Radio Shack

TRS-80® MODEL 16

TRSDOS™-16 DISK OPERATING SYSTEM

CUSTOM MANUFACTURED IN U.S.A. BY RADIO SHACK, A DIVISION OF TANDY CORPORATION

Limited Warranty Information

I. CUSTOMER OBLIGATION

A. CUSTOMER assumes full responsibility that this computer hardware, (the "Equipment") and/or software (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 is 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 ticket, RADIO SHACK warrants to the original CUSTOMER that the Equipment and the cassettes and/or diskettes containing software programs are free from defects. This warranty is only applicable to purchases from RADIO SHACK company-owned Computer Centers, retail stores and through RADIO SHACK franchisees and dealers. The warranty is void if the unit's case or cabinet has been opened, or if the unit has been subjected to improper or abnormal use. If a defect occurs during the warranty period, the defective Equipment must be returned to a Radio Shack Computer Center, a Radio Shack retail store, participating franchisee or dealer for repair, along with a copy of the sales ticket 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 complete refund, 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 Equipment or Software. Software is licensed on an "AS IS" basis, without warranty. CUSTOMER'S exclusive remedy, in the event of a software defect is its repair or replacement within thirty (30) calendar days of the date of purchase upon its return to a Radio Shack Computer Center, Radio Shack retail store, participating franchisee or dealer along with the sales ticket.

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.

F. 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 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, 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 any Equipment or Software.

C. No action arising our 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 ticket for the Equipment or Software whichever first occurs.

D. Some states do not allow the exclusion or limitation 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 application or system Software and/or the RADIO SHACK system Software (including firmware) installed in or provided with the Equipment 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 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.

D. 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.

E. 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.

F. 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 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 and/or Software to a third party for lease to CUSTOMER.

B. The limitations of liability and warranty provisions herein shall insure to the benefit of RADIO SHACK, the owner and/or licensor of RADIO SHACK Software to RADIO SHACK, and any author or manufacturer of computer hardware or Equipment sold or Software licensed by RADIO SHACK.

VII. 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.

TRSDOS-16

THE MODEL 16

DISK OPERATING SYSTEM

OWNER'S MANUAL

INTRODUCTION

WHAT IS AN OPERATING SYSTEM?

An operating system is a low level program which monitors and controls your entire computer system.

This includes the Model 16's two microprocessors, as well as the disk drives, keyboard, printer, and all other equipment.

It is the operating system which enables you to use application programs (such as PAYROLL, PROFILE, and MAILING LIST II.)

WHAT IS TRSDOS-16?

TRSDOS-16 is one of the three operating systems you can use with your Model 16.

It uses both of the Model 16's microprocessors -- the Z8Ø and the MC68ØØØ. All Model 16 application programs are designed to run under TRSDOS-16.

If you've read the Operator's Manual, you know about the other two operating systems -- TRSDOS and TRSDOS-II. They use only the Z8Ø microprocessor and are for Model II mode programs.

ABOUT THIS MANUAL

This manual shows how you can use the TRSDOS-16 operating system to:

- write Model 16 programs
- store, retrieve or manipulate information on disk.

In the Operator's Read Me First Manual, we covered all the essential information to get you started. This means if you're not a programmer, you do not need to read this manual.

If you are a programmer, you'll find a lot of useful information in this manual:

SECTION I/ USING TRSDOS-16 describes how to start-up TRSDOS-16, what TRSDOS-16 Ready means, and some general information on how TRSDOS-16 works.

SECTION II/ TRSDOS-16 SYSTEM COMMANDS contains a number of commands you will find helpful.

SECTION III/ TECHNICAL INFORMATION explains how to use TRSDOS-16 on a technical level. Besides providing useful technical information, this section lists many TRSDOS-16 routines (called supervisor calls or SVCs) which you can call from your machine-language program.

SECTION IV/ THE APPENDICES contains more information, including memory maps, error messages, and codes useful in writing programs.

This manual describes TRSDOS-16 only. If you have a new Model 16 and need information on TRSDOS and TRSDOS-II (the Model II operating systems), see your Owner's Manual/Model II Mode.

If you have an Enhanced Model II, see the Model II Owner's Manual for information on TRSDOS and the Supplement to the Model II Owner's Manual for information on TRSDOS-II.

NOTATIONS

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

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

<KEYBOARD CHARACTER>
indicates keys you press.

lowercase underlined
represent words, letters, characters or values.

H'nnnn
specifies nnnn as a hexadecimal (base 16) number. All other numbers in the text of this book are in decimal (base 1Ø) form, unless otherwise noted.

TERMS

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

command — represents the TRSDOS-16 command you want to execute.

comment — is an optional field used to document the purpose of the command line.

{options} — is a list of one or more parameters that may be needed by the command. Some commands have no options. The braces { } around options can usually be omitted if you don't use a comment at the end of the command line.

filespec — is a standard TRSDOS-16 file specification having the general form:
filename/ext.password:drive(disk name)

hard disk refers exclusively to a hard disk (Drives 4 - 7).

diskette refers exclusively to a floppy diskette (Drives Ø - 3).

disk refers to a disk that can be either a hard disk or a floppy diskette (Drives Ø - 7).

primary drive refers to the disk drive that contains the operating system information Drive Ø or 4).

TABLE OF CONTENTS

PAGE

INTRODUCTION 1
- About this Manual 3
- Notations and Terms 5

SECTION I/ USING TRSDOS-16
- How the Computer Uses TRSDOS-16 11
- Loading TRSDOS-16 11
- What Does TRSDOS-16 Ready Means 12
- Disk Files 14
- Sample Exercise 18
- Swapping Diskettes 21

SECTION II/ TRSDOS-16 SYSTEM COMMANDS
- Introduction 25
- How To Use This Section 26
- Listing of Commands 29

SECTION III/ TECHNICAL INFORMATION
- Memory Requirements 105
- Disk Organization 108
- Disk Files 113
- Supervisor Calls 122
- Calling Procedure 125
- Programming With User Interrupts 127
- SVC Listings 129

SECTION IV/ APPENDICES
- A/ Error Messages 221
- B/ The Configuration Command File 232
- C/ Memory Map 238
- D/ ASCII Character Codes 239
- E/ Graphics Codes 245
- F/ Specifications 246
- G/ SVC Quick Reference List 253

INTRODUCTION

HOW THE COMPUTER USES TRSDOS-16

Whenever you are using a Model 16 program, your computer will, from time to time, need to referencé TRSDOS-16. It always looks for TRSDOS-16 on the primary drive.

The primary drive is:

Drive Ø -- if you start up your system under floppy disk control

Drive 4 -- if you start up your system under hard disk control. (It may be your only hard disk drive)

For this reason, if you have a floppy disk system (that is, you don't have the optional hard disk) or if you have a hard disk system and are operating under floppy disk control, you must at all times have a diskette containing TRSDOS-16 in Drive Ø.

