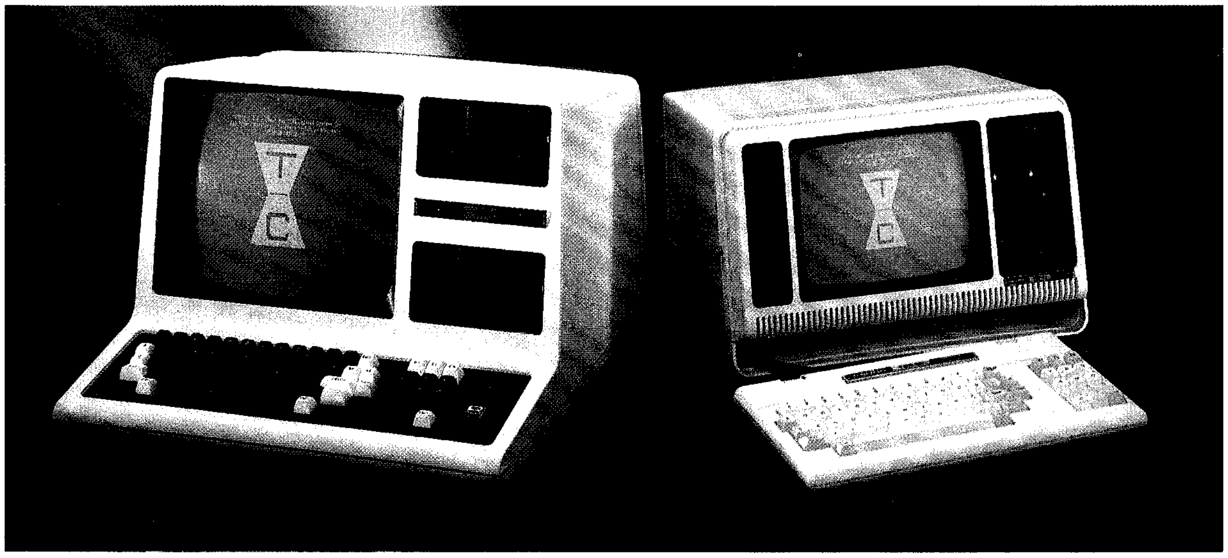
Radio Shack®

DISK SYSTEM OWNER'S MANUAL

TRS-80® Model 4/4P



Custom Manufactured in USA by RADIO SHACK, A Division of TANDY CORPORATION

TERMS AND CONDITIONS OF SALE AND LICENSE OF RADIO SHACK COMPUTER EQUIPMENT AND SOFTWARE
PURCHASED FROM A RADIO SHACK COMPANY-OWNED COMPUTER CENTER, RETAIL STORE OR FROM A
RADIO SHACK FRANCHISEE OR DEALER AT ITS AUTHORIZED LOCATION

LIMITED WARRANTY

I. CUSTOMER OBLIGATIONS

A. CUSTOMER assumes full responsibility that this Radio Shack computer hardware purchased (the "Equipment"), and any copies of Radio Shack software included with the Equipment or licensed separately (the "Software") meets the specifications, capacity, capabilities, versatility, and other requirements of CUSTOMER.

B. CUSTOMER assumes full responsibility for the condition and effectiveness of the operating environment in which the Equipment and Software are to function, and for its installation.

II. RADIO SHACK LIMITED WARRANTIES AND CONDITIONS OF SALE

A. For a period of ninety (90) calendar days from the date of the Radio Shack sales document received upon purchase of the Equipment, RADIO SHACK warrants to the original CUSTOMER that the Equipment and the medium upon which the Software is stored is free from manufacturing defects. THIS WARRANTY IS ONLY APPLICABLE TO PURCHASES OF RADIO SHACK EQUIPMENT BY THE ORIGINAL CUSTOMER FROM RADIO SHACK COMPANY-OWNED COMPUTER CENTERS, RETAIL STORES AND FROM RADIO SHACK FRANCHISEES AND DEALERS AT ITS AUTHORIZED LOCATION. The warranty is void if the Equipment's case or cabinet has been opened, or if the Equipment or Software has been subjected to improper or abnormal use. If a manufacturing defect is discovered during the stated warranty period, the defective Equipment must be returned to a Radio Shack Computer Center, a Radio Shack retail store, participating Radio Shack franchisee or Radio Shack dealer for repair, along with a copy of the sales document or lease agreement. The original CUSTOMER'S sole and exclusive remedy in the event of a defect is limited to the correction of the defect by repair, replacement, or refund of the purchase price, at RADIO SHACK'S election and sole expense. RADIO SHACK has no obligation to replace or repair expendable items.

B. RADIO SHACK makes no warranty as to the design, capability, capacity, or suitability for use of the Software, except as provided in this paragraph. Software is licensed on an "AS IS" basis, without warranty. The original CUSTOMER'S exclusive remedy, in the event of a Software manufacturing defect, is its repair or replacement within thirty (30) calendar days of the date of the Radio Shack sales document received upon license of the Software. The defective Software shall be returned to a Radio Shack Computer Center, a Radio Shack retail store, participating Radio Shack franchisee or Radio Shack dealer along with the sales document.

C. Except as provided herein no employee, agent, franchisee, dealer or other person is authorized to give any warranties of any nature on behalf of RADIO SHACK.

D. Except as provided herein, **RADIO SHACK MAKES NO WARRANTIES, INCLUDING WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.**

E. Some states do not allow limitations on how long an implied warranty lasts, so the above limitation(s) may not apply to CUSTOMER.

III. LIMITATION OF LIABILITY

A. EXCEPT AS PROVIDED HEREIN, RADIO SHACK SHALL HAVE NO LIABILITY OR RESPONSIBILITY TO CUSTOMER OR ANY OTHER PERSON OR ENTITY WITH RESPECT TO ANY LIABILITY, LOSS OR DAMAGE CAUSED OR ALLEGED TO BE CAUSED DIRECTLY OR INDIRECTLY BY "EQUIPMENT" OR "SOFTWARE" SOLD, LEASED, LICENSED OR FURNISHED BY RADIO SHACK, INCLUDING, BUT NOT LIMITED TO, ANY INTERRUPTION OF SERVICE, LOSS OF BUSINESS OR ANTICIPATORY PROFITS OR CONSEQUENTIAL DAMAGES RESULTING FROM THE USE OR OPERATION OF THE "EQUIPMENT" OR "SOFTWARE". IN NO EVENT SHALL RADIO SHACK BE LIABLE FOR LOSS OF PROFITS, OR ANY INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY BREACH OF THIS WARRANTY OR IN ANY MANNER ARISING OUT OF OR CONNECTED WITH THE SALE, LEASE, LICENSE, USE OR ANTICIPATED USE OF THE "EQUIPMENT" OR "SOFTWARE".

NOTWITHSTANDING THE ABOVE LIMITATIONS AND WARRANTIES, RADIO SHACK'S LIABILITY HEREUNDER FOR DAMAGES INCURRED BY CUSTOMER OR OTHERS SHALL NOT EXCEED THE AMOUNT PAID BY CUSTOMER FOR THE PARTICULAR "EQUIPMENT" OR "SOFTWARE" INVOLVED.

B. RADIO SHACK shall not be liable for any damages caused by delay in delivering or furnishing Equipment and/or Software.

C. No action arising out of any claimed breach of this Warranty or transactions under this Warranty may be brought more than two (2) years after the cause of action has accrued or more than four (4) years after the date of the Radio Shack sales document for the Equipment or Software, whichever first occurs.

D. Some states do not allow the limitation or exclusion of incidental or consequential damages, so the above limitation(s) or exclusion(s) may not apply to CUSTOMER.

IV. RADIO SHACK SOFTWARE LICENSE

RADIO SHACK grants to CUSTOMER a non-exclusive, paid-up license to use the RADIO SHACK Software on **one** computer, subject to the following provisions:

A. Except as otherwise provided in this Software License, applicable copyright laws shall apply to the Software.

B. Title to the medium on which the Software is recorded (cassette and/or diskette) or stored (ROM) is transferred to CUSTOMER, but not title to the Software.

C. CUSTOMER may use Software on one host computer and access that Software through one or more terminals if the Software permits this function.

D. CUSTOMER shall not use, make, manufacture, or reproduce copies of Software except for use on **one** computer and as is specifically provided in this Software License. Customer is expressly prohibited from disassembling the Software.

E. CUSTOMER is permitted to make additional copies of the Software **only** for backup or archival purposes or if additional copies are required in the operation of **one** computer with the Software, but only to the extent the Software allows a backup copy to be made. However, for TRSDOS Software, CUSTOMER is permitted to make a limited number of additional copies for CUSTOMER'S own use.

F. CUSTOMER may resell or distribute unmodified copies of the Software provided CUSTOMER has purchased one copy of the Software for each one sold or distributed. The provisions of this Software License shall also be applicable to third parties receiving copies of the Software from CUSTOMER.

G. All copyright notices shall be retained on all copies of the Software.

V. APPLICABILITY OF WARRANTY

A. The terms and conditions of this Warranty are applicable as between RADIO SHACK and CUSTOMER to either a sale of the Equipment and/or Software License to CUSTOMER or to a transaction whereby RADIO SHACK sells or conveys such Equipment to a third party for lease to CUSTOMER.

B. The limitations of liability and Warranty provisions herein shall inure to the benefit of RADIO SHACK, the author, owner and/or licensor of the Software and any manufacturer of the Equipment sold by RADIO SHACK.

VI. STATE LAW RIGHTS

The warranties granted herein give the **original** CUSTOMER specific legal rights, and the **original** CUSTOMER may have other rights which vary from state to state.

TRS-8Ø® Model 4/4P
Disk System Owner's Manual

The FCC Wants You to Know . . .

This equipment generates and uses radio frequency energy. If not installed and used properly, that is, in strict accordance with the manufacturer's instructions, it may cause interference to radio and television reception.

It has been type tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient the receiving antenna
- Relocate the computer with respect to the receiver
- Move the computer away from the receiver
- Plug the computer into a different outlet so that computer and receiver are on different branch circuits.

If necessary, you should consult the dealer or an experienced radio/television technician for additional suggestions. You may find the following booklet prepared by the Federal Communications Commission helpful: *How to Identify and Resolve Radio-TV Interference Problems.*

This booklet is available from the US Government Printing Office, Washington, DC 20402, Stock No. 004-000-00345-4.

Warning

This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC Rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

Model 4P Only

The TRS-80 Model 4P Computer (Catalog Number 26-1080) is equipped with an external I/O bus to enable connection of a Radio Shack TRS-80 Five Meg Disk System (Catalog Number 26-1130). If you wish to connect the Five Meg Disk System to this I/O bus, the classification of the Model 4P is changed to Class A and the following warning applies:

Warning

This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instructions manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interferences.

10 9 8 7 6 5 4 3 2 1

Introduction

About the Model 4/4P

Congratulations on the purchase of your TRS-80 Model 4 or Model 4P Disk System. Your new Model 4 or Model 4P is a compact computer that is perfect for business needs, as well as personal use. You will find it to be a valuable tool which will save you work as well as give you hours of enjoyment. Its basic features include:

- 64K (65536 characters) of random access memory, expandable to 128K
- High-speed Z-80A microprocessor, the "brains" of the computer
- Upper and lower case text display of 80 characters by 24 lines or 64 by 16 (software selectable)
- Compatible with Radio Shack's Model III Software Library.
- One or two built-in drives that let you use single-sided, double-density floppy disks.
- Sound generation.
- 70-key console keyboard which includes three function keys and numeric keypad.
- Built-in printer interface.

As your needs grow — you can expand your Model 4 or Model 4P to include hard disks, external floppy disk drives (Model 4 only), high-resolution graphics, printers, RS-232C communications and more.

About This Manual

This manual shows how you can use your disk system to:

- Store, retrieve or manipulate information on disk (using TRSDOS).
- Write programs for Model 4 and Model 4P (using BASIC).

The Model 4/4P's operating system is TRSDOS Version 6. Throughout this manual, we refer to this system simply as TRSDOS.

In the *Introduction To Your Disk System* manual, we covered all the essential information to get you started. As you learn more about TRSDOS and programming, you can take advantage of its many features explained in this manual.

Since this is a reference manual, you don't have to read it from front to back. If you are a programmer, you'll find a lot of useful information in this manual. If you are an advanced programmer, you'll find additional "advanced" information included for you in this manual. Additional technical information is available in *The Technical Reference Manual* (Cat. No. 26-2110). This manual is available at your local Radio Shack store.

Part I/ TRSDOS

Section I/ Using TRSDOS describes how to start up TRSDOS, what TRSDOS Ready means, and some general information on how TRSDOS works.

Section II/ TRSDOS Commands contains a number of commands and utilities you will find helpful.

PART II/ BASIC

Section III/ Operations explains how to start up and operate BASIC.

Section IV/ BASIC Language describes (1) BASIC concepts, (2) how to store data on disks, and (3) each BASIC statement and function.

Table of Contents

Part I/TRSDOS Version 6

Section I/ Using TRSDOS

Section I/ Using TRSDOS

How The Computer Uses TRSDOS

Whenever you are using a program which runs under TRSDOS, your computer will, from time to time, need to reference TRSDOS. It always looks for TRSDOS on Drive 0.

For this reason, you must at all times have TRSDOS in Drive 0.

TRSDOS Notations

For clarity and brevity, we use some special notations and type styles in this section.

CAPITALS and punctuation
indicates material that you must enter exactly as it appears or material that you see on your computer's video display.

(KEYBOARD CHARACTER)
indicates key you press.

italics
represents words, letters, characters, or values that you supply.

TRSDOS Terms

Below is a listing of terms which we use frequently in this section. The italicized words represent variable information which you must supply.

command	represents the TRSDOS command you want to execute. *command* can be in upper or lowercase letters.
(*parameters*)	is a list of one or more values that may be needed by the command. Some commands have no parameters. Most parameters are optional. Brackets [ ] around any word in a command line indicate that it is optional.
filespec	is a standard TRSDOS file specification having the general form: *filename/ext.password:drive*
devspec	is (1) one of six standard TRSDOS device specifications, or (2) a user created device specification having the general form: **two-letter abbreviation*
diskette	refers exclusively to a floppy diskette.
disk	refers to a floppy diskette.

disk ID	refers to the disk NAME, creation date, and Master Password.
I/O	refers to a transfer of data (Input/Output).

TRSDOS Abbreviations

You can abbreviate a *parameter* to its first letter (unless otherwise stated in the *command* explanation). You can also abbreviate YES to Y and NO to N.

Loading TRSDOS

When you install and power up your system, you'll see the TRSDOS start-up logo. This means you're in the TRSDOS Version 6 Operating System. You then need to enter the current date in the form *mm/dd/yy*. For example, for June, 14, 1983, type:

```
06/14/83 (ENTER)
```

The system displays the date in expanded form (for example, Tue, Jun 14, 1983).

TRSDOS Ready

Whenever you see the TRSDOS Ready prompt you know that you are in control of TRSDOS — not COBOL, PAYROLL, or any of your application programs. Being in control of TRSDOS allows you to do one of these operations:

- execute a TRSDOS system command or utility program
- execute an application program

When an error occurs, it came from one of two places: TRSDOS or the application program that you are running. If it came from TRSDOS, look in Appendix D or the TRSDOS command section for an explanation of the error message. If it came from the application program that is running, you'll need to see the manual which came with the application program for an explanation of the error message.

Executing A Command

You can execute a TRSDOS system command whenever you see the TRSDOS Ready prompt. The command you type can consist of up to 79 upper or lower case characters. You must enter the command by pressing (ENTER).

For example, if you want to see the TRSDOS system commands, type:

```
    LIB (ENTER)
```

TRSDOS displays a list of all the available system commands and returns to TRSDOS Ready:

```
Library <A>
  Append Copy   Device Dir    Do     Filter Lib
  Link   List   Load   Memory Remove Rename Reset
  Route  Run    Set

Library <B>
  Attrib Auto   Build  Create Date   Debug  Dump
  Free   Purge  Time   Verify

Library <C>
  Forms  Setcom SetKi  Spool  Sysgen System
```

If you wanted this display to print on the printer, type (CTRL)(:). Whenever you press this key sequence, what is displayed on the screen is printed on the printer.

Disk Files

You can keep a record of anything you type into your Model 4 by storing it on disk in a "disk file." A disk file can contain a program, a collection of data, a project report you intend to make, or almost anything you want it to contain. But whatever it is, if you want to keep it permanently, you'll have to store it in a disk file.

When the computer stores the file, it records the name of the file and its disk location in a special place on the disk called the disk's directory. Whenever you want to access the file, the computer can immediately find its location by using this directory.

Filespec

Whenever you create a disk file, you need to give it a name. This name is just one part of a file specification — filespec, for short. The filespec is the standard TRSDOS format you'll use every time you reference your file:

filename/ext.password:drive

filename
The name of your file can be most anything you like, as long as it is one to eight alphanumeric characters, the first of which must be a letter. (The only names you cannot use are TO, ON, USING and OVER.)

/extension
If you want to further identify your file, you can give it a second name by adding an extension. An extension (indicated by */ext* on our filespec) is a sequence of one to three alphanumeric characters (the first of which must be a letter) with a preceding slash (/).

You can use an extension to provide additional information on a file, or you can use an extension to indicate the type of file you have.

.password
A password protects a file by limiting access to it. You can accomplish this protection via a password either when you create the file or with the ATTRIB command.

A password is a sequence of up to eight alphanumeric characters, the first of which must be a letter.

:drive
Often when you're using your computer, you'll have more than one disk drive in use. You can speed up the file access time by specifying the drive the desired file is on.

If you omit a drive number on the filespec, your computer automatically starts looking for the file on all available drives, beginning with Drive 0.

Here are some examples of valid TRSDOS filespecs:

```
DOPROG.OPEN
CLR/BAS:1
MOD16:4
STL12/TXT.ARCH:1
GAME1
THESIS/OLD:2
CONTEMP:3
```

You cannot use TO, ON, OVER, or USING as TRSDOS filespecs.

Partspecs

Certain system commands and utilities allow you to specify a collection of files by using a "partspec." A partspec is used with a "wildcard" mask ($). When you use a wildcard in a partspec, it represents a wildcard field and means "any character." For example, suppose the following files exist on a disk in Drive 1:

```
A                         ACORN
ADVANCE/DAT               ADVISE/DAT
BILLING/CMD               BILLING
BILLING/BAK               BILLING/DAT
```

If you issue the command:

```
DIR A:1 (ENTER)
```

TRSDOS displays these files:

```
A                         ACORN
ADVANCE/DAT               ADVISE/DAT
```

All files on the disk that begin with the letter A are displayed because when you specify a partspec, TRSDOS treats the command as "DIR of all files that begin with 'A'."

If you issue the command:

```
DIR BILLING:1 (ENTER)
```

TRSDOS displays:

```
BILLING                   BILLING/CMD
BILLING/DAT               BILLING/BAK
```

Because you did not specify an extension, TRSDOS assumed that all extensions are acceptable.

If you issue the command:

```
DIR /$A:1 (ENTER)
```

TRSDOS displays the files on the disk which have an A as the second character in their extension:

```
ADVANCE/DAT               ADVISE/DAT
BILLING/DAT               BILLING/BAK
```

Because you did not specify a filename, TRSDOS assumed that all filenames are acceptable.

A wildcard character must always have at least one character to its right. The following partspecs are identical:

```
A                         A$
A$$$                      A$/$
A$/$$$
```

Devices

There are two kinds of TRSDOS devices: physical and logical.

A physical device is a piece of your computer hardware: the video display, the keyboard, the printer, etc.

A logical device (devspec) is a connection between TRSDOS and a physical device.

TRSDOS lets you treat your devices independently, which means you can sometimes substitute a device for another one. You can also substitute a file for a device. See the LINK, ROUTE, and SET library commands.

Devspec

When you want to access a device, you use its device specification or *devspec*.

TRSDOS devices already have devspecs assigned to them. You assign devspecs to devices that you create. The devspec is the standard TRSDOS format you'll use every time you reference your device:

two-letter abbreviation

Your original TRSDOS master diskette is configured with six devices. They are:

Devspec	Device
*KI	Keyboard Input
*DO	Display Output (Video)
*PR	Printer
*SI	Standard Input
*SO	Standard Output
*JL	Job Log

Drivers And Filters

Each device is controlled by its own driver program, filter program, or both. You can change a device's I/O by manipulating its driver or filter program. For more information on drivers and filters see the SET and FILTER commands; see also Appendices I and K.

Section II/ TRSDOS Commands

Section II/ TRSDOS Commands

TRSDOS commands and utilities (typed in at the TRSDOS Ready level) perform a variety of helpful operations:

Diskette Handling commands allow you to prepare your blank diskettes for use or make copies of existing diskettes. Any time you use a blank diskette, you should use this command:

FORMAT

If you want to change the way your computer system starts up and initializes its parameters, you can use *Initialization* commands. For example, you can use the FORMS commands to set your printer's parameters; or you can use the AUTO command to set your computer to AUTOmatically perform a particular function at start-up. The Initialization commands are:

AUTO	BOOT
DATE	FORMS
SETCOM	SETKI
SYSGEN	SYSTEM
TIME	

You might find the *Auxiliary* commands helpful for such functions as seeing what is on your disk or simply seeing what system commands are available. They include:

DEVICE	DIR
DO	FREE
LIB	LIST
LOG	SPOOL
VERIFY	

The *File Handling* commands and utilities allow you to copy, rename, delete, or convert your disk files. These commands include:

APPEND	ATTRIB
BACKUP	BUILD
COMM	CONV
COPY	CREATE
DEBUG	PATCH
PURGE	REMOVE
RENAME	REPAIR
TAPE100	

The *Device Handling* commands allow you to set, filter, route, or reset your devices. Be sure you have a good understanding of devices before you use these commands! These commands include:

FILTER	LINK
MEMDISK	RESET
ROUTE	SET

Machine Language File Handling commands create and execute machine language disk files. These commands include:

DUMP LOAD
MEMORY RUN

How to Use This Section

This section contains an alphabetic listing of all TRSDOS commands and utilities. The commands and utilities for advanced programmers are marked as "Advanced Programmer's Utilities" and "Advanced Programmer's Commands."

Commands

Commands are system operations that can be used at TRSDOS Ready.

To see a list of all library commands, use the LIB command. Type:

```
LIB (ENTER)
```

and the following list is displayed:

```
Library <A>

  Append Copy   Device Dir    Do     Filter Lib
  Link   List   Load   Memory Remove Rename Reset
  Route  Run    Set

Library <B>

  Attrib Auto   Build  Create Date   Debug  Dump
  Free   Purge  Time   Verify

Library <C>

  Forms  Setcom SetKi  Spool  Sysgen System
```

Utilities

Utilities use some or all of user memory. They return to TRSDOS Ready; under most conditions you cannot use them effectively within programs.

The utilities are:

BACKUP COMM
CONV FORMAT
LOG PATCH
REPAIR

Entry Organization

Each entry in this section is identified as either a command or a utility.

The command's "syntax" is the first line you see after the keyword. Use it as your guide to type in a command. (See "Syntax" below for details.) If a word or value in the syntax is **highlighted**, you need to type in that word or value for the command to work.

A description of the command or utility follows the syntax. This description tells you what the command or utility does. Next, the entry includes additional information on the parameters of the command. A command may require you to supply some values.

The definition also may offer several "options" that customize the command to your needs. These optional parameters increase the usefulness of the commands but are not necessary for normal operation. Values and options are discussed in the additional parameter information.

Finally, each entry gives examples of the command's use.

Syntax

The command's syntax tells you what format to use when you type the command.

For example, here is the syntax for the REMOVE command:

REMOVE *filespec*, *filespec* . . .

italicized words in a command's syntax indicate values that you supply. In this case, the value that you supply is a filename. Remember that a command's required parameters are **highlighted**, so the beginner can skip values not boldfaced.

If you want to remove the disk file named SAMPLE in Drive 1, type:

```
REMOVE SAMPLE:1 (ENTER)
```

The syntax for the COPY command is:

COPY *source* [TO] ***destination*** [(*parameters*)]

Here, you must supply the name of the source filespec you wish to copy and the destination filespec to which you want it copied. Remember that TO is optional and cannot be used as a TRSDOS filespec. For example:

```
COPY NEW/DAT:1 TO NEWDAT/ONE:2 (ENTER)
```

copies the Drive 1 file NEW/DAT onto the diskette in Drive 2 and names the new file NEWDAT/ONE.

