 COPYTAPE Parameters

This section describes all of the command-line parameterss available for COPYTAPE. These parameters tell the program to take special actions and must be given on the DOS command line when you run the program.

Syntax: **copytape** [*drive-selectors*] [*options*]

14.5.1 Drive Selectors

The drive selectors allow you to specify drives for source and destination from the command line. Doing so will bypass the drive selection process previously described, where you select drives from a list of available drives.

NOTE

The option of choosing drives from the command line is intended mainly for batch operation of COPYTAPE. In most cases it will be easier and less confusing not to use this option, and to simply select drives from the list displayed.

Drive selection from the command line is done using selectors in the following forms:

src=*tapestar-drive-selector*[**,***tapestar-drive-selector*] [...]

or

dest=*tapestar-drive-selector*[**,***tapestar-drive-selector*] [...]

(where **src** is for *source* (input) and **dest** is for *destination* (output). Each *tapestar-drive-selector* is a standard drive selector, as described in Chapter 7 (*Selecting Drives*).

NOTE

No spaces are allowed anywhere inside of the **src=** or **dest=** specifier, while at least one space is required between all options on the command line.

Notice that you can specify multiple drive selectors for both "src" and "dest" by stringing together multiple Tapestar drive selectors separated by commas. For example, the COPYTAPE command line

```
copytape src=9t:1,9t:2,9t:4 dest=3480:1
```

sets up three source drives—the first, second, and fourth nine-track drives in the system—and one destination drive, the first (or only) 3480 drive in the system.

See Section 14.3 for a more detailed explanation of how multiple tape drives are handled.

14.5.2 COPYTAPE Options

All of the following options require a forward slash (/) before each option:

1. **DB#** - Destination tape will be written with fixed length blocks of # bytes each. This option forces the output tape to be written in the fixed length block mode as previously described.

   This option cannot be used with the **/SB#** option.

2. **DN** - Do not rewind the destination tape drive after the copy operation. This will leave the tape ready to be appended to.

3. **format**=*format/density-specifier* - Specifies the tape format or density to be used for the output tapes. Using this option forces the selected density to be used, which may be different than the default format or density of the destination drive. The format specifier can be any of the following, but it must apply to the tape drive being used as shown in Table 14-2 (i.e., you can't specify a QIC format for a nine-track tape drive):

QIC DRIVES	NINE-TRACK DRIVES
QIC-11	800
QIC-24	1600
QIC-120	3200
QIC-150	6250
QIC-320	-
QIC-1000	-

Table 14-2 Format/Density Specifier

   Other types of tape drives (3480, TK-50, etc.), have only one possible format-/density which can be used.

4. **L** - Do not write labels to destination when processing labeled tapes. This will create an unlabeled tape when copying labeled tapes.

   This option will cause COPYTAPE to fail with an error message if the source tape does not contain fixed length blocks. If this happens, retry the copy operation without this option.

5. **OB#** - Reblock variable length output to # bytes/block. This option forces the tape data to be reblocked as described in Section 14.4.

6. **SB#** - Specifies that the source tape has fixed length blocks of # bytes each. This option is provided mainly for the sake of speed. Tapes can be read and written in either a variable length block or a fixed length block mode. Tapes written with fixed length blocks can be read in either mode; however, it is much faster to read such tapes in fixed length mode. Typically, fixed length blocks

are found on cartridge tapes (notably QIC tapes) and are usually (though not always) 512 bytes long. This option cannot be used with the **/DB#** option.

7. **SN** - Do not rewind the source tape after the copy operation.

8. **V** - Do not validate volume serial numbers (VSNs). Use this option if you get error messages which tell you that the VSNs are incorrect when copying multivolume tapes. This is because some systems which create tapes don't put the correct numbers in the VSN fields.

15. Using TCLONE.EXE

TCLONE is used to copy a tape directly to another tape. If you only have one tape drive on the system,TCLONE will create an intermediate disk file which can then be copied to one or more tapes. When using this method, there must be enough free space on a single disk drive to hold the entire tape contents. Refer to the Tape Capacity tables in Chapter 19. TCLONE will also verify tapes against each other or against intermediate disk files.

Syntax: `tclone [/v] [/ccomment] [filename] [options]`

15.1 Command Line Parameters

15.1.1 TCLONE Verify Switch

The **/V** switch tells TCLONE to verify a tape against either another tape or a disk image of a tape.

15.1.2 TCLONE Comments

You may enter comments by preceding them with a **/C**.

15.1.3 TCLONE Filename

The TCLONE *filename* parameter names the disk file when copying a tape to or from an intermediate disk file, or when verifying a tape against a disk file. Do not specify this part of the command if copying or verifying from one tape to another.

NOTE

If using an intermediate disk file, specify only the path and file name. Do not specify the file extension.

15.1.4 TCLONE Options

The following are valid options for TCLONE (use the forward slash only where7 indicated):

1. **source=***source-drive* - Specifies a tape drive to read from.
2. **dest=***dest-drive* - Specifies a tape drive to write to.
3. **/F** - Forces TCLONE to flag tape read errors and continue copying. Normally, such errors terminate the copy operation. If this option is specified, the block in the output file which contains the block read in error will be flagged with an error message for later identification.
4. **format=***format/density-specifier* - This specifies a tape format or density to be used for the output tapes. Using this option forces the selected density to be used, which may be different than the default format or density of the destination drive. The format specifier can be any of the standard TAPESTAR tape formats (as long as it corresponds to the tape drive being used. You cannot specify a QIC format for a 9-track tape drive).
5. **/L** - ("loud") Tells TCLONE to display block lengths and file numbers while it runs. Normally, TCLONE only displays the number of blocks transferred for each tape file.
6. **/Q** - ("quiet") Tells TCLONE to report nothing to the screen except errors.

7. */S#blocks* - Tells TCLONE to stop after the specified number of blocks have been read from the source tape or from the disk image file.

8. **/U** - Tells TCLONE to unload the tape(s) at the end of the operation if that operation was successful. This only works for drives which are actually capable of unloading tapes on their own.

9. **/X** - Forces TCLONE to set all tape drives being used to the fixed block mode. Use this option if you know for sure that the tape being copied contains fixed length blocks of a size which all tape drives in use can handle as fixed length blocks.

   Using this option can speed up the copying operation considerably, since most tape drives will run much faster in the fixed length block mode, especially when copying tapes with small block sizes.

To halt the program while it is running, press ESC.

Use Table 15-1 to choose the correct command line parameters for TCLONE, depending upon the desired activity.

DIRECTION (COPY OR VERIFY)	PARAMETERS NEEDED
Tape to Tape	**source=***drive*, **dest=***drive*
Tape to Disk	**source=***drive*, *filename*
Disk to Tape	**dest=***filename*, *filename*

Table 15-1 TCLONE Command Line Parameters

15.2 TCLONE Examples

The following examples demonstrate copying from one tape to another:

1. `tclone source=qic:1 dest=qic:2`

   Copies the contents of the first QIC tape to the second.

2. `tclone tape1 source=dat`

   Makes a disk image of the contents of the tape in the first (or only) DAT drive in the system.

3. `tclone tape1 dest=dat /u`

   Creates a tape copy from the previously created disk dataset in the preceding example. The tape will be unloaded when done.

4. `tclone /v dest=dat tape1`

   Verifies a tape copy against the previously created disk dataset in the preceding example.

15.3 Copying Using an Intermediate Disk File

Use this method if you only have one tape drive. When this method is used, two files are created on the disk with the same root name. One has the extension .DAT and the other has the extension .LEN. Be sure to copy both of these files if it is necessary to move the intermediate file.

Your disk must have enough free space to hold the entire tape's data. This can be large, since some cartridge drives can hold more than 1000 megabytes of data if full.

Whenever you create an intermediate disk file, two files are actually created on the disk. Both have the same root name, one with the extension .DAT and the other with the extension .LEN. Be sure to copy both of these files if you need to move the intermediate file.

The following example demonstrates how to copy one tape to another using an intermediate disk file.

First, copy the original tape to a dataset on drive D: called TEMPFILE (TEMPFILE.DAT and TEMPFILE.LEN are created) by typing

```
tclone d:tempfile source=ot
```

Next, copy the data from the temporary file to the new tape by typing

```
tclone d:tempfile dest=ot
```

Finally, delete the disk files by typing

```
del tempfile.*
```

16. Using TAR.EXE

The TAR program included in the TAPESTAR software package reads and writes tapes in the UNIX *tar* format, which is widely used for storing files on UNIX systems. Such tapes are referred to in the UNIX world as *archives*. TAR archives are useful for making backup copies of files or sets of files, and for packaging groups of related files together to move them between computer systems. TAR archives can be stored on any storage medium: Hard disk, floppy disk, or tape.

16.1 TAR Command Line Parameters

Syntax: **`tar`** *`action-specifier [options] [filenames]`*

IMPORTANT NOTE ABOUT CASE SENSITIVITY WITH TAR

> Unlike the other TAPESTAR programs, all command line elements in TAR are ***case sensitive.*** This means that you must type them as either upper-case or lower-case as shown in the syntax and examples. For example: The **b** option is not the same as the **B** option.

16.1.1 Action Specifiers

You must include one of the following action specifiers on the command line:

1. **c** - Tells TAR to create a new archive. Files will be written starting at the beginning of the archive.

2. **t** - Tells TAR only to list the files on the tape. If any file names are specified on the command line, only files matching this specifier are listed; otherwise, all files on the tape are listed. No other actions (reading or writing files on disk) occur with this action specifier.

3. **x** - Tells TAR to extract files from an existing archive. The files will be restored either to their original subdirectories or to the current subdirectory, depending on whether absolute or relative path names were used when the archive was created. See Section 16.2 for more information.

16.1.2 Commonly Used Options

The options described in this section are the more commonly used ones.

1. **b** *blocking-factor* - Selects the blocking factor to use when writing tape archives. This controls the size of the blocks which will be written to the tape. In general, larger blocks will run faster and let you fit more data on a tape. The unit measure for the blocking factor is 512 bytes, so the resulting block size will be 512 times the blocking factor. The default blocking factor is 20, resulting in 10,240-byte blocks. The upper limit on block size is 65,024 bytes, or a blocking factor of 127.