If you have a hard disk system and want to operate under the control of your hard disk, you need to copy TRSDOS-16 onto your primary drive. The Operator's Read Me First Manual shows how.

LOADING TRSDOS-16

When you install and power up your system, you'll see the TRSDOS-II start-up logo. This means you're in the TRSDOS-II 4.1 Operating System (the Model II mode). You then need to enter the date and time. Enter the date in the form MM/DD/YYYY form. For example, type:

Ø8/21/1982 <ENTER>

for August 21, 1982. You now will be prompted to enter the time. You can skip this question by simply pressing <ENTER>. The time will start at ØØ.ØØ.ØØ.

If you want to set the time, type the time in the 24-hour format -- HH.MM.SS. The seconds are optional. For example, type:

14.3Ø <ENTER>

for 2:3Ø pm.

After entering the date and time, the TRSDOS-16 Operating System AUTOmatically loads and displays:

TRSDOS-16 Ready

This indicates that you are at the 68ØØØ Disk Operating System's command level.

NOTE: A factory-set AUTO command loads the TRSDOS16/SYS program -- the 68ØØØ operating system. It then loads a configuration command file named CONFIG16/SYS which links certain extra operating system modules into memory. If you want to save memory or change the configuration command file, see Appendix B.

To override this factory-set AUTO command, see AUTO in SECTION II.

You can also load TRSDOS-16 from the TRSDOS-II Ready prompt by typing:

BOOT16 TRSDOS16/SYS <ENTER>

WARNING: DO NOT press <BREAK> when you are loading the TRSDOS-16 operating system. If you do, your computer may become confused if interupted while doing the configuration.

If you accidentally press <BREAK>, you will have to power down your system before trying to load TRSDOS-16 again.

WHAT DOES TRSDOS-16 READY MEAN?

Whenever you see the TRSDOS-16 Ready prompt you know that you are in control of TRSDOS-16 -- not COBOL, PAYROLL, or any of your application programs. Being in control of TRSDOS-16 allows you to do one of these operations:

- execute a system command
- execute a program

If you want to perform any other operation, you need to be in control of an application program.

When you are in control of TRSDOS-16 and an error occurs, you'll get one of the error messages listed in Appendix A.

If you get an error message not listed, it came from an 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-16 system command whenever you see the TRSDOS-16 Ready prompt. The command you type can consist of up to 80 characters. You must end the command by pressing <ENTER>.

For example, if you want to see the TRSDOS-16 system commands, type:

```
LIB <ENTER>
```

TRSDOS-16 displays a list of all the available system commands and returns to TRSDOS-16 Ready:

```
APPEND  ASSIGN  ATTRIB  AUTO    BACKUP   CLEAR   CLS     COPY
CREATE  DATE    DEBUG   DIR     DISMOUNT DO      DRIVE   DUMP
EXEC    FCOPY   FILES   FLOPPY  FORMAT   FORMS   FREE    HELP
KILL    LIB     LIST    LOAD    MOUNT    MOVE    MSG     PATCH
PAUSE   PRINT   LIST    PURGE   RELEASE  RENAME  RESET   RESTORE
SAVE    SETCOM  SIZE    SPOOL   TERMINAL TIME    VERIFY  VERSION
```

EXECUTING A PROGRAM

You can also execute a program (such as the Editor) at the TRSDOS-16 Ready prompt. If what you enter is not a recognized system command, TRSDOS-16 checks to see if it is the name of a 68000 program. It checks for the program file on all drives, beginning with the primary drive (Drive 0 or Drive 4) unless you specify drive number.

If TRSDOS-16 finds a matching 68000 program file, it loads and executes the file. Otherwise, you get an error message. For example:

EDIT16 <ENTER>

loads the Editor program. Now you can create a program or edit an existing one.

DISK FILES

You can keep a record of anything you type into your Model 16 by storing it on disk (hard or floppy) 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 indexes the name of the file and it's 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.

If you want to see how a disk file is created and stored, see the SAMPLE EXERCISE/ CREATING A DISK FILE later in this chapter.

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-16 format you'll use every time you reference your file:

filename/ext.password:drive(disk name)

filename
The name of your file can be anything you like, as long as it is one to eight alphanumeric characters, the first of which must be a letter. For example, if you want to save a file containing an inventory list, you could simply name it:

INVNTRY

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 with a preceding slash (/).

You can use an extension to provide additional information on a file. For example, with an extension such as /NEW, /IRS, or /PAY, you could distinguish files with the same name or divide files into categories.

You can also use an extension to indicate the type of file you have. The extension /BAS indicates a BASIC program file; /DAT indicates that a file contains data only; or /SRC defines a SOURCE file.

With the extension /DAT, the new name of our inventory file is:

```
INVNTRY/DAT
```

password
Some files allow you to protect them. You can accomplish this protection via a password either when you create the file or with the command ATTRIB.

A password is a sequence of up to eight alphanumeric characters, the first of which must be a letter, with a period (.) preceding it to separate it from the filename.

There are two levels of passwords and the protection they provide -- access passwords and update passwords. These passwords not only can inhibit entry to a file, they also can provide protection at varying levels.

When you initially create a file and assign a password, the access and update passwords are the same. Later, if you choose, you can change these values with the system command ATTRIB and thus provide the additional protection to your files. (See ATTRIB for details.)

With the password Sesame, the new name of our inventory file is:

```
INVNTRY/DAT.Sesame
```

drive
Often when you're using your computer, you'll have more than one disk in use. Whether these disks are floppy or hard, you can speed up the file access time by specifying the

drive the desired file is on. Use the form :drive for the drive number.

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

To indicate your inventory file's location on the filespec, type:

```
INVNTRY/DAT.Sesame:2
```

See your Operator's Read Me First Manual (Model 16), Hard Disk Owner's Manual, or Operations Manual (Enhanced Model II) if you are not sure what your drive numbers are.

disk name
You may want to indicate the name of the disk that a file is on. The disk name was assigned when you formatted or backed up the disk.

It takes the form (disk name), which is a field of up to eight alphanumeric characters, the first character being a letter, with parentheses ( ) surrounding the entire name. If you specify the disk name, you must also specify the drive number.

Now, if you're ready to reference your inventory file, enter this complete filespec to ensure that you're getting the right one:

```
INVNTRY/DAT.Sesame:2(WREHSE)
```

Of course, every filespec you enter won't include all of these optional specifications, however, you can use any combination of the fields as long as you follow the guidelines indicated.

Here are some more examples of valid TRSDOS-16 filespecs:

```
DOPROG.OPEN
CLR/BAS:1
MOD16:4(TRSDOS16)
STL12/TXT.Arch:1(TRAVL82)
GAME1
THESIS/OLD:2
CONTEMP:3
```

Wildcards

Certain system commands and SVC's allow you to specify a collection of files by using a "wildcard" mask. An asterisk "*" in a file specification represents a wildcard field and means "any sequence of zero or more characters". For example:

```
*/BAS:1
```

represents all the files stored on the diskette in Drive 1 having the extension /BAS.

```
D*
```

represents all the files on the disk in the primary drive, beginning with D, without extensions.