The COPY command offers four optional parameters. They are:

LRL
CLONE
ECHO
X

If you need the LRL parameter (the LRL parameter tells TRSDOS to assign a specific record length to a destination filespec), type:

```
COPY NEW/DAT:1 TO NEWDAT/ONE:2 (LRL=128) (ENTER)
```

Every command uses some variation of the syntax lines discussed above.

APPEND

Command

APPEND ***source*** [TO] ***destination*** [(*parameters*)]

Appends the contents of the *source* onto the end of the contents of the *destination*. (The contents of the source file remain the same.)

You can use APPEND to combine two files on a disk.

The *source* is a valid TRSDOS filespec or input devspec, and *destination* is a valid TRSDOS filespec.

The parameters are:

ECHO echoes the characters to the screen when appending a device to a file.
STRIP backspaces the destination file one byte before the append begins. Use this parameter with files, such as SCRIPSIT files, that have an "internal end of file marker."

Examples

```
APPEND EAST/DAT:1 TO WEST/DAT:0 (ENTER)
```

adds the information in EAST/DAT on Drive 1 onto the end of the information in WEST/DAT on Drive 0.

```
APPEND *KI TO WEST/DAT:0 (ENTER)
```

appends the information that you type on the keyboard to the end of WEST/DAT on Drive 0. Press (CONTROL)(SHIFT)(@) (at the same time) to end the append.

```
APPEND *KI TO WEST/DAT:0 (ECHO) (ENTER)
```

displays what you are appending to WEST/DAT as you type it. Press (CONTROL)(SHIFT)(@) (at the same time) to end the append.

Error Conditions

If the records in *source* and *destination* are not the same length, then an error message appears.

APPEND is mainly for data files since it works only with files containing ASCII text. You cannot APPEND program files that are in the "load module format." **Also, you cannot APPEND BASIC programs unless they are saved in the ASCII format.**

Some programs (such as SCRIPSIT) place a special marker at the end of a file. If this marker is in the file, you must use the STRIP parameter when appending to it. If you do not use the STRIP parameter, the program ignores the appended section of the file. Consider the following example:

```
APPEND EAST/DAT:1 TO WEST/DAT:0 (STRIP) (ENTER)
```

copies the information from EAST/DAT to WEST/DAT as in the above example, except that the first byte of EAST/DAT to overwrite the last byte of WEST/DAT.

Error Conditions

When an APPEND aborts with an error, the destination file involved in the APPEND may be left open. Use the RESET library command to close the file.

Sample Uses

Suppose you have two data files, PAYROLL/A and PAYROLL/B.

PAYROLL/A	PAYROLL/B
Atkins, W. R.	Lewis, G. E. ....
Baker, J. B.	Miller, L. O. ....
Chambers, C. P.	Peterson, B. ....
Dodson, M. W.	Rodriguez, F. ....
Kickamon, T. Y.	

You can combine the two files with the command:

```
APPEND PAYROLL/B TO PAYROLL/A (ENTER)
```

PAYROLL/A now looks like this:

```
        PAYROLL/A
Atkins, W. R. ................
Baker, J. B. .................
Chambers, C. P. ............
Dodson, M. W. ..............
Kickamon, T. Y. .............
Lewis, G. E. ................
Miller, L. O. .................
Peterson, B. ................
Rodriguez, F. ...............
```

PAYROLL/B is unaffected. To see the APPENDed file, type LIST PAYROLL/A.

Assigns protection passwords and attributes to a particular file or a group of files.

You can use ATTRIB to protect a file with passwords.

	Command
ATTRIB *filespec* (*parameters*)	

For filespec ATTRIBs, the parameters are:

USER = "*password*" sets the user password to *password*. If this parameter is omitted, the user password remains the same. If USER = is specified with no *password*, then any current user password is removed.

OWNER = "*password*" sets the owner password to *password*.If this parameter is omitted, the owner password remains the same. If OWNER = is specified with no *password*, then any current owner password is removed.

PROT = *level* specifies the protection level that is enforced if the user password is specified. If this parameter is omitted, the level is unchanged. You have to give a file an OWNER password before you can set PROT. The optional levels for access to a file are:

EXEC	Execute only
READ	Read and execute
UPDATE	Update, read, and execute
WRITE	Write, update, read, and execute
RENAME	Rename, write, update, read, and execute
REMOVE	Remove, rename, write, update, read, and execute (allows total access except for changing attributes with the ATTRIB command.
FULL	Allows total access

VIS specifies the filespec as visible in the directory

INV specifies the filespec as invisible in the directory. Use the INV parameter to reduce the number of files that TRSDOS displays when you issue a DIR command.

You can abbreviate the levels of PROTection to their first two letters except for RENAME and REMOVE, which you can abbreviate to RN and RM respectively.

	Command
ATTRIB [:*drive*] **(*disk* parameters)**	

For disk ATTRIBs, the parameters are:

LOCK protects all visible files not currently protected by setting their user and owner passwords to the disk master password.

UNLOCK removes the user and owner passwords from visible files if their passwords match the disk master password.

MPW="*password*" states the disk's current master password. If you don't specify this option, TRSDOS prompts you for it, if the password is not PASSWORD.

NAME[="*disk name*"] specifies the new disk name. If this parameter is omitted, the disk name remains the same.

PW[="*password*"] sets the new disk master password to *password*. If this parameter is omitted, the disk master password remains the same.

PW cannot be abbreviated.

drive defaults to Drive 0.

Assigning Protection Attributes To a File

Using the Owner and User Passwords. Passwords are first assigned when the file is created. At that time, the **owner** and **user passwords** are set at the same value (either the password you specified, or a blank password if you did not specify one).

ATTRIB allows you to assign a file two different passwords. The user password could be for the operator. It protects a file's contents at a certain protection level (set by PROT). For example, if you want an operator to have limited access to a file, you can set the PROTection level to READ. Then, using the user password, the operator will be able only to read (list) and execute the file, not change, rename, re-attrib, or remove it.

In the same manner, the owner password could be for the programmer. Using the owner password, the programmer could change, remove, re-attrib, or rename the same file. (When you use the owner password to access a file, TRSDOS ignores the PROTection level.)

In short, the user password allows limited access to a file and the owner password allows total access.

Examples

```
ATTRIB CUSTFILE/DAT:1
(USER= ,OWNER="BOSSMAN" ,PROT=READ) (ENTER)
```

sets the user password blank (so no password is necessary to access the file), sets the owner password to BOSSMAN, sets the protection level to read and execute only.

```
ATTRIB CUSTFILE/DAT.BOSSMAN
(USER="SECRET",PROT=EXEC,INV) (ENTER)
```

re-attribs CUSTFILE/DAT. Note that the owner password BOSSMAN was required to re-attrib the file. Now, CUSTFILE/DAT has the user password SECRET, keeps owner password BOSSMAN, has the protection level of execute only, and is invisible in the directory.

Assigning Protection Attributes To a Disk

The ATTRIB command also allows you to change the disk name, the disk master password, and the password protection of all visible filespecs.

Examples

```
ATTRIB (UNLOCK,NAME="MYDISK") (ENTER)
```

removes all user and owner passwords from the visible filespecs on Drive 0 if the filespecs' current password matches the disk master password. It also changes the disk name to MYDISK. Since the current master password was not specified with the MPW parameter, your computer asks you for it (if it is other than PASSWORD) before it executes this command.

```
ATTRIB :1
(NAME="DATA",PW="SECRET",MPW="BOSSMAN") (ENTER)
```

sets the disk name in Drive 1 to DATA, changes the master password to SECRET if the current disk master password is BOSSMAN.

```
ATTRIB (LOCK) (ENTER)
```

prompts you for the disk's master password (if other than PASSWORD) and changes the user and owner passwords of all visible, non-password protected files to the disk's current master password. Since no drive was specified, the command is carried out on Drive 0.

```
ATTRIB :1 (NAME) (ENTER)
```

prompts you for Drive 1's disk master password (unless it is PASSWORD). It then prompts you for the new disk name.

Sample Uses

Suppose you have a data file, PAYROLL, and you want an employee to use the file in preparing paychecks. You want the employee to be able to read the file but not to change it. Then use a command like:

```
ATTRIB PAYROLL (I,USER="PAYDAY", OWNER="BANANA"
PROT=READ) (ENTER)
```

Now tell the clerk to use the password PAYDAY (which allows read only); while only you know the password, BANANA, which grants total access to the file.

Command

AUTO [*parameters*] [*][***command line***]

Stores an AUTO *command line*. This *command line* automatically executes whenever you start up or reset TRSDOS. (That is, after you enter the date and time, TRSDOS loads, executes the command line, and displays the TRSDOS Ready or BASIC prompt.)

You can use AUTO to automatically run a program after you type in the date.

command line is limited to 74 characters in length.

The *parameters* are:

:*drive* specifies which drive to store the AUTO command line on.
?[:*drive*] displays any AUTO command line stored on *drive*. (The default is Drive 0.)
=[:*drive*] executes any AUTO command line stored on *drive*. (The default is Drive 0.)

In most cases, you can override the AUTO command during start-up or reset by (1) holding down the ENTER key, or (2) pressing BREAK while the auto command is executing.

The exception to this is when you store the AUTO command with the * parameter (which disables the BREAK key and the ability of the ENTER key to override AUTO).

If the AUTO command disables the BREAK key and the program is non-functional, gaining control of the disk requires several steps. To regain control:

1. Start up the system with another non-AUTOed disk in Drive 0.
2. When TRSDOS Ready appears, place the non-functional disk in Drive 0.
3. Type AUTO and press ENTER, and the runaway AUTO command is removed from the disk.

Use the :*drive* parameter to place an AUTO command on a drive other than Drive 0.

Examples

```
AUTO BASIC ENTER
```

loads the BASIC program whenever you start up or reset on Drive 0.

```
AUTO ENTER
```

Turns off the AUTO function currently stored on Drive 0.

```
AUTO *DO INIT/JCL:1 ENTER
```

executes the DO file on Drive 1 named INIT/JCL whenever you start up or reset. Notice that the * parameter is used. This means the operator cannot use ENTER to halt the auto command; BREAK is also disabled.

```
AUTO :1 DEVICE ENTER
```

places the AUTO command DEVICE on Drive 1.

```
AUTO ?:1 ENTER
```

displays the AUTO command on Drive 1.

```
AUTO =:1 ENTER
```

executes the AUTO command on Drive 1.

Error Conditions

To place an AUTO command on a disk, it must be write-enabled.

The system does not check the command line for errors when you first enter the AUTO command line. Errors are detected when the command is executed.

Sample Use

Suppose you want the DEVICE library command to execute automatically when you restart your computer.

Do this by issuing the command:

```
AUTO DEVICE ENTER
```

Utility

BACKUP [*partspec*][*:source drive*][TO][*:destination drive*]
[(*parameters*)]

Duplicates (backs up) all or some of the files from *source drive* to *destination drive*.

You can use BACKUP to copy the contents of one disk to another.

Use the *parameters* to tell the system which group of files to duplicate. Use the *partspec* option to choose a group of files by name or extension. (See the Examples.) If no *parameters* or *partspec* are specified, all visible files are duplicated (unless the diskette types are the same, in which case TRSDOS performs a mirror-image backup).

Note: In most cases, you can see which files a BACKUP command would duplicate by issuing a DIR command of *source drive* using the same *partspec* and *parameters* as the BACKUP command.

If you do not specify *source drive* and *destination drive*, the system prompts you for them. If the source disk has a Master Password other than PASSWORD, and you do not state it with the MPW= parameter, the system prompts for it as well.

If the destination drive is not ready, the message `"Insert DESTINATION disk <ENTER>"` is displayed. Insert the destination disk and press ENTER to continue, or press BREAK to return to TRSDOS Ready.

The destination disk must be formatted before the backup begins. To format a disk, see the FORMAT command.

The parameters are:

MPW="*password*" specifies the source disk's Master Password
SYS backs up system files as well as the visible files
INV backs up invisible files as well as the visible files
MOD backs up files that have been modified since the last backup
QUERY=YES questions you about each file before it is backed up
OLD backs up only those files that already exist on the destination disk
NEW backs up only those files that do not already exist on the destination disk
X allows backups with no system disk in Drive 0
DATE="*M1/D1/Y1-M2/D2/Y2*" backs up files with modify dates between the two specified dates, inclusive. *M1/D1/Y1* must be before *M2/D2/Y2*.

="*M1/D1/Y1*" backs up files with modify dates equal to the specified date
="*-M1/D1/Y1*" backs up files with modify dates before or equal to the specified date
="*M1/D1/Y1-*" backs up files with modify dates after or equal to the specified date

MPW cannot be abbreviated.

When you specify QUERY=YES, the system questions you for each file before it is copied. Answer by pressing:

(Y)	to copy the file.
(N) or (ENTER)	to bypass the file and move on to the next one.
(C)	to copy the file, turn off the Query function, and automatically copy all remaining files.

After you type the BACKUP command, TRSDOS automatically performs one of the three types of backups: "mirror image," "backup by class," and "backup reconstruct." The difference between the three is of technical interest and is discussed in "General Information."

NOTE: A backup by class and backup reconstruct require two disk drives.

For information on backing up "backup limited" diskettes, see Appendix M, "Backup Limited Diskettes."

Backups With the (X) Parameter

When you specify the (X) parameter, you do not have to have a system disk in Drive 0 when you back up a disk. TRSDOS prompts you to insert the proper disks in the proper drive.

Examples

```
    BACKUP $:0 :1 (SYS,INV) (ENTER)
```

examines all files on the disk in Drive 0 and copies all files to Drive 1, because all files match the $ partspec. The partspec causes a backup by class.

NOTE: You can use this command to force a backup by class in situations where a mirror image would normally be performed. For example, it reduces fragmentation of files on the source disk by copying them in a more contiguous manner onto a newly formatted destination disk.

```
    BACKUP :0 :1 (MOD,QUERY=YES,MPW="SECRET") (ENTER)
```

copies all visible files from Drive 0 to Drive 1 that have been modified (written to) since the last backup. It questions you for each file before it is copied, showing the file's mod date and flag. The (MPW=) parameter states the Master Password, so the system does not prompt you for it.

```
BACKUP $/CMD:0 :1 (ENTER)
```

copies all visible files with the extension /CMD from Drive 0 to Drive 1. If the file already exists on Drive 1 it is overwritten. No other files on Drive 1 are touched. A backup by class is performed.

```
BACKUP /$$S:1 :2 (ENTER)
```

backs up all files whose extensions are three characters long and end with the letter S. The $ wildcard masks the first two characters of the extension, so extensions such as /BAS, /TSS, and /TRS form a match. A backup by class is performed.

```
BACKUP :1 :1 (ENTER)
```

backs up between two disks in Drive 1. You are prompted to switch the source disk and destination disk at the appropriate times.

The disks used in this type of backup must allow a mirror image backup, or the backup aborts.

This command and the following command could be used to back up a data disk.

```
BACKUP :0 :1 (X) (ENTER)
```

backs up the disk in Drive 0 to the disk in Drive 1. Its main use is to back up non-system disks, such as data disks, in a two-drive system.

When you use this parameter, you are prompted to insert the proper disk in Drive 0. You may be prompted to re-insert a system disk into Drive 0 during certain backups.

When the backup is complete, you are prompted to insert a system disk back in Drive 0.

```
BACKUP -/CMD:0 :1 (ENTER)
```

backs up all visible files from Drive 0 to Drive 1, EXCEPT those files that have a /CMD extension. A backup by class is performed.

```
BACKUP :1 :2 (NEW,QUERY=YES) (ENTER)
```

backs up only those visible files from Drive 1 that do not already exist on Drive 2. You are prompted before each file is moved.

```
BACKUP /ASM:3 :2 (DATE="05/06/82-05/10/82")
(ENTER)
```

backs up all visible files with the extension /ASM, whose modify dates fall on or between the specified dates.

Error Conditions

The destination disk must be formatted before the backup begins. To format a disk, see the FORMAT command.

For a backup by class, if the backup is to include system files, the destination disk must be newly formatted. BACKUP can't create a system disk if the destination disk contains data files. (Existing files may be using certain areas needed by the system.)

If you are backing up the entire disk, TRSDOS compares the source and destination disk Disk ID's to make sure they are identical. If the master passwords or disk names differ, you see the following message:

```
Destination disk ID is different --
NAME=disk name
     DATE=mm/dd/yy
Are you sure you want to backup to it <Y,N>?
```

Press (N) to abort the BACKUP or (Y) to continue.

If the disks' master passwords differ, the following message appears:

```
Destination disk ID is different -- NAME=disk
name
     DATE=mm/dd/yy
Enter its Master Password or <Break> to abort:
```

Press (BREAK) to abort the BACKUP or enter the password to continue.

If the source and destination disks have a different number of cylinders, the following message appears:

```
Cylinder counts differ - Attempt mirror-image
backup ?
```

Answer this question with (Y) to attempt a mirror image backup or with (N) to force a backup reconstruct.

If a mirror image backup is not possible, you get the error:

```
Backup aborted, destination not mirror-image
```

This appears if the destination disk is missing a cylinder that contains information on the source disk. This might be the case if the destination disk was formatted with fewer cylinders than the source disk, or if cylinders were locked out on the destination disk when it was formatted. You can use the FREE library command to check the destination disk for locked out cylinders.

After all the cylinders that contain data are copied to the destination disk, BACKUP attempts to remove the modification flags from the files on the source disk. If the disk is write protected, the following message appears:

```
Source disk is write protected; MOD flags not
updated
```

Backup by class may NOT be done on a single drive.

General Information

Mirror Image Backup. A mirror image backup is basically a cylinder-for-cylinder copy from the source to the destination disk. (Only those cylinders that actually contain data are copied.) When the backup is complete, the destination disk is an exact copy, or mirror image, of the source disk.

Backup By Class. A backup by class takes place if you specify a partspec or any parameter except X or MPW in the command line.

Backup Reconstruct. A backup by class and a backup reconstruct function identically. The only difference is that while **you** initiate a backup by class, the **system** initiates a backup reconstruct.

On certain TRSDOS application programs, you can only make a limited number of backups. And, when you make a backup on one of these programs, the source disk has to be write-enabled during the backup or the backup fails.

Backups With the (X) Parameter. This parameter allows you to back up data disks of different sizes or capacities on a two-drive system (using backup reconstruct). One-drive system can perform mirror-image backups only. When you use the (X) parameter to backup non-system disks of different sizes or capacities, system modules 2, 3 and 10 must first be put into memory with the SYSTEM (SYSRES=*number*) command. Remember that the (X) parameter is used only when there is a non-system disk in Drive 0.

Mirror Image Backup. TRSDOS makes a mirror image backup if the source and destination disks' size and density are identical, and if you specify no partspec or parameters (except X or MPW) in the command line. The number of cylinders doesn't need to be identical as long as the destination disk has at least as many cylinders as the source disk.

The date on the destination disk shown with the DIR or FREE library commands is changed to the current system date.

After the backup, the destination disk has its directory on the same track as the source disk regardless of where it was before the backup. The information on the destination disk is updated to reflect its true cylinder count and available free space.

Backup By Class. This type of backup does a file-for-file copy from the source to the destination disk. Files that are fragmented (spread over more than one extent) on the source disk are consolidated (if possible) on the destination disk.

Unlike a mirror image backup, files that exist on the destination disk but are not on the source disk are not touched in the backup. When the backup is complete, the destination disk contains all files moved from the source disk plus any other files that existed on the destination disk before the backup began.

The destination Disk ID is not changed by the backup.

When the file SYS0/SYS is included in a backup by class, the destination disk is configured in the following manner:

1. The state of the SYSGEN (on or off) is changed to match that of the source disk.
2. The initial date and time prompts (on or off) on power-up are set to match those of the source disk.
3. The default drive configurations match those of the source disk.

Backup Reconstruct. The system performs a backup reconstruct when the size or the density differs between the source and destination disks.

DIR/SYS and BOOT/SYS are not moved to the destination disk in this type of backup.

When a backup by class or a backup reconstruct occurs, only the type of files specified will be moved. (This is a departure from the Model III hard disk operating system LDOS 5.1.3.) For example, if you are moving files to a hard disk with the command BACKUP :5 :2, a backup reconstruct is invoked because the two disks are of different sizes. This command moves only the visible files.

If you want all of the files to be moved, then you must use the command BACKUP :5 :2 (SYS,INV). This moves visible, invisible, and system files to the destination disk.

Hard disk users should note that system files are stored in specific places in the directory. If you use BACKUP to move the visible files and then repeat the command with the SYS option, the backup aborts if the directory positions required for the system files are already in use. If this happens, you can use the PURGE command to delete the files that were moved to the disk, and then give the BACKUP command with the SYS option.

Sample Use

Suppose you have a payroll disk where all of the new employees have a file with an extension of /NEW and all of the old employees have a file with an extension of /OLD.

Now suppose you want to have two separate disks: one with old employee files and one with new employee files. You could issue the command:

```
BACKUP /NEW:0 :1 (ENTER)
```

to move all of the files of new employees from the master disk in Drive 0 to another disk in Drive 1.

Command

BOOT [*keys*]

Resets (boots) TRSDOS by returning it to its original start-up condition.

You can use BOOT to return your computer to the TRSDOS copyright and startup message.

The keys are:

CLEAR allows no sysgened configuration to take place.
ENTER allows no breakable AUTO commands to occur.
D enters the system debug. No sysgened configuration is loaded.

Note: When you use one or more of these keys, you must press and hold them down when the screen is erased and keep them down until the TRSDOS Ready message or the DEBUG display appears. If you are prompted for the date, you must hold down the keys as soon as you type the date and press ENTER. If you don't press the keys in time, simply reset and hold the keys down as soon as the screen clears.

BOOT loads the TRSDOS system in floppy Drive 0 back into the computer. It returns the computer back to its normal power-up configuration as if the system had been turned off and then turned on again.

Examples

Remember to hold down the *key* after you press ENTER until you see TRSDOS Ready or the debug display.

```
BOOT ENTER
```

resets the system.

```
BOOT ENTER CLEAR
```

returns the system to its original start-up condition and ignores any sysgened configuration.

```
BOOT ENTER ENTER
```

returns the system to its original start-up condition and ignores any breakable AUTO commands.

```
BOOT ENTER D
```

returns the system to its original start-up condition and enters the system debug. No sysgened configuration is loaded, and any AUTOed command is not executed. Note: If the AUTOed command is unbreakable, D is ignored.

Command

BUILD *filespec* [(*parameters*)]

Lets you enter data (such as commands) and save it on disk as *filespec*. If you do not specify an extension to *filespec*, it defaults to /JCL.

You can use BUILD to make a file on a disk.

The parameters are:

HEX accepts data in hexadecimal format only.
APPEND appends the BUILD data to the end of *filespec*.

Although you can build any type of data file with this command, it is mainly for creating files to be executed with the DO command, KSM/FLT, or the PATCH utility.

The HEX parameter lets you input data in hexadecimal form (see Appendix C for a listing of hexadecimal characters). You can use hex to generate control characters and graphics symbols which are not available from the keyboard.

The APPEND parameter lets you add data to the end of an existing file.

Some programs (such as SCRIPSIT) place their own marker at the end of a file. If this marker is in the file, you cannot append BUILD data to it unless you:

- Use the BUILD command to create a new file containing the information you wish to append.
- Use the APPEND library command with the STRIP parameter to properly append the new information to the existing file.

Building a File

When you enter the BUILD command with a non-existing *filespec*, BUILD creates the file and then allows you to insert lines.

You can enter a command line of up to 255 characters. JCL files are limited to 79 characters per line. To end a line, press (ENTER).

To end the file, press (CONTROL)(SHIFT)(@) at the beginning of a new line. The system returns you to TRSDOS Ready.