   To use this option, you must follow the **b** option by a space and the desired blocking factor, for example:

   ```
   tar cvb 40
   ```

2. **f** *archive* - Specifies the file or drive to use for the archive. This is where the archive itself will be written or read, not the files which are being archived or extracted. This can be one of three possible things:

a. A regular disk file - This can be any legitimate DOS file on any DOS drive known to the system, like, C:\ANIMALS\ZOOS.TAR. Any TAR archive can be written to or read from such a DOS file. In this case, the full name of the file (or at least enough to fully qualify it) is given following the **f** option. For example:

```
tar cvf a:floppy.tar *.doc
```

b. A Qualstar tape drive - Any Qualstar tape drive installed in the system can be specified as the source or destination of a TAR archive. The drive is named using the standard TAPESTAR drive selector descriptors as described in Chapter 7. The drive descriptor follows the **f** option. For example:

```
tar cvf drive=9t:1 *.doc
```

c. If the archive is specified as " **-** " (a single dash), the archive is read from standard input or written to standard output. Standard input and output are normally the computer's keyboard (input) and screen display (output), but both can be redirected on the command line. For more information about input and output redirection, consult a DOS manual.

NOTE

If the f option is not specified, the default source or destination for the archive will be the first Qualstar tape drive found in the system (i.e., the drive with the lowest SCSI ID or drive address on the first tape controller or host adapter in the system).

3. **v** - Tells TAR to be verbose about the files that are being processed or listed. Normally, archive creation or file extraction are silent, and archive listing just gives file names. The **v** option causes additional file information (like creation dates and sizes) to be shown.

16.1.3 Rarely-Used Options

The following options are less commonly used:

1. **B** - Tells TAR to reblock an archive as it is read. Normally, TAR reads each block with a single read operation. With this option, TAR will do multiple reads until it gets enough data to fill the specified block size. **B** can also be used to speed up the reading of tapes which were written with small blocking factors, by specifying a large blocking factor with the **b** option and having TAR read many small blocks into memory before it tries to process them.

2. **D** - If TAR generates a message, this option tells TAR to display the record number within the archive where the message occurs. This option is especially useful to pinpoint damaged sections when reading corrupted archives.

3. **h** - If a symbolic link is encountered while creating an archive, this option tells TAR to dump the file or directory to which it points, rather than dumping it as a symbolic link. This option has little or no use under DOS.

4. **i** - Tells TAR to ignore blocks of zeros while reading an archive. Normally, a block of zeros indicates the end of the archive, but in a damaged archive, or in one which was created by appending several archives, this option allows TAR

to continue. This option is disabled by default because the normal behavior of the UNIX TAR program is to write "garbage" after the zeroed blocks.

5. **k** - Tells TAR to keep existing files when extracting files from an archive, rather than overwriting them with the version from the archive.

6. **m** - Tells TAR to set each file's modified time stamp to the current time when extracting files from an archive, rather than extracting each file's modified time from the archive.

7. **n** - Do not rewind the tape after the TAR read or write operation is finished. By default, TAR rewinds the tape when it is done. This option will leave a tape being written ready to have data appended to it. When reading tape archives, there may be multiple archives on a tape. Using this option will allow archives behond the first one to be read.

8. **o** - Tells TAR to write an *old* format when creating an archive. The old format does not include information about directories, pipes, or device files, and specifies file ownership by *uids* and *gids* (user and group id's) rather than by user names and group names. In most cases, a new format archive can be read by an old TAR program without serious trouble, so this option should seldom be needed.

9. **p** - Tells TAR to restore files to the same *permissions* they had in the archive.

10. **s** - When specifying a list of filenames to be listed or extracted from an archive, this option specifies that the list is to be sorted into the same order as the files occur on the tape. This allows a large list to be used, even on computers with limited memory, because the list need not be read into memory all at once. Such a sorted list can easily be created by running **tar t** on the archive and editing the resulting output.

11. **S** *drive-size* - This option is to be used together with the **v** option to specify that the capacity of the disk drive to be used is *drive-size* kilobytes. The default is 360. Since DOS has no way to detect end-of-volume in the "raw" disk I/O mode, this option lets TAR know when it's time to switch disks in the multi-volume raw disk mode.

12. **T** *file-file* - Rather than specifying the file names to use on the TAR command line, this option allows the file names to be read from *file-file*. This file must be a plain ASCII text file containing the desired file names, one per line. If the *file-file* given is "**-**" (a single dash), the list of files is read from standard input. This option, together with the (lower-case) **s** option, allows an arbitrarily large list of files to be processed, and allows the list to be piped to TAR.

13. **u** *extension* [...] - This option adds specified file extensions (the characters after the period in a DOS filename) to the list of extensions which TAR implicitly recognizes as being binary files. Files without one of these extensions are treated as plain ASCII (text) files. ASCII files undergo certain standard conversions when being read into an archive, and the inverse conversions are applied when the files are being extracted from an archive. Therefore, this option protects non-ASCII files from being tampered with by TAR during archive creation or extraction.

14. When the verbose listing mode is selected (with the **v** option,) binary files have a single dash (**-**) as the first character of the mode flags, while other files have an "**a**" (for ASCII). By default, the extensions .COM, .EXE, and .OBJ are treated as binary files; these extensions cannot be overridden, and will always be treated as binary, since they are extensions having special meaning to DOS.

extension can be the single character " . " (a period), meaning that files with no extension should be treated as binary.

Example: The following command line tells TAR to treat the extensions .DAT, .LOG, and .DBF as binary files:

```
tar cvu dat log dbf
```

NOTE

When specifying DOS file extensions using the **u** *extension* [...], leave off the initial period and give the extension as DAT, not as .DAT.

15. **V** *drive* - This tells TAR to write the archive to floppy disk drive *drive* in the multi-volume raw disk mode using direct BIOS I/O. The diskettes must already be formatted under DOS. The archive will be written onto the entire disk, overwriting any DOS directory and other information. When the diskette is full, TAR will ask you to insert another disk; no volume labels or other header information will be written.

    The data is written as if the floppy disks were a single, larger, contiguous disk. Floppy disks written in the raw mode with this option under DOS may be read under UNIX by specifying the proper floppy device. This allows you to transfer large numbers of files between DOS and UNIX. The default disk size for this option is 360K; use the (upper-case) **S** option to specify a different capacity.

Example:

The following command line will list the names of files on an archive diskette in drive b:, a 720K 3-1/2 inch disk drive:

```
tar tvVS b 720
```

NOTE

This option may cause some memory-resident virus-detection software to generate a warning message).

16. **Z** or **z** - These options tell TAR to decompress the archive while extracting from or listing the archive, using the **-d** option of the compress program. The archive itself is not modified.

16.1.4 TAR Filename

The TAR *filename(s)* parameter is a user-specified list of one or more files to be copied, extracted from, or listed in an archive. File names can contain DOS "wildcard" characters (* and **?**).

16.2 How TAR Treats Path Names

TAR has two ways of looking at path names (i.e., the location of files in a subdirectory structure). This is determined by the way the files are named when creating the archive.

1. Absolute path names - If the file names start with a backslash (\ ), then the path is considered to be an *absolute* path name. This means that the path name becomes part of the file name in the archive, and that the file will be restored to its original place when extracted from the archive (if possible).

2. Relative path names - Any file name which does not start with either a drive letter or a backslash is considered to have a *relative* path name. This means that the file's placement when it is restored will be relative to the current directory in effect when extracting from the archive. For example, assume that the following command was used to create an archive:

```
tar cv *.doc
```

Later the following command is used to extract the files:

```
tar xv .
```

The files will be written to the current directory as specified by the single period.

A drive letter may be specified as part of a file name. However, the drive letter and its colon (**c:**) become part of the file name embedded in the archive, and will probably cause problems if the archive is read on a UNIX or XENIX system.

NOTE

TAR includes all subdirectories included in any subdirectories, including all files in these subdirectories, when creating an archive. It will create the same set of subdirectories when extracting from the archive. If the archive was created with relative path names, then this directory structure will be recreated starting at whatever place was named as the destination.

In general, relative path names are easier to deal with, since they present a choice of where to put the extracted files.

17. TABLEDIT and Translation Tables

Beginning with version 4.2, the ability to create and load customized translation tables to translate between the ASCII and EBCDIC character sets is included with *TAPESTAR for DOS*. Prior to version 4.2, the translation tables were embedded and not changeable. Now, you can either use the built-in tables as before, or you can specify a custom table to use by naming the file containing the table.

In addition to the "official" ASCII and EBCDIC character sets, many variations exist. The built-in tables used in *TAPESTAR for DOS* follow the IBM standard, but computer systems which deviate from this standard will require different tables.

The Windows application TABLEDIT allows you to build customized translation tables. TABLEDIT runs under Microsoft Windows and lets you easily modify the standard ASCII-to-EBCDIC or EBCDIC-to-ASCII tables or create new ones from scratch. This chapter describes how to use TABLEDIT.

17.1 Types of Translation Tables

There are two sets of translation tables:

- **EBCDIC-to-ASCII** - These are usually used when *reading* tapes (typically, tapes generated by a mainframe or minicomputer system) into a disk file on the PC;
- **ASCII-to-EBCDIC** - Usually used when *writing* tapes from ASCII (text) files on the PC. The tapes can then be read by a mainframe or minicomputer system.

Not all TAPESTAR programs use translation tables. Those which do are shown in Table 17-1.

PROGRAM	ASCII TO EBCDIC	EBCDIC TO ASCII
DSK2TAPE	Yes	No
TAPE2DSK	No	Yes
TCONVERT	Yes	Yes
DEX	No	Yes
TLABEL	Yes	No

Table 17-1 TAPESTAR Programs Using Translation Tables

17.2 Loading Custom Translation Tables

To load a custom translation table, include one of the following options on the DOS command line which starts the program:

/xe*table-name* - Loads an EBCDIC-to-ASCII table

/xa*table-name* - Loads an ASCII-to-EBCDIC table

In both cases, *table-name* is the path and filename of the file which contains the translation table. There must be no spaces between the **/XA** or **/XE** and the name of the file.