As an example use, if you want a DIRectory of all the files with an extension that begin with the letter D, type:

```
DIR D*/* <ENTER>
```

TRSDOS-16 returns a listing of all the files beginning with D and having extensions:

```
Disk Name:TRSDOS                    Drive:4          04/09/82        00.25.35
 File Name       Created  Updated        Atrb Fil  Rec     # of     -----Sectors-- -
                 MM/DD/YY MM/DD/YY             Typ  Len     Records   Alloc     Used
DATM32/TXT       04/09/82 04/09/82        D*X0  V   +++         +++       1        1
DOBUDGET/SRC     04/09/82 04/09/82        D*X0  V   +++         +++       1        1
DEMOPROG/1       04/09/82 04/09/82        D*X0  V   +++         +++       1        1
DIRECAC/FLE      04/09/82 04/09/82        D*X0  V   +++         +++       1        1
4 Files Displayed
```

Super Wildcard

Besides the wildcard "*", TRSDOS-16 has a super wildcard -- "!". You can use it to specify all files, with and without extensions.

For example, if you want to FCOPY all files from a diskette to hard disk, you can use the super-wildcard. This accomplishes in one step what it would take the wildcard two steps to do:

```
FCOPY */*:1 TO 4 <ENTER>
FCOPY *:1 TO 4 <ENTER>
```

Here, the first command FCOPYs files with extensions. The second then FCOPYs files without extensions. But, if you use the super wildcard and type:

```
FCOPY !:1 TO 4 <ENTER>
```

TRSDOS-16 FCOPYs all files on the diskette in Drive 1 to hard disk in one step.

You can use the wildcard and super wildcard with these system commands:

```
DIRectory
FCOPY
KILL
MOVE
```

SAMPLE EXERCISE/ CREATING A DISK FILE

This is an example of how to create a command file and then save it to disk. When run, the file will automatically clear the screen and print the date and time in the top left hand corner of the screen. This command file -- known as a DO-file -- is a program made up of one or more system commands or programs that is executed with the command DO.

To create this file, you must first enter the Editor and its Insert mode by typing:

```
EDIT16 <ENTER>
IN <ENTER>
```

(See the DO command in this manual for further information.) Now you should be in the insert mode (with the prompt I?) where you can type in the command lines:

```
CLS <ENTER>
MSG "TODAY'S DATE IS:" <ENTER>
DATE <ENTER>
```

<ENTER>

At this point, the command file is in the computer's memory. If you want to keep or run it, you'll have to save it in a disk file. Type:

SAVE MYPROG <ENTER>

and the Editor writes your program, with the name MYPROG, to disk. Now you have a permanent copy.

Exit the Editor by typing QU <ENTER>.

To run your DO-file, type:

DO MYPROG <ENTER>

TRSDOS-16 will clear your screen and display:

```
TODAY'S DATE IS:

Fri May 14 1982 134 -- 10.24.30

TRSDOS-16 Ready
```

Sometime in the future, you may want to run MYPROG again, but you've forgotten the exact filename you gave it. You can look at the directory (as we mentioned earlier) to see how it's filed. Type:

DIR <ENTER>

for a directory listing of the disk in Drive 4. TRSDOS-16 returns a DIRectory listing like this:

```
Disk Name:TRSDOS                    Drive:4          04/09/82        00.25.52
 File Name       Created  Updated        Atrb Fil  Rec    # of      ----Sectors----
                 MM/DD/YY MM/DD/YY             Typ  Len    Records   Alloc     Used
DESIGN2          04/05/82 04/05/82        P*X0  F   256          2       2        2
READ/ME          04/05/82 04/05/82        D*X0  F   256         77      77       77
TRSDOS16/SYS     03/26/82 04/16/82        D*X0  F   256        107     107      107
CONFIG16/SYS     03/26/82 04/16/82        D*X0  V   +++        +++       1        1
ASM16            02/25/82 04/16/82        D*X0  F   256        148     148      148
LINK16           02/25/82 04/16/82        D*X0  F   256         64      64       64
EDIT16           03/27/82 04/16/82        D*X0  F   256         50      50       50
BOOT16           04/03/82 04/16/82        P*X0  F   256          6       6        6
IFC              04/03/82 04/16/82        P*X0  F   256         13      13       13
RUNCOBOL/SYS     03/27/82 04/16/82        D*X0  F   256         68      68       68
VIDTEX           10/13/80 04/07/82        D*X0  F     1        +++      13      +++
SAMPLE1/PRO      04/09/82 04/09/82        D*X0  V   +++        +++       2        2
DATM32/TXT       04/09/82 04/09/82        D*X0  V   +++        +++       1        1
DOBUDGET/SRC     04/09/82 04/09/82        D*X0  V   +++        +++       1        1
DEMOPROG/1       04/09/82 04/09/82        D*X0  V   +++        +++       1        1
DIRECAC/FLE      04/09/82 04/09/82        D*X0  V   +++        +++       1        1
MYPROG           04/09/82 04/09/82        D*X0  V   +++        +++       1        1
17 Files Displayed
```

You can also use a wildcard to find the filename you want. For example, if the filename does not have an extension, you can avoid a long directory listing that consists of all files with and without extensions by typing:

```
DIR * <ENTER>
```

TRSDOS-16 returns a directory listing that is substantially shorter because it only lists the files without extensions:

```
Disk Name:TRSDOS                    Drive:4          04/09/82        00.26.21
 File Name       Created  Updated        Atrb Fil  Rec    # of      ----Sectors-- -
                 MM/DD/YY MM/DD/YY             Typ  Len    Records   Alloc     Used
DESIGN2          04/05/82 04/05/82        P*X0  F   256          2       2        2
ASM16            02/25/82 04/16/82        D*X0  F   256        148     148      148
LINK16           02/25/82 04/16/82        D*X0  F   256         64      64       64
EDIT16           03/27/82 04/16/82        D*X0  F   256         50      50       50
BOOT16           04/03/82 04/16/82        P*X0  F   256          6       6        6
IFC              04/03/82 04/16/82        P*X0  F   256         13      13       13
VIDTEX           10/13/80 04/07/82        D*X0  F     1        +++      13      +++
MYPROG           04/09/82 04/09/82        D*X0  V   +++        +++       1        1
8 Files Displayed
```

SWAPPING DISKETTES

Whenever you want to change the diskettes in any of your floppy drives, you must perform the DISMOUNT/MOUNT operations. (NEVER change a diskette when a file on that diskette is in use or open.)