Examples

```
BUILD DISPLAY:2 (ENTER)
```

creates a new file named DISPLAY/JCL on Drive 2. TRSDOS allows you to insert lines. Type:

```
DEVICE (ENTER)
FREE :0 (ENTER)
FREE (ENTER)
(CONTROL)(SHIFT)(@)
```

The first three lines insert the DEVICE, and FREE :0 and FREE commands into the "DISPLAY" file. Pressing (CONTROL)(SHIFT)(@) tells TRSDOS that you are finished entering command lines. The system returns to TRSDOS Ready.

Now, whenever you type:

```
DO DISPLAY (ENTER)
```

TRSDOS executes the file by displaying the device table, the free space map of Drive 0, and the free space information for all enabled drives.

```
BUILD MYKEYS/KSM (ENTER)
```

builds MYKEYS on the first available drive. Since the /KSM extension was used, a KSM file is built. See the KSM/FLT filter in Appendix I for more information.

```
BUILD SPECIAL/:0 (ENTER)
```

builds SPECIAL on Drive 0. Adding the "/" allows SPECIAL to be built without an extension.

```
BUILD MYJOBS/JCL  (APPEND)  (ENTER)
```

searches all available drives for MYJOBS/JCL (until it is found) and adds the information from this build to the end of the file. If MYJOBS/JCL is not found, the file is built on the first available drive.

```
BUILD MYPROGA/FIX:0 (ENTER)
```

builds MYPROGA/FIX, which is to be used with PATCH. See the PATCH utility for more information.

```
BUILD DISPLAY/BLD (HEX) (ENTER)
```

builds a file on the first available drive, allowing data to be entered in hexadecimal format. Information is entered into this file as hexadecimal bytes (with no spaces or other delimiters between them).

The HEX parameter allows you to enter characters not directly available from the keyboard, such as control, printer control, and graphics characters. You can enter any one-byte character value.

A hex build allows 127 hex byte representations (254 characters) per logical line. Logical lines may continue on more than one physical line as long as a "0D" logical line terminator does not appear. Also, more than one logical line can appear on one physical line.

To create a character string containing graphics characters, type:

```
    818A90A10D (ENTER)
```

This line contains the hexadecimal bytes 81, 8A, 90, and A1. Notice that the byte values are packed together. "0D" ends a logical line, and (ENTER) ends a physical line.

If a non-hex digit is entered, the error message "Bad hex digit encountered" is displayed and the build aborts.

Error Conditions

If the filespec you specify already exists, you will get the error message "File already exists" (unless you specify the APPEND parameter).

Sample Use

Suppose you want to build a file to be used with the PATCH command. Issue the command:

```
    BUILD PROG/FIX (ENTER)
```

and enter the patch lines.

Utility

COMM *devspec* [(*parameters*)]

Lets two computers communicate via a device, usually the RS-232 communications line.

You can use COMM to let your computer talk with another computer.

COMM lets your computer:

- be used as a terminal in communicating with another computer.
- transfer files to and from another computer.
- spool output from the other computer to your printer.

Using COMM, you can access:

- Bulletin Board Systems
- News and Information Systems
- Timesharing Systems
- Electronic Mail Services

COMM can also communicate with systems that support XON/XOFF (Proceed/Pause) protocol. This is a protocol that uses two control codes named Device Control 1 and Device Control 3. (The Device Control codes are discussed in the (CLEAR) (SHIFT)(*) command section.)

devspec is usually *CL, the RS-232C communications line.

Note: Before you can use *CL, you have to SET it to its driver program COM/DVR. See Appendix I.

The *parameters* are:

XLATES=*X'aabb'* translates a character being sent.
XLATER=*X'aabb'* translates a character being received.
XON=*X'cc'* changes the XON code.
XOFF=*X'cc'* changes the XOFF code.

aa is the character to be translated.
bb is the character *aa* is translated into.
cc is the new value of XON or XOFF.

Enter hexadecimal values in the format X'nnnn'.

NULL=OFF	prevents any nulls (ASCII value 0) from being received.

XLATES and XLATER can be abbreviated to XS and XR.

XLATES and XLATER let you translate a character that you send and a character that you receive from another computer. (Only one character can be translated in each direction at any one time.)

Example

Suppose you are using COMM as a terminal to communicate with another computer, and you want to print a right bracket ( ]). It appears that you can't because there is no key on your keyboard that produces this character.

Use the XLATES parameter to produce a ( ]) by entering another character from the keyboard. Type:

```
    XLATES=X'Ø25D'
```

Now when you press (CTRL)(B) (hexadecimal Ø2), your computer sends the code for a right bracket (hexadecimal 5D) to the other computer. Since the Model 4 can display a right bracket when it receives a hex 5D, there is no need to use XLATER to translate the received character.

Characters that you receive from another computer can be translated to a different symbol using XLATER. Use the same method that we used for XLATES.

See Appendix C for a list of characters and their hex values.

The Function Keys

The Function Keys are used to:

1. Direct the flow of data (text or software) from device to device.
2. Enable and disable certain functions of COMM, including XON/XOFF and full/half duplex operation.

The Function Keys are divided into two groups: (1) the Application keys and (2) the Action keys.

The Application and Action keys are achieved by holding down all of the keys in the sequence, such as (CLEAR)(6) (hold down (CLEAR) and then press (6).

The Application Keys

The Function Keys (CLEAR)(1) through (CLEAR)(6) designate what device an action applies to.

Function Key	Device	Abbreviation
(CLEAR)(1)	Keyboard Device	(*KI)
(CLEAR)(2)	Display Device	(*DO)
(CLEAR)(3)	Printer Device	(*PR)
(CLEAR)(4)	Communications Line Device	(*CL)
(CLEAR)(5)	"Data Send" Device	(*FS)
(CLEAR)(6)	"Data Receive" Device	(*FR)

Action Keys

The remaining Function Keys perform an action. Some action keys require you to specify an application key (CLEAR 1 through CLEAR 6) before you can perform the action.

CLEAR 7

Causes the contents of the "Data Received" (*FR) area in memory to be written to disk. This is called "Dump-To-Disk" or DTD. DTD may be turned ON before or after a file is received. (You must turn DTD ON if a file will exceed the size of the *FR memory area.)

When you start COMM, DTD is ON. When you perform an *FR RESET (CLEAR 6 CLEAR 0), DTD is turned OFF. To turn it ON again, press CLEAR 7 followed by CLEAR :

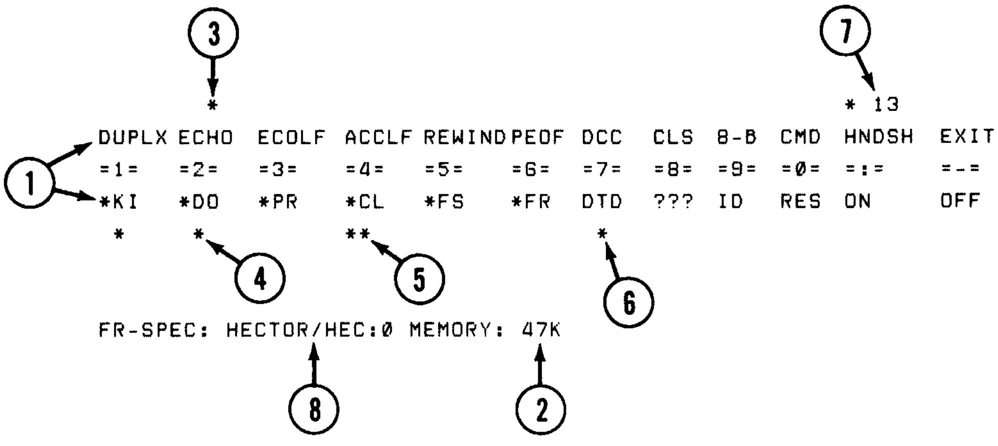
If you are writing data to floppy disks and the RS-232 port is running at a speed higher than 300 baud, you have to wait until you receive an entire file before turning DTD ON.

CLEAR 8

Displays the MENU of Function Keys on the display. You can use this command any time.

The display goes from left to right. This is not intended to be a complete menu, but a built-in "quick reference" card.

The screen display is altered to display the menu. Any data you receive while the menu is displayed is not lost because COMM saves the data in a special area of memory. This data appears on the screen after the menu is displayed.



```
         *                                                    * 13
DUPLX ECHO  ECOLF  ACCLF REWINDPEOF  DCC   CLS  8-B  CMD  HNDSH  EXIT
=1=   =2=   =3=    =4=   =5=    =6=   =7=   =8=  =9= =0=  =:=    =-=
*KI   *DO   *PR    *CL   *FS    *FR   DTD   ???  ID   RES  ON     OFF
 *     *           **                  *

FR-SPEC: HECTOR/HEC:0 MEMORY: 47K
```

1. The devices and functions. (The asterisks above and below the function keys indicate that the function is active.)
2. The amount of available memory.
3. Asterisks for the shifted function keys.
4. Asterisks for the unshifted functions keys.
5. Two asterisks denote a device capable of both input and output.
6. One asterisk denotes a device capable of either input or output.
7. If HANDSHAKE is active, the auto XOFF character selected is shown in hex.
8. Displayed filespec of any *FS or *FR filespec.

CLEAR 9

Specifies what file to use when you send or receive data. After you specify the file, your computer opens it. If you specify a file that does not exist, COMM creates it.

To specify the name of a receiving file, press CLEAR 6 followed by CLEAR 9 and answer the following prompt:

```
File Name:
```

COMM opens the file but does not set aside an area of memory to receive the data, so any incoming data is ignored.

To save incoming data, enter the command CLEAR 6 followed by CLEAR :. Now, data received is placed in the "Data Received" (*FR) area of memory, and the data is eventually placed in the file you specified. (See CLEAR : for more information on activating devices.)

If a file is already open, the system aborts your CLEAR 9 command and prints the warning message:

```
File Already Open
```

This warning prevents you from opening another file before closing this one. This protection also applies to files associated with the "Data Send" (*FS) area of memory.

CLEAR 0

Closes either a receive file or a send file. You must close a receive file so its directory can be updated, and so you can receive another file. If you reset a device, its buffer is cleared.

You must turn OFF the *FR or *FS device before you can close the associated file. (See CLEAR – for information on turning OFF devices.)

CLEAR :

Turns ON a device. This is the second command of a two-command sequence.

For example, if you want to turn ON the printer, first press [CLEAR][3] to indicate that you want to do something with the printer, and then press [CLEAR][:] to indicate that you want to turn it ON. Press [CLEAR][3] followed by [CLEAR][-] to turn the printer OFF.

[CLEAR][-]

Turns OFF a device. This is the second command of a two-command sequence.

For example, if you want to turn the printer OFF, first press [CLEAR][3] to indicate that you want to do something with the printer, and then press [CLEAR][-] to indicate that you want to turn it OFF. Press [CLEAR][3] followed by [CLEAR][:] to turn the printer ON.

[CLEAR][SHIFT][!]

This is the DUPLEX control, which allows you to select Full-Duplex or Half-Duplex.

Full-Duplex and Half-Duplex indicate how data is sent from one computer to another on an RS-232C line.

- Half-Duplex is used with a computer that cannot read data while it is sending it or send data while it is receiving it. A [CLEAR][SHIFT][!] followed by a [CLEAR][:] indicates Half-Duplex.
- Full-Duplex is used with a computer that can read data while it is sending it or send data while it is receiving it. A [CLEAR][SHIFT][!] followed by a [CLEAR][-] indicates Full-Duplex operation.

When you start COMM, it is set to Full-Duplex (Duplex off).

Some computers and terminals combine Duplex and Echo functions, so you should know something about the computer or terminal you are communicating with.

[CLEAR][SHIFT]["]

Controls character "Echo"ing. Press [CLEAR][SHIFT]["] followed by [CLEAR][:] to turn Echo ON.

You should turn Echo ON if you are communicating with a computer or terminal that is operating in full-duplex, but does not display a copy of characters it is transmitting to you (called Local Copy).

Turning Echo ON causes your computer to transmit each character it receives back to the computer that sent it. This lets the person operating the other computer see what is being transmitted.

Caution: If both ends are set for Echo ON, then the first character sent is echoed back and forth indefinitely — or until one end turns Echo OFF.

(CLEAR)(SHIFT)(#)

This command controls Echoing linefeeds. When enabled, any carriage return your computer receives causes a linefeed character to be transmitted back to the other computer.

This command is useful since there are a large number of terminals and computers that treat a carriage return (ASCII 13) and linefeed (ASCII 10) as separate functions.

When you are communicating with another TRS-80 computer, you can turn OFF this function by pressing (CLEAR)(SHIFT)(#) followed by (CLEAR)(-).

(CLEAR)(SHIFT)($)

Controls the ability of your computer to accept a linefeed. COMM usually ignores the first linefeed after a carriage return, since most computers send both a carriage return and a linefeed.

In most cases, this command is not necessary on TRS-80's where an (ENTER) is treated as both a carriage return and linefeed.

(CLEAR)(SHIFT)(%)

Positions to the start of an *FR or *FS file, so you can start again. For example, if you are receiving a file and it aborts with an error, you can startover by pressing (CLEAR)(6) followed by (CLEAR)(SHIFT)(%). Then you can attempt to receive the file again.

(CLEAR)(SHIFT)(&)

Appends new data to the end of a file. This command applies to the "Data Received" (*FR) area of memory only. If you open an existing file and then press (CLEAR)(6) followed by (CLEAR)(SHIFT)(&), you can append new data to the end of the file.

(CLEAR)(SHIFT)(')

Displays control characters that are being received or sent. You can use this command to detect if you are receiving unwanted control characters. If you are receiving unwanted control characters, you can use the XLATER to translate them.

(CLEAR)(SHIFT)(()

Erases the contents of the screen and places the cursor in the upper left corner. No data is transmitted.

(CLEAR)(SHIFT)())

When followed by an ON command ((CLEAR)(:)), your computer uses all 8 bits of a character it receives. Normally, bit 8 is either not present or invalid, so COMM removes it from each character it receives.

Do not turn this option ON unless the RS-232C word length is set to 8. You can use the SETCOM library command to set the word length before you enter COMM.

(CLEAR)(SHIFT)(0)

Allows you to enter a TRSDOS **library command** from COMM.

For example, when you type:

```
    DEVICE (ENTER)
```

the device table is displayed on your screen. The message "Command complete" is displayed below the device table.

NOTE: If the specified library command attempts to change HIGH$, the command aborts and TRSDOS returns you to COMM.

(CLEAR)(SHIFT)(*)

This command controls the handshaking on the data line.

Handshaking is the agreed-upon method that two communicating computers use to control the flow of data between them. If this option is turned ON, COMM responds to the following codes when your computer receives them from the communications line:

Symbol	Value			Description
DC1	17	X'11'	—	Resume transmission (XON or Proceed character)
DC2	18	X'12'	—	Turns the *FR device ON
DC3	19	X'13'	—	Pause transmission (XOFF or Pause character)
DC4	20	X'14'	—	Turns the *FR device OFF

NOTE: You can use the XON and XOFF parameters to change the codes to other values when entering COMM.

The above codes are part of the ASCII standard. DC1 and DC3 control the device that is transmitting data. They tell it when to start and stop transmitting.

DC2 and DC4 control the recording device. They tell it when to start and stop recording. These codes use the "Received Data" device (*FR) to accumulate the received data. If you use DC2 and DC4, be sure you have created an *FR file with (CLEAR)(6) and (CLEAR)(9).

When a DC3 is received, it suspends transmission through the *CL device until a DC1 is received. Any characters that are received are accepted, but none are transmitted. You can override a received DC3 with the *CL ON command ((CLEAR)(4) (CLEAR)(:)).

Handshaking can also be activated when a given character is transmitted. Do this by pressing (CLEAR)(SHIFT)(*) followed by the character you want to pause on.

Typically, the (ENTER) key is specified so that line-at-a-time transmission occurs with automatic pausing at the end of each line.

When handshaking is ON, COMM pauses transmission whenever the specified character is transmitted until it (1) receives a DC1 from the other system, or (2) receives a *CL ON command from the keyboard.

(CLEAR)(SHIFT)(=)

Exits to TRSDOS Ready. It does not require any ON or OFF code.

Before COMM stops running, it checks the "Data Received" device (*FR) to see if any open files exist. If there is an open file, COMM closes it before it exits to TRSDOS. This feature prevents you from having unclosed files in your system.

Quick Reference Label

If you are a beginning COMM user, you may find it helpful to make a label containing each key's function and place the labels directly above the keyboard. Label the keys as follows:

Key	Unshifted	SHIFTed
1	*KI	Duplex
2	*DO	Echo
3	*PR	Echo-Linefeed
4	*CL	Accept-Linefeed
5	*FS	Rewind File
6	*FR	Position @ EOF
7	DTD	DCC
8	Menu	Clear Screen
9	ID	8-bit mode
0	Reset	Command
:	On	Handshaking
-	Off	Exit

Logging-On to CompuServe

You can use your Model 4 and COMM to log-on to CompuServe. To log on to CompuServe, you must first buy a Universal Sign-Up Kit (Radio Shack Cat. No. 26-2224). Next, follow these steps:

1. First, use the SET command to SET *CL to COM/DVR (see Appendix I). Then issue the command:

```
SETCOM (WORD=8,PARITY=OFF,STOP=1) (ENTER)
```

2. Type:

```
COMM *CL (ENTER)
```

3. Now you need to dial CompuServe's number that comes in the Universal Sign-Up Kit. Depending on which modem you are using, you either dial the number on a phone or you must enter the commands that cause the modem you are connected to dials the number for you. See your modem manual for the correct procedure.
4. After the number is dialed, wait for a "carrier tone" that CompuServe sends to tell you that you are connected.
5. Now press (CONTROL)(C) to send a hexadecimal value of 03 to CompuServe.
6. CompuServe prompts you on your video display with:

   ```
   User ID:
   Password:
   ```

   Answer each prompt with the numbers supplied in the Universal Sign-Up Kit and (ENTER).
7. You are now logged-on to CompuServe.

COMMunicating with Bulletin Board Systems

A Bulletin Board System (BBS) is typically a small computer used by individuals, schools, or companies that provides a communication link between its users.

With some TRS-80 Bulletin Board Systems, you can receive graphics characters. For you to be able to accept these graphics, the COM/DVR driver has to be initialized at 8 bits per word (see the SETCOM library command) and you have to use 8-bit mode in COMM ((CLEAR)(SHIFT)(])) followed by (CLEAR)(:)).

COMMunicating with Other Computers

This section shows you how to use COMM to communicate with other computers. The first example describes how a TRS-80 communicates with a mainframe computer. The second example describes how two TRS-80's can communicate.

COMMunicating with a Mainframe

When a TRS-80 communicates with a mainframe computer, in most cases it is not necessary to change the default device or function settings when you enter COMM. Most mainframes operate as the host computer while you operate as a terminal, and the mainframe provides echo functions for you. You must be sure to specify the RS-232C parameters when setting up the COM/DVR driver to match those expected by the mainframe.

To transfer a file from a mainframe to your TRS-80 computer, use the following procedure:

1. Type in the command which causes the mainframe to list the file, but do not press (ENTER).
2. Specify your receive file by pressing (CLEAR)(6) followed by (CLEAR)(9). Type in the filename in response to the prompt.
3. Press (CLEAR)(6) followed by (CLEAR)(:) to open the receive area of memory. If the file you wish to receive is larger than your available area of memory, you should then press (CLEAR)(7) followed by (CLEAR)(:). This causes the file to be written to the disk as it is being received.
4. Press (ENTER) to start the file listing.
5. When the listing is complete, press (CLEAR)(6) followed by (CLEAR)(-). to turn OFF the *FR and if you have not already done so, press (CLEAR)(7) followed by (CLEAR)(:) to write the file to disk.
6. When the disk write is complete, type (CLEAR)(6) followed by (CLEAR)(0) to turn off DTD and to close the receive file.

To transfer a file from your TRS-80 computer to a mainframe, use the following procedure:

1. Designate the file that you want to send by pressing (CLEAR)(5) followed by (CLEAR)(9) and entering the name of the file in response to the prompt.
2. Turn on the handshake mode by pressing (CLEAR)(SHIFT)(*) followed by (ENTER) (assuming that the line terminating character in your file is (ENTER)).

   If the mainframe does not support handshaking, first try to transfer the file without the handshake mode. If this doesn't work, contact the mainframe's computer site and find out how to send files to that mainframe.
3. Open the file at the host end and ready it for receiving information by whatever command process your host requires.
4. Turn on your file send by pressing (CLEAR)(5) followed by (CLEAR)(:).

   Note that one line of your file is transmitted and then your machine pauses. Once the host sends you the XON, the next line of the file is automatically transmitted.

   If you are operating in half-duplex, you may see the entire file displayed without any pauses. The file is being read from your disk and put in an area of memory where it waits to be transmitted.
5. When the transmission is complete, turn off the handshake mode by pressing (CLEAR)(SHIFT)(*) followed by (CLEAR)(-).

6. Close the file at the host end by whatever command process the host accepts. You may then close your file send by pressing (CLEAR)(5) followed by (CLEAR)(0) (which turns off the *FS and closes the file).

   If you want to force the transmission to resume after a line is ended, you may turn the *CL back on by pressing (CLEAR)(4) followed by (CLEAR)(:).

COMMunicating Between Two TRS-80's

When you use COMM to communicate between two TRS-80's, one end has to run on half-duplex ((CLEAR)(SHIFT) (!) followed by (CLEAR)(:)) and echo ((CLEAR)(SHIFT)(") followed by (CLEAR)(:)). If files are to be sent and received, the RECEIVING end should run half-duplex and echo.

To transfer files between two TRS-80's, use one of the following two methods. Use Method A if you are operating above 300 baud. Use Method B if you are operating at 300 baud.

Method A

1. The sending end presses (CLEAR)(5) followed by (CLEAR)(9) and enters the name of the file to be sent.
2. The receiving end presses (CLEAR)(6) followed by (CLEAR)(9) and enters in the name of the file to be received. Turn the dump-to-disk (DTD) OFF by pressing (CLEAR)(7) followed by (CLEAR)(-). This stores the file in memory as it is received.

   If the sending end supports XON/XOFF handshaking, then you should turn HANDSHAKE ON by pressing (CLEAR)(SHIFT)(*) followed by (CLEAR)(:).
3. When both ends are ready, the receiving end presses (CLEAR)(6) followed by (CLEAR)(:), after which the sending end presses (CLEAR)(5) followed by (CLEAR)(:).

   If your free area of memory decreases to less than 2K during receipt of the file, a warning message is issued and an XOFF is automatically sent to the sending end.

   Transmission from the sender should cease. Once it does, dump the receive area of memory to disk by turning on DTD by pressing (CLEAR)(7) followed by (CLEAR)(:).

   You can observe the increase in available memory space by displaying a menu as the area of memory is written to disk. Once ample space is available, turn off the DTD by pressing (CLEAR)(7) followed by (CLEAR)(-).

   You can then manually restart the sender's file by transmitting an XON from your keyboard with (CONTROL)(Q).

4. The receiving end presses CLEAR 6 followed by CLEAR - when it has received all of the file. The last receive area of memory should be dumped to disk by turning on DTD (CLEAR 7 followed by CLEAR :).

   The sending end presses CLEAR 5 followed by CLEAR - and then CLEAR 5 followed by CLEAR 0.

5. When the receiving end has finished writing the information to the disk, close the file by resetting the *FR (CLEAR 6 followed by a CLEAR 0). This performs an *FR OFF and a DTD OFF, and it closes the file just received.

Method B

1. The sending end presses (CLEAR 5 followed by (CLEAR 9 and enters in the name of the file to be sent.
2. The receiving end presses (CLEAR 6 followed by (CLEAR 9 and enters in the name of the file to be received.

   The dump-to-disk (DTD) must be turned ON by pressing CLEAR 7 followed by CLEAR :. Check to see if it is already ON by displaying a menu (CLEAR 8) and noting if an asterisk is displayed beneath its key.
3. When both ends are ready, the receiving end presses CLEAR 6 followed by CLEAR :. The sending end then presses CLEAR 5 followed by CLEAR : to turn ON the receive and send files.
4. When the receiving end has received all of the file and it is written to the disk, close the file by resetting the *FR. Press CLEAR 6 followed by CLEAR 0. This performs an FR OFF and a DTD OFF, and it closes the file just received. The sending end then presses CLEAR 5 followed by CLEAR 0.