Example:

```
dsk2tape /tr /e /xad:burrough.xlt newfile.dat
```

The preceding command line will load the custom ASCII-to-EBCDIC translation table file D:BURROUGH.XLT and use it to translate the disk file data as it is written to tape.

17.3 Installing TABLEDIT

NOTE

The following discussion assumes that you have a basic understanding of Windows. All of the operations mentioned here are adequately covered in the Windows *User's Guide* which comes with the software.

If you haven't already done so, copy the following files from the TAPESTAR distribution diskettes to the directory which contains your TAPESTAR files.

- **TABLEDIT.EXE** - The executable program
- **ETOA.TBL** - The standard EBCDIC-to-ASCII translation table
- **ATOE.TBL** - The standard ASCII-to-EBCDIC translation table

17.3.1 Starting TABLEDIT

There are three ways to start TABLEDIT:

1. By selecting the Run command in the Program Manager's File menu, and entering the name of the program (TABLEDIT.EXE) in the Command Line: field. You can use the Browse button in this dialog box to locate TABLEDIT.EXE using standard Windows file finders.
2. By running File Manager, locating TABLEDIT.EXE, and double-clicking on the filename.
3. By adding TABLEDIT to a program group, using the New command in the Program Manager's File menu. Once this is done, TABLEDIT's icon will appear in the program group's window, and you can simply double-click on the icon to run it.

NOTE

TABLEDIT includes three icons from which to choose.

When you run TABLEDIT, a window like the one shown in Figure 17-1 will appear.

17.3.2 The TABLEDIT Window

The TABLEDIT window consists of the following components:

- A title bar
- A menu bar
- A tool bar
- A table area
- A vertical scroll bar

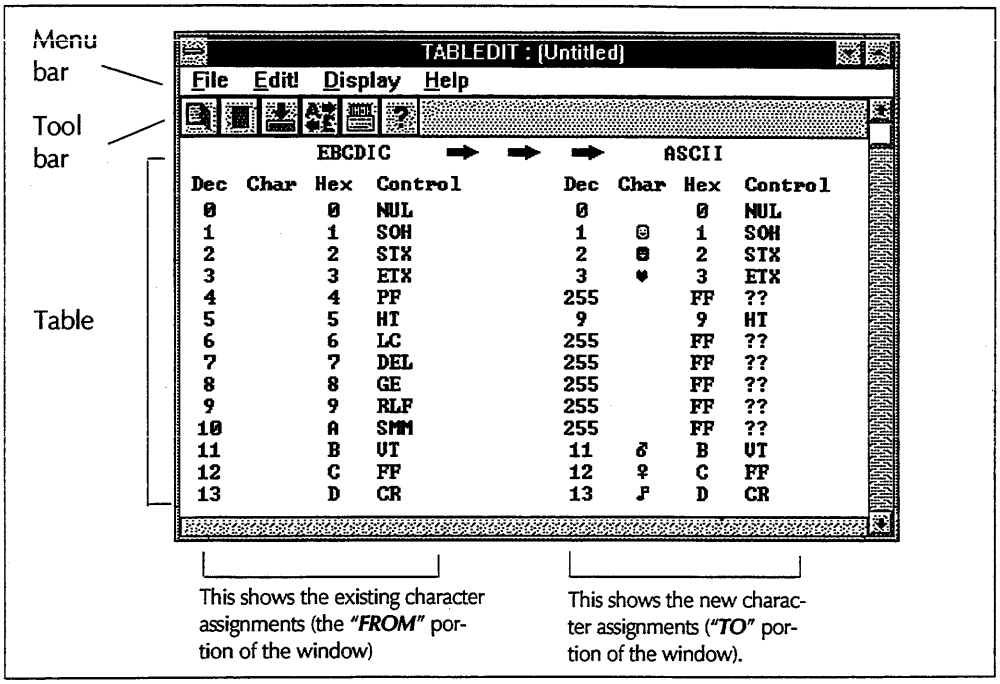


Figure 17-1 TABLEDIT Window

The title bar is the standard windows title bar, displaying the program name and the currently open file. It also contains the standard window close, minimize and maximize boxes.

The menu bar is a standard windows menu bar consisting of File, Edit, Display and Help menus.

The tool bar contains short-cut buttons which duplicate functions found in the menus. Clicking on one of these buttons is the same as pulling down the appropriate menu and selecting the function. From left to right, the tools are:

- Open file
- Close file
- Save File
- Translation direction
- Print
- Help

17.3.3 The Table Area

The table area displays the translation table entries. All translation tables have 256 entries (rows), one for each possible byte value. The display area is divided into two sections: The incoming characters to be translated are shown in the four columns on the left, and the outgoing characters (translated) are shown in the four columns on the right. When you edit an entry, you change to right half of the table only. Use the elevator box to scroll through the table.

The incoming and outgoing characters are shown in four different representations: decimal value, "printable" character value, hexadecimal value, and control code. Notice that parts of the character and control code columns are blank, since not all byte values correspond to printable character or control codes.

17.3.4 Changing a Standard Translation Table

The following procedure may be used to make changes to a standard translation table.

1. Make sure that the correct translation direction is selected. The arrows at the top of the table area indicate the direction of translation. The window shown in Figure 17-1 here indicates an EBCDIC-to-ASCII translation. To change the direction of translation, click on the translation direction button in the toolbar.

2. Click on the Open tool in the tool bar, or select **OPEN** from the File menu, and load a standard table by specifying one of the following standard translation table files:

   a. **ATOE.TBL** - ASCII-to-EBCDIC

   b. **ETOA.TBL** - EBCDIC-to-ASCII

   In order not to overwrite the standard table file, first save it under a new name by selecting **SAVE AS...** in the File menu. Once you have saved the file under a new name, continue with the next step.

3. To make changes in a particular row, either double-click on that row or select the row and then select the Edit! menu. An editing dialog box like the one shown in Figure 17-2 will appear.

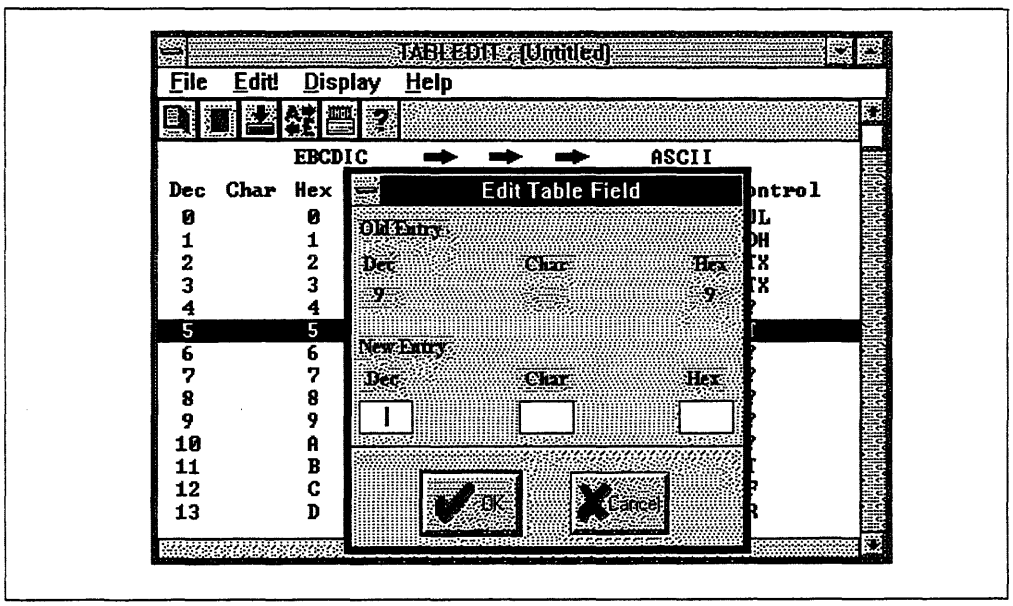


Figure 17-2 TABLEDIT Table Editing Dialog Box

   a. In the example shown, the entry being edited is the sixth one (decimal 5 in the left column,) which happens to be the EBCDIC "horizontal tab" character (notice the **HT** in the Control column). So, for this entry, all incoming EBCDIC tab characters (with a decimal value of 5) will be translated into ASCII tab characters with a decimal value of 9).

   b. You can enter the new value in any of three forms: a decimal value, any keyboard character, or as a hexadecimal value.

c. When you are satisfied with the information in the New Entry portion, click the OK button, or click the Cancel button to close the dialog box without changing that row.

4. When you are finished editing the table, save the changes by clicking the Save button (third from the left in the tool bar) or by selecting **SAVE** in the File menu.

5. After saving the table file, exit by selecting **EXIT** from the File menu, or by double-clicking on the system-menu button in the upper-left hand corner of the TABLEDIT window.

17.4 TAPESTAR Translation Table File Structure

TAPESTAR translation table files are simply 256-byte files, where each byte is the value to be output for an incoming character *at that position in the table* . In other words, it's an indexed table. The incoming ("**from**") character determines the *position* in the table: the byte at that position is the "**to**" character. For example, for an EBCDIC-to-ASCII table, the sixth entry contains the character which will be output for any incoming EBCDIC tab characters (decimal value of 5), since the table starts at zero. The standard ASCII tab character has a decimal value of 9, so the standard table contains a byte value of 9 in the sixth slot.

There is no validation of translation table files by TAPESTAR applications, except that they must be at least 256 bytes long. Any characters beyond 256 are simply ignored.

18. Error Messages

18.1 COPYTAPE Error Messages

`Bad device specifier for drive (drive #).`

A drive specifier given on the command line was incorrect, or no such drive was found in the system.

`Bad VOL1 label on source tape.`

A block which should be a valid VOL1 label on the source tape contained incorrect data.

`*** Canceled by user ***`

This message appears when the user cancels COPYTAPE by pressing ESC.

`Device must be same type as previous 'source' selections.`
`Device must be same type as previous 'destination' selections.`

All drives selected as source or destination tape drives must be of the same type.

`Different kinds of drives specified for source OR dest.`

All drives selected as source or destination tape drives on the command line must be of the same type.