To swap diskettes, first remove the diskette(s) you want to change and type:

DISMOUNT <ENTER>

This informs TRSDOS-16 that you have just removed diskettes from the drives.

TRSDOS-16 returns the prompt:

INIT DONE

Insert other diskettes, close the drive door, and type:

MOUNT <ENTER>

Again TRSDOS-16 will acknowledge the change of diskettes with:

INIT DONE

You can begin using the diskettes.

(See the MOUNT and DISMOUNT commands in the next section for more information.)

SYSTEM COMMANDS

TRSDOS-16 SYSTEM COMMANDS

TRSDOS-16 system commands (typed in at the TRSDOS-16 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. Anytime you use a blank diskette, you should use one of these commands:

FORMAT
BACKUP

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 command 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	MOUNT
DATE	SETCOM
DISMOUNT	TIME
FORMS	

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

CLEAR	PAUSE
CLS	PRINT
DIR	SIZE
DO	SPOOL
FREE	T
HELP	TERMINAL
LIB	VERIFY
LIST	VERSION

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

APPEND	MOVE
ATTRIB	PROT
COPY	PURGE
CREATE	RENAME
FCOPY	RESTORE

KILL SAVE

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

DEBUG EXEC
DUMP LOAD

HOW TO USE THIS SECTION

This section contains an alphabetical listing of each system command, with each listing divided into several parts.

The command syntax is the first line you'll see after the command keyword. Use it as your guide to type in a system command. (See SYNTAX below for a detailed explanation)

Following each syntax is a definition of the system command. This tells you exactly what the specific command does.

Next is additional information on the parameters of the command; i.e., what are the values you must supply, what is optional information, and what these options do when included in the command.

After this optional information, you'll find further explanation of the command, including special instructions on the command, switches, and how best to use the command for your purposes.

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

SYNTAX

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

For example, the syntax for the CLS (CLear Screen) command is simply:

CLS

CLS <ENTER> is all you need to type to execute this command.

The syntax for the KILL command includes an optional parameter (a value you supply):

KILL filespec

In this case, the parameter is a TRSDOS-16 filespec (discussed in Chapter 1). For example, if you want to kill the disk file named SAMPLE in Drive 1, you could type:

KILL SAMPLE:1 <ENTER>

Still other commands require additional parameters, such as:

COPY source filespec TO destination {option}

Here you must supply the name of the source filespec you wish to copy and the destination where you want it copied. For example:

COPY NEW/DAT:1 TO NEWDAT/1:2 <ENTER>

makes a copy of the file NEW/DAT, from the diskette in Drive 1, onto the diskette in Drive 2, and then names the new file NEWDAT/1.

Sometimes additional information is required; sometimes it is optional. This optional information is indicated inside braces { }. In the COPY example above, there is one option:

{ABS}

When typing this command, you must decide if you need this option which tells TRSDOS-16 to overwrite any existing files with the same name. If so, type:

COPY NEW/DAT:1 TO NEWDAT/1:2 ABS <ENTER>

You can usually omit the surrounding braces { } unless you include a comment or if the second filespec is optional and omitted.

Although the variable "comment" is not included in every syntax statement, you can add one at any time. Comments are for your information only. For example:

COPY NEW/DAT:1 TO NEWDAT/1:2 {ABS} Latest version

documents the purpose of the COPY command.

You might want to use the comment if you are calling the command from a DO-file (see the DO command) or a program.

Every system command uses some variation of the syntaxes discussed above. Pay attention to each command to know the appropriate parameters and options to use.

APPEND

APPEND source filespec TO destination

Copies the contents of the source filespec onto the end of the contents of destination. (The contents of the source file remain the same.)

The destination can be a filespec or a drive number

If it is a drive number alone, TRSDOS-16 will append only if that drive contains a disk file with the same name as the source filespec.

The types of the two files must match, i.e., both must be variable length records (VLRs) or both must be fixed length records (FLRs).

You cannot use the APPEND command with ISAM (indexed access files used by some Compilers such as the COBOL Compiler) files, program files, or TRSDOS-II DO files.

Example

```
APPEND EMPLFILE TO STAFF/LST:3 <ENTER>
```

copies the contents of EMPLFILE onto the end of STAFF/LST on Drive 3.

```
APPEND DOC/NEW:1 TO 2
```

copies the contents of DOC/NEW on Drive 1 to the file of the same name on Drive 2.

ATTRIB

ATTRIB filespec {options}

Assigns or changes the password and protection level of an existing filespec.

Passwords are initially assigned when the file is created. At that time, the update and access passwords are set at the same value (either the password you specified or a blank password). See TRSDOS-16 FILESPECS in Section 1 of this manual for further explanation of passwords.

The options are:

ACC=password sets the access password to password. If omitted, the access password remains the same.

UPD=password sets the update password to password. If omitted, The update password remains the same.

PROT=level specifies the protection level for access. If omitted, the level is unchanged. The optional protection levels for access to a file are:

NONE	No access
EXEC	Execute only
READ	Read and execute
WRITE	Read, execute and write
RENAME	Rename, read, execute and write
KILL	Kill, rename, read, execute and write (gives access word total access)

This command allows you to assign a file two passwords. The access 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 access password, the operator will only be able to read and execute the file; not change, rename or kill it.

In the same manner, the update password could be for the programmer. Using the update password, the programmer could change, kill, or rename the same file. (When you use the update password to access a file, TRSDOS-16 ignores the PROTection level.)

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

Examples

```
ATTRIB DATAFILE  ACC=JUNE1Ø, UPD=NEWDAT <ENTER>
```

sets the access password to JUNE1Ø and the update password to NEWDAT. The PROTection level remains at the previous setting.

```
ATTRIB PAYFILE ACC= ,PROT=READ <ENTER>
```

sets the access password to blanks, leaves the update password the same, and sets the level of protection to allow only reading and execution of PAYFILE.

AUTO

AUTO command line

Stores command line. This command line will automatically execute whenever you start-up TRSDOS-16. (That is, after you enter the date and time, TRSDOS-16 will load, execute the command line, and then display the TRSDOS-16 Ready prompt.)

command line is optional. If you omit it, TRSDOS-16 deletes the AUTO command line currently stored.

The system doesn't check the command line for errors when you first enter the AUTO command line. Errors are detected when the automatic command is actually executed.

Examples

```
AUTO DIR <ENTER>
```

executes the DIRectory command whenever you start-up. The system then returns to TRSDOS-16 Ready.

```
AUTO <ENTER>
```

Turns off the AUTO function currently stored.

```
AUTO DO MYPROG <ENTER>
```

executes the DO-file named MYPROG.

BACKUP

BACKUP drive1 TO drive2 (options)

(FOR FLOPPY DISKETTE USE ONLY)
Makes an exact copy of the source diskette in floppy drive1 to the destination diskette in floppy drive2.

If the destination diskette is unformatted, the BACKUP command will format it before copying the source diskette to it.