Technical Information

This section describes some of the more technical aspects of COMM operation. This information allows you to predict how COMM will perform during higher speed I/O operations.

Main memory usage

COMM uses all available memory below the top of memory mark (HIGH$) for dynamic buffering of device I/O. You can see or set this value with the MEMORY command.

The amount of buffer space devoted to each logical device dynamically expands and shrinks according to how quickly data is sent to a device and how fast the device can process the data it receives. Each buffer is essentially a variable length First-In, First-Out (FIFO) storage compartment.

The amount of free space available for the buffers is noted in the bottom line of the menu display. When this free space shrinks to less than 2K (2048 characters), a warning message is displayed and an XOFF is automatically sent to the communications line (*CL).

This function is useful when you are receiving a file from a system that supports handshaking. (The (CLEAR)(SHIFT)(*) command describes the supported handshaking.)

Break commands

COMM generates a modem break (long space) when you press the (BREAK) key. A modem break is used on many mainframe systems to indicate you want to abort a function that is occurring at the other computer.

However, for small computers, detecting a modem break is more difficult, so you have to select a control character to be treated as a "break" command.

To transmit a break character to another computer, press (CONTROL)(C) if the other computer is a Model II, 12 or 16. Press (CONTROL)(A) if the other computer is a Model I or III.

Escape code sequences

Some systems transmit control codes to indicate that a cursor movement or action is to be performed. Many systems have adapted a two-character sequence, which does not perform the intended function in COMM.

If you are working with one of these systems, you should contact the operators of the other system and ask if there is a way to prevent these control sequences from being sent to your system.

Some systems support several different types of terminals and computers, so with a little experimenting, you should be able to find a terminal setting that suits your needs.

Receiving large files from another system

If you receive files that won't fit into memory in one piece, you may have to use handshaking to reduce the possibility of losing data.

CONV (CONV/CMD)

Utility

CONV [*partspec*]:***source drive*** [:*destination drive*] [(*parameters*)]

Allows you to move (convert) data files from a TRSDOS 1.3 (Model III) diskette onto a TRSDOS Version 6 formatted diskette.

Use this command with data or BASIC ASCII files. TRSDOS 1.3 application programs will not work on TRSDOS Version 6. To use TRSDOS 1.3 programs on the Model 4, start up your system with a TRSDOS 1.3 system diskette in Drive 0.

The *parameters* are:

VIS moves visible files
INV moves invisible files
SYS moves system files
NEW moves only those files that do not already exist on the destination disk.
OLD moves only those files that already exist on the destination disk.
QUERY=NO specifies that you are not to be questioned before each file is moved to the destination disk.
DIR displays a short directory of a TRSDOS 1.3 disk. If you do not specify *destination drive*, a short directory is displayed. If you specify DIR, no files are moved.

If you don't specify VIS, INV, or SYS, TRSDOS moves all three types of files.

The TRSDOS 1.3 disk must be a non-limited backup disk. Some programs such as SCRIPSIT and VISICALC are limited backup disks.

If you have data files on a TRSDOS 1.3 limited backup disk, you must COPY these files (under TRSDOS 1.3) to a non-limited backup disk before you can CONVert them.

The *source drive* cannot be Drive 0.

When you specify a *partspec*, only those files matching the partspec are moved to the destination disk.

When you do not specify the QUERY=NO parameter, you are questioned before each file is moved. Answer the prompt by pressing:

Y	to copy to file.
N or ENTER	to bypass the file and show the next one.

Caution: Do Not move BASIC/CMD or any other existing TRSDOS system files.

Examples

```
CONV :2 :1 (ENTER)
```

moves all files from Drive 2 onto Drive 1. You are questioned before each file is moved. If the file already exists in Drive 1, you are asked again before it is copied.

```
CONV :1 :Ø (VIS,Q=NO) (ENTER)
```

moves all visible files from Drive 1 onto Drive Ø. You are not questioned before each file is moved.

```
CONV :2 :Ø (NEW) (ENTER)
```

moves only those files from Drive 2 that do not already exist on Drive Ø.

```
CONV $$$DATA:1 :2 (OLD) (ENTER)
```

moves any file whose filename is seven or eight characters long, the 4th through 7th characters are DATA, and that already exists on Drive 2. You are questioned before each file is moved.

```
CONV :1 (DIR) (ENTER)
```

displays the directory of the TRSDOS 1.3 disk in Drive 1.

```
CONV :1 (INV,DIR) (ENTER)
```

displays the invisible files of the TRSDOS 1.3 disk in Drive 1.

Command
COPY *source* [TO] ***destination*** [(*parameters*)]

Copies the *source* to the *destination*.

Source and *destination* can be a filespec or a devspec. *Destination* can also be a drive number.

The parameters are:

LRL=*nnn* specifies the logical record length (1 to 256) for *destination*. If omitted, *destination* will have the same LRL as *source*.
CLONE=NO specifies that *destination* is not to have the attributes of *source*.
ECHO causes any character copied from a devspec to be printed on the screen.
X allows a single drive copy.

The LRL parameter lets you restructure files to make them compatible with other programs. It is also useful when converting a source file from one format to another.

If you wish to append two files with different LRLs, this parameter can be used to make the LRLs match. If LRL is not specified, it defaults to the LRL of *source*.

If CLONE is not specified, the directory entry as well as the contents of *source* copies to *destination*. The owner and user passwords are copied, along with the assigned protection level, the visibility in the directory, the create flag, the last written-to date, and the modified status of the file.

If CLONE=NO is specified, the system date becomes the last written-to date for the destination file. If an existing destination file was copied over, the attributes of the destination file (except for the date) are unchanged. If the COPY command creates a new file, any password included becomes both the user and owner password of the destination file and the file's Mod Flag is set. The destination file is visible, even if the source file was invisible. See the ATTRIB library command for more information on file attributes.

Examples

```
COPY TEST/DAT TO :1 (ENTER)
```

searches the disk drives to find TEST/DAT and copies it to Drive 1.

```
COPY TEST/DAT.PASSWORD:0 TO :1 (ENTER)
```

copies the protected file TEST/DAT.PASSWORD from Drive 0 to

Drive 1. All parts of the destination file, including the password, are the same as those of the source file.

```
    COPY TEST/DAT:Ø TO MYFILE:1 (ENTER)
```

copies TEST/DAT on Drive Ø to MYFILE/DAT on Drive 1. Since the destination filespec does not contain an extension, it defaults to /DAT to match the source.

```
    COPY DATA/NEW:Ø TO /OLD:Ø (ENTER)
```

copies DATA/NEW on Drive Ø to DATA/OLD on Drive Ø. Since the destination filespec does not contain a filename, it defaults to DATA to match the source.

```
    COPY TEST/DAT:Ø TO  TEST/DAT.CLOSED:1
    (CLONE=NO)  (ENTER)
```

copies TEST/DAT from Drive Ø to Drive 1 and assigns the user and owner passwords CLOSED. To assign a password to a destination filespec, the CLONE parameter must be turned off.

```
    COPY DATA/V56:Ø TO  DATA/V28:1  (LRL=128)  (ENTER)
```

copies DATA/V56 on Drive Ø to DATA/V28 on Drive 1. The LRL of DATA/V28 is set to 128.

```
    COPY *KI TO *PR (ECHO) (ENTER)
```

copies from the keyboard to the printer. As keys are pressed, they are sent to the line printer. The keystrokes are visible on the video because the ECHO parameter is specified. Pressing (CONTROL)(SHIFT)(@) or (BREAK) terminates the copy.

When copying from devspec to devspec, it is very important that all devices specified be assigned and active in the system. Any routing or setting affecting the devices may affect the copy.

It is *very* important to be aware that you can generate non-ending loops that lock up the system when copying between devices. Be sure to have a good understanding of this type of copy before you use it.

```
    COPY *KI TO KEYIN/NOW:Ø (ENTER)
```

sends the keystrokes entered from the keyboard to the file KEYIN/NOW on Drive Ø. If the file already exists, it is written over. To view the characters as you type them, use the ECHO parameter. Pressing (CONTROL)(SHIFT)(@) terminates the copy.

```
    COPY TEST/DAT.SECRET:Ø (X) (ENTER)
```

copies TEST/DAT.SECRET from one disk to another.

The destination file TEST/DAT.SECRET is visible, and its owner and user passwords are set to SECRET.

When you use the (X) parameter, a TRSDOS system disk is not required in the copy if the proper system modules (1, 2, 3, and 4) are loaded into memory (see the SYSTEM (SYSRES) library command).

During the copy, the following disk swap prompts are repeated until the copy is complete. The prompts are:

```
    Insert SOURCE disk (ENTER)
```

— Contains the file to be copied.

```
    Insert SYSTEM disk (ENTER)
```

— Any TRSDOS SYSTEM disk. If system modules 1, 2, 3, and 4 are loaded into memory, press (ENTER) at this prompt.

```
    Insert DESTINATION disk (ENTER)
```

— Receives the file being copied. May appear twice in a row. The disk must have a different Disk ID (disk name, master password, or date) from the source diskette. (If it is a system disk, use ATTRIB if you need to change its Disk ID.)

You cannot use the (X) parameter in copies involving logical devices.

Sample Use

Whenever a file is updated, use COPY to make a backup file on another disk. You can also use COPY to restructure a file for faster access. Be sure the destination disk is already less segmented than the source disk; otherwise the new file could be more fragmented than the old one. (See FREE for information on file fragmentation.)

To RENAME a file on the same disk, use RENAME, not COPY.

Advanced Programmer's Command

CREATE *filespec* [(*parameters*)]

Creates a file named *filespec* and pre-allocates space for its future contents.

You can use CREATE to prepare a file which will contain a known amount of data. This usually speeds up file write operations. File reading is also faster, since pre-allocated files are less segmented or dispersed on the disk — requiring less motion of the read/write mechanism to locate the records.

When a file is CREATEd, TRSDOS does not recover unused space at the end of the file (each time you finish using it). If you exceed the created size, TRSDOS allocates extra space for your file as you write to it.

The parameters are:

LRL=*number* assigns *number* as the record length of *filespec*. *number* can be from 1 to 256. If you omit this parameter, the record length defaults to 256.
REC=*number* assigns the specified *number* of fixed-length records to the file.
SIZE=*number* allocates disk space to the file as *number* (in K).

You have to specify (1) SIZE or (2) LRL and REC.

(For more information about record lengths and types, see "Disk Files" in the Technical Reference Manual.)

CREATE also lets you permanently assign additional space to a file that already exists. Use the appropriate parameters for the new file size.

Examples

```
CREATE NEWFILE/DAT:Ø (LRL=128,REC=1ØØ) ENTER
```

creates a file named NEWFILE/DAT on Drive Ø and allocates space for one hundred 128-byte records.

```
CREATE GOOD/DAT (REC=5Ø) ENTER
```

creates a file named GOOD/DAT on the first available drive and allocates space for fifty 256-byte records.

```
CREATE INVENT/DAT (SIZE=2Ø) ENTER
```

takes the already existing file named INVENT/DAT and increases its size to 2ØK.

Error Conditions

If the specified SIZE (or LRL times REC) would cause an existing file to become smaller than it presently is, the error message "File exists larger" appears.

Sample Use

Suppose you are going to store personnel information on no more than 250 employees, and each data record will look like this:

Name (Up to 25 letters)
Social Security Number (11 characters)
Job Description (Up to 92 characters)

Your records would need to be 25 + 11 + 92 = 128 bytes long.

You could create an appropriate file with this command:

```
CREATE PERSONNL/TXT  (REC=250,LRL=128) (ENTER)
```

Once created, this pre-allocated file would allow faster writing than would a dynamically allocated file, since TRSDOS would not have to stop writing periodically to allocate more space (unless you exceed the pre-allocated amount by adding more than 250 employees).

Command

DATE [*mm/dd/yy*]

Resets the date or displays the current system date.

You can use DATE to see today's date. You can also use DATE to change the date that TRSDOS uses.

You first set the date when you start up your computer. DATE lets you change it without turning your computer off and on.

Use *mm/dd/yy* to reset the date. *mm* is a 2-digit month specification; *dd* is a 2-digit day of the month specification; *yy* is a 2-digit year specification. If you omit *mm/dd/yy*, TRSDOS displays the current date.

It is important to set the date because TRSDOS uses the date when creating and accessing files, making backups, and formatting disks.

TRSDOS restricts the date to between 01/01/80 and 12/31/87. If you enter a date outside of this range, an "Illegal date" error results.

Examples

```
DATE (ENTER)
```

displays the current date, such as:

```
Fri, Oct  8, 1982
```

for Friday, October 8, 1982.

```
DATE 10/09/82 (ENTER)
```

resets the date to October 9, 1982 and displays the new date.

Advanced Programmer's Command

DEBUG [([*switch*] [,] [*parameter*])]

The DEBUG command sets up the debug monitor, which allows you to enter, test, and debug machine-language programs.

The *switches* are:

ON	turns on DEBUG
OFF	turns off DEBUG

If *switch* is not specified, ON is assumed.

The *parameter* is:

EXT	specifies the extended debugger

EXT, ON, and OFF can be abbreviated to E, Y, and N.

Once you have turned on DEBUG, you automatically enter the debug monitor whenever you do one of the following:

1. Press the (BREAK) key (provided (BREAK) is enabled)
2. Load and execute a user program (as long as the file's protection is not execute only)

You can also automatically activate the debugger by holding down the (D) key while the system is booting.

While in the DEBUG monitor, you can enter any of a special set of single-key commands to study how your program is working (as detailed under Command Description below).

EXT loads a separate block of the system debugger into high memory. While DEBUG is on, TRSDOS automatically protects this area of memory from being overlaid by BASIC or other user programs.

If you execute a program with execute only protection, DEBUG turns off.

Examples

```
DEBUG (ENTER)
```

turns on the standard DEBUG and waits for it to be activated.

```
DEBUG (EXT) (ENTER)
```

turns on extended DEBUG (loads it into high memory) and waits for it to be activated.

```
DEBUG (OFF) (ENTER)
```

turns off standard or extended DEBUG.

```
DEBUG (OFF,EXT) (ENTER)
```

Turns off DEBUG and attempts to reclaim the high memory occupied by the extended debugger. (If anything is loaded in high memory below the extended debugger, this area is not reclaimed.)

To enter the monitor when DEBUG is on, type:

```
filespec (ENTER)
```

TRSDOS loads the program (as long as the file's protection level is not execute only) and transfers control to DEBUG. (You can also enter the monitor by pressing (BREAK).)

Following is a sample display of the debugger screen.

```
af = Ø1 93 S--H--NC
bc = Ø1 Ø7 => FC 1B 13 Ø6 2D Ø6 23 Ø6  3Ø Ø6 ØE Ø6 28 Ø6 7B 19 .....-.#. Ø...(.{.
de = Ø2 Ø8 => Ø5 68 Ø8 ØØ ØØ ØØ 4B 49  Ø7 6Ø ØB ØØ ØØ ØØ 44 4F .h....KI .@....DO
hl = ØA DD => FF 19 19 2D 2D 2D 2D 2D  2D 2D 2D 2D 2D 2D ØA ØA ...----- ------..
af'= FF FF SZ1H1PNC
bc'= ØØ 1Ø => C9 ØØ ØØ FF ØØ FF ØØ FF  C9 ØØ ØØ 81 FF FF FF 2F ........ ......./
de'= 4A B1 => Ø3 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø  2Ø 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø 2Ø .
hl'= 4D 92 => F7 4E 64 69 72 2F 73 79  73 2C 33 32 ØD FF ØØ FF .Ndir/sy s,32....
ix = Ø2 Ø8 => Ø5 68 Ø8 ØØ ØØ ØØ 4B 49  Ø7 6Ø ØB ØØ ØØ ØØ 44 4F .h....KI .@....DO
iy = ØØ 6A => ØØ ØØ Ø4 Ø2 ØØ ØØ ØØ ØØ  ØØ ØØ Ø5 ØØ 78 ØØ 87 ØØ ........ ....x...
sp = Ø3 EA => 88 Ø8 5Ø Ø6 Ø8 Ø2 2Ø Ø4  4A 2B 4F 4F 16 Ø6 78 Ø5 ..P... . J+OO..x.
pc = Ø8 E1 => 38 Ø7 28 Ø5 79 FE Ø3 28  6A E5 23 7D 23 BE 28 2Ø 8.(.y..( j.#}#.(
        FFØØ => 2D 2D 2D 2D 2D 2D 2D 2D  2D 2D 2D 2D 2D 2D 2D 2D -------- --------
        FF1Ø => 2D 2D 2D 2D 2D 2D 2D 2D  2D 2D 2D 2D 2D 2D 2D 2D -------- --------
        FF2Ø => 2D 2D 2D 2D 2D 2D 2D 2D  2D 2D 2D 2D 2D 2D 2D 2D -------- --------
        FF3Ø => 2D 2D 2D 2D 2D 2D 2D 2D  2D 2D 2D 2D 2D 2D 2D 2D -------- --------
```

The debug display contains information about the Z-8Ø microprocessor registers. The display is set up in the following manner:

The register pairs are shown along the left side of the display, from top to bottom. The current contents of each register pair are shown immediately to the right of the register labels.

The AF and AF' pairs are followed by the current status of the flag registers to the right of the register contents. The other register pairs are followed by the contents of the 16 bytes of memory they are pointing to. The contents are shown in both hexadecimal and ASCII representations. Non-displayable ASCII characters are represented by periods.

The PC register shows the memory address of the next instruction to be executed. The display to the right of that address shows the contents of that address and the next 15 addresses.

The bottom four lines of the screen show the contents of the memory locations indicated by the address at the left of each line. These locations vary depending upon which command is used.

Command Descriptions

When the DEBUG monitor is displayed, you can enter one of the following single-key commands.

Any value entered as an address or a quantity must be entered as a hexadecimal number.

To cancel an incorrect command, press (X).

A (ASCII Modify)

A*address*

Enter the above command line to modify *address*. If *address* is displayed in the monitor, vertical bars appear around it.

After you enter *address*, it and the current byte appear in the lower left corner of the screen. To modify the byte, type the new ASCII value and press:

- (SPACE BAR) to modify the byte and move to the next address.
- (ENTER) to modify the byte and exit from the A command.

If you do not specify *address*, TRSDOS uses the current "memory modification address" (shown by the vertical bars).

Example:

```
AD004 (SPACE BAR)
```

causes address D004 and the current byte value to appear in the lower left corner of the screen. TRSDOS allows you to enter the new byte value. Then press (SPACE BAR) to continue, or (ENTER) to end.

B (Move Block of Memory)

B*starting address,destination address,number of bytes*

Enter the above command line to move a block of memory from *starting address* to *destination address*.

Always specify a non-zero *number of bytes*. If you enter *number of bytes* as 0, TRSDOS moves a block of 65,535 bytes to *destination address*, causing the extended debugger to function improperly.

Example:

```
B3E04,4E34,14E (ENTER)
```

moves the 14E-byte block of memory from 3E04 to 4E34.

C (Call Instruction)

Press C to single-step through the instructions pointed to by the PC register. If a call instruction is encountered, the routine that it calls is executed.

D (Display)

D*address*

Enter the above command line to display memory beginning at *address*.

Example:

```
DE4Ø4 ENTER
```

displays memory beginning at address E4Ø4.

F (Fill Memory)

F*first address,last address,byte*

Enter the above command line to fill the block of memory from *first address* to *last address* with the value *byte*.

Example:

```
F3DØ8,3E14,ØØ ENTER
```

fills the block of memory from 3DØ8 to 3E14 with the value ØØ.

G (Go to an Address and Execute)

G*address,breakpoint1,breakpoint2*

Enter the above command line to begin execution at *address*. If *address* is omitted, execution begins at the PC address.

breakpoint1 and *breakpoint2* are optional breakpoint addresses where execution stops. The breakpoints must be in memory. The system removes them when you return to debug.

Example:

```
GEØFF,FØØ1,F2Ø1 ENTER
```

begins execution at EØFF. Stops execution at breakpoint addresses FØØ1 and F2Ø1.

H (Hex Modify)

H*address*

Enter the above command line to modify *address*. If *address* is displayed in the monitor, vertical bars appear around it.

After you enter *address*, it and the current byte appear in the lower left corner of the screen. To modify the byte, type the new hexadecimal value and press:

- (SPACE BAR) to modify the byte and move to the next address.
- (ENTER) to modify the byte and exit from the H command.
- (X) to exit from the H command without modifying the byte.

If you do not specify *address*, TRSDOS uses the current "memory modification address" (shown by the vertical bars).

Example:

```
HDØØ4 (SPACE BAR)
```

causes address DØØ4 and the current byte value to appear in the lower left corner of the screen. TRSDOS allows you to enter the new byte value. Then press (SPACE BAR), (ENTER), or (X) to continue.

I (Single-Step Execution)

Press (I) to single-step through the instructions pointed to by the PC register. This command is identical to the C command except that any calls encountered are stepped through instruction by instruction. (Note that RST 28H, RST 3ØH, and RST 38H instructions automatically convert the I command to a C command.)

J (Jump)

Press (J) to increment the program counter (PC) by 1.

O (Return to TRSDOS Ready)

Press (O) to return to TRSDOS. DEBUG is not turned off. (Use the DEBUG (OFF) command to turn DEBUG off.)

Q (Port)

There are two kinds of ports — input and output. You read an input port and you write to an output port.

Q*port*

Enter the above command line to read the byte at *port* and display its value. There are 256 input ports (ØØ - FF).

Example:

```
Q45 (ENTER)
```

displays the value of port 45 in the lower left corner of the screen.

Q*port,byte*

Enter the above command line to write the value of *byte* to *port*. There are 256 output ports.

Example:

```
Q45,04
```

writes the byte value of 04 to port 45.

R (Register Pair)

R*register pair code contents*

Enter the above command line to change the specified register pair's contents to the new *contents*. There must be a space between *register pair code* and *contents*.

The *register pair codes* are:

AF	for	AF	AF'	for	AF'
BC	for	BC	BC'	for	BC'
HL	for	HL	HL'	for	HL'
DE	for	DE	DE'	for	DE'
IX	for	IX			
IY	for	IY			
SP	for	SP			

Example:

```
RBC 3D01
```

changes the contents of register pair BC to the value 3D01.

S (Full Screen Mode)

Press (S) to change the monitor format from the register display mode to full screen mode. The full screen mode displays a page of memory (256 bytes) beginning with the current display address (see the D command).

U (Update)

Press (U) to constantly update the display and to show any active background tasks. To cancel this command, hold down any key for several seconds.

X (Return)

Press (X) to return the display to the normal register display mode.

; (Advance Memory)

Press (;) to advance the memory display 64 bytes in the register mode and 256 bytes in the full screen mode.

– (Decrement Memory)

Press (–) to decrement the memory display by 64 bytes in the register mode and 256 bytes in the full screen mode.

Disk Read/Write Utility

Lets you read or write to a specified block of memory. The command line is:

> *disk drive,cylinder,starting sector,operation,address,number of sectors*

address is the starting address in memory where the information read from the disk is to be placed, or where information written to the disk is to be taken from.

Specify operation as: R for Read, W for Write, or * for a Directory Write.

If you do not specify *cylinder*, the system uses the directory track. If you do not specify *starting sector*, the system starts with sector 0. If you do not specify *number of sectors*, the system reads the whole cylinder.

If an error occurs during a disk function, the error number appears on the screen surrounded by asterisks. Hold down the (ENTER) key to abort the disk function.

Example:

```
2,0,0,R,6000,2 (ENTER)
```

reads into memory (beginning at address X'6000') sectors 0 and 1 of cylinder 0 from the disk in Drive 2. This block of memory is displayed on the monitor in the full screen mode.

Extended Command Descriptions

The following commands are available only with the extended debugger.

E (Enter Data)

> E*address*

Enter the above command to enter data directly into memory beginning at *address*. The contents of *address* are displayed and you can then type in two hex characters to replace the current contents. After typing the byte, press:

- (SPACE BAR) to modify the byte and move to the next address.
- (ENTER) to modify the byte and exit from the E command.
- (X) to exit from the E command without modifying the byte.