`Drive already selected as SOURCE.`
`Drive already selected as DESTINATION.`

An attempt was made to select a drive already selected.

`Error on COPYTAPE command line.`

Something on the DOS command line given to run COPYTAPE was incorrect.

`Error reading source tape (reason)`

An error occurred while trying to set tape drive modes for the destination drive. (See above).

`Error setting source drive mode: (reason)`

An error occurred while trying to set tape drive modes for the source drive. Check the connections to the drive, drive termination or any switch or other drive configuration settings.

`Error specifying fixed-length block sizes.`

The option specifying fixed-length block sizes was given incorrectly on the COPYTAPE command line.

`Error writing destination (reason)`
`Error writing labels (reason).`

A write error occurred on the destination tape.

```
Error writing filemark after file (tape file)
Error writing filemark after trailer labels (reason)
Error writing filemark before trailer labels (reason)
```

A write error occurred while attempting to write a filemark to the destination tape.

```
Error writing first block (reason)
```

A write error occurred while attempting to write the first block of the destination tape.

```
Initialization failed: (reason)
```

The program tried to initialize MCS-Tape but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

```
Not enough memory to run program.
```

Not enough DOS conventional memory is available in the system to run CopyTAPE. Free more memory and retry the program.

```
No usable drives found in system.
```

Tape drives were found in the system, but none could be used because they were not the correct type for the specified operation. (The most likely reason is that the installed drives(s) cannot handle variablelength blocks.)

```
Only 1 drive found in system.
Only one MCS drive found.
```

COPYTAPE requires two tape drives (one source and one destination).

```
Source tape is labeled, but destination is a fixed-block drive.
```

Since the tape to be copied is a labeled tape, the destination tape drive must be capable of handling variable-length tape blocks. Try using a different destination drive.

```
Tape labels will be discarded--only file data will be copied.
```

This is an informational message explaining that only the file data from the source tape will be copied, since the tape is written in an unknown labeled-tape format.

```
Tape volume ID doesn't match previous tape
```

The volume ID of the tape just mounted doesn't match that of the previous tape. All tapes in a multi-volume set must have the same volume ID.

```
That number is out of range--please re-enter:
```

The number given for a source or destination drive number was out of the range of existing tape drives in the system.

```
There are no drives left for use as destination.
```

All available tape drives in the system have already been selected for use as source or destination drives.

```
Unknown header label on tape.
```

A tape block was read which should have been a header label, but was something else instead.

`Unknown trailer label on tape.`

A tape block was read which should have been a trailer label, but was something else instead.

`You may want to use the /l command-line option.`

Use the **/L** option to create an unlabeled tape when copying a labeled tape.

18.2 DEX Error Messages

`Control file error: (reason)`

A syntax error was found in the control file. The offending line of the file is shown, with a caret (^) underneath the error.

`Couldn't find control file (filename)`

The specified control file (using the **/CF** option) couldn't be found or opened.

`Couldn't open output file: (filename)`

The named file to hold the extracted data couldn't be created. Possible reasons are not enough free disk space or other disk problems.

`Error initializing: (reason)`

The program tried to initialize TAPESTAR but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

`Error: Selected tape drive can't read variable-size blocks.`

DEX requires that the tape drive used be capable of reading variable size blocks. The selected drive can only handle fixed length blocks.

`Error skipping files`

An error occurred while skipping tape files.

`Error: Wrong size tape block`

A tape block of a size different than that expected was read.

`No known MCS-Tape drives were found.`

No TAPESTAR tape drive could be found in the system. Possible reasons are incorrectly installed or missing controllers (SCSI or Pertec), device drivers for the controllers, or tape drives not installed or online.

`No tape drive specified: Using first tape drive found.`

This informational message simply indicates that since no tape drive was specified, the first available tape drive in the system will be used.

`Read error in block (block #)`

A tape read error occurred in the block indicated.

`Unknown option '(option)' ignored`

An incorrect option was given, but will be ignored.

18.3 DSK2TAPE Error Messages

`Bad drive specifier given`

A tape drive specifier on the command line (**drive**=*drive-specifier*) was given incorrectly.

`Block size must be a multiple of record size`

The tape block size specified on the command line must be an exact multiple (one or more) of the specified logical record size.

`Block size too large`

The block size specified on the command line is too big. The largest allowable block size is 48K (49,152 bytes).

`Could not open file '(filename)': skipping`

The indicated file couldn't be opened (because of network conflicts or disk I/O problems). The file is simply being skipped, and the program continues.

`Error initializing: (reason)`

The program tried to initialize the MCS-1 but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

`Error reading disk (reason)`

A disk I/O error occurred while reading the source file.

`Error setting drive modes: (reason)`

An error occurred while trying to set drive modes for the tape drive. Check the connections to the drive, drive termination or any switch or other drive configuration settings.

`Error writing logical EOT (reason)`

A physical write error occurred while attempting to write filemarks to mark the logical end-of-tape on the tape drive.

`Error writing tape (reason)`

A tape write error occurred while writing the tape file.

`File names cannot contain '/'`

A filename was given which contained the slash character.

`Invalid record size given`

The record size specified with the **/R** option was incorrect or out of range.

`Mutually exclusive options given /Z and /E`

You can only specify one of these two command line options. **/E** translates from ASCII to EBCDIC, while **/Z** strips the high bit from each byte of disk file data.

`No block size given`

You must specify the tape block size to create on the output tape, using the **/B** option.

`No files found because of disk error`

Because of problems reading the disk directory, no files could be located on the specified drive.

No record size given

For the transfer mode specified, you must give the tape record size using the **/R** command line option.

`No tape drives found`

No MCS-1 tape drives could be found in the system. Possible reasons are incorrectly installed or missing controllers (SCSI or Pertec), device drivers for the controllers, or tape drives not installed or online.

`No transfer mode given`

You must specify the tape transfer mode by using the **/T** option.

`No valid files specified.`

No filenames specified could be found or opened.

`Not enough memory to run program`

Not enough DOS conventional memory is available in the system to run DSK2TAPE. Free more memory and retry the program.

`Record exceeded given length`

A variable-length record in the source diskfile exceeded the record size given on the command line. The program will either proceed or stop, depending on the transfer mode specified.

`Skip/limit count can't be zero`

A zero skip (**/SB** or **/Sf**) or limit (**/CR**) count was given. These options only make sense with non-zero counts.

`Too many filenames in buffer. Continue with those that fit? (y/n)`

There were too many filenames specified on the command line to fit into the filename buffer. This includes the results of expanding any wildcarded filenames, such as *.DAT.) If you answer yes to this prompt, the program will proceed with all those filenames which did fit in the buffer. Answering no will cancel the program.

`Unknown option: '(option)'`

The indicated option given on the command line is invalid.

`Unknown transfer mode: '(mode)'`

An incorrect transfer mode was specified as part of the **/T** option on the command line.

`Waiting for tape to be ready + online. Hit <Esc> to quit:`

This informational message indicates that the tape drive isn't ready and online. The program will wait until either the drive becomes ready or the user cancels with the ESC key.

`Waiting for tape to rewind. Hit <Esc> to quit:`

This informational message indicates that the tape drive isn't rewound. The program will wait until either the drive is positioned at BOT or the user cancels with the ESC key.

18.4 MCSTEST Error Messages

`Bad controller #!`

The controller number given for the Inq command was out of range (must be from 0 to [#-of-controllers minus 1]).

`Bad destination specifier!`

The source and destination specifiers given for the Copy command must be either a disk filename, a valid drive number or " * " for the currently selected tape drive.

`Bad hexadecimal #!`

An invalid hexadecimal number was given.

`Bad LUN!`

The LUN (logical unit number) given for the Inq command was out of range (must be from 0 to 7).

`Bad source specifier!`

The source and destination specifiers given for the Copy command must be either a disk filename, a valid drive number or " * " for the currently selected tape drive.

`Bad starting block #!`

The starting block number given for the Show command was greater than the number of blocks in the read buffer.

`Can't create destination file!`

The file named as the output for the Copy command couldn't be created. One possible reason is that the disk for the file is full.

`Destination drive # out of range!`

The drive number given for the source or destination tape to be used with the Copy command was out of range.

`Drive number out of range - maximum: (#)`
`Bad drive number!`

The specified drive number was out of range. Drive numbers start at 1. Use the List or Rescan commands to show a list of currently active drives.

`Error allocating memory.`

MCSTEST couldn't allocate DOS conventional memory for internal use.

`Error: byte mismatch expected (expected), got (actual) @ location (loc.)`

While verifying a block of tape data against the read buffer, a mismatch was found at the specified location. The expected and actual bytes are shown as hexadecimal values, while the location is shown in decimal, starting at 0.

`Error: can't specify block size if drive is in fixed-block mode.`

When using the Copy command to write to a tape drive in fixed block mode, the output tape block size can't be given.

`Error reading file`

When using the Copy command to copy from a disk file, a disk I/O error occurred while reading the file.

`Error reading tape (reason)`

A tape read error occurred.

`Error writing tape (reason)`

A tape write error occurred.

`Error--wrong block size read (count) bytes.`

When verifying a tape block, the block size read didn't match the size of the block in the read buffer.

`"get status" not supported by this drive.`

The Stat command only works on non-SCSI tape drives connected to an MCS-1 tape controller.

`Initialization failed: reason was (reason)`

The program tried to initialize TAPESTAR but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

`No active drives found in system.`

No TAPESTAR tape drives could be found in the system. Possible reasons are incorrectly installed or missing controllers (SCSI or Pertec), device drivers for the controllers, or tape drives not installed or online.

`No can do: MCSTape module couldn't be initialized.`

The program tried to initialize TAPESTAR but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

`No drives found to select--try RESCAN.`

No TAPESTAR tape drives could be found in the system. Possible reasons are incorrectly installed or missing controllers (SCSI or Pertec), device drivers for the controllers, or tape drives not installed or online.

`No drive selected: use SEL <drive>`

You must select the current drive (using the Sel command) before using any MCSTEST commands (except for Inquiry, List, and a few other commands which don't require a currently selected drive).

`Not a valid controller type.`

The controller type specified for the Inq command was not a know controller type.