The options are:

ID=id assigns the name id to the new diskette. If omitted, TRSDOS-16 gives the new diskette the same name as the source diskette.

PW=password indicates the master password of the source diskette. TRSDOS-16 won't duplicate a diskette unless you give the correct password. (All diskettes distributed by Radio Shack use PASSWORD as the master password.)

NEW=password assigns password to the destination diskette. If omitted, TRSDOS-16 uses the password of the source diskette.

ABS instructs TRSDOS-16 to overwrite the data on the destination diskette without prompting.

To make a backup copy, you need at least two floppy diskette drives. If you don't have two floppy drives, use the COPY, FCOPY, or MOVE system commands. (See the appropriate command for further information.)

If you have a new Model 16 (not an Enhanced Model II), you can BACKUP a single-sided diskette to a double-sided diskette. This gives you twice as much disk space. You don't need to tell TRSDOS-16 whether the diskettes are single-sided or double-sided.

You cannot BACKUP a double-sided diskette to a single-sided diskette.

If you want to make a copy of a system diskette, BACKUP is

the only way to do this.

Examples

BACKUP Ø TO 1 <ENTER>

makes an exact copy of the floppy diskette in Drive Ø to the floppy diskette in Drive 1.

BACKUP Ø TO 3 {ID=MODEL16 NEW=FEB23} <ENTER>

duplicates the diskette in Drive Ø to the diskette in Drive 3, naming the new diskette "MODEL16" and assigning the master password "FEB23" to it.

BACKUP 1 TO Ø <ENTER>

allows you to copy a data diskette in Drive 1 to a diskette in Drive Ø. After entering this command, TRSDOS-16 will prompt you to insert the destination diskette in Drive Ø.

CLEAR

CLEAR

Clears user memory.

Example

CLEAR <ENTER>

CLS

CLS

Clears the display and positions the cursor at the top left-hand corner of the display.

Example

CLS <ENTER>

COPY

COPY source filespec TO destination {option}

Copies the source filespec to the destination.

destination can be either a filespec or a drive number. If you use a filespec as the destination, you must also specify the drive number of the disk containing that file, Otherwise, it automatically goes to the primary drive.

If you specify a drive number only as destination, TRSDOS-16 will copy the source file to the disk in that drive, giving the destination file the same name as the source file.

The option is:
ABS tells TRSDOS-16 to overwrite any existing file with the same name without prompting

The source filespec must be a file you can use with TRSDOS-16 or TRSDOS-II. If you cannot, you must FCOPY rather than COPY it. (See FCOPY for details.)

You can make single drive copies of a file. If so, be sure to assign a different filespec for the destination.

Example

COPY OLDFILE:3 TO NEWFILE:4 <ENTER>

makes a copy of OLDFILE from the diskette in Drive 3 to the diskette in Drive 4, naming the new file NEWFILE.

COPY NEW/DAT TO DEFUNCT/DAT:2 ABS <ENTER>

copies NEW/DAT to DEFUNCT/DAT (in Drive 2). If you already have a disk file named DEFUNCT/DAT in Drive 2, this command overwrites it with the new file.

```
COPY NEW/DAT TO 2 <ENTER>
```

copies NEW/DAT to the diskette in Drive 2. The newly copied file is also named NEW/DAT.

```
COPY FILE/BAS:4 TO NEW/BAS:1 <ENTER>
```

copies FILE/BAS from the diskette in Drive 4 to the diskette in Drive 1, naming the new file NEW/BAS.

```
COPY INCTAX/IRS:1 TO INCTAX/82:1 <ENTER>
```

copies the file INCTAX/IRS on the diskette in Drive 1 to the same diskette, renaming the file INCTAX/82.

CREATE

CREATE filespec {options}

Creates a file named filespec and preallocates space for its contents. Without CREATE, TRSDOS-16 allocates space for your file dynamically as you write to it.

The available options are:

- **NGRANS=n** allocates n X 5 sectors to the file. For example, if you want to allocate 1ØØ sectors to the file, set NGRANS to 2Ø.
- **NRECS=n** assigns n fixed length records to the file. LRL must accompany NRECS.
- **LRL=n** assigns n as the logical record length. n can be 1 to 256. If LRL is omitted, the record length defaults to 256.
- **TYPE=t** specifies the record type as t. t is either F, a fixed length record, or V, a variable length record. If TYPE is omitted, t defaults to F.

NGRANS and NRECS are mutually exclusive. If you use NGRANS, don't use NRECS. If you use NRECS, don't use NGRANS.

When you use CREATE to preallocate a file, TRSDOS-16 does not deallocate unused space at the end of the file. Without CREATE, TRSDOS-16 deallocates the unused space upon closing the file.

Examples

```
CREATE NAMEFILE NGRANS=45Ø,TYPE=F <ENTER>
```

creates a fixed length record file named NAMEFILE with 225Ø sectors.

```
CREATE DATMAS/NJ2 NRECS=1ØØ,LRL=2Ø <ENTER>
```

creates a fixed length record file named DATMAS/NJ2 with 1ØØ logical records of 2Ø bytes each.

```
CREATE MARKET/WST:3 NGRANS=1ØØ,TYPE=V <ENTER>
```

creates a variable length record file on Drive 3 named MARKET/WST and allocates 5ØØ sectors to it.

```
CREATE EMPLY/LST NGRANS=2ØØ,TYPE=F
```

creates a 1ØØØ sector fixed length record file on the diskette in Drive Ø.

DATE

DATE

Displays the date and time in the format:

```
WED MAR 25 1981 84 -- 16.24.34
```

for Wednesday, March 25, 1981, the 84th day of the year, 4:24:34 p.m. Note that leading zeroes are not shown.

Example

```
AUTO DATE <ENTER>
```

automatically displays the date and time upon startup.