If you do not specify *address*, TRSDOS uses the current memory modification address (shown by the vertical bars).

L (Locate)

L*address,byte*

Enter the above command to locate the first occurrence of *byte*, starting the search at *address*.

If you don't specify *address*, DEBUG uses the current memory modification address (shown by the vertical bars). If you don't specify *byte*, DEBUG uses the last byte given in a previous L command.

Example:

```
L470E,0D (ENTER)
```

searches for the first occurrence of 0D after address 470E.

N (Next Load Block)

Enter the above command to position the vertical bars to the next load block. This command is used to move logically through a block of memory that has been loaded directly from disk using DEBUG.

To use this command, position the vertical bars over the file type byte at the beginning of any block. Press (N)(ENTER) and DEBUG advances to the next load block header.

Example:

Position the vertical location bars over the beginning byte of a load block and type:

```
N (ENTER)
```

DEBUG advances to the beginning byte of the next load block.

P (Print)

P*first address,last address*

Enter the above command line to print out the block of memory from *first address* to *last address*.

Example:

```
PFC80,FC90 (ENTER)
```

prints out the block of memory from FC80 to FC90 in the following format:

```
aaaa   bb bb . . . bb    cccccccccccccccc
```

aaaa	represents the current address
bb bb . . . bb	represents 16 locations in hex notation
cccc	represents the ASCII equivalents of the 16 hex locations

T (Type ASCII)

T*address*

Enter the above command line to type ASCII characters directly into memory, starting at *address*. If you omit *address*, DEBUG uses the current memory modification address (shown by the vertical bars).

Example:

```
TCBØ1 (SPACE BAR)
```

displays the address CB01 and its current contents in ASCII code. If the contents of the address are out of the ASCII character range, then a period is displayed.

DEBUG then prompts you to enter the new ASCII contents for CB01. Type:

```
A
```

to enter the hex value for A, which is 41, in address CB01.

Pressing (SPACE BAR) advances memory one byte without changing its contents. DEBUG continues to prompt you to add ASCII values until you press (ENTER) to exit the command.

V (Compare)

V*first address,second address,length*

Enter the above command line to compare a block of memory beginning at *first address* to the block of memory beginning at *second address*. The compare is for the specified *length* in bytes (X'0001' – X'FFFF').

Example

```
VCBØØ,EFØ2,45 (ENTER)
```

compares a 45-byte long block of memory beginning at CB00 to a 45-byte long block of memory beginning at EF02. The first byte of the block of memory beginning at CB00 that does not match is displayed as the first byte of memory in the DEBUG monitor. The corresponding byte in the block of memory beginning at EF02 becomes the current memory modification address that is used by the H, A, E, and T commands.

W (Word)

W*address,word*

Enter the above command to search memory for *word*, beginning at *address. word* must be in the least significant byte, most significant byte format.

If you do not specify *address*, DEBUG uses the current memory modification address. If you do not specify *word*, DEBUG uses the last word given in a previous W command.

Example:

```
    WABØ6,3412 (ENTER)
```

searches memory for word (1234) beginning at address ABØ6. The address where *word* is found is displayed in the DEBUG monitor with the vertical location bars positioned one byte before it.

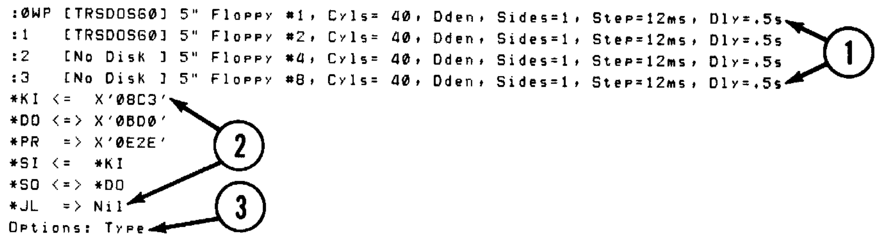
DEVICE [(*parameters*)]	**Advanced Programmer's Command**

Displays the status of the drives, the options selected, and the data paths for the logical devices that have been set, routed, or linked.

It also logs in disks in the available disk drives.

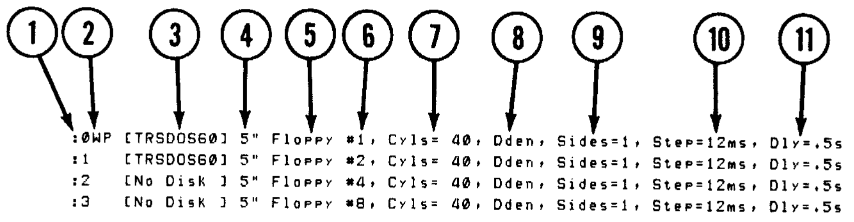
The parameters are:

D=NO	suppresses the drive portion of the display. Any new drives or disks are not detected.
B=YES	enables the logical device portion of the display.
S=NO	suppresses the options status portion of the display.
P=YES	duplicates the display to the printer.



```
:0WP [TRSDOS60] 5" Floppy #1, Cyls= 40, Dden, Sides=1, Step=12ms, Dly=.5s
:1   [TRSDOS60] 5" Floppy #2, Cyls= 40, Dden, Sides=1, Step=12ms, Dly=.5s
:2   [No Disk ] 5" Floppy #4, Cyls= 40, Dden, Sides=1, Step=12ms, Dly=.5s
:3   [No Disk ] 5" Floppy #8, Cyls= 40, Dden, Sides=1, Step=12ms, Dly=.5s
*KI <=  X'08C3'
*DO <=> X'0BD0'
*PR  => X'0E2E'
*SI <=  *KI
*SO <=> *DO
*JL  => Nil
Options: Type
```

1. The DRIVE section shows the current configuration of the disk drives.
2. The DEVICE section shows the devices (displayed when B=YES).
3. The STATUS section displays the status of some user selected options.



```
:0WP [TRSDOS60] 5" Floppy #1, Cyls= 40, Dden, Sides=1, Step=12ms, Dly=.5s
:1   [TRSDOS60] 5" Floppy #2, Cyls= 40, Dden, Sides=1, Step=12ms, Dly=.5s
:2   [No Disk ] 5" Floppy #4, Cyls= 40, Dden, Sides=1, Step=12ms, Dly=.5s
:3   [No Disk ] 5" Floppy #8, Cyls= 40, Dden, Sides=1, Step=12ms, Dly=.5s
```

1. Logical drive number — The number of the drive accessed. (See the SYSTEM command.)
2. Disk write protect status — The write protect status assigned to the drive by the SYSTEM (DRIVE=,WP=) command. A disk can also be write protected by placing a foil tab over the write-protect notch on the diskette.
   WP=Write Protected
3. Disk name — The name of the disk accessed in the drive.

4. Disk size — The size of the floppy or hard disk.
5. Type of drive — Either floppy or hard.
6. For floppy disk systems, the physical location of the drive.
   1 - lower
   2 - upper
   4 - middle of the disk expansion cable
   8 - end of cable
7. Number of cylinders — The number of cylinders available in the drive.
8. Density — The data density of the disk accessed in the drive.
   Dden = Double density
   Sden = Single density
9. Number of sides — The number of sides the disk being accessed has.
   1 = One side
   2 = Two sides
10. Step rate — The step rate of the drive in milliseconds. Step rate is the speed at which the disk drive head is moved from cylinder to cylinder.
11. Delay time — The delay time when accessing a 5″ floppy disk. Delay time is the amount of time the system waits after starting the drive motor before it attempts to access the disk.

NOTE: The Step rate and Delay time are preset for this system. Information needed to change these settings can be found in the *Technical Reference Manual* (Cat. No. 26-211Ø).

```
*KI  <=  X'Ø893'
*DO  <=>X'ØB8A'
*PR   =>*LØ: *TD & => X'ØDE8'
*SI  <=  Nil
*SO  <=>Nil
*JL   =>Nil
*FF  <#> [Inactive] X'ØFD4'
*TD  <=>PRINT/TXT: Ø
*LØ   =>X"ØDE8'
```

In the logical device portion of the device table:

>=	indicates an input device
=>	indicates an output device
<=>	indicates a device capable of input and output
#	indicates a filter
|	indicates a link

If you add a driver or filter to the default system, the DEVICE command shows the address where a device transfers control to its driver or filter. If more than one filter or driver is associated with the same device, the first driver's address is displayed. It also shows the interaction between devices and/or files.

```
Options: Type
System modules resident: 1, 2, 4,
```

The options line shows you the active system parameters. These parameters are usually established with the FILTER, LINK, ROUTE, SET, SPOOL, and SYSTEM library commands.

It also shows the resident system overlays. (See the SYSTEM (SYSRES=) library command.)

Examples

```
DEVICE (ENTER)
```

displays the device table.

```
DEVICE (D=NO) (ENTER)
```

displays the TRSDOS options, and turns the drive portion of the display off. Any new drives or disks are not detected.

```
DEVICE (B=YES) (ENTER)
```

enables the device portion of the device table, and displays the entire table.

```
DEVICE (S=NO) (ENTER)
```

displays the drive table, and turns the options status portion of the display off.

```
DEVICE (P) (ENTER)
```

displays the device table, and sends it to the printer.

Command

DIR [*partspec*] [:*drive*] [(*parameters*)]

Displays the specified disk's directory.

You can use DIR to see the files on a disk.

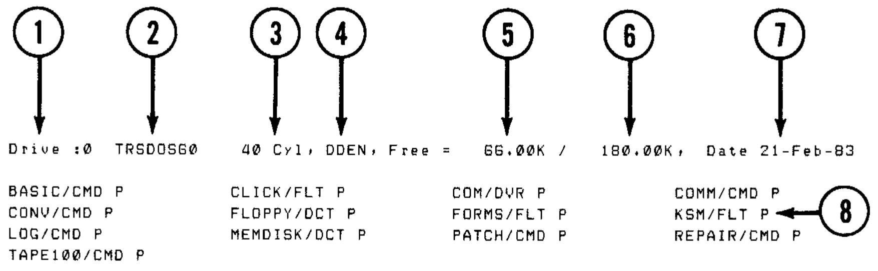
When you specify a *partspec*, only those files matching the partspec are displayed.

If you omit the drive number, TRSDOS displays the directories of all enabled drives.

The parameters are:

ALL displays all directory information for the specified files, including space used on the disk.
INV displays the non-system, invisible files along with the non-system visible files.
MOD displays the files modified since the last backup.
NON enables the non-stop display mode.
PRT outputs the directory display to the printer.
SYS displays the system files along with the visible files.
DATE displays the files with today's date.
DATE="*M1/D1/Y1-M2/D2/Y2*" displays the files with modify dates between the two specified dates, inclusive.
="*M1/D1/Y1*" displays the files with modify dates equal to the specified date.
="*-M1/D1/Y1*" displays the files with modify dates before or equal to the specified date.
="*M1/D1/Y1-*" displays the files with modify dates after or equal to the specified date.
SORT=NO specifies that the directory entries are not to be sorted alphabetically.

SORT can be abbreviated to O.

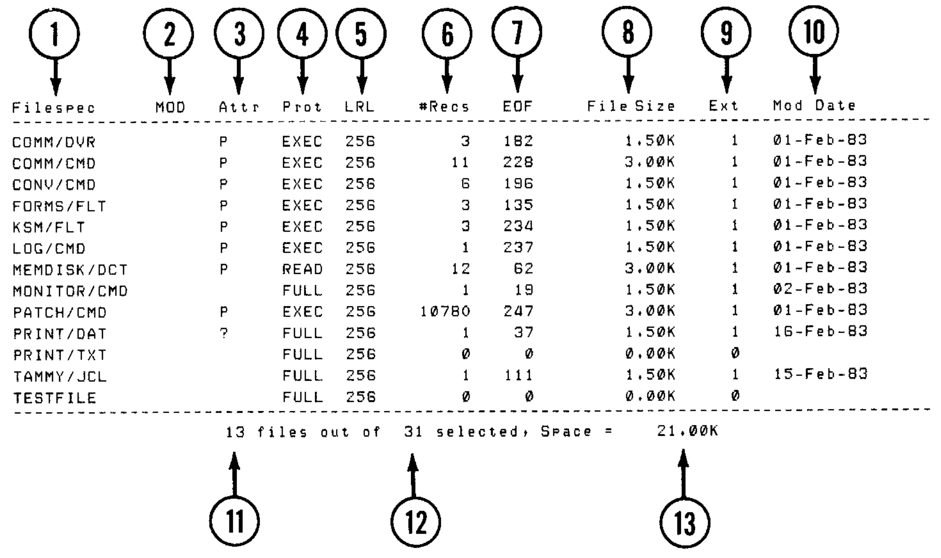


```
Drive :0  TRSDOS60   40 Cyl, DDEN, Free =   66.00K /   180.00K,  Date 21-Feb-83

BASIC/CMD P         CLICK/FLT P          COM/DVR P            COMM/CMD P
CONV/CMD P          FLOPPY/DCT P         FORMS/FLT P          KSM/FLT P
LOG/CMD P           MEMDISK/DCT P        PATCH/CMD P          REPAIR/CMD P
TAPE100/CMD P
```

1. Drive number.
2. Disk name.

3. Number of cylinders on the disk.
4. Density of the disk.
   DDEN = double density
   SDEN = single density
5. Free space — The amount of space on the disk currently available to the user.
6. Total free space — The total amount of space (used and unused) on the disk.
7. Creation date — The date on which the disk was created.
8. Disk files — The disk files are sorted alphabetically.



```
Filespec        MOD   Attr  Prot  LRL     #Recs    EOF      File Size   Ext   Mod Date
-----------------------------------------------------------------------------------------
COMM/DVR              P     EXEC  256         3    182          1.50K     1   01-Feb-83
COMM/CMD              P     EXEC  256        11    228          3.00K     1   01-Feb-83
CONV/CMD              P     EXEC  256         6    196          1.50K     1   01-Feb-83
FORMS/FLT             P     EXEC  256         3    135          1.50K     1   01-Feb-83
KSM/FLT               P     EXEC  256         3    234          1.50K     1   01-Feb-83
LOG/CMD               P     EXEC  256         1    237          1.50K     1   01-Feb-83
MEMDISK/DCT           P     READ  256        12     62          3.00K     1   01-Feb-83
MONITOR/CMD                 FULL  256         1     19          1.50K     1   02-Feb-83
PATCH/CMD             P     EXEC  256     10780    247          3.00K     1   01-Feb-83
PRINT/DAT             ?     FULL  256         1     37          1.50K     1   16-Feb-83
PRINT/TXT                   FULL  256         0      0          0.00K     0
TAMMY/JCL                   FULL  256         1    111          1.50K     1   15-Feb-83
TESTFILE                    FULL  256         0      0          0.00K     0
-----------------------------------------------------------------------------------------
                13 files out of  31 selected, Space =     21.00K
```

1. Filespec — The name and extension assigned to a file when it is created.

2. Modification Status — The modification status of the file.
   + indicates the file was modified since it was last backed up.

3. Attribute — The file's attributes.

I	indicates an invisible file.
P	indicates the file has an owner password.
S	indicates a system file.
C	indicates the file was created with the CREATE library command.
?	indicates an open file (see RESET library command)

4. Protection Level — The level of access assigned to the user password. See the ATTRIB library command for a list of these levels.

5. Logical Record Length (LRL) — The length of each logical record in the file.

6. Number of Records — The number of logical records in the file.

7. End of File (EOF) — Shows the last byte number of the file.
8. File Size — The amount of space in K (1K=1024 bytes) that the file takes up on the disk.
9. Extents — The number of extents (blocks of space) that the file is stored in. The higher the number, the more fragmented the file is on the disk.
10. Modify Date — The date that the file was created or last written to. Since the system uses the date you enter at start-up, we recommend that you do not disable the start-up DATE prompt.
11. Specified Files — The number of files on the disk that match the parameters you specify in the command line.
12. Total Files — The total number of files on the disk.
13. Space — The amount of space used by the specified files.

Examples

```
DIR (ENTER)
```

displays the visible files of all enabled drives.

```
DIR :1 (I,S) (ENTER)
```

displays all files (visible, invisible, and system) on Drive 1. Specifying a drive that is not enabled or ready causes an "Illegal drive number" error message or a [No Disk] message to appear.

```
DIR :1 (I,S,P) (ENTER)
```

sends the directory display of Drive 1 to the printer as well as the screen. The NON parameter is automatically set to YES and the entire directory prints without pause. Pressing the (SHIFT)(@) keys pauses the display. Press any key to continue.

```
DIR :2 (A) (ENTER)
```

displays the directory of Drive 2 in the allocation format. The display pauses after every 24 lines. Pressing (BREAK) terminates the display, while pressing any other key displays the next 24 lines.

```
DIR (DATE="10/01/82-") (ENTER)
```

displays the files modified on or after October 1, 1982.

```
DIR (M,SORT=NO) (ENTER)
```

displays the files modified since the last backup in an unsorted order.

```
DIR B:0 (ENTER)
```

displays the files that begin with the letter B on Drive 0.

```
DIR T$$T/CMD:1 (ENTER)
```

displays any file that matches partspec T$$T/CMD on Drive 1.

```
    DIR -/CMD:0 (ENTER)
```

displays all visible files on Drive 0 except for those that have a CMD extension.

Command

DO [*control character*] ***filespec*** [(*parameters*)] [;]

Compiles and executes a DO file.

You can use DO to run a file of commands each time TRSDOS starts up.

A DO file is a user created Job Control Language (JCL) file that contains one or more library commands. TRSDOS executes the commands as if you had typed them in from the keyboard.

In addition to executing TRSDOS commands, you can load and execute user programs from a DO file.

You can create a DO file with the BUILD command. Command lines in this file can include library commands or filespecs. See the Job Control Language section for more information on DO files.

The *control characters* are:

$ compiles your DO file without actually executing the commands.
= executes your DO file without compiling it.
* reruns the last DO command that was compiled.

When you specify a control character, you must leave a space between DO and the character or TRSDOS ignores the character.

The *parameters* are:

@label lets you create JCL files with multiple entry points (an entry point is the place where processing begins). A label consists of the @ symbol followed by one to eight alphanumeric characters.
parm[=*value*] lets you pass *value* to *filespec* during execution.

When you specify the *@label* parameter, *filespec* does not execute until the label is reached. Execution continues until it reaches the next label or the end of the JCL file.

The *@label* parameter, by building many different functions into one file, reduces the number of individual files on the disk (conserving space in the directory).

Use the semicolon (;) parameter when you need to specify a command line longer than 79 characters.

When a DO command line exceeds 79 characters:

1. Enclose as many parameters as will fit on one line in parentheses. Close the parentheses, insert a (;), and press ENTER.

2. When a question mark appears on the screen, enter the remaining parameters (enclosed in parentheses).

Examples

```
DO DRIVE/JCL (ENTER)
```

compiles and executes the file named DRIVE/JCL.

```
DO = DRIVE/JCL (ENTER)
```

executes the file named DRIVE/JCL without compiling it.

```
DO $ DRIVE (ENTER)
```

compiles the file named DRIVE/JCL without executing it. Since you did not specify an extension to DRIVE, it defaulted to JCL. You can LIST the SYSTEM/JCL file to see if the JCL compiled properly.

```
DO MY/JCL (@THIRD) (ENTER)
```

compiles and executes the program named MY/JCL. All instructions in the program are ignored up to the label (@THIRD). Compilation begins at the line following the label and continues until the next label or the end of the file is reached.

```
DO * (ENTER)
```

executes SYSTEM/JCL, which contains the last DO file that was compiled.

```
DO TEST/NEW:2 (D=5,E=6) (ENTER)
```

compiles and executes the file TEST/NEW on Drive 2. The variable parameters D=5 and E=6 are passed as needed during compilation.

Error Conditions

If you specify *@label*, you must compile the DO file, or an error is generated.

The occurrence of any error aborts the DO execution.

When you specify the * *control character*, a "File not in directory" error occurs if there is no previously compiled DO file to rerun.

When you specify the $ *control character*, the system compiles the JCL file and informs you of any errors that occur. This lets you see if the file compiles properly before you actually execute it. When you compile a JCL file, a disk in your system must be write-enabled, so the system can write the compiled information to a file named SYSTEM/JCL.

When you use the = character, you cannot use some of the JCL features. See the JCL section of this manual.

Sample Uses

Suppose you want to set up the following TRSDOS functions to execute by typing one command:

```
FORMS (MARGIN=8)
TIME (CLOCK=ON)
```

Use BUILD to create such a file. If you called the file BEGIN, then use the command:

```
DO BEGIN (ENTER)
```

to perform the commands.

Advanced Programmer's Command
DUMP *filespec* (*parameters*)

Copies an area of memory to a disk file named *filespec*.

You can use DUMP to store a machine-language program from memory to a file.

DUMP can produce a program or a core ASCII file. A program produced with DUMP can then be loaded or executed at any time. (An ASCII file cannot be loaded with the LOAD command or executed with the RUN command.)

The default extension for program dumps is /LMF, and the default extension for ASCII dumps is /TXT.

You can use some or all of the following *parameters*:

START = *address* starts the dump at *address*. You must include this parameter. The address must be above 2FFF hexadecimal or 12287 decimal.
END = *address* ends the dump at *address*. You must include this parameter. END must be greater than or equal to START, and can be either a hexadecimal or decimal number.
TRA = *address* sets the address at which your program begins executing after you load it. If you omit this parameter, any subsequent run of the file will only load the program and return you to TRSDOS Ready. TRA can be either a hexadecimal or a decimal number.
ASCII specifies that the dump is to an ASCII file. ASCII files contain program code only. No system loading information is written to *filespec*.
ETX = *value* specifies that the character at the end of an ASCII file is equal to *value*. *value* is a hexadecimal number in the format x'nn'. When you specify ETX, you must also specify ASCII.

ETX cannot be abbreviated.

When you DUMP to an ASCII file, you create a file that has the identical file structure as a SCRIPSIT file. The system writes a special character at the end of the file which can be changed with the ETX parameter.

Examples

```
DUMP ROUTINE/CMD  (START=X'7000',END=X'8000',TRA=
X'7000') (ENTER)
```

dumps the area of memory starting at hexadecimal 7000 and ending at hexadecimal 8000. This block of memory is written to a disk file named ROUTINE/CMD. If the file already exists, it is overwritten. If it does not exist, it is created on the first available drive. The transfer address (starting address for execution) of ROUTINE/CMD is hexadecimal 7ØØØ.

```
DUMP ROUTINE/CMD  (START=28672,END=32768,TRA=
28672) (ENTER)
```

is identical to the above command except that START, END, and TRA have decimal values.

```
DUMP TEST:1  (S=X'9ØØØ',E=X'BCØF') (ENTER)
```

dumps the specified block of memory to a disk file named TEST/LMF on Drive 1. Since you did not specify a file extension to TEST, it defaulted to /LMF. Also, since you did not specify a transfer address, it is written to the file as a return to TRSDOS Ready.

```
DUMP WORD/IMG:Ø  (S=X'7ØØØ',E=X'AØØØ', ASCII)
(ENTER)
```

dumps the specified block of memory to a disk file named WORD/IMG on Drive Ø. Since the ASCII parameter is specified, an ASCII file is created.

```
DUMP WORD (S=X'7Ø5Ø',E=X'AØØØ', ETX=X'FF',
ASCII)  (ENTER)
```

dumps the specified block of memory to a disk file named WORD/TXT. An ASCII file is created, and the special character at the end of the text (end of text marker) is written as hexadecimal FF. Since you did not specify an extension for WORD, it defaulted to /TXT.

Advanced Programmer's Command

FILTER *devspec* [USING] ***phantom devspec***

Connects a filter program to *devspec* which modifies or "filters" data as it is read from or written to *devspec.*

You can use a filter to change data as it passes from *devspec* to *phantom devspec* (and vice versa).

A filter is a program that controls the flow of data to or from any device or file.

devspec is any valid, active TRSDOS device. *phantom devspec* is the name of a device which is connected to the filter program established in memory with the SET command.

You can apply as many filter programs to *devspec* as you want to. If there is not any more space in memory for the filter connection, the error message "No device space available" appears.

*See the SET command for more information on FILTER.