`Nothing in buffer to show or verify.`

The Show or Verify command was given without a previous Read command to place tape data into the read buffer.

`Operation not supported by drive.`

The requested operation isn't supported by the currently selected drive. Use the Info command to show a list of the drive's capabilities.

`Requested controller not found in system.`

The controller number and type given for the Inq command wasn't found installed in the system.

`Rescan failed: reason was (reason)`

The Rescan command failed. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

`Source drive # out of range!`

The drive number given for the source or destination tape to be used with the Copy command was out of range.

`Source file not found!`

The disk file given as the source for the Copy command couldn't be found or opened.

`Starting offset  * bytes in block!`

The starting byte offset given for the Show command was greater than the number of bytes in the read buffer.

`Too many drives to test  maximum: (count)`

Too many drives were specified for the Exe command.

`Unknown command '(command)'`

The command given wasn't a valid MCSTEST command.

`You must specify a valid output block size for this transfer!`

When using the Copy command to copy to a tape drive in variable-block mode, you must specify the tape block size. The maximum block size is 48K (49,152 bytes).

`# of blocks to transfer can't be 0!`

A zero block transfer count was given for the Copy command.

18.5 TAPE2DSK Error Messages

`Bad drive specifier given`

A tape drive specifier on the command line (**drive=***drive-specifier*) was given incorrectly.

`Canceled by user.`

This message appears when the user cancels TAPE2DSK by pressing ESC.

`Can't append: Output file doesn't exist`

The **/A** option was given to append output to an existing disk file, but the specified file couldn't be found or opened.

`Error: Disk full`

A full disk was encountered while writing the output disk file.

`Error initializing: (reason)`

The program tried to initialize the MCS-1 tape drive but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

`Error opening/creating output file`

The specified output file couldn't be found or created. Possible reasons are not enough free disk space or other disk problems.

`Error setting drive modes: (reason)`

An error occurred while trying to set drive modes for the tape drive. Check the connections to the drive, drive termination or any switch or other drive configuration settings.

`Error skipping blocks (reason)`

An error occurred while skipping tape blocks.

`Error skipping files (reason)`

An error occurred while skipping tape files.

`Error: Tape block not a multiple of record size block (block #)`

When doing fixed-length record transfers, each tape block must be an exact multiple of the specified record length. This means that a block of the wrong size was read.

`Error transferring to disk (reason)`

A write error occurred while writing the output disk file.

`File name not specified`

No filename for the output disk file was given on the command line.

`Invalid record size given`

The record size specified with the **/R** option was incorrect or out of range.

`Mutually exclusive options given: /Z and /E`

You can only specify one of these two command line options. **/E** translates from EBCDIC to ASCII, while **/Z** strips the high bit from each byte of tape data.

`No record size given`

For the transfer mode specified, you must give the tape record size using the **/R** option.

`No tape drives found`

No MCS-1 tape drives could be found in the system. Possible reasons are incorrectly installed or missing controllers (SCSI or Pertec), device drivers for the controllers, or tape drives not installed or online.

`No transfer mode given`

You must specify the tape transfer mode by using the **/T** option.

`Not enough memory to run program`

Not enough DOS conventional memory is available in the system to run TAPE2DSK. Free more memory and retry the program.

`Only 1 filename allowed`

Only one filename (for the output file) can be given on the TAPE2DSK command line.

`Output file already exists`

The file named to hold the tape data already exists. Use the **/O** command line option to force the program to overwrite an existing file.

`Output file already exists: OK to overwrite? (y/n)`

This message indicates that the specified output disk file exists, and asks your permission to overwrite it. If you respond yes, the file will be overwritten. Otherwise, the program will be canceled.

`Skip/limit count can't be zero`

A zero skip (**/SB** or **/SF**) or limit (**/CR**) count was given. These options only make sense with non-zero counts.

`Transferred (count) records`

This informational message at the end of the program run tells you how many logical records were transferred from tape to disk.

`Unknown option: '(option)'`

The indicated option given on the command line is invalid.

`Unknown transfer mode: '(mode)'`

An incorrect transfer mode was specified as part of the **/T** option on the command line.

`Waiting for tape to be ready and online. Hit <Esc> to quit:`

This informational message indicates that the tape drive isn't ready and online. The program will wait until either the drive becomes ready or the user cancels with the ESC key.

`Waiting for tape to rewind. Hit <Esc> to quit:`

This information message indicates that the tape isn't rewound to the load point. The program will wait until either the is positioned at the load point or the user cancels with the ESC key.

18.6 TAPEDIAG Error Messages

`Attempted to write to write-protected tape`

The tape for testing must not be write-protected in order for the tape write tests to run.

`Card number requested not installed`

The requested MCS-1 controller is not installed in the system.

`Compare failed in block (block #)`

Data previously written to the tape in the indicated block number was not read back as expected.

```
Could not allocate memory for pattern (pattern #)
Could not allocate memory for tape buffer.
Could not allocate memory for window structure (#)
```

Not enough DOS conventional memory is available in the system to run TAPEDIAG. Free more memory and retry the program.

`Drive timed out`

The allotted time-out period for the current operation expired before the operation completed.

`Error in block (block #)`

A tape read/write error occurred in the indicated block number.

```
Insufficient room in cache for block
Queue is full
```

This indicates a problem with the MCS-1 cache system (hardware or device driver).

`Recoverable soft error occurred`

This warning message indicates that a read or write error occurred, but was recovered by the tape drive.

`Tape drive not ready`

The tape drive being tested is not ready or online.

`Tape unit not open or not available`

The requested tape drive is not installed or available for testing.

18.7 TAPELIST Error Messages

`Error: bad drive specifier given.`

A tape drive specifier on the command line (**drive**=*drive-specifier*) was given incorrectly.

`Error initializing: (reason)`

The program tried to initialize TAPESTAR but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

```
Error reading tape (reason)
```

A tape read error occurred.

```
Error setting drive modes: (reason)
```

An error occurred while trying to set drive modes for the tape drive. Check the connections to the drive, drive termination or any switch or other drive configuration settings.

```
Expected EOF1 label, read filemark instead.
Expected EOF1 label, read incorrect size block instead (count) bytes
Expecting EOF1 label, found something else.
```

At the point where a valid EOF1 label was expected on the tape, the indicated condition occurred.

```
Expected HDR1 label, read filemark instead.
Expected HDR1 label, read incorrect size block instead (count) bytes
Expecting HDR1 label, found something else.
```

At the point where a valid HDR1 label was expected on the tape, the indicated condition occurred.

```
Expected HDR2 label, read filemark instead.
Expected HDR2 label, read incorrect size block instead (count) bytes
Expecting HDR2 label, found something else.
```

At the point where a valid HDR2 label was expected on the tape, the indicated condition occurred.

```
Treating tape as unlabeled.
```

While TAPELIST at first sensed that the tape being read was labeled, an unexpected condition occurred, and the program is reverting to reading the tape as unlabeled. The sizes and number of blocks in each tape file will be reported, but no filenames or other information which depends upon tape labels will be shown.

```
Using drive: (drive name)  ID:LUN  (ID #:LUN #)
```

This informational message shows which tape drive is being used.

18.8 TAPEMENU Error Messages

```
Could not duplicate stdout.
```

This internal error may occur if not enough file handles are available. Increase the number of FILES= in CONFIG.SYS and reboot the computer to retry.

```
Could not execute TAPELIST. (reason)
```

The TAPELIST program couldn't be run to fulfill the *list data on tape* menu item request.

```
Could not open batch file '(filename)' for writing.
```

The indicated batch file requested to be written couldn't be created. Possible reasons include a full disk in the indicated drive.

```
Error writing to batch file '(filename)'.
```

An error occurred while writing the requested batch file. Possible reasons include a full disk in the indicated drive.

`Program file TAPELIST.EXE not found.`

The indicated file wasn't found on disk. Execution of the *list data on tape* menu item requires this program.

`The file '(filename)' already exists.`

The file named to hold the tape data for a tape-to-disk transfer already exists.

`The specified file '(filename)' does not exist.`

The specified file for a disk-to-tape transfer couldn't be found or opened.

`The specified file '(filename)' is not readable.`

A disk I/O error occurred while trying to read the indicated disk file.

18.9 TAR Error Messages

`absio: error (op) drv (drive) block (block #) address (addr.): bios code (code)`

A disk error occurred while performing low-level I/O to the indicated drive. Treat this as any other disk read/write error.

`Argument name too long: (arg)`

The name given for the specified command line argument was too long.

`Attempting to read a block smaller than block size.`

While reading a TAR archive on tape, a block smaller than the expected block size was read. The archive may be damaged.

`Attempting to write a block smaller than block size.`

A write error occurred while writing a tape archive.

`Bad device specifier.`

An invalid TAPESTAR drive specifier was given on the command line.

`Can't open (filename) for I/O`

The specified file couldn't be found or opened.

`Could not make directory`

TAR couldn't create the specified subdirectory. Check that all outer-level directories in the specified path exist on the given drive.

`(error): error opening directory`

The specified subdirectory couldn't be found on the disk.

`Error parsing device.`

An invalid TAPESTAR drive specifier was given on the command line.

`Error (op) drive (drive), sector: (sector #), code: (error code)`

A disk error occurred while performing low-level I/O to the indicated drive. Treat this as any other disk read/write error.

`(filename): name too long`

The specified filename is too long.

`tar: (filename): Not a valid DOS filename.`

The specified filename cannot be used as a valid DOS filename. (Remember the differences between UNIX and DOS filenames.)

`tar: (filename) not found in archive`

The specified filename couldn't be located in the tar archive.

`tar: too many args with -T option`

Too many arguments following the **-T** option were given on the command line.

`tar: Volume full. Change volumes and press [Enter]:`

This message indicates that a tape (or disk) being used to write an archive is full. Mount the next tape (or insert the next disk) and press ENTER to proceed with the next volume.

`This device may not flush data at the end of the tape.`

This warning message indicates that since the tape drive being used may not flush buffered data at the end of a tape, some data may be lost at the end of a tape in a multivolume archive.

`Too few args. on command line`

One or more options needed to run tar are missing from the command line.