DIR

DIR source {options}

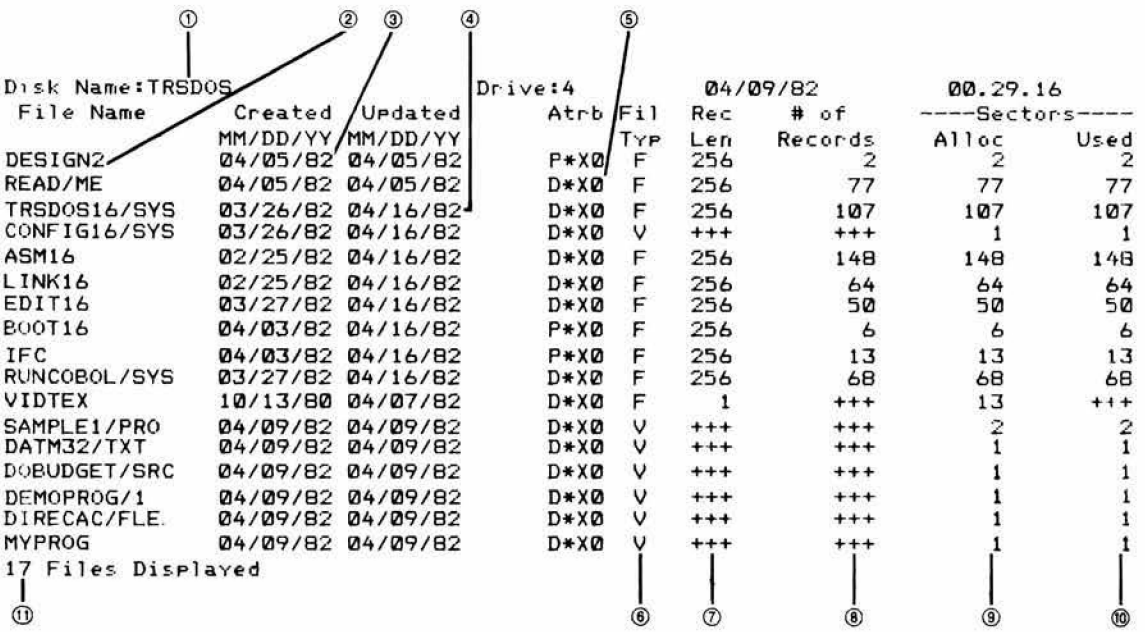
Displays the disk's directory.

source can be a standard TRSDOS-16 filespec, a wildcard, or a drive number (Ø-7). If drive number is omitted, TRSDOS-16 goes to the first available drive.

The options are:

PRT prints the directory listing on the line printer.

SYS displays only the system files (certain Radio Shack files). If you don't use the SYS option, TRSDOS-16 displays only the user files.



What the column headings mean:

1 Disk Name -- The name assigned to the disk when it was formatted or backed-up.

2 File Name -- The name and extension assigned to a file when it was created. (Insert A)

3 Creation Date -- When the file was created.

4 Update -- When the file was last modified.

5 Attributes -- A four-character field.

The first character is either P for Program file or D for Data file.

The second character is either S for System file or * for User file.

The third character gives the password protection status.

- X The file is unprotected (no passwords)
- A The file has an access word but no update word.
- U The file has an update word but no access word.
- B The file has both update and access words.

The fourth character specifies the level of access assigned tot he access word:

- Ø,1 Kill file and everything listed below.
- 2 Rename file and everything listed below.
- 3 Not used.
- 4 Write and everything listed below.
- 5 Read and everything listed below.
- 6 Execute only.
- 7 None.

The ATTRIB command explains how to change the access password, update password, and protection level.

6 File Type -- Indicates the record type for the file.

F Fixed-length records.

V Variable-length records.

7 Record Length -- Assigned when the file was created (applies to fixed-length record files only.)

8 Number of Records -- How many logical records have been written. Plus signs (+) signify none have been written or file has variable length records and number written cannot be calculated. True number of records can be inferred from Sectors Used column.

9 Sectors Allocated -- How many sectors (256 byte blocks) have been allocated to the file.
1Ø Sectors Used -- Shows how many sectors have data written into them. Plus sign (+) means no data in file.
11 Files Displayed -- the number of files on the DIRectory listing.

Figure 1. Sample DIRectory Display

Examples

DIR BAST/ASM <ENTER>

lists the directory for the file BAST/ASM on the display.

DIR 3 PRT <ENTER>

lists the directory of the diskette in Drive 3 to the line printer.

DIR B*/* <ENTER>

displays all files in the directory beginning with the letter B and having an extension.

DISMOUNT

DISMOUNT

Tells TRSDOS-16 that you have removed diskettes in the floppy drives.

The DISMOUNT command informs TRSDOS-16 that you have just taken diskettes out of the drives. Once you enter the DISMOUNT commmand and TRSDOS-16 acknowledges, you can insert different diskettes. See MOUNT for details.

Example

DISMOUNT <ENTER>

informs TRSDOS-16 that you have removed diskettes from the floppy drives. TRSDOS-16 acknowledges the DISMOUNT by displaying:

```
INIT DONE
```

Now you can insert other diskettes

DO

DO filespec

Executes a "DO-file" -- a file containing one or more system commands or programs.

You can create this DO-file with the Editor. The example below shows how.

A DO-file cannot include the SPOOL command. You can load and execute programs from a DO-file and chain DO-files together.

Example

This example creates a sample DO-file that prints the current date and amount of free sectors on the disk in the primary drive whenever you execute it.

You need to use the Editor to create this file. Type:

```
EDIT16 <ENTER>
```

to get into the Editor. The C? prompt is displayed, signifying the Editor's command level.

To begin writing your program, type:

```
IN <ENTER>
```

This puts you in the insert mode, with the prompt I?. You can begin entering command lines. Type:

```
DATE <ENTER>
```

```
FREE <ENTER>
```

These are your command lines. Now save this DO-file under the name HELLO. But first, exit the insert mode. Type:

```
! <ENTER>
```

and then, to save the file, type:

```
SA HELLO <ENTER>
```

Exit the Editor by typing:

```
QU <ENTER>
```

and you can run your DO-file, listed on your directory as HELLO. Type:

```
DO HELLO <ENTER>
```

and TRSDOS-16 executes the DO-file named HELLO and prints the date and free list on your display (see the DATE and FREE commands).

DRIVE

DRIVE drive {options}

Allows you to:

1. Gain optimum use of a floppy disk drive by changing the following disk drive settings:
   - seek rate (the rate the computer is able to access the diskette)
   - diskette swap detection
   - wait (for a drive ready condition)
2. Turn secondary floppy or hard disk drives offline.

If you include no options, DRIVE returns the current settings for the specified drive.

The following information offers a thorough explanation of the DRIVE command and all its options. Please read it before using this command.

The options are:

RATE=n (used for floppy drives only.) Sets the seek rate of the floppy disk drive. n may be:
Ø = 3 milliseconds
1 = 6 milliseconds
2 = 1Ø milliseconds
3 = 15 milliseconds
If omitted, setting is not changed.

DETECT (used for floppy drives only.) Sets the diskette swap detection. This causes TRSDOS-16 to check the drive hardware for a "door opened" condition. Set DETECT for Push-Button and Thinline drives.

NODETECT (used for floppy drives only.) Sets the diskette swap to "no detection". This causes TRSDOS-16 to ignore any "door opened" conditions received from the drive hardware. Set the latch drives for NODETECT.

WAIT (used for floppy drives only.) Sets TRSDOS-16 to wait for the drive to gain proper motor speed if a "Drive Not Ready" error occurs, then try again. If the error occurs again, then the drive is considered not ready and an error code is generated. Set WAIT for Thinline drives.

NOWAIT (used for floppy drives only.) Sets TRSDOS-16 to not wait if a "Drive Not Ready" error occurs. Generate error code immediately. Set Push-Button and Latch Drives to NOWAIT.