Example

Suppose you create a filter program named CONVERT/FLT that converts a linefeed character to a "null", and you establish it in memory with the SET library command to a device, *LF.

```
    FILTER *PR USING *LF (ENTER)
```

filters I/O directed to the line printer through the CONVERT/FLT program. As a result of this filter program, all linefeed characters in output directed to the printer are converted to the null character (ASCII 0).

Sample Use

You can use a filter to control a printer working with non-standard size paper (see Appendix I and the FORMS library command).

FORMAT

Utility

FORMAT [:***drive*** [(***parameters***)]]

Prepares a blank or old disk for use by defining the tracks and sectors and writing system information onto it. (For more information, see "Diskette Organization" in the *Technical Reference Manual* (Cat. No. 26-2110).)

You can use FORMAT to organize a disk so you can store information on it.

drive specifies the drive in which the blank or old disk is to be formatted. If you omit the *drive*, TRSDOS prompts you for it.

The *parameters* are:

ABS overwrites any existing data without prompting. The ABS parameter is used primarily when you execute a FORMAT from a JCL file. See the JCL section for more information.
NAME="*disk name*" assigns a name to the disk being formatted.
MPW="*password*" assigns the master password to the disk. The master password allows limited access to all user files.
SDEN specifies the density of the disk as single.
DDEN specifies the density of the disk as double.
CYL=*number* specifies the number of cylinders (tracks) for the disk. *number* can be 35 to 96.
QUERY=NO turns off the prompts for density, number of cylinders, name, and password.

QUERY is the only parameter that can be abbreviated.

When to Format

To prepare a new disk. Before you can use a new disk, you must format it. After formatting, record the disk name, date of creation, and password. Store this information in a safe place. It helps you estimate how long a diskette has been in use. And, if you forget the master password, it ensures continued access.

To erase all data from a disk. To "start over" with a disk, you can reformat it. This erases all old information and locks out all flawed sectors which have developed. It puts the system information back on the disk and leaves the "good" sectors available for information storage.

The Format Prompts

If the destination drive is not ready, the following message is displayed:

```
Load destination diskette <ENTER>
```

Insert the destination diskette and press (ENTER) to continue, or press (BREAK) to return to TRSDOS Ready.

If you specify the drive number, disk name, or master password in the command line, you are not prompted for them.

If you specify either the DEN or CYL parameters, the system uses the default values for the other parameters.

If you enter a FORMAT command without specifying any parameters, you are prompted for them in the following order:

```
Which drive is to be used?
```

Enter the number of the drive you are formatting in.

```
Diskette name?
```

Enter any name with up to eight alphanumeric characters. The first character must be a letter. Press (ENTER) and the disk name defaults to DATADISK.

```
Master password?
```

Enter any password with up to eight alphanumeric characters. The first character must be a letter. Pressing (ENTER) causes the master password to default to PASSWORD.

The remaining prompts concern the type of diskette you are using. Press (ENTER) in answer to each of them if you are using standard Radio Shack diskettes.

```
Single or Double density <S,D>?
```

Enter (S) for single density or (D) for double density. Press (ENTER) and the value defaults to double density.

```
Number of cylinders?
```

Enter any number from 35 to 40 on TRS-80 hardware. Pressing (ENTER) causes the system to default to the value set with the SYSTEM (CYL=) command. If this value is not set, the default is 40 cylinders.

If you are formatting in Drive 0, the following prompt appears after you answer the cylinders question:

```
Load destination disk (ENTER)
```

It is important that you do not remove the system disk and insert the disk to be formatted until this prompt appears. Several seconds after you swap disks, you are prompted to put the system disk back in Drive 0 with the message:

```
Load SYSTEM disk (ENTER)
```

The format is now complete.

Examples

```
FORMAT (ENTER)
```

prompts you for the drive number, the diskette name, the master password, the density, and number of cylinders, and checks to see if the destination disk is already formatted.

```
FORMAT :1 (NAME="DATA3",MPW="SECRET") (ENTER)
```

prompts you for the DEN and CYL parameters, and checks to see if the disk in Drive 1 is already formatted. The disk in Drive 1 is assigned the name DATA3 and the master password SECRET.

```
FORMAT :0  (NAME="FILES", MPW="FILE01", Q=N)
(ENTER)
```

displays the message:

```
Load destination disk (ENTER)
```

When you insert the destination disk in Drive 0, the system checks to see if the disk is already formatted. When the message:

```
Load SYSTEM disk (ENTER)
```

appears, insert the system disk in Drive 0 and the format is completed.

```
FORMAT :1 (QUERY=NO,ABS) (ENTER)
```

formats the disk in Drive 1 (with the default options) even if the disk already contains data. Because you specified the ABS parameter, you don't have the opportunity to abort the FORMAT (if the disk is already formatted and its master password is PASSWORD).

Error Conditions

After you enter a format command and before the actual formatting begins, the system checks the destination diskette to see if it is already formatted.

If the disk is formatted and its MPW is PASSWORD, the following message appears:

```
Disk contains data -- NAME=disk name
  DATE=mm/dd/yy
Are you sure you want to format it?
```

Press (N) to abort the FORMAT or (Y) to continue. If you specified the ABS parameter in the command line, you see the DISK CONTAINS DATA message, but you are not prompted to abort the format.

If the disk is formatted and the master password of the destination disk is not PASSWORD, the following message appears:

```
Disk contains data -- NAME=disk name
  DATE=mm/dd/yy
Enter its Master Password or <BREAK> to abort:
```

Press (BREAK) to abort the format or enter the master password to continue.

If the disk contains an incomplete or non-standard format, one of the following messages may appear in place of NAME=*disk name*:

```
Unreadable directory
Non-standard format
Non-initialized directory
```

When the format begins, you see the cylinder numbers appear as the necessary information is written to them. After all cylinders are written, FORMAT verifies that the proper information is actually on each cylinder.

If the verify procedure detects an error, an asterisk and the cylinder number are shown on the screen. That cylinder is locked out, so that no files can be written to the defective area. Use the FREE library command to see the locked out cylinders on a diskette.

During parts of the format operations, the system real time clock is turned off.

If you are formatting in Drive 0, the following prompt appears after you answer the cylinders question:

FORMS [(***parameters***)] **Command**

Sets up forms filter (*FF) parameters.

You can use FORMS to print a form larger or smaller than a standard-size page.

Before you can use FORMS, you have to SET *FF to its filter program FORMS/FLT and FILTER the printer to *FF. See Appendix I.

The *parameters* are:

DEFAULT returns all parameters to their start-up values.
ADDLF issues a linefeed after every carriage return.
CHARS=*number* sets the number of characters per printed line.
    number is 1 - 255.
FFHARD issues a form feed (Top of Form) character instead of a
    series of linefeeds.
INDENT=*number* sets the number of spaces a line is to be
    indented if the line length exceeds CHARS. The default
    value for *number* is 0.
LINES=*number* sets the number of lines to be printed per page.
    The default value for *number* is 66.
MARGIN=*number* sets the left margin.
PAGE=*number* sets the physical page size as *number* of lines.
    The default value for *number* is 66.
QUERY prompts you for each parameter.
TAB specifies that tab characters are to be translated into the
    appropriate number of spaces.
XLATE=*X'aabb'* specifies a one-character translation to be
    performed by the filter.
    aa is the character in hex format to be translated.
    bb is the character in hex format *aa* is translated to.

To determine the parameters to set for:

page size	multiply form length in inches by the number of lines per inch.
lines per page	determine the number of blank lines at the bottom of every page. The default number of blank lines is 0. If LINES = PAGE, then text can be written on every line of each page. LINES cannot exceed PAGE.
characters per line	multiply form width in inches by the number of characters per inch (10 or 12). Use CHARS to set the maximum number of printable characters per line. If a line is greater than CHARS, then TRSDOS

automatically breaks the line at the maximum length, and continues printing at the next line. The line is indented if you have specified INDENT.

Examples

Be sure that you have SET *FF to its filter program FORMS/FLT and you have FILTERed the printer to *FF with the commands:

```
SET *FF TO FORMS/FLT (ENTER)
FILTER *PR *FF (ENTER)
```

```
FORMS (ENTER)
```

displays the current parameter values.

```
FORMS (CHARS=80,INDENT=6,PAGE=51,
LINES=45,FFHARD) (ENTER)
```

allows a maximum of 80 characters per printed line. If a line contains more than 80 characters, the excess is printed on the next line and indented 6 spaces. The physical page size is set to 51 lines, and 45 lines can be printed on a page.

FFHARD causes the printer to advance 6 linefeeds continuously rather than individually when the line count reaches 45. Be aware that certain printers cannot respond to FFHARD.

```
FORMS (MARGIN=10,CHARS=80,INDENT=16) (ENTER)
```

causes all lines to start 10 spaces in from the normal left-hand starting position. Any line longer than 80 characters is indented 16 spaces (6 spaces after the margin) when wrapped around, so it is printed starting at position 16.

```
FORMS (TAB,ADDLF) (ENTER)
```

specifies that tab characters are to be translated into the appropriate number of spaces. Also, a linefeed is sent to the printer every time a carriage return is sent.

```
FORMS (XLATE=X'2A2E') (ENTER)
```

translates all hexadecimal 2A characters (asterisks) to hexadecimal 2E characters (periods).

Sample Uses

Suppose you have a payroll program that contains all of your employees' payroll information, and that prints checks of the size 4" × 7".

To instruct your computer to print a form 4" × 7", issue the following commands:

```
SET *FF TO FORMS/FLT (ENTER)
FILTER *PR *FF (ENTER)
FORMS (CHARS=55,LINES=20,PAGE=24) (ENTER)
```

Now when you run your payroll program, you can print the checks on the proper size form.

Command

FREE [:*drive*] [(*parameter*)]

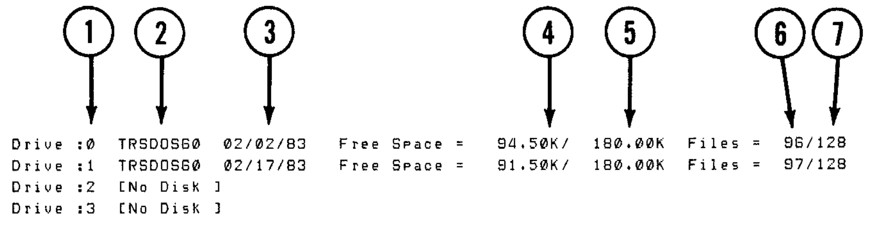
Lists the amount of space that is free (available for use) and the number of files on each drive, if no *drive* is specified. If *drive* is specified, displays a free-space map of the disk in that drive.

You can use FREE to see how many files are on a disk. You can also use FREE to see a table containing information about each disk in your computer.

The parameter is:

PRT sends output to the printer.

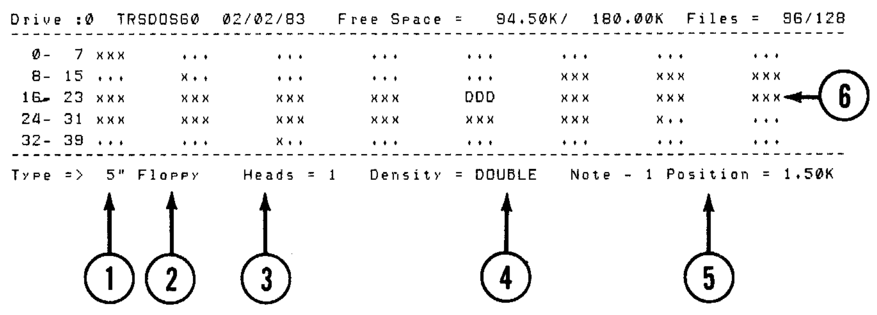
FREE displays free-space information about each enabled disk in the following format:



```
Drive :Ø  TRSDOS6Ø  Ø2/Ø2/83  Free Space =  94.5ØK/  18Ø.ØØK  Files =  96/128
Drive :1  TRSDOS6Ø  Ø2/17/83  Free Space =  91.5ØK/  18Ø.ØØK  Files =  97/128
Drive :2  [No Disk ]
Drive :3  [No Disk ]
```

1. Drive number — The number of the drive whose free space map is displayed.
2. Disk name — The name of the disk.
3. Date — Creation date.
4. Free space in K — The amount of space (in K) that is available for use. 1K = 1Ø24 bytes.
5. Total space in K — The total amount of space (in K) on the disk.
6. Number of files — The number of directory entries you have available. Each file uses one or more directory slots.
7. Total number of files — The total number of directory slots available on the disk.

FREE displays a free space map of the specified disk in the format shown below.



```
Drive :Ø  TRSDOS6Ø  Ø2/Ø2/83   Free Space =   94.5ØK/  18Ø.ØØK  Files =  96/128
-----------------------------------------------------------------------------------
  Ø-  7 xxx       ...       ...       ...       ...       ...       ...       ...
  8- 15 ...       x..       ...       ...       ...       xxx       xxx       xxx
 16- 23 xxx       xxx       xxx       xxx       DDD       xxx       xxx       xxx
 24- 31 xxx       xxx       xxx       xxx       xxx       xxx       x..       ...
 32- 39 ...       ...       x..       ...       ...       ...       ...       ...
-----------------------------------------------------------------------------------
Type =>  5" Floppy    Heads = 1   Density = DOUBLE   Note - 1 Position = 1.5ØK
```

1. Disk size — The size of the floppy or hard disk.
2. Type of drive — Either floppy or hard.
3. Number of heads — The number of surfaces on the disk that contain data for this logical drive.
4. Density — The density of the disk (SINGLE or DOUBLE).
5. Note — 1 Position = 1.5Ø K. — The minimum allocation of disk space for the disk type (hard or floppy). This value will change depending on the type of drive.
6. Detailed space allocation map — The organization of data on the disk.

   The numbers on the left represent the cylinders on the disk, and these cylinders are divided into granules (grans). The grans for the specified cylinders run across each line.

   Each gran is represented by one of the following characters:

.	Unused — The gran is available for use.
*	Locked out — The gran is flawed and not available for use.
X	Used — The gran is currently used for system or user files.
D	Directory — The gran is used for the system's directory files.

Examples

```
FREE (ENTER)
```

displays free space information about each enabled disk.

```
FREE :Ø (PRT) (ENTER)
```

displays a free space map of the disk in Drive Ø. The map is also sent to the printer because you specified the PRT parameter.

LIB	Command

Displays a listing of all system commands in Libraries <A>, <B>, and <C>.

You can use LIB to see a list of TRSDOS commands.

Library <A> contains the primary TRSDOS commands, Library <B> contains the secondary commands, and Library <C> contains the machine-dependent commands.

Example

```
LIB (ENTER)
```

displays a list of the TRSDOS library commands.

Library <A>

Append	Copy	Device	Dir	Do	Filter	Lib
Link	List	Load	Memory	Remove	Rename	Reset
Route	Run	Set				

Library <B>

Attrib	Auto	Build	Create	Date	Debug	Dump
Free	Purge	Time	Verify			

Library <C>

Forms	Setcom	Setki	Spool	Sysgen	System

Technical Information

Library <A> is located in the SYS6/SYS system module, Library <B> is located in the SYS7/SYS system module, and Library <C> is located in the SYS8/SYS system module. You can remove any of the three system modules if you will not be using their commands. (Use the PURGE or REMOVE library commands to delete system modules.)

Advanced Programmer's Command
LINK *devspec1* [TO] ***devspec2***

Links together two logical devices; both must be enabled in the system.

You can use LINK to get a printout of the data displayed on your video display. You can also use LINK to write data displayed on the screen to a disk file.

To "unlink" the devices, use the RESET command.

Be careful if you make several links to the same device. You could create an endless loop and hang up the system.

Examples

```
LINK *DO *PR (ENTER)
```

links the video display (*DO) to the line printer (*PR). All output sent to the display (*devspec1*) is also sent to the line printer (*devspec2*).

NOTE: Although all output to the video display is also sent to the printer, any output sent individually to the printer (such as an LPRINT from BASIC) is *not* sent to the video display. This is because the order of the devices in the link command line is important. Once linked, any information sent to *devspec1* is also sent to *devspec2*, and any information requested from *devspec1* can also be supplied by *devspec2*. However, information sent to *devspec2* is not sent to *devspec1*, nor can information requested from *devspec2* be supplied by *devspec1*.

```
LINK *PR *DO (ENTER)
```

links the line printer to the video display. All output sent to the printer (*devspec1*) is also sent to the video display (*devspec2*).

Linking a Device To a File

It is not possible to directly LINK a device to a file. To link a device to a file, follow this procedure:

- Use the ROUTE library command to create a "phantom" device and route it to the file.
- Link the device to the phantom device using the LINK library command.

NOTE: Do not use the SYSGEN library command if you currently have a device linked to a file. The linked file is shown as open every time you power up or reset the system. You can overwrite other files very easily if you switch disks with the linked file open.

The following example shows how to link your line printer to the disk file PRINT/TXT on Drive 0 using a phantom device.

```
ROUTE *DU TO PRINT/TXT:0 (ENTER)
```

creates the phantom device *DU and routes it to the disk file PRINT/TXT on Drive 0. If PRINT/TXT does not exist, it is created. If it already exists, data sent to the file is appended onto its end.

```
LINK *PR *DU (ENTER)
```

links the printer to *DU, which in turn is routed to PRINT/TXT. All output sent to the line printer is also sent to *DU (that is, written to PRINT/TXT).

NOTE: PRINT/TXT remains open until you issue a RESET *DU command. To break the link between the printer and PRINT/TXT without closing the file, use the RESET *PR command. See the ROUTE and RESET library commands; also see the "Using the Device-Related Commands" section.

Sample Use

Suppose you want to use your computer to communicate with another computer. First, set the Communications Line device (*CL) and use SETCOM to specify WORD=8 and PARITY=NO with the commands:

```
SET *CL TO COM/DVR (ENTER)
SETCOM (WORD=8,PARITY=NO) (ENTER)
```

then issue commands:

```
LINK *DO *CL
LINK *KI *CL
```

to link the video display and keyboard to the RS-232C interface. This lets your Model 4 act as a "host" and be accessed by a remote terminal via the RS-232C hardware.

LIST *filespec* [(*parameters*)]	**Command**

Lists the contents of *filespec*.

You can use LIST to see the contents of a file on a disk.

The parameters are:

ASCII8 displays the graphic characters and special characters in a file, along with the text.
NUM numbers the lines in ASCII text files.
HEX specifies hexadecimal output format. When you specify the HEX parameter, NUM and LINE are ignored.
TAB=*number* specifies that tab stops are to be placed every *number* of spaces apart for ASCII text files. Each tab character (hex Ø9) encountered causes a jump to the next tab stop. The default value for *number* is 8.
PRT directs output to the printer.
LINE=*number* sets the starting line to *number*. If you omit the LINE= parameter, TRSDOS uses 1. This parameter works only with ASCII files.
REC=*number* sets the starting record number to *number*. If you omit the REC= parameter, TRSDOS uses Ø. The REC= parameter is used only with the HEX parameter.
LRL=*number* sets the logical record length to be used to display a file with a record length of *number*. If you omit the LRL= parameter, TRSDOS uses the logical record length of the file. The LRL= parameter is used only with the HEX parameter.

LINE cannot be abbreviated, and the abbreviation for ASCII8 is A8.

When you enter a LIST command, LIST first searches for *filename*/TXT. If it cannot find *filename*/TXT, it searches for *filename*.

Press (SHIFT)(@) to pause a list. Press (ENTER) to continue. Press (BREAK) to abort the list.

When you use the HEX parameter, *filespec* is listed in the following format:

```
1    2              3                                    4
ØØØØ:ØØ =  41 42 43 44 45 46 47 48  49 4A 4B 4C 4D 4E 4F 5Ø  ABCDEFGH IJKLMNOP
ØØØØ:1Ø =  51 52 53 54 55 56 57 58  59 5A ØD                 QRSTUVWX YZ.
```

1. Current logical record of the file in hex notation, starting with record Ø.

2. Offset from the first byte of the current logical record (in hex notation).
3. Hex representation of the byte listed.
4. ASCII representation of the byte. A period is used for all non-displayable bytes.

Examples

```
    LIST TESTFILE:0 (ENTER)
```

searches Drive 0 for TESTFILE/TXT. If not found, it searches for TESTFILE.

```
    LIST MONITOR/CMD (HEX,LRL=8)
```

lists MONITOR/CMD in the hexadecimal mode using an LRL of 8.

```
    LIST REPLY/TXT (NUM,TAB=10,P) (ENTER)
```

prints a listing of REPLY/TXT on the printer, numbering each line that is printed. Lines are numbered beginning with 00001. Any tab character encountered causes a jump to the next tab position (every 10th column).

```
    LIST TESTFILE/OBJ (HEX,REC=5) (ENTER)
```

list TESTFILE/OBJ in the hex mode, beginning with record 5.

Sample Use

Suppose you used BUILD to create a file named HEXFILE/TXT which contains hexadecimal characters that are not available from the keyboard. Issue the command:

```
    LIST HEXFILE/TXT (ENTER)
```

and the special characters are displayed on the screen.

Advanced Programmer's Command

LOAD [(***parameter***)] ***filespec***

Loads a machine-language program file (without executing it) and then returns to TRSDOS Ready.

You can use LOAD to pre-load assembly language routines that programs written in a language such as BASIC can call.

The parameter is:

X	loads a file from a non-system disk.

The file must be in load module format. Do not use it to load BASIC program files. The default file extension for the LOAD command is /CMD.

Programs to be loaded must reside at or above the address X'3000'.

Examples

```
LOAD STATUS/CMD ENTER
```

loads the file STATUS/CMD into memory.

```
LOAD (X) PROGRAM/CIM ENTER
```

loads PROGRAM/CIM from a non-system disk. The system prompts you to insert the disk with the desired file on it with the message:

```
Insert SOURCE disk ENTER
```

After the file is loaded, you are prompted to put the system disk back in Drive 0 with the message:

```
Insert SYSTEM disk ENTER
```

The load is now complete.

Sample Use

Often several program modules must be loaded into memory for use by a master program. For example, suppose PAYROLL/PT1 and PAYROLL/PT2 are modules, and MENU/CMD is the master program. Then you could use the commands:

```
LOAD PAYROLL/PT1 ENTER
LOAD PAYROLL/PT2 ENTER
```

to get modules into memory, and then type: MENU to load and execute MENU.

LOG/CMD

Utility

LOG/CMD is used in Radio Shack hard disk installations. See your hard disk manual for an explanation on how to use this utility.

Advanced Programmer's Command

MEMORY [(*parameters*)]

allows you to reserve a portion of memory, display or change the current HIGH$ and LOW$, modify a memory address, or begin executing at a specified memory location. HIGH$ has to be higher that LOW$.

You can use MEMORY to find out which area of memory you can use.

The *parameters* are:

CLEAR=*value* fills memory from hex 2600 to HIGH$ with *value*. *value* in the format X'nn'. If you do not specify *value*, memory is filled with the hexadecimal value 00 (null).
HIGH=*address* sets *address* as HIGH$. *address* must be less than the current HIGH$.
LOW=*address* sets *address* as LOW$. If you specify LOW, the current LOW$ is displayed.

Note: Subsequent MEMORY commands (and any other system level commands) will reset LOW$ to its default value.
ADD=*address* displays the word at *address* and specifies the address of WORD and BYTE. If you specify ADD, the current contents of the address are displayed and WORD and BYTE are not necessary.
WORD=*word* changes the contents of ADD and ADD + 1 to *word*.
BYTE=*byte* changes the contents of ADD to *byte*.
GO=*address* transfers control to *address*. If more than one parameter is specified, the GO parameter is always executed last.

address is any memory address in hexadecimal or decimal notation. *word* is any value in the range 0000 - FFFF hexadecimal or 0 - 65535 decimal. *byte* is any value in the range 00 - FF hexadecimal or 0 - 255 decimal.