`Too many errors, quitting.`

Since tar has encountered too many tape or disk I/O errors, it is abandoning any further attempts.

`Too much data was buffered when EOT was hit.`

A write error occurred at the end of a tape being written to. There wasn't enough room left on the tape to hold all of the unwritten data stored in the drive's buffer.

`Volume overflow on device.`

The tape drive being written to returned a SCSI error indicating Volume Overflow. This means that an attempt to write past EOT failed.

`Unexpected EOF on archive file`

An unexpected filemark was read on the TAR archive being read

`(command line argument): unknown option (option)`

The unknown option character shown was found in the command line argument shown.

18.10 TCLONE Error Messages

`BAD BLOCK! Error was (error)`

The indicated tape read error occurred while reading the source tape.

`Bad destination specifier on command line`

A tape drive specifier on the command line (using either **source=***drive-specifier* or **dest=***drive-specifier*) was given incorrectly.

`Bad drive specifier given`

A tape drive specifier on the command line (using either **source=***drive-specifier* or **dest=***drive-specifier*) was given incorrectly.

`Bad format specifier given`

A tape format specifier on the command line was given incorrectly.

`Bad source specifier on command line.`

A tape drive specifier on the command line (using either **source=***drive-specifier* or **dest=***drive-specifier*) was given incorrectly.

`Bad stop count (count)`

The stop count specified on the command line was incorrectly given (must be numeric and greater than 0).

`Block size must be a multiple of record size`

The tape block size specified on the command line must be an exact multiple (one or more) of the specified logical record size.

`Block size too large`

The tape block size specified on the command line was too large. The limit is 48K (49,152 bytes).

`Could not open file '(filename)': skipping`

The named file couldn't be opened. Possible reasons include network file conflicts (another user may have the file opened with exclusive access rights,) or problems reading the drive the file is on.

`Data error at offset (location), bytes: (expected) & (actual)`

The byte at the indicated location in the tape block was not what it was supposed to be. The expected and actual bytes are shown in hex; the location is shown in decimal (offsets start at 0).

`Early filemark on destination.`

An unexpected filemark on the destination tape drive was read while verifying against a source tape or disk file dataset.

`EOD on source but not on destination.`

End-of-data was hit on the source tape drive, but not on the destination tape drive or disk file dataset during verification.

`EOD on destination but not on source.`

End-of-data was hit on the destination tape drive, but not on the source tape drive or disk file dataset during verification.

`Error (error) on source drive`

The indicated tape read error occurred while reading the source tape.

`Error flagging is incompatible with verification.`

The error flagging and verification options cannot be used at the same time.

`Error initializing: (reason)`

The program tried to initialize the MCS-1 but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

`Error opening disk data file.`

The file containing the tape dataset's actual data couldn't be found or opened. (The data file always has an extension of .DAT.)

`Error opening disk length file.`

The file containing the tape dataset's Length data couldn't be found or opened. (The length file always has an extension of .LEN.)

`Error reading disk (reason)`

A physical I/O error occurred while trying to read a file from disk.

`Error reading length file.`

A disk I/O error occurred while reading the length file.

`Error setting drive modes: (reason)`

An error occurred while trying to set drive modes for the tape drive. Check the connections to the drive, drive termination or any switch or other drive configuration settings.

`Error writing logical EOT (reason)`

A physical write error occurred while attempting to write filemarks to mark the logical end-of-tape on the tape.

`Error writing output data file`

A disk I/O error occurred while writing the intermediate disk data file.

`Error writing tape (reason)`

A physical write error occurred while writing to the tape drive.

`(count) errors in block, last error at offset (location)`

The indicated number of errors were encountered in the current tape block, with the last error occurring at the indicated offset. All numbers are given in decimal (offsets start at 0).

`File names cannot contain '/'`

A filename was given which contained the slash character.

`Flagging error and continuing.`

A tape read error occurred on the source tape. The error has been flagged (logged on the output tape), and the copy operation continues.

`Improper mode selection, QIC-24 on TDC-3800 not allowed:`

The indicated tape format is not supported by this tape drive.

`Invalid record size given`

The logical record size given on the command line is out of range or contains some other error.

`Missing filemark on destination.`

An expected filemark on the destination tape drive was not found while verifying the tape in that drive against a source tape or disk file dataset.

`No block size given`

You must specify a tape block size on the command line.

`No files found because of disk error`

Because of problems reading the disk directory, no files could be located on the specified drive.

`No record size given`

You must specify the logical record size on the command line.

`No tape drives found`

No MCS-1 tape drives could be found in the system. Possible reasons are incorrectly installed or missing controllers (SCSI or Pertec), device drivers for the controllers, or tape drives not installed or online.

`No transfer mode given`

You must specify a transfer mode on the command line.

`No valid files specified.`

No filenames specified could be found or opened.

`Not a valid TDUP control file!`

The specified disk file is not a valid TDUP control file.

`Not enough memory to run TCLONE.`

Not enough DOS conventional memory is available in the system to run TCLONE. Free more memory and retry the program.

`Please select quiet OR loud.`

The *quiet* and *loud* command line options cannot both be used.

`(count) read errors flagged.`

This informational message at the end of the TCLONE run shows how many tape read errors were flagged during the run.

`Record exceeded given length`

A tape record exceeded the specified logical record size.

`Too many filenames in buffer. Continue with those that fit? (y/n)`

There were too many filenames specified on the command line to fit into the filename buffer (This includes the results of expanding any wildcarded filenames, such as *.DAT.) If you answer **yes** to this prompt, the program will proceed with all those filenames which did fit in the buffer. Answering **no** will cancel the program.

`Unknown transfer mode: '(mode specified)'`

The transfer mode specified on the command line is unknown.

`Using drive: (drive name) ID:LUN (ID #:LUN #)`

This informational message shows which tape drive is being used.

`Waiting for tape to be ready + online. Hit <Esc> to quit:`

This informational message indicates that the tape drive isn't ready and online. The program will wait until either the drive becomes ready or the user cancels with the ESC key.

`Waiting for tape to rewind. Hit <Esc> to quit:`

This informational message indicates that the tape isn't rewound to the load point. The program will wait until either the tape is positioned at the load point or the user cancels with the ESC key.

`Wrong block length: source = (length), dest. - (length)`

While verifying a tape, a tape block of the wrong size was found.

`You must specify a file to copy to or from.`

When using an intermediate disk file to copy tapes, you must specify the filename, using only the root name of the file and any needed drive or path specifiers.

`You must specify either a source or destination drive.`

No drive was specified on the command line for either source or destination use.

18.11 TCONVERT Error Messages

`A block was read that was bigger than expected.`

While reading from a tape with a given block size, a block larger than the indicated size was read.

`Attempt to set invalid mode.`

An error occurred while trying to set drive modes for the tape drive. Check the connections to the drive, drive termination or any switch or other drive configuration settings.

`Could not open '(filename)' for reading.`

The indicated disk file couldn't be found or opened.

`Could not open '(filename)' for writing.`

The indicated disk file couldn't be created. Possible reasons include a full disk on the indicated drive.

`Drive encountered physical end of medium.`

While writing to a tape drive, the physical end-of-tape was hit.

`Drive is not online.`

The tape drive currently being used is not online.

`Drive is not ready.`

The tape drive currently being used is not ready.

`Drive is write protected.`

An attempt was made to write to a tape which is write-protected.

`Drive timed out.`

A tape drive took too long to complete a requested operation. Check to see that the drive is online.

`ERROR - Cannot convert from Tape file to Tape file`

TCONVERT cannot transfer data directly from tape to tape. Use TCLONE instead.

`ERROR - Cannot find sufficient memory`

Not enough DOS conventional memory is available in the system to run TCONVERT. Free more memory and retry the program.

`ERROR - Cannot find sufficient memory for disk buffer`

Not enough DOS conventional memory is available in the system to run TCONVERT. Free more memory and retry the program.

`ERROR - Cannot read from file (filename)`

A disk I/O error occurred while attempting to read from the indicated file.

`ERROR - Cannot write to file (filename)`

A disk I/O error occurred while attempting to write to the indicated file.

`ERROR - Filename too long: (filename)`

The specified DOS filename is too long for DOS to handle.

`ERROR - No data found, tape mark encountered`

A filemark was encountered instead of a tape block containing data.

`Error reading header of labeled tape`

A tape read error occurred while reading a block expected to contain a tape label.

`Insufficient memory for buffers.`

Not enough DOS conventional memory is available in the system to run TCONVERT. Free more memory and retry the program.

`No controllers found in system.`

No known TAPESTAR controllers were found in the system. Either the controllers themselves, or the device drivers required to run them are not installed, or are not correctly installed.

`No tape drives found in system.`

No TAPESTAR tape drives could be found in the system. Possible reasons are tape drives not installed or online.

`Out of memory: unable to allocate menu structure (count) bytes.`

Not enough DOS conventional memory is available in the system to run TCONVERT. Free more memory and retry the program.

`Out of memory: unable to allocate space for dialog entries.`

Not enough DOS conventional memory is available in the system to run TCONVERT. Free more memory and retry the program.

`Out of space on disk`

The disk on which the output file is being written is full.

`Program error: operation not supported by drive.`

A requested operation is not supported by the tape drive.

`Tape medium is flawed.`

A tape read/write error occurred because of flawed tape media.

`Tape must be at load point to set density`

The density or format of a tape cannot be changed unless the tape is rewound to load point or BOT.

`Tape must be at load point to set speed`

The speed of a tape drive cannot be changed unless the tape is rewound to load point or BOT.

`Unable to allocate (count) bytes for entry '(entry)'`

Not enough DOS conventional memory is available in the system to run TCONVERT. Free more memory and retry the program.

18.12 TINSTALL Error Messages

`ERROR: An error occurred while reading the device driver.`

A disk read error occurred while reading the file containing the device driver for the MCS-1.

`ERROR: The address (address) is not valid.`

The indicated base I/O address is not valid. The address is given in hexadecimal.

`ERROR: The file '(filename)' was not found on drive (drive.`

The required file wasn't found on the disk in the indicated drive.

`ERROR: Unable to open the file '(filename)'.`

The required file couldn't be found or opened.