OFFLINE (all secondary drives) Sets a drive offline. TRSDOS-16 ignores that drive entirely.

ONLINE (all secondary drives) Sets a drive online.

1. GAINING OPTIMUM USE OF FLOPPY DISK DRIVES

When TRSDOS-16 starts up, it initializes each of your floppy drives to the following seek, swap, and wait/nowait settings:

DRIVE	SEEK RATE	SWAP DETECT	WAIT/NOWAIT STATUS
Ø	1Ø ms	DETECT	WAIT
1 - 3	15 ms	NODETECT	WAIT

Any type of Model 16/Model II floppy drive can operate under these settings. However, to get the optimum use out of your particular drive, we suggest you try different settings.

There are three types of drives that could be on your Model 16 or Enhanced Model II computer. Each type of drive has its own set of specifications that determines how it can be set-up.

The three types of drives are:

Push-Button	used as Drive Ø in most Model II's and as the secondary drives in some Model II Expansion Bays
Latch	used as the secondary drives in some Model II Expansion Bays
Thinline	used in the Model 16

We suggest you try the following settings for each of these drives.

Drive	Minimum Rate	Swap Detect	Wait / Nowait
Push-Button	1Ø ms	DETECT	NOWAIT
Latch	15 ms*	NODETECT*	NOWAIT
Thinline	3 ms	DETECT	WAIT*

* These settings are required for these particular drives and are set this way at start-up.

You can determine the type of drive you have by looking at the pictures below.

Push-Button Latch Thinline

Figure 2. The three types of drives

When using the DRIVE command with the seek rate, swap detect, and wait options, be sure to note the following:

When reset, TRSDOS-16 always returns to the start-up settings. Use the AUTO command (or a DO file) to implement the DRIVE command automatically upon power-up or reset.

If you receive numerous I/O errors on disk reads/writes after changing the seek rate, you probably set it too fast for that particular drive. To remedy this, either issue the DRIVE command again with the proper seek rate or reset the computer.

Latch drives cannot properly detect if a drive door has been opened since the last disk access. Always set Latch drives with the NODETECT option.

- Thinline drives have a built in feature to reduce the wear on the floppy diskette. If a Thinline drive is not accessed for 2Ø seconds or more, the drive motor shuts off until the next drive access. At the next disk access, it takes approximately 8/1Ø of a second for the motor to reach proper speed.

- Always set Thinline Drives with the WAIT option. If a Thinline Drive is run with the NOWAIT option, a "Drive Not Ready" error will occur since the motor could not reach proper speed before the access.

Examples

If your Drive Ø is a Thinline drive, this command:

```
DRIVE Ø {RATE=Ø,DETECT,WAIT} <ENTER>
```

allows you to get the optimum use out of Drive Ø.

If your drive is Push-Button, this command:

```
DRIVE 1 {RATE=2,DETECT,NOWAIT} <ENTER>
```

allows you to get the optimum use out of Drive .

If your drive is a Latch, this command:

```
DRIVE 1 {RATE=3,NODETECT,NOWAIT} <ENTER>
```

allows you to get the optimum use out of Drive 1

2. TURNING THE DRIVES OFFLINE

The OFFLINE option turns a secondary disk drive OFFLIINE; ONLINE turns it back ONLINE. You can use both options with both hard or floppy secondary disk drives:

Floppy Drives -- By turning a non-existing or unused secondary floppy drive OFFLINE, TRSDOS-16 will access your disks much more quickly.
Hard Disk Drives -- If you have more than one hard disk drive, you can MOVE or COPY files to your secondary drive(s), turn these drives OFFLINE, and thus protect your files from access and/or change.

The default is ONLINE. When you turn a drive back ONLINE after it was OFFLINE, you must also use the MOUNT command to reinitialize the drive.

Examples

```
DRIVE 5 {OFFLINE} <ENTER>
```

allows you to protect files on Drive 5 from access.

```
DRIVE 3 {OFFLINE} <ENTER>
```

allows you to tell TRSDOS-16 not to attempt to access Drive 3; this speeds up access time.

```
DRIVE 2 {ONLINE} <ENTER>
```

tells TRSDOS-16 to attempt to access Drive 2.

If you have a Hard Disk, you might find it helpful to create a DO-file (see DO command) containing these commands:

```
DRIVE 6 {ONLINE}
MOVE !:5 TO 6 {ABS}
DRIVE 6 {OFFLINE}
```

This will cause TRSDOS-16 to turn Drive 6 ONLINE, MOVE all the files on Drive 5 to Drive 6 and then turn Drive 6 back OFFLINE.

DUMP

DUMP filespec {options}

Copies filespec, a machine-language program, from memory to disk. You can then load and execute the program at any time.

The options can be any or all of the following:

START=address sets address as the program's starting address.

END=address sets address as the program's ending address.

TRA=address sets address as the transfer address. This is the address where your program begins executing after you load it. If omitted, DUMP uses the address set by RELO. The transfer address must be less than the ending address.

RELO=address sets address as the starting address for loading the program back into memory. If omitted, **START** is used.

RORT=c specifies the program as c, directly executable from TRSDOS-16 Ready mode. c can be:
R(eturn) loads filespec, but doesn't execute it.
T(ransfer) loads and executes filespec from TRSDOS-16 Ready. If omitted, **RORT=T** is used.

NOTE: All addresses are 24-bit hexadecimal numbers (up to six digits).

Examples

```
DUMP TEST/FIL START=64FØ, END=6AFØ, TRA=67F2, RORT=R <ENTER>
```

creates filespec TEST/FIL which contains the program in memory location 64FØ hex to 6AFØ hex. When loaded, it occupies the same memory location. Since this specifies RORT=R, you can't execute the program from TRSDOS-16 Ready mode.

```
DUMP INTCOM/DMA START=6ØØØ, END=67FF, TRA=31Ø8, RELO=3ØØØ <ENTER>
```

creates filespec INTCOM/DMA. It contains the program in memory location 6ØØØ hex to 67FF hex. When loaded, the program resides from 3ØØØ to 37FF and execution starts at 31Ø8 hex. You can execute program from the TRSDOS-16 Ready mode by typing:

```
INTCOM/DMA <ENTER>
```

EXEC

EXEC filespec

Executes filespec.

The keyword EXEC is optional.

Examples

EXEC JOBLIST <ENTER>

executes JOBLIST.

NAMEDUMP <ENTER>

executes the program NAMEDUMP.

FCOPY

FCOPY source TO destination {options}

Copies disk files that were created with the Model II TRSDOS Operating System to a disk formatted by TRSDOS-16 (or TRSDOS-II) and vice versa.