Examples

```
MEMORY (ENTER)
```

displays HIGH$ (the highest unused memory location) and LOW$ (the lowest reserved memory location) in the hexadecimal X'nnnn' format.

```
MEMORY (HIGH=X'E100') (ENTER)
```

sets HIGH$ to hexadecimal memory address E100, as long as the existing HIGH$ is above X'E100'. The MEMORY command moves HIGH$ lower in memory.

```
MEMORY (ADD=X'65ØØ') (ENTER)
```

displays the contents of hexadecimal memory addresses 65ØØ and 65Ø1 in the following format:

```
X'65ØØ' = 25856 (X'6745')
High = X'E1ØØ' Low = X'2FFF'
```

1 The address specified in hexadecimal notation.
2 The decimal equivalent of the address.
3 The contents of address and address + 1, in MSB-LSB format.
4 The current HIGH$ address.
5 The current LOW$ address.

```
MEMORY (ADD=X'E1ØØ',WORD=X'3EØA') (ENTER)
```

modifies hexadecimal memory locations ADD (E1ØØ) and ADD+1 (E1Ø1), changing them to the value of WORD. The following display appears:

```
X'E1ØØ' = 576ØØ (X'ØØØØ' => X'3EØA')
High = X'E1ØØ' Low = X'2FFF'
```

1 The address specified in hexadecimal notation.
2 The decimal equivalent of the address.
3 The new contents of address and address + 1, in MSB-LSB format.
4 The current HIGH$ address.
5 6The current LOW$ address.

```
MEMORY (ADD=X'E1ØØ',BYTE=X'C9') (ENTER)
```

changes the BYTE of memory at hexadecimal address E1ØØ to hexadecimal C9. The display after executing this command is:

```
X'E1ØØ' = 576ØØ (X'ØØ' => X'C9')
High = X'E1ØØ' Low = X'2FFF'
```

The display is identical to the last example, except that the modified BYTE is shown instead of the modified WORD.

```
MEMORY (GO"X'E1ØØ') (ENTER)
```

transfers control to hexadecimal memory address E1ØØ.

Error Conditions

If you set LOW$ equal to or higher than HIGH$, you get the error message "Range error."

Lets you make minor corrections in any disk file by (1) typing in the patch code directly from the command line (Method A), or (2) creating an ASCII file containing patch information (Method B).

You can use PATCH to make minor corrections in any disk file.

You can use PATCH to make minor changes in your own machine-language programs. You need not change the source code, reassemble it, and recreate the file. You can use PATCH to make minor replacement changes in data files, also.

Method A

Advanced Programmer's Utility
PATCH *filespec* (*patch commands*)

Makes minor corrections in *filespec* by letting you type in the patch code directly from the command line. You can use only the *address* patch command for this type of patch.

Method B

Advanced Programmer's Utility
PATCH *filespec1* USING ***filespec2*** [**(*parameters*)**]

Makes minor changes (which are contained in *filespec2*) to *filespec1*.

filespec1 is the file to be changed and /CMD is its default extension. *filespec2* contains the patch commands. *filespec2* can contain only ASCII characters and /FIX is its default extension.

The *patch commands* are:

address = *value* identifies the PATCH as a patch by "memory load location." It changes the contents of memory beginning with *address* to *value*.

D*record,byte* = *value* identifies the PATCH as a "direct modify patch." *record* tells which record contains the data to be changed. It is a hexadecimal number from 00 to FF. *byte* specifies the position of the first byte to be changed. It is a hexadecimal number from 00 to FF.

F*record,byte* = *value* lets you make sure that a patch is applied to the correct place in memory, when used in conjunction with the D *patch command*. F*record,byte* follows D*record,byte*. If the location specified with the D *patch command* does not contain the data specified with F*record,byte* , then the PATCH aborts. F*record,byte* is also used with the REMOVE parameter to remove a patch and replace it with the original data.

L*code* identifies the PATCH as a "library mode patch." The PATCH applies to either the SYS6/SYS, SYS7/SYS, or

SYS8/SYS library command module. *code* is the binary coded location in the format *nn* where the change begins.

address is a four-digit hexadecimal value in the format X'*nnnn*' which is the memory load address for the change.

value can be either a series of hexadecimal bytes in the format *nn nn nn...*, or a string of ASCII characters in the format "*string*."

The *parameters* are:

YANK removes the PATCH specified by *filespec2* from *filespec1*. The specified PATCH contains code in the *address* format.
REMOVE removes the PATCH specified by *filespec2* from *filespec1*. The specified PATCH contains code in the D*record,byte* format.

An **address** *patch command* changes a file by "memory load location." It adds the patch code to the end of the *filespec* and then makes the changes beginning at *address* each time the file is loaded. You can use YANK to remove the added code from *filespec*. This type of patch can be applied only to files that can be loaded with the LOAD or RUN library command.

A **Drecord,byte:Frecord,byte** *patch command* changes a file by "direct modify patch." It changes a file by directly applying the patch code to the specified record and byte of *filespec*. When you BUILD a file containing *patch commands* in this format, you can REMOVE the patch.

An easy way for you to find the record and byte of *filespec* that you want to patch is to list *filespec* using the LIST library command with the HEX parameter. Remember that the first record in a file is record 0, not record 1.

Lcode *patch command* patches are supplied by Radio Shack for you to implement changes to TRSDOS. You have to BUILD a patch file to apply this type of PATCH.

You can specify more than one line of patch code from the command line by placing a colon (:) between the lines of patch code.

Examples

These examples are used to show the syntax and development of the PATCH command, so do not execute them.

```
PATCH MONITOR/CMD (X'E100'=C3 66 00 CD 03 40)
ENTER
```

patches the file MONITOR/CMD by the memory load location method. The six bytes beginning at hexadecimal E100 are changed. During the PATCH operation, the following message is displayed:

```
Installing patch
```

When the operation is completed, this message appears:

```
Patch function completed
x patch lines installed
```

x is the number of lines of patch code that were installed.

Since there is no filespec used for the patch code, the name CLP (Command Line Patch) is assigned to the patch code. You can use this name to later YANK the patch from MONITOR/CMD.

Error Conditions

It is important that you do not patch a file more than once using the same name for *filespec2* because you cannot YANK the second patch from the file.

We recommend that you name all patch files in a systematic manner and that you use the extension /FIX for all of the files.

Using BUILD To Create a PATCH File

You can use the BUILD library command to create a PATCH file. (See the "Building a File" section of the BUILD library command.) A PATCH file can contain only ASCII characters.

Each line in a patch file is either a *patch command* or a comment. Comment lines begin with a period and are ignored by the patch utility. Use comments in patch files to document the changes that you make. You can append a comment onto the end of a *patch command* by using a semicolon to separate the two parts.

You can also use SCRIPSIT to create a PATCH file. When you create a PATCH file with SCRIPSIT, use the S,A type of save. SCRIPSIT sometimes leaves extra spaces after the last carriage return in a file. To remove the extra spaces, position the cursor just after the last carriage return in the file and do a delete to end of text.

Examples

These examples are used to show the syntax and development of the PATCH command, so do not execute them.

```
    PATCH BACKUP/CMD:0 USING SPECIAL/FIX (ENTER)
```

The data in BACKUP/CMD on Drive 0 is changed to 23 3E 87 beginning at hexadecimal 6178. The data beginning at hexadecimal 61A0 is changed to FF 00 00. This is an example of a memory load location patch, and since the patch is added onto the end of BACKUP/CMD, you can use the YANK parameter to remove it.

Use the BUILD library command to create the following PATCH file named TEST/FIX:

```
    .This patch modifies the SYS2 module.
    D0B,49=EF CD 44 65:F0B,49=DD 3A 33 44
    D0B,55=C3 00 00:F0B,55=EF 44 55
    .End of patch
```

Now, type in the command line:

```
    PATCH SYS2/SYS.PASSWORD USING TEXT/FIX (ENTER)
```

changes the data specified in SYS2/SYS.PASSWORD to the data in TEST/FIX. Since the data beginning at record 0B, byte 49 and 55 is directly changed on disk, this is an example of a direct disk modify patch. The Find *patch commands* let you make sure you are patching the correct place in memory.

Using PATCH on a TRSDOS System File

When Radio Shack releases a modification to TRSDOS, you receive a printout of the exact *patch commands* that you must use to make the change.

Suppose Radio Shack sends you the patch information for a file named LIB1 that contains the following patch code:

```
.USE LIB1 TO PATCH SYS6/SYS.
L54
X'5208'=32 20 DE AF 00 C3 66 00
```

Use the BUILD library command to create the file LIB1, and type in the lines exactly as they appear on the printout. After you end the file, type in the command line:

```
PATCH SYS6/SYS:1 USING LIB1 (ENTER)
```

This changes the data specified in SYS6/SYS on Drive 1 to the data in LIB1/FIX. Since you did not specify an extension to LIB1, it defaulted to /FIX. This patch is in the memory load location mode. Library patches can also be done with the direct disk modify mode. To be sure that you do not patch the disk in Drive 0, specify the drive number in the filespec (such as SYS6/SYS:2) or write protect the disk in Drive 0.

PATCH lets you implement any changes to TRSDOS that may be supplied by Radio Shack. This way, you do not have to wait for a new release of TRSDOS.

To make a Radio Shack change, follow these general steps:

1. Make a backup copy of the diskette to be patched.
2. Insert the TRSDOS diskette to be changed into one of the drives. (Make sure the diskette is "write-enabled.")
3. In the TRSDOS mode, use the BUILD library command to create a PATCH file containing the patch commands specified in the information provided by Radio Shack.
4. Issue the appropriate PATCH command.
5. After the patch is complete, test the patched diskette in Drive 0 to see that it is operating as a TRSDOS system diskette. You have to reset the computer before you can test the diskette.

PURGE

	Command
PURGE [*partspec*] ***:drive*** [(***parameters***)]	

Quickly deletes files from the disk in *drive*.

You can use PURGE to remove all or some of the files on a disk with one command.

The parameters are:

QUERY = NO automatically removes files without prompting for each one.
MPW = "*password*" states the disk master password
INV removes the invisible files as well as the visible files.
SYS removes the system files as well as the visible files.
DATE = "*M1/D1/Y1-M2/D2/Y2*" removes the files with modify dates between the two specified dates, inclusive.
= "*M1/D1/Y1*" removes the files with modify dates equal to the specified date.
= "*-M1/D1/Y1*" removes the files with modify dates before or equal to the specified date.
= "*M1/D1/Y1-*" removes the files with modify dates after or equal to the specified date.

Once you enter the PURGE command, TRSDOS prompts you for the disk's password (if it is not PASSWORD), unless you specified it with the MPW parameter.

Then, the system displays the files one at a time. It prompts you to remove the file or keep it. Respond with (Y) to remove the file or (N) to keep it. Pressing (ENTER) skips the file.

NOTE: BOOT/SYS and DIR/SYS cannot be purged and do not appear during execution of any PURGE command.

Examples

```
PURGE :0 (MPW="SECRET") (ENTER)
```

purges all visible files on Drive 0 if the master password of the disk is SECRET. If SECRET is not the master password, the purge aborts with an error message. The purge shows each file and waits until (Y), (N), or (ENTER) is pressed before displaying the next file.

```
PURGE :1 (QUERY=NO,INV,SYS) (ENTER)
```

purges ALL files from Drive 1. You are not questioned before each file is removed, as QUERY is specified as NO. Be careful with this command because it produces a blank formatted disk.

```
PURGE /BAS:1 (Q=NO) (ENTER)
```

purges all visible files with the extension /BAS. You are not questioned before each file is removed, as QUERY is specified as NO.

```
PURGE /$$S:2 ENTER
```

purges all visible files on Drive 2 whose file extension contains 3 characters and ends in the letter S.

```
PURGE -/CMD:Ø (INV) ENTER
```

purges all non-system files from Drive Ø except those with the extension /CMD.

```
PURGE :1 (DATE="Ø2/Ø1/81-") ENTER
```

purges all visible files on Drive 1 with modify dates of February 1, 1981 or later. You are questioned before each file is removed.

Sample Use

Refer to the Sample Use example in the BACKUP command section. Now that you have moved all of the new files to the disk in Drive 1, you can remove all the new files on the disk in Drive Ø by issuing the command:

```
PURGE /NEW:Ø ENTER
```

Now you have two separate disks: one with new employee files on it and one with old employee files on it.

REMOVE

REMOVE ***filespec*** **[*filespec* ...]** — **Command**

Deletes *filespec* from the directory and frees the space allocated to it.

REMOVE ***devspec*** **[*devspec* ...]** — **Command**

Removes *devspec* from the device table.

You can use REMOVE to remove a file that you don't need to use anymore. You can also REMOVE a device that is no longer needed.

Examples

```
REMOVE ALPHA/DAT:Ø  BREAKER/DAT:Ø  (ENTER)
```

deletes ALPHA/DAT and BREAKER/DAT from the directory on Drive Ø and frees all space allocated to them.

```
REMOVE MIDWEST/DAT.SECRET (ENTER)
```

deletes MIDWEST/DAT. If the file is protected at a level of RENAME or higher, the owner password must be used to remove the file. If you supply the user password, the error message "Illegal access attempted to protected file" is displayed. If you supply the wrong password, the error message "File access denied" is displayed.

```
REMOVE *LU (ENTER)
```

removes the user-created device *LU from the device table.

NOTE: A device can be removed only if it is pointed NIL in the device table. If a device is not pointed to NIL, it must first be reset with the RESET library command before it can be removed.

TRSDOS does not permit the removing of the system devices: *JL, *KI, *DO, *SI, *SO, and *PR. Attempting to remove these devices produces the error message "Protected system device."

Sample Use

Suppose you have a file of temporary employees that you hire for inventory. All of the temporary employees' files are in a file named EMPLOYE/TEM. When you complete inventory, you can remove this file with the command:

```
REMOVE EMPLOYE/TEM (ENTER)
```

RENAME

Command

RENAME ***filespec1*** [TO] ***filespec2***
RENAME ***devspec1*** [TO] ***devspec2***

Changes a file's name and/or extension from *filespec1* to *filespec2*. It also changes a device name from *devspec1* to *devspec2*.

You can use RENAME to change the name of a file or a device.

RENAME does not change the file's password, contents, or position on the disk. (See the ATTRIB command to change the password.) If *filespec1* is password protected, the password must be specified or an error message will result.

RENAME does not change a device's routing, filtering, linking, or setting. *devspec1* must be an existing device, and *devspec2* must be an unused device name.

You cannot RENAME the system devices: *KI, *DO, *PR, *SI, *SO, and *JL.

Examples

```
RENAME TEST/DAT:0 TO OLD/DAT (ENTER)
```

renames TEST/DAT on Drive 0 to OLD/DAT.

```
RENAME TEST/DAT:0 TO REAL (ENTER)
```

renames TEST/DAT on Drive 0 to REAL/DAT. Since you did not specify an extension for *filespec2*, it defaulted to the extension on *filespec1* (DAT).

```
RENAME TEST/DAT:0 TO REAL/ (ENTER)
```

renames TEST/DAT on Drive 0 to REAL (without an extension).

```
RENAME DATA/NEW.SECRET:1 TO /OLD (ENTER)
```

renames the password protected DATA/NEW.SECRET on Drive 1 to DATA/OLD.SECRET. Since you did not specify a filename for *filespec2*, it defaulted to that of *filespec1*. RENAME does not change or delete passwords, so the password defaulted also.

```
RENAME *UD TO *TX (ENTER)
```

renames the device *UD to *TX.

Error Conditions:

Suppose you want to rename TESTER/BAS to TESTER (without an extension). Typing:

```
RENAME TESTER/BAS TO TESTER (ENTER)
```

produces the error message "Destination spec required" because you did not add a "/" to TESTER.

Suppose you want to rename TEST/JCL to TEST/ABS, but the file TEST/ABS already exists. Typing:

```
    RENAME TEST/JCL TO TEST/ABS (ENTER)
```

produces the error message "Duplicate file name" because TEST/ABS already exists.

REPAIR (REPAIR/CMD)

REPAIR :*drive*	**Utility**

Updates and modifies information on floppy disks produced on other Radio Shack operating systems and computers so TRSDOS Version 6 can use them.

drive is any floppy drive currently enabled in the system (except Drive 0).

After REPAIR is complete, you should be able to copy any file off the modified diskette.

Use REPAIR to read any disk formatted by any disk operating system other than TRSDOS Version 6, LDOS 5.1.3, or their successors.

Once a disk is modified by TRSDOS, the operating system that created it may not be able to read it.

TRSDOS 1.2 and 1.3 disks should NEVER be repaired. Use the CONV utility to copy programs from them.

Using REPAIR, you can convert the following list of operating system diskettes to TRSDOS Version 6 diskettes:

- Model I TRSDOS 2.0, 2.1, 2.2, 2.3, 2.3a

Examples

```
REPAIR :1 (ENTER)
```

updates information on the diskette in Drive 1 so that TRSDOS can use it.

Advanced Programmer's Command

RESET *devspec*
RESET *filespec*

Returns a *devspec* to its original start-up condition. Closes an open *filespec*.

You can use RESET to set a system device to its normal condition in the DEVICE table. You can also use RESET to close a file that has not been properly closed.

Resetting a Device

A RESET *devspec* command removes any filtering, linking, or routing that has been set to the device. Any open disk file that is connected to the device is closed.

If *devspec* is a device you created (see the LINK and ROUTE library commands), it is pointed at NIL when reset.

If *devspec* is a system device (*KI, *DO, *PR, *SI, *SO, and *JL), it returns to its start-up condition when reset.

To see that *devspec* has been pointed at NIL or returned to its start-up condition, issue a DEVICE library command and examine the device table that is displayed.

You can use the REMOVE library command to remove the device from the device table once it points at NIL.

Example

Suppose you have used the FORMS command to specify printer parameters, or you have filtered, linked, or routed *PR.

```
RESET *PR (ENTER)
```

returns *PR to its start-up condition and disconnects the printer filter.

Resetting a Filespec

You can RESET an open *filespec* that has not been properly closed.

Improperly closed files result when (1) your system loses power and files are left open, (2) you remove a disk from a drive and files are left open, or (3) you reset your system while files are open, or (4) a command aborts while files are open.

To see if any files on a disk are not properly closed, issue a DIR library command. Any file that appears with a question mark (?) after it needs to be RESET before you can access it. Receiving the error "File already open" may also indicate a file is not properly closed.

Example

Suppose that your system lost power and there is a file named PRINTER/DAT that is not properly closed.

```
    RESET PRINTER/DAT (ENTER)
```

closes the file named PRINTER/DAT and lets you access it.

Advanced Programmer's Command
ROUTE *devspec1* [TO] ***devspec2***
ROUTE *devspec1* [TO] ***filespec*** [(REWIND)]
ROUTE *devspec1* (NIL)

Routes devspec1 to one of the following:

- another device (*devspec2*)
- a disk file (*filespec*)
- nothing (NIL)

You can use ROUTE to create a device. You can also use ROUTE to alter the flow of data from one device to another.

If *devspec1* does not already exist, ROUTE creates it.

To see how your devices are routed, use the DEVICE command. To return the non-system devices to their normal start-up state, use the RESET and REMOVE commands.

Examples

```
ROUTE *PR *DO
```

routes the printer (*PR) to the video display (*DO). All data normally sent to the printer will be displayed on the screen.

```
ROUTE *PR TO PRINTER/DAT
```

routes the printer (*PR) to a disk file (PRINTER/DAT). All data normally sent to the printer will be stored in a disk file named PRINTER/DAT. If PRINTER/DAT already exists, the data is appended to the end of the file.

```
RESET *PR
```

closes the PRINTER/DAT file and any subsequent output to *PR goes to the printer. The PRINTER/DAT remains open until you execute the above command.

```
ROUTE *PR TO PRINTER/DAT (REWIND)
```

routes the printer to PRINTER/DAT. If PRINTER/DAT already exists, the system "rewinds" the file to the beginning. The contents of the file will be replaced with the new printer data.

```
ROUTE *PR (NIL)
```

routes *PR to NIL. TRSDOS ignores all output to the printer.

```
ROUTE *DU TO TEST/TXT:1
```

routes a user device (*DU) to a disk file named TEST/TXT in Drive 1.

Error Condition

If you ROUTE *CL to a file and you are receiving data from a communications line (*CL), you might lose data if it is coming in at a high speed. If so, use CREATE to preallocate file space before using ROUTE. This makes data loss less likely because the system no longer has to spend the time allocating more space.

Sample Use

Suppose you want to route a report-producing program to a file (instead of printing the report). Issue the command:

```
ROUTE *PR TO REPORT/DAT (ENTER)
```

Now you can run the program and the report that it produces is routed to the file REPORT/DAT. This means that you can print the report in the file REPORT/DAT whenever you want by using the LIST command.

NOTE: Remember, you must reset *PR before listing the file so that it will be properly closed.

RUN

[**RUN**] [**(X)**] ***filespec*** [***command text***]	**Command**

Loads a program named *filespec* into memory and executes it.

You can use RUN to execute a program.

Typing RUN is optional. You can load and execute a program from the TRSDOS Ready prompt by simply typing in the name of the program (without the RUN).

The default extension for *filespec* is CMD.

X executes a program from a non-system disk for the single drive user.

Command text is an optional value which the program you specified may require.

When running a program, observe the following address restrictions:

RUN	*filespec* must load above X'2FFF'.
RUN (X)	*filespec* must load above X'2FFF'.

Examples

```
RUN CONTROL/CMD (ENTER)
```

loads a program named CONTROL/CMD and executes it.

```
CONTROL/CMD (ENTER)
```

loads a program named CONTROL/CMD and executes it.

```
RUN PROG (ENTER)
```

loads a program named PROG/CMD and executes it. Since you did not supply a file extension, it defaulted to /CMD.

```
RUN (X) TRADERS/CMD:0 (ENTER)
```

loads TRADERS/CMD from a non-system disk. First you are prompted with the message:

```
Insert SOURCE disk (ENTER)
```

Insert the disk containing the program into Drive 0 and press (ENTER). After the program is loaded into memory, you are prompted with:

```
Insert SYSTEM disk (ENTER)
```

Insert the system disk back into Drive 0 and press (ENTER); program execution begins.

Sets a driver or filter program to a device. This command is for advanced programming applications.

You can use SET to tell TRSDOS the name of a filter or driver program that you are going to use.

A driver program channels data to or from a device. If it is outputting to a device, it converts data to the device's format. If it is inputting from a device, it converts the data to the computer's format.

A filter program filters data before it is sent out or after it is received.

Once the device is SET, it remains SET until it is RESET. You cannot SET an active device.

Setting a Driver Program

Advanced Programmer's Command

SET *devspec* [TO] ***driver filespec*** [***parameters***]

Loads *driver filespec* into memory and sets it to *devspec*. The driver filespec can be one of your own driver programs.

Once you set a system device to a driver, the device is controlled by your own driver program, rather than the TRSDOS driver program.

The parameters are values sent to the driver filespec. These parameters are totally independent of the SET command. They are determined only by the needs of your driver program.

Example

Within TRSDOS is a driver program that sends printer output to the parallel port. Suppose you write a driver program named SERIAL/DVR that sends printer output to the serial port.

```
SET *SP TO SERIAL/DVR (ENTER)
```

loads SERIAL/DVR into memory and sets it to the device *SP.

```
ROUTE *PR TO *SP (ENTER)
```

routes data going to the printer to the device *SP. Now any input to the printer goes to the SERIAL/DVR program. The SERIAL/DVR program, in turn, sends the output to the serial port.

By using a LINK command instead of the ROUTE command, data sent to *PR is sent to the TRSDOS parallel printer driver. It is also sent (via the LINK) to the serial driver and then to the serial printer.

Advanced Programmer's Command
SET *phantom devspec* [TO] ***filter filespec*** [USING]
[***parameters***]

loads the driver or filter specified by *filter filespec* into memory and connects it to *phantom devspec* (a non-existing device). You must then use the FILTER command to connect the phantom device to a real device.

The filter filespec can be the KSM/FLT program (listed in Appendix I) or one of your own filter programs.

Once you have (1) set a phantom device to a filter, and then (2) filtered a device to the phantom device, the device is connected to your filter program. All data input and output to the device is filtered through your program.

The parameters are values sent to the filter program. These parameters are totally independent of the SET command. They are determined only by the needs of your filter program.

Example

Suppose you write a filter program named TRAP/FLT to change some characters sent to *DO, the video display.

First, you need to load your TRAP/FLT program and set it to a phantom device (this example uses *LC as the name of the phantom device):

```
SET *LC TO TRAP/FLT
```

This causes *LC to point to TRAP/FLT.