`ERROR: Unable to select drive (drive)`

The indicated drive doesn't exist or can't be selected.

`ERROR: Unable to write to drive (drive)`

A disk write error occurred while writing a file to the indicated drive.

18.13 TLABEL Error Messages

`Couldn't open input file (filename)`

The indicated disk file couldn't be found or opened for reading.

`Error (error). Aborting program.`

A fatal error occurred, and program execution is halting.

`Error. Could not open tape device '(drive name)'.`

The indicated tape drive couldn't be located or initialized.

18.14 TMERGE Error Messages

`BAD BLOCK! Error was: (error)`

This message indicates that a read or write error occurred at the indicated block number.

`Couldn't find specified 9-track set`

The tape set specified couldn't be found on the current tape.

`END of archive or ERROR on archive tape`

An unexpected end of archive or a read error occurred while reading the archive tape.

`Error at block (block #)`

This message indicates that a read or write error occurred at the indicated block number.

`Error initializing: (reason)`

The program tried to initialize the MCS-1 but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

`Error initializing archive tape: (reason)`

The program tried to initialize the MCS-1 but couldn't. Probable reasons are controllers (SCSI or Pertec) not being installed, device drivers for controllers not being installed, or tape drives not connected or online.

`Error initializing first 9-track drive.`

An error occurred while writing initialization information to the first 9-track tape drive in the system.

`Error setting drive modes: (reason)`

An error occurred while trying to set drive modes for the tape drive. Check the connections to the drive, drive termination or any switch or other drive configuration settings.

`Error: The archive tape already contains more than two gigabytes of data. No more may be added.`

TMERGE imposes a 2-gigabyte (2 billion byte) limit on the amount of data on each tape. This message indicates that tape is full.

```
Error: This is not an initialized ARCHIVE tape. Please prepare all archive tapes
with the 'initialize' operation.
```

All tapes to be used with TMERGE must be initialized (have control information written to them) before being used.

```
Error writing archive label: (reason)
```

A write error occurred on the destination tape while attempting to write control information.

```
(count) read errors flagged.
```

This informational message indicates the number of physical tape read errors which occurred during this run of TMERGE.

```
Error locating end of archive: (reason)
```

While trying to locate the end of the archive on tape, a read error occurred.

```
Error (error) on source drive
Error reading archive: (reason)
```

A tape read error occurred.

```
Flagging error and continuing.
```

This informational message indicates that a read error occurred on the source tape, but it is simply being flagged, and the program will continue reading.

```
Not enough memory to run TMERGE.
```

Not enough DOS conventional memory is available in the system to run TMERGE. Free more memory and retry the program.

```
Out of sync! Re-syncing may take a while...
```

TMERGE has discovered that it is "out of sync" on the current tape, and is trying to locate the proper place on the tape. This can take a significant amount of time.

```
Searching for end of data, this may take a minute...
```

This informational message indicates that the program is searching for EOD on the tape cartridge. This process takes a significant amount of time to complete.

```
Waiting for source tape drive to come online
```

This informational message indicates that the source drive is not ready and online, and TMERGE is waiting for it to become ready.

19. Appendix A

Appendix A provides help in situations requiring unusual tape processing and presents additional background information about tape subsystems.

The chapter is divided into the following sections:

1. Physical tape characteristics, Section 19.1
2. Labeled and Unlabeled Tape, Section 19.2
3. Tape Data and File Formats, Section 19.3
4. Tape Terminology, Section 19.4
5. Nine Track Tape Capacity Tables, Section 19.5
6. Tapestar Errors, Section 19.6
7. Using SEISMIC.EXE for Large Block Transfers, Section 19.7

19.1 Physical Tape Characteristics

Each type of tape medium is described in turn below.

19.1.1 Nine-Track Tape

Nine-track magnetic tape is one-half inch wide and is wound onto reels like audio recording tape. Reel sizes from 6 to 10.5 inches are available, providing total tape lengths of from 600 to 3600 feet.

19.1.1.1 BOT and EOT Reflective Markers

Each reel of tape has two strips of reflective tape placed on the outer side of the tape. The first reflective strip is located sixteen feet from the beginning of the tape and is called the BOT marker. The second reflective strip is located about 25 feet from the end of the tape and is called the EOT marker. A reflection sensor in the tape path detects the passing of these strips to provide an indication of BOT and EOT. If the strips are worn, dirty, or missing, there may be problems loading tapes, or tapes may run off the end of the supply reel. These strips can be replaced by new self-adhesive markers available wherever computer accessories are sold.

19.1.1.2 Packing Densities

Packing density refers to how close together the bytes of data are on the tape. Density is measured in characters per inch (cpi), sometimes also called BPI (bits per inch). There are four standard nine-track tape densities in use today, grouped into three recording methods:

DENSITY	RECORDING FORMAT
800	NRZI (Non-return to zero inverted)
1600	PE (Phase Encoded)
3200	DPE (Double Phase Encoded)
6250	GCR (Group Code Recording)

The two most common densities in use today are 1600 and 6250 cpi. A tape drive set to one density cannot read tapes recorded (written) at another density. Many tape drives allow the selection of several densities. Some drives will automatically sense the density of tapes being read.

19.2 Labeled and Unlabeled Tape

There are many ways of writing data to tape, but most of them fall into two broad categories:

1. **Labeled tapes** - with special records (labels) on tape. These *header* and *trailer* records contain control and bookkeeping information about the data on the tape.

2. **Unlabeled tapes** - where data is simply written to the tape, usually with no headers or trailers.

There are, of course, lots of variations and exceptions to these general rules.

19.2.1 Labeled Tape Formats

Tapestar is capable of reading and writing labeled tapes in today's three most commonly used formats. These formats all follow the general layout in Figure 19-1. The volume label, file header, EOF trailer and EOV trailer each consist of one or more 80-byte tape blocks. Tape labels are usually self-identifying, containing the name of the block (VOL1, HDR1, EOF1, etc.) as the first characters in the label. Labeled tapes can have one or more files on each volume. They can also include more than one tape reel in a multi-volume set. Like unlabeled tapes, the logical end of a labeled tape is marked by two consecutive filemarks.

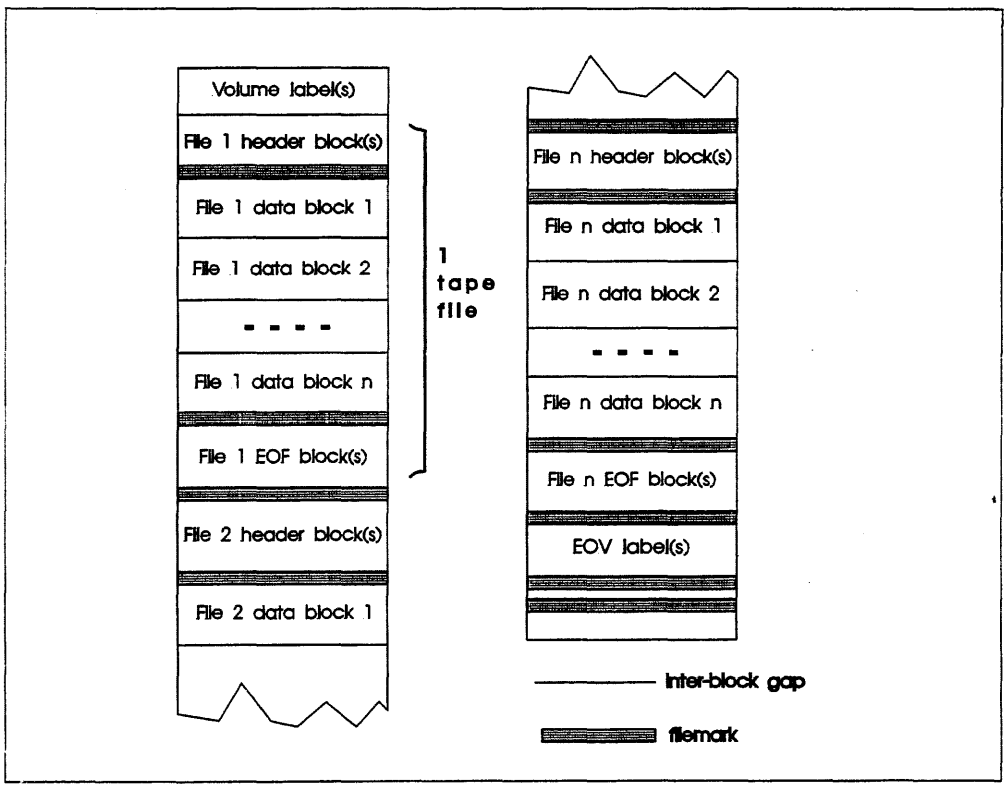


Figure 19-1 Standard Labeled Tape Format

1. IBM OS/VS Labeled Tapes

   a. An OS/VS labeled tape has a single VOL1 label at the beginning of the tape.

   b. Each file on an OS/VS labeled tape is preceded by one or more HDR1 and HDR2 blocks, and is followed by one or more EOF1 and EOF2 blocks.

c. The fields in OS/VS tape labels are written in EBCDIC.

2. IBM DOS/VSE Labeled Tapes

   a. A DOS/VSE labeled tape has a single VOL1 label at the beginning of the tape.

   b. Each file on a DOS/VSE labeled tape is preceded by one HDR1 block and is followed by one EOF1 block.

   c. The fields in DOS/VSE tape labels are written in EBCDIC.

3. ANSI Labeled Tapes

   a. An ANSI labeled tape has a single VOL1 label at the beginning of the tape.

   b. Each file on an ANSI labeled tape is preceded by one or more HDR1 and HDR2 blocks, and is followed by one or more EOF1 and EOF2 blocks.

   c. The fields in ANSI tape labels are written in ASCII.

19.3 Tape Data and File Formats

19.3.1 Files

In general, on both unlabeled and labeled tapes, data is written in the form of *files*. In tape terms, a file is defined as a set of data blocks with special marks, called filemarks (or tape marks) separating tape files.

Usually, the logical end of a tape is marked by two successive filemarks, and may not necessarily be the same as the physical end of the tape (which is marked by the reflective EOT marker). The double filemark is used to indicate that there is no more data on the tape.