Then, you need to use *LC (which points to the TRAP/FLT program) to filter the data output to the video display:

```
FILTER *DO *LC (ENTER)
```

Now, all data output to the video display is filtered through your filter program.

```
SET *DU KSM/FLT USING FILEDAT/KSM (ENTER)
FILTER *KI *DU (ENTER)
```

loads the Keystroke Multiply filter into memory and sets it to the keyboard.

Advanced Programmer's Command

SETCOM [(*parameters*)]

Adjusts the parameter values of the RS-232C driver program COM/DVR. Before you can use SETCOM, you have to install the driver using the SET command (see Appendix I).

You can use SETCOM to adjust your computer so that it can communicate with another computer or a piece of computer equipment.

The RS-232C port lets you communicate with:

- another computer
- a modem
- a serial printer

The *parameters* configure the RS-232C port, and establish line conditions.

The *parameters* are:

DEFAULT returns all parameters to their start-up values.
BAUD=*number* sets the BAUD rate to any supportable rate. *number* can be 110, 135, 150, 300, 600, 1200, 2400, 4800, 9600. The default value for BAUD is 300.
WORD=*number* sets the word length to *number*. *number* can be 5, 6, 7, or 8. The default value for *number* is 7.
STOP=*number* sets the *number* of stop bits per word. *number* is either 1 or 2. The default value for *number* is 1.
PARITY=*switch* sets the parity switch to either YES or NO. If you specify YES, you can also use EVEN or ODD. The default values for PARITY are YES and EVEN.
QUERY prompts you for each parameter.
BREAK=*value* sets the logical BREAK to *value*. The default value for BREAK is hexadecimal 03 (CONTROL C).
EVEN sets parity to even, if parity switch is YES.
ODD sets parity to odd, if parity switch is YES.

BREAK cannot be abbreviated.

Examples

```
SETCOM ENTER
```

displays the current configuration of the RS-232C port in the following format:

RS232 parameters:

Baud	= 300
Word Length	= 7
Stop Bits	= 1
Parity	= ON
Even	= ON
Break value	= X'03'

Output control line status:

DTR=ON
RTS=OFF

Input control line conditions observed:

RI	=IGNORE
DSR	=IGNORE
CD	=IGNORE
CTS	=IGNORE

```
    SETCOM (BAUD=300,WORD=8,STOP=1, PARITY=NO) (ENTER)
```

configures the RS-232C using the values specified. Notice that PARITY is specified as NO.

Technical Information

This command allows you to set the parameters to values that match any other RS-232C devices. The receiving side of the driver is interrupt driven and contains an internal one-character buffer to prevent loss of characters during disk I/O and other lengthy operations. The system usually uses the *CL devspec to communicate with the RS-232C port.

If you are using a serial printer, (1) use SET to set *CL to COM/DVR (see Appendix I), (2) use SETCOM to set the proper parameters, and (3) use the command:

```
    ROUTE *PR TO *CL (ENTER)
```

to direct output to the RS-232C (rather than the standard parallel port). Radio Shack printers do not require this procedure since they use the parallel printer port.

The line condition parameters let you set up the conventions required by most communicating devices.

The RS-232C line output *parameters* are:

DTR=*switch* Data Terminal Ready
RTS=*switch* Request To Send

The RS-232C line input *parameters* are:

DSR=*switch* Data Set Ready
CD=*switch* Carrier Detect
CTS=*switch* Clear To Send
RI=*switch* Ring Indicator

switch is either YES or NO. You cannot abbreviate any of the RS-232C line parameters.

As specified by standard RS-232C conventions, a TRUE condition means a logical 0, or positive voltage. A FALSE condition means a logical 1, or negative voltage.

DTR and RTS can be set to a constant TRUE by specifying the YES switch. If DSR, CD, CTS, or RI is specified YES, the driver observes that signal and waits for a TRUE condition before sending each character. If specified NO, the driver waits for a false condition before sending a character. If not specified, that signal is ignored.

The BREAK parameter allows you to set a logical BREAK character.

This is useful in "host" type applications. The BREAK parameter causes the serial driver to set the system break bit whenever a modem break (extended null) or an ASCII logical BREAK is received. The system pause bit is set whenever the hex code 60 is received. The system enter bit is set whenever a carriage return (0D) is received.

The default for BREAK is 3, so a (CONTROL)(C) sets the break bit. Use BREAK=*value* to to set another character as the logical break.

Technical Examples

```
SETCOM (BREAK) (ENTER)
```

configures the RS-232C port to the default values. Specifying BREAK with no value assigns the default value of 3 as the logical break value.

```
SETCOM (CTS) (ENTER)
```

configures the RS-232C port to the default values. Because CTS is specified, the driver looks at the CTS line for a TRUE condition before it sends a character.

Sample Use

Suppose you want to log-on to CompuServe. First, you have to SET *CL to COM/DVR, and then you have to use SETCOM to set the parameters of the RS-232C port so that your Model 4 can communicate with CompuServe. See the Logging-On to CompuServe section in the COMM utility description.

SETKI [(*parameters*)]	**Advanced Programmer's Command**

Sets keyboard repeat *parameters*. If you do not specify a *parameter*, the current delay and repeat rate settings are displayed.

You can use SETKI to adjust how your keyboard reacts when you press a key.

The parameters are:

DEFAULT returns the parameters to their start-up values.
RATE=*number* sets the repeat rate as *number*. *number* is any number greater than or equal to 1. *number* equals 2 when the system is started or reset.
WAIT=*number* sets the initial delay between the time a key is first pressed and the first repeat of that key as *number*. *number* is any number greater than or equal to 10. *number* equals 22 when the system is started or reset.
QUERY prompts you to enter new values for RATE=*number* and WAIT=*number*.

Examples

```
SETKI (WAIT=15) ENTER
```

sets the delay rate to 15.

```
SETKI ENTER
```

displays the current delay and repeat rate settings in the format:

Wait=15, Rate=2

Note: Both the RATE and WAIT parameters use modulo 128. For example, entering 138 has the same effect as entering 10.

	Command
SPOOL [***devspec***] [TO] [***filespec***] **(*parameters*)**	

establishes a First-In, First-Out buffer for a specified device (usually a line printer).

You can use SPOOL to print data while you perform other operations on your computer (such as create a BASIC program). However, you can not spool using BANK 1 or 2 when a BASIC program is running.

If you do not specify *devspec*, it defaults to *PR.

The *parameters* are:

NO turns off the spooler and resets *devspec*.
MEM=*number* specifies *number* as the amount of memory buffer (in K) to be used by the spooler. The value of *number* is 1 - 32.
BANK=*number* selects one of three 32K banks of memory to be used as the spool buffer. *number* can be a 0, 1, or 2. The default value of *number* is 0.
DISK=*number* specifies *number* as the amount of disk space (in K) to be used by the spooler. The value of *number* cannot be larger than the amount of available space (in K) on the disk.
PAUSE temporarily suspends output to *devspec*.
RESUME restarts *devspec* after a PAUSE.
CLEAR clears the spool buffer.

How Data Is Spooled To a Device

All data sent to *devspec*, such as a printer, is placed in an output buffer where it waits until the device is again available to accept the data.

There are two kinds of output buffers: memory and disk. You can set up a spooler with both types of buffers. Or, you can set up a spooler with a memory buffer only.

The minimum space allocation for the memory buffer depends on which BANK you select. If you specify BANK 0, a minimum of 1K (1024 bytes) is allocated for the memory buffer. If you specify BANK = 1 or BANK = 2, the entire 32K bank is automatically used for the memory buffer.

If you specify both buffers, data is sent first to the memory buffer. When the memory buffer is full, the data is sent to a disk buffer named *filespec*, where it waits to be sent to the device. If you specify a memory buffer only, data is sent to a memory buffer until the device is ready to accept it.

When you specify *filespec*, you may also use the DISK parameter to specify the amount of disk space to be used by the spooler. TRSDOS creates a file of the size specified. If you do not specify DISK=, approximately 5K of disk space is automatically allocated to *filespec*.

To prevent TRSDOS from allocating any disk space to SPOOL, specify DISK=0.

filespec remains open as long as SPOOL is on. Do not REMOVE this file or remove the disk from the drive without closing the file (by issuing a SPOOL *devspec* (NO) command).

You cannot issue a SYSGEN library command if the spooler is on.

Once the spooler is turned off, you can turn it on again. The same memory locations are used, but the following restrictions apply:

- The original parameters are not affected by turning *devspec* off and then on.
- Any parameter specified the second time cannot exceed the memory or disk parameters originally given. If they do, an error occurs.

Examples

```
SPOOL *PR TO TEXTFILE:0 (MEM=5,DISK=15) ENTER
```

allocates 5K of memory and 15K of disk space in a file named TEXTFILE/SPL on Drive 0. Since you did not specify an extension to TEXTFILE, it defaulted to /SPL.

Any output for the printer is buffered and sent to the line printer (*PR) as fast as the printer can accept the characters. If the 5K memory buffer is filled, the data is written to the disk file TEXTFILE/SPL on Drive 0.

```
SPOOL *PR (BANK=1,DISK=0) ENTER
```

creates a 32K memory buffer for data sent to *PR. Any output for the printer is sent to the memory buffer and then spooled to *PR when it is available to accept the data. Since the parameter DISK= is specified without any size, none of the spooled data is sent to a disk file.

If the memory buffer is filled, TRSDOS does not process any more printer data until *PR has printed enough data to bring the number of characters waiting to be printed below 32K (the size of the memory buffer).

```
SPOOL (CLEAR) ENTER
```

clears the information in the spool buffer.

```
SPOOL *PR (NO) ENTER
```

turns off the spooler and closes the associated disk file. Any filtering, linking, setting, or routing done to *PR is reset.

You cannot close the disk file by issuing a RESET or REMOVE library command. SPOOL must be turned off to close the file.

Sample Use

Since most programs produce reports faster than the printer can print the data, you can use SPOOL to let the programs run at top speed without having to stop and wait on the printer. That is, while the first program's report is still printing, you can begin executing the second program.

Advanced Programmer's Command
SYSGEN [([*switch*] [,] [DRIVE=*drive*])]

creates a configuration file on *drive* to store information about your system.

You can use SYSGEN to create a file of current device and driver configurations that you want TRSDOS to execute each time you return your computer to the TRSDOS copyright and start-up message.

If you do not specify *drive*, it defaults to Drive 0.

The *switch* is either YES or NO.

If you specify *switch* as YES, then your system creates a configuration file. If you don't specify *switch*, then YES is assumed.

If you specify *switch* as NO, then your system removes the configuration file. However, your system's current configuration does not change until you reset your computer.

When you issue a SYSGEN command, all current device and driver configurations are stored in a file named CONFIG/SYS. The file is invisible in the directory.

Each time you reset your computer, the configuration stored in this file is loaded into memory. If you don't want this configuration loaded into memory, hold down the (CLEAR) key when you reset the system (see the BOOT command).

When you start up or reset your computer, it is configured before any AUTO command executes.

The configuration file CONFIG/SYS contains:

- All active background tasks (such as CLOCK, DEBUG, TRACE, etc.).
- All filtering, linking, routing, and device setting (including RS-232C and KI settings).
- All programs that were loaded into high memory above HIGH$. All memory from HIGH$ to the top of memory is written to CONFIG/SYS. HIGH$ can be set with the MEMORY command or with the @HIGH$ supervisor call (see the Technical Reference Manual).
- The present state of the VERIFY library command (YES or NO).
- All Device Control Blocks (see the Technical Reference Manual for more information).

- The present state of the CAPS lock for the keyboard.

Any ROUTE or SET involving a file should never be SYSGENed. SYSGENing open files can result in lost data if the disks are switched in the drives without the files being closed. Switching a disk with open files can also cause existing data to be overwritten.

Examples

```
SYSGEN (YES) (ENTER)
```

creates a configuration file on Drive 0 and writes the system configuration to it.

```
SYSGEN (NO) (ENTER)
```

removes the configuration file from Drive 0.

Sample Use

Suppose you want to create a file of commands that automatically execute each time you startup TRSDOS.

Issue the commands:

```
TIME (CLOCK=YES) (ENTER)
SYSTEM (TRACE=YES) (ENTER)
SYSGEN (YES) (ENTER)
```

to create a CONFIG/SYS file that contains CLOCK and TRACE information.

Advanced Programmer's Command

SYSTEM (*parameters*)

configures certain areas of your TRSDOS system.

You can use SYSTEM to determine some of the commands that are put in the file that SYSGEN creates.

You can see the current configuration of your TRSDOS system by issuing a DEVICE or a MEMORY command.

Using the SYSTEM command, you can set or change the disk drive configuration and turn on or off different keyboard, video, and hardware drivers.

When you configure your system, you can store the configuration on the disk in Drive 0 with the SYSGEN library command.

Certain SYSTEM commands load driver routines into high memory. Be aware that your overall free memory decreases accordingly when you use driver routines.

SYSTEM Commands

```
    SYSTEM (ALIVE[=switch]) (ENTER)
```

displays an "alive" character in the upper right corner of the screen.

If the alive bug is moving, the task processor is running. Note that the alive bug may continue moving (indicating an alive system) even when the SYSTEM (TRACE) library command display has stopped.

If the alive bug is not moving, then the system is "hung up." Also, the bug stops during FORMAT operations and disk I/O.

The *switch* is either YES or NO. If you do not specify *switch*, YES is assumed. The ALIVE parameter uses some RAM in high memory.

```
    SYSTEM (BLINK=switch) (ENTER)
    SYSTEM (BLINK=number) (ENTER)
    SYSTEM (BLINK,parameter) (ENTER)
```

controls the TRSDOS cursor character.

switch is either YES or NO. Specifying YES turns the blinking cursor ON. Specifying NO turns the blinking of the cursor OFF.

number is an ASCII value in decimal. For example, if you specify SYSTEM (BLINK=42), the blinking cursor character is an asterisk (character 42).

The *parameters* are:

LARGE turns on a large blinking cursor.
SMALL turns on a small blinking cursor.

Specifying SYSTEM (BLINK) returns the cursor to its start-up character and size.

```
    SYSTEM (BREAK[=switch]) (ENTER)
```

enables or disables the (BREAK) key.

The *switch* is either YES or NO. If you do not specify *switch*, YES is assumed.

Once the (BREAK) key is disabled by issuing a SYSTEM (BREAK=NO) COMMAND, pressing it has no effect.

You can re-enable it at any time by issuing a SYSTEM (BREAK=YES) command. This command also enables the (BREAK) key if it was disabled by the AUTO library command.

If you switch Drive 0 disks or change the logical Drive 0 with the SYSTEM (SYSTEM) command, the default value you select for BSTEP is taken off the new Drive 0 disk.

```
    SYSTEM (DATE[=switch]) (ENTER)
```

enables or disables the prompt for the date when you power up your system.

The *switch* is either YES or NO.

If you specify the *switch* as YES, the system enables the date prompt if it has been disabled with the NO switch. If you do not specify *switch*, YES is assumed.

If you specify the *switch* as NO, the date prompt is disabled when you power up or restart your system.

Since the date is used extensively throughout the TRSDOS system, we recommend that you never disable the initial date prompt.

```
    SYSTEM (DRIVE=drive,[parameters]) (ENTER)
```

sets certain parameters for the disk drives in your system.

drive is any valid drive number in your system.

The *parameters* are:

CYL=*number* sets the default number of cylinders that is used with the FORMAT utility. *number* is in the range of 35 to 96.
DELAY=NO/YES sets the DELAY time for floppy disks. DELAY time is the time allowed between the drive motor start-up and the first attempted read of the diskette in that drive. If you specify DELAY=NO, the DELAY time is set to .5 seconds (the standard DELAY time). If you specify DELAY=YES, the DELAY time is set to 1 second.
DISABLE removes access to the specified drive.
ENABLE allows access to a drive that has been disabled.

Error Conditions

CYL is written to the system information sector on Drive 0. If you switch the disk in Drive 0, these values are taken from the new disk in Drive 0 if you format a disk.

If you try to access a disk after you have DISABLEd it, the error "Illegal drive number" appears.

If you ENABLE a drive that is not available on your system or that is not set up with the SYSTEM (DRIVE=,DRIVER) command, unpredictable results follow.

```
    SYSTEM (DRIVE=drive,DRIVER="filespec") (ENTER)
```

To access the disk drives, TRSDOS uses information stored in memory in the Drive Code Table (DCT). Since these drives are preset in your system, no special configuration should have to be done. This command is used for Radio Shack hard disk installation.

```
    SYSTEM (DRIVE=drive,WP[=switch]) (ENTER)
```

sets the software Write Protect status for the disk in *drive*.

The *switch* is either YES or NO. If you do not specify either one, YES is assumed.

If you specify WP=YES, you cannot write to the disk, although you can still read from it. If you specify WP=NO, you can write to and read from the disk (assuming the disk is not hardware write protected).

If you do not specify *drive*, the *switch* that you specify applies to all drives enabled in your system.

If there is a write-protect tab on a disk, specifying WP=NO does not allow you to write to the disk.

```
    SYSTEM (RESTORE [=switch]) (ENTER)
```

determines whether all drives are to be restored when the system is reset. Drives are restored by moving the drive heads to track 0.

If you specify RESTORE=YES, the drives are restored on reset. If you do not specify *switch*, YES is assumed. If you specify RESTORE=NO, the drives are not restored on reset.

```
    SYSTEM (SYSRES=number) (ENTER)
```

adds TRSDOS system "overlays" into high memory. You can load SYS files 1, 2, 3, 4, 5, 9, 10, 11, and 12 with this command.

Adding certain of these SYS overlays to memory speeds up most disk I/O, because these overlays do not have to be loaded from disk every time they are needed. This also allows you to purge these overlays from your system disk using the PURGE library command, providing more room for data and programs.

SYS0 and SYS2 must remain on any booting disk if a configuration file created with the SYSGEN library command is loaded.

To see which overlays are currently resident in high memory, issue a DEVICE library command.

You can add only one system overlay to high memory in a command line.

```
    SYSTEM (SYSTEM=drive) ENTER
```

assigns a drive other than Drive 0 as your system drive.

drive is any drive currently enabled in your system. There must be a system disk in *drive* when you issue the command. If *drive* is not ready, the message "Insert SYSTEM diskette in drive *x*!" appears. Insert a system diskette in the indicated drive, or press BREAK to abort the command and return to TRSDOS Ready.

Every time you issue this command, the logical drive numbers of the drives change. You can repeat this command as many times as you wish, but be careful not to lose track of which drive is assigned to which logical drive number.

```
    SYSTEM (TIME[=switch]) ENTER
```

turns on or off the start-up time prompt.

The *switch* is either YES or NO.

If you specify TIME=YES, you enable the start-up time prompt. If you do not specify *switch*, YES is assumed. If you specify TIME=NO, you disable the start-up time prompt.

When you issue this command, you must not have a write-protected disk in Drive 0.

```
    SYSTEM (TRACE[=switch]) ENTER
```

displays the contents of the Program Counter (PC) in the upper right corner of the video display. The display is a hexadecimal address and is constantly updated as a "low priority background task" (see the explanation of the @ADTSK supervisor call in the Technical Reference Manual).

The *switch* is either YES or NO.

Use the TRACE command to help you debug assembly language programs.

```
    SYSTEM (TYPE[=switch]) ENTER
```

turns on or off the task processing of the keyboard type-ahead feature. Type-ahead is active when you startup your computer.

The *switch* is either YES or NO.

You can restart the type-ahead task processing with the SYSTEM (TYPE=YES) command.

TAPE1ØØ

(Model 4 only)

TAPE1ØØ [*file*] [(*parameters*)] **Utility**
TAPE1ØØ [*file1* [TO] *file2*] [(*parameters*)]

Lets TRSDOS (1) read a cassette tape file and write it to a disk file, or (2) read a disk file and write it to a cassette tape.

You can use TAPE1ØØ to read files from cassette tape as well as from Model 4 disks.

The cassette tape must have been made with the Model 1ØØ computer.

file, file1, and *file2* are each either a TRSDOS filespec or a Model 1ØØ filename.

A Model 1ØØ filename is 1 - 6 alphanumeric characters long and it must begin with a letter. For example, ACCT61, LETTER, and ABFILE can be Model 1ØØ filenames.

If you do not specify *file* or *file1* and *file2*, you will be prompted to enter the source and destination filespecs if the operation is a WRITE, or just the destination filespec if the operation is a READ.

The *parameters* are:

READ specifies that you want to read a file (*file* or *file1*) from tape and write it to a file (*file* or *file2*) on disk. If you specify READ, you do not have to specify *file1*. TRSDOS simply reads the first text file it sees on the tape.
WRITE specifies that you want to read a disk file (*file* or *file1*) and write it to a tape as *file* or *file2*.

If you do not specify the READ or WRITE parameter, you will be prompted for it.

Examples

```
TAPE1ØØ PRNTER TO PRINT/DAT:Ø (READ) ENTER
```

TRSDOS reads the Model 1ØØ file PRNTER and writes it to the disk in Drive Ø as PRINT/DAT.

```
TAPE1ØØ ACCTING/TXT:1 (READ) ENTER
```

TRSDOS reads the first text file it finds on a Model 1ØØ tape and writes it to the disk in Drive 1 as ACCTING/TXT.

```
TAPE1ØØ WEST/DAT:Ø TO WESTRN (WRITE) ENTER
```

TRSDOS reads the Drive Ø disk file WEST/DAT and writes it to a file on a Model 1ØØ tape named WESTRN.

Error Conditions

Any file (disk or tape) must fit in available memory or the error message "File too large to fit in available memory" appears.

Command

TIME [***hh:mm:ss***] [**(*parameter*)**]

You can use TIME to see the current time. You can also use it to reset the time.

If you specify *hh:mm:ss*, TRSDOS resets the time. If you do not specify it, TRSDOS displays the current time.

The parameter is:

CLOCK=YES/NO turns the clock display on or off. YES is the default.

The real time clock turns off while TRSDOS does some of its disk I/O functions, such as BACKUP and FORMAT, so do not depend on the clock for constantly accurate time and date information.

You can enable and disable the prompt for time on power-up or reset with the SYSTEM (TIME=) command.

Examples

```
TIME (ENTER)
```

displays the real time of the system. The clock is reset to 00:00:00 every time you power up.

```
TIME (CLOCK=YES) (ENTER)
```

displays the real time clock in the upper right corner of the screen. Note: CLOCK will print over whatever TRSDOS attempts to print at the location occupied by the clock display.

```
TIME 12:29:34 (ENTER)
```

sets the clock to 12:29:34 p.m. The latest acceptable time is 23:59:59, as the clock runs in the 24-hour mode. When the clock reaches 23:59:59, the date is automatically updated.

The time lag between pressing (ENTER) and the time set on the clock is approximately 2 seconds. So, when setting the clock with the correct time, remember to adjust for the 2-second time lag.

VERIFY

VERIFY [(*switch*)] **Command**

Controls the verify function.

You can use VERIFY to assure you that data was properly written to a disk.

When VERIFY is on, TRSDOS reads the data it writes to the disk to verify that the data is readable.

The *switch* is either ON or OFF.

A TRSDOS floppy disk system starts up with VERIFY off. A hard disk system starts up with VERIFY on.

Although having the VERIFY switch turned on provides a reliability check during disk I/O, it also increases overall processing time when you write to a disk file. You must determine if the increase in reliability warrants the increase in processing time.

All disk writes are automatically verified during any BACKUP utility function, whether or not the VERIFY switch is on.

The state of the VERIFY command can be saved in the configuration file with the SYSGEN library command. (You can check the present status of VERIFY using the DEVICE command.)

Examples

```
    VERIFY (ON) (ENTER)
```

turns on the verify function.

```
    VERIFY (OFF) (ENTER)
```

turns off the verify function.

```
    VERIFY (ENTER)
```

turns on the verify function.

Sample Use

Suppose you are writing a tax file named TAX/TXT to disk and it is extremely important that the information in the file be correct.

Using VERIFY causes TRSDOS to produce an informative message when data written to TAX/TXT is written incorrectly. An informative message could indicate that the disk needs to be replaced or the drives need to be cleaned.