19.3.2 Tape File and Record Formats

Both labeled and unlabeled tapes usually consist of one or more tape files. A file consists of a number of *blocks*; each block contains one or more *records*. Each record contains one or more fields, and each field contains one or more characters or bytes of data.

Files on tape (and disk) are either fixed length or variable length:

1. **Fixed Length** - Files with fixed length records have records which are all the same length. The majority of tape files are fixed length.

2. **Variable Length** - Files with variable length records have records which are not necessarily the same length. The end of each record is marked by one or more *delimiters*, which are special characters which cannot occur in the data of the record. Text files on the PC are a common example of variable length records. Each line of text can be any size (up to some arbitrary limit depending on which program is reading or writing the text file), and the end of the record (line) is marked by a carriage return/line feed combination.

NOTE

Under UNIX, text files use only a single line feed character as a delimiter, also called a *newline* character.

Other, less often used types of variable length records are sometimes encountered on tape:

1. **IBM variable-length records** - These have a special prefix containing a length field at the beginning of each record, specifying the length of that record.

2. **Variable blocked files** - These tape files consist of variable length blocks, one record per block. This type of file can be transferred correctly to a disk file by using the TAPE2DSK program. Refer to Chapter 8 for more information.

19.4 Tape Terminology

ASCII - American Standard Code for Information Interchange. One of the two standard character sets widely used for small and large computer systems.

Block - The smallest integral unit of data which can be transferred to or from tape during one operation. A block can contain any number of characters and is followed by a short erased area called an interblock gap. Often erroneously interchanged with *record.*

BOT - Beginning of Tape, as indicated by a reflective marker on the tape.

Bpi - Bits per inch. Unit of data packing density on the media, sometimes erroneously interchanged with *cpi.*

Byte - The smallest piece of information which can be accessed in a disk file. One byte contains eight bits.

Character - The smallest piece of information which can be accessed in a tape file. It consists of an eight bit byte plus a parity bit.

Character set - The set of rules in use which govern how bytes are interpreted and represented. The two most widely used character sets are ASCII and EBCDIC.

Cpi - Characters per inch. Unit of data packing density on tape media, sometimes erroneously interchanged with *Bpi.*

DAT - Digital Audio Tape (4mm helical-scan cartridge)

Delimiter - A certain character or sequence of characters which marks the end of each data record.

Density - The term used to specify close to one another characters are physically written on magnetic media. The four standard nine-track tape densities are 800, 1600, 3200 and 6250 cpi. 1/4-inch cartridge densities are identified as different QIC formats.

DPE - Double PE. A recording format identical to PE except that the recording density is 3200 cpi.

EBCDIC - Extended Binary Coded Digital Interchange Character. One of the two standard character sets for computers, created by IBM.

EOD - The logical end of data marker on a tape.

EOT - The physical end of tape as determined by a reflective marker strip.

Field - A part of a data record which holds a certain piece of information (e.g., name, address, etc.)

File - A contiguous group of data records (disk) or blocks (tape).

Filemark - A unique kind of tape block used to identify the end of a file. Also called tape mark.

Fixed length records - Data records which are always the same length in the file, and which have no delimiters.

GCR - Group Coded Recording. A high-density (6250 cpi) tape recording format.

Helical Scan - Recording method where the tape is wrapped at an angle around a rotating head, making parallel tracks which are diagonal to the tape.

IBG - Interblock gap. An erased area of tape (0.6 inch for NRZI, PE and DPE, 0.3 inch for GCR) which follows every tape block. It separates data into blocks and provides an area from which to stop and start the tape during block operations. Often erroneously interchanged with *IRG*.

IRG - Inter-record gap. The erased area between adjacent sectors on a disk. Often erroneously interchanged with *IBG*.

Label - A special tape block which contains information about the data on the tape.

Labeled tape - A tape which contains tape labels in addition to data files.

Loadpoint - The beginning of the tape (BOT).

NRZI - Non Return to Zero Inverted. One of the earliest nine-track tape recording formats (800 cpi).

PE - Phase Encoded. A widely-used recording format (1600 and 3200 cpi).

QIC - An acronym for Quarter Inch Cartridge

Record - When used in connection with databases, a record refers to a group of data fields. When used in connection with disk drives, the smallest contiguous of data which can be transferred between the system and the disk. It occupies one complete sector on one surface of a platter.

Serpentine - Recording method used for QIC tapes, where data is recorded on parallel tracks, back and forth from one end of the tape to the other, one track at a time.

Tape mark - A special kind of tape block used to identify the end of a file. Also called *filemark*.

Variable length record - A record which is not restricted to a specified length. The end of a variable length record is marked by delimiter characters.

Volume - A group of one or more files.

19.5 Nine-Track Tape Capacity Tables

The following tables show the capacity, in megabytes, of various lengths of tape and various block sizes. There are four tables, one for each density (800, 1600, 3200 and 6250 cpi). If a tape has variable block sizes, use the average block size in the tables.

LENGTH (FT):	300	600	1200	2400	3600
Block Size			Megabytes		
80	0.4	0.8	1.6	3.2	4.8
128	0.6	1.2	2.4	4.9	7.4
256	1.1	2.2	4.4	8.9	13.3
512	1.8	3.7	7.4	14.9	22.3
1024	2.8	5.6	11.2	22.5	33.7
2048	3.8	7.5	15.1	30.2	45.3
4096	4.6	9.1	18.3	36.5	54.9
8192	5.1	10.2	20.4	40.7	61.1
16384	5.4	10.8	21.6	43.2	64.8
32768	5.6	11.2	22.3	44.6	66.9

Table 19-1 Formatted tape capacity @ 1600 cpi.

LENGTH (FT):	300	600	1200	2400	3600
Block Size			Megabytes		
80	0.4	0.9	1.7	3.4	5.1
128	0.7	1.3	2.7	5.4	8.1
256	1.2	2.5	5.1	10.2	15.2
512	2.3	4.6	9.2	18.3	27.4
1024	3.8	7.6	15.3	30.5	45.8
2048	5.7	11.5	22.9	45.0	68.8
4096	7.7	15.3	30.6	61.2	91.9
8192	9.2	18.4	36.8	73.6	110.4
16384	10.2	20.5	40.9	81.8	122.8
32768	10.8	21.7	43.3	86.7	130.0

Table 19-2 Formatted tape capacity @ 3200 cpi.

LENGTH (FT):	300	600	1200	2400	3600
Block Size			Megabytes		
80	0.6	1.3	2.7	5.3	8.0
128	1.0	2.1	4.2	8.4	12.6
256	1.9	3.9	8.0	16.0	24.0
512	3.6	7.2	14.7	29.4	44.2
1024	6.2	12.4	25.3	50.6	75.9
2048	10.1	20.2	39.5	79.0	118.5
4096	13.9	27.9	54.9	109.8	164.7
8192	17.2	34.5	68.2	136.4	204.6
16384	19.5	39.0	77.6	155.2	232.8
32768	20.9	41.8	83.3	167.7	250.0

Table 19-3 Formatted tape capacity @ 6250 cpi.

19.6 Tapestar Errors

ERROR #	MEANING
0	No errors.
1	M-Init has not been called.
2	Device entry number was out of range.
3	No known adapters found.
4	No active drives found.
5	Insufficient RAM to allocate buffers.
6	Error occurred, no more information is available.
7	Device was not ready.
8	Device was busy when command sent.
9	Mode select data was rejected.
10	An illegal parameter was passed into the module.
11	Physical end-of-medium hit; data not written to tape.
12	An error occurred writing to tape.
13	An error occurred reading from tape.
14	Tape flaw.
15	Device hardware failure.
16	Device timed out.
17	Device is write-protected.
18	Requested operation is not supported by device.
19	Wrong size or number of blocks read/verified.
20	Violation of MCS-1 "freeze/thaw" protocol.
The following group includes recoverable "alerts" - not really errors:	
21	Physical end-of-medium hit; all data written to tape.
22	Device encountered end-of-data.
23	Device encountered a filemark.
24	"noWait" command in progress, not completed.
25	"Unit attention" - tape just loaded or drive powered on.
26	Data error on verify.
27	No tape found in autoloader position.
28	No autoloader present on drive.
29	Invalid slot # or other autoloader error.
30	No such adapter (# out of range).
31	Error on ganged operation.

Table 19-4 Errors reported by Tapestar

19.7 Using SEISMIC.EXE for Large Block Transfers

Normally, tape block lengths are limited to 64K (65,536 bytes). 64K is the maximum block length which can be used with the other MCS-1 programs (and some have a lower maximum). The majority of tapes rarely have block lengths exceeding 32K (32,768 bytes).

The geophysical industry is an exception, and tapes with seismologic data may have block lengths which exceed one megabyte. Hence the name of the program, SEISMIC. This utility copies data to and from tape one block at a time. Going from disk to tape, the entire disk file becomes one tape block. Conversely, SEISMIC will write a single disk file from a large tape block.

19.7.1 Disk Speed Requirements for SEISMIC

Because the data transfer can not be interrupted while writing blocks larger than 64K, the data transfer rate from the computer to the disk must be greater than or equal to the data transfer rate from the tape drive to the computer. The disk transfer rate depends both on the physical transfer rate of the hard disk and the overhead imposed by DOS. SEISMIC will automatically select low speed on the tape drive (if possible) to allow the disk to keep up.

19.7.2 Errors during SEISMIC Operation

If an error is reported while transferring a block to or from tape, it is probably because the system throughput is slower than the tape drive transfer rate. If this happens, select a lower tape speed and retry the transfer. The only other possible errors would be actual tape read or write errors resulting from bad spots on the tape.

19.7.3 SEISMIC Command Syntax

Syntax: **`seismic`** *`filename direction`* [**n**]

- *filename* is a name of a disk file you specify from which to read or write the next block;
- *direction* is specified by you and tells SEISMIC the direction of the transfer:
  a. Specify **R** to copy from tape to disk;
  b. Specify **W** to copy from disk to tape.
- **N** is an option which tells SEISMIC not to rewind the tape after transferring the block. This option is required when transferring multiple blocks to or from a single tape.

END OF DOCUMENT