You must FCOPY any disk files created with Model II TRSDOS before you can use them with TRSDOS-16 or TRSDOS-II. You will get the error -- Illegal I/O Attempt -- if you attempt to use a Model II diskette while operating under TRSDOS-16.

source can be a filespec, wildcard, or drive that you want to copy.

destination can be the drive number that you are FCOPYing to or the filespec you are FCOPYing to. destination can be a filespec only if source is a filespec.

options can be one of the following:

ABS tells TRSDOS-16 to overwrite any data that already exists on the disk

PROMPT tells TRSDOS-16 to prompt you before it copies a file. You should press <Y> yes, <N> no, <Q> quit, or <S> stop asking for prompting.

ALL tells TRSDOS-16 to copy all files. (ALL won't transfer system files, use SYS.) If you use

drive as source, you must use **ALL**.

SYS allows you to FCOPY language and applicaton programs. If used, destination must be primary drive.

In addition to using FCOPY to copy disk files and change their format, you can use FCOPY to get a DIRectory listing of your Model II TRSDOS files while you are in the control of TRSDOS-16.

The syntax is:

FCOPY drive {DIR,SYS,PRT

When you use the FCOPY command with DIR and SYS, TRSDOS-16 will return the DIRectory according to the way the diskette in drive is formatted. For example, if you enter the command:

FCOPY 1 {DIR,SYS} <ENTER>

and the diskette in Drive 1 is in TRSDOS 2.ØA format, it will return a directory consisting of both System and User files.

On the other hand, if you use the same command on a diskette formatted under TRSDOS-II, it will return a DIRectory of only the System files.

Examples

FCOPY NEWFILE/TXT TO :3 <ENTER>

copies NEWFILE/TXT (contained on a Model II TRSDOS formatted diskette) to the diskette in Drive 3 (a TRSDOS-16 formatted diskette).

FCOPY 1 TO Ø {ALL} <ENTER>

copies all files on the TRSDOS-formatted diskette in Drive 1 to the TRSDOS-16 formatted diskette in Drive Ø.

FCOPY ! TO 1 <ENTER>

copies all files, with and without extensions, from the disk in the primary drive to Drive 1. (See "Wildcards" in Section 2.)

```
FCOPY TRN/TXT:1 TO TRNTXT/OLD:4 <ENTER>
```

copies the file TRN/TXT on the diskette in Drive 1 to the disk in Drive 4, renaming the file TRNTXT/OLD as it copies and reformats.

```
FCOPY 2 {DIR} <ENTER>
```

lists the DIRectory of the Model II TRSDOS diskette in Drive 2.

FILES

FILES source {options}

Returns an alphabetical listing of the filenames that are stored on the specified source.

source can be a standard TRSDOS-16 filespec, a wildcard, or a drive number (Ø-7). If drive number is omitted, TRSDOS-16 goes to the primary drive.

The options are:

SYSlists all system files. SYS is optional; if omitted, TRSDOS-16 lists only the user files.

PRTtells TRSDOS-16 to print the files. PRT is optional; if omitted, lists files on the video display.

This command will list filenames that are stored on the specified drive. This is not the same as DIRectory because only filenames are listed with FILES. FILES lists the filenames alphabetically in five columns (from left to right) across the screen.

FILES allows full wildcarding. For details, see Section I of this manual.

Example

```
FILES */BAS:4 {PRT} <ENTER>
```

lists all files with the extension /BAS on Drive 4 to the printer.

```
FILES 0 {SYS} <ENTER>
```

lists all System files on Drive 0. The System directory is on Drive 0.

FLOPPY

FLOPPY (options)

Tells TRSDOS-16 to ignore all references to floppy drive numbers within filespecs. This is useful when a program includes a reference to a file specification where a drive number is included.

options are:

ON	sets FLOPPY ON. TRSDOS-16 does not ignore references to drive number within filespecs
OFF	sets FLOPPY OFF. TRSDOS-16 ignores any references to drive number within filespecs

If you do not specify option, TRSDOS-16 displays the current status of FLOPPY.

When you enter FLOPPY OFF, TRSDOS-16 ignores any reference to a floppy drive number (0-3) within a system command or program and follows the normal file search sequence (going to the primary drive first).

For example, assume a COBOL program references a file named SAMPLE:2. By turning FLOPPY OFF, TRSDOS-16 will treat this reference as simply SAMPLE (ignoring the drive reference) and look for it on the primary drive first.

The system commands which require a filespec and are affected by FLOPPY are:

ATTRIB	DO	LOAD
APPEND	DUMP	MOVE
COPY	KILL	OPEN
CREATE	LIST	RENAME

Examples

FLOPPY {OFF} <ENTER>

sets FLOPPY OFF so that TRSDOS-16 ignores any references to drive numbers within a filespec entered with a system command or program.

FLOPPY {ON} <ENTER>

turns FLOPPY ON so that TRSDOS-16 uses the drive number referenced in any filespec.

FLOPPY <ENTER>

returns the status of FLOPPY.

FORMAT

FORMAT drive {options}

Prepares a blank disk for use by defining the tracks and sectors, and writing system information onto it. (For more information, see TECHNICAL INFORMATION.)

drive specifies the drive to use for the format operation and can be Drives Ø - 3 and 5 - 7. If omitted, TRSDOS-16 prompts you to enter the drive number.

The options are:

ABS tells TRSDOS-16 to overwrite existing data without prompting. If omitted, TRSDOS-16 warns you before overwriting a disk that contains data.

ID=id assigns a name to the disk being formatted. If omitted, TRSDOS is used.

PW=pw assigns the master password to the disk. If omitted, PASSWORD is used. The master password allows access to all user files (via the PROT command), and also allows full BACKUP privileges.

DIR=nn places the primary directory on cylinder nn. If omitted, TRSDOS-16 uses cylinder 44, (single-sided and double-sided floppy); or

cylinder 13Ø (hard disk). You can put the primary directory on any cylinder from 1 - 71 (single-sided or double-sided floppy); or 1 - 253 (hard disk).

ALT=nn places the alternate directory on cylinder nn, which is a backup of the primary directory. If ALT=ØØ, no alternate directory is created. If you omit the ALT option, it will be at the location of the primary directory plus 3 cylinders (single-sided and double-sided floppy, or hard disk). You can put the alternate directory on cylinders 1 - 71 (single- or double-sided floppy); or 1 - 253 (hard disk).

SIZ=nn tells TRSDOS-16 to allow nn filenames in the initial directory. For hard disks and floppy diskettes, nn can be any number between 1-122Ø. If omitted, the default is 18Ø (single- or double-sided floppy); or 336 (hard disks).

FULL/NONE is the verification level. FULL instructs TRSDOS-16 to read each sector and compare the value against what was written during initialization. NONE instructs TRSDOS-16 not to perform verification.

The disk you format can be blank or already formatted. If it is already formatted, all information is lost when you reformat the disk.

Examples

```
FORMAT 1 {ID=ACCOUNTS,PW=IRS} <ENTER>
```

formats the diskette in Drive 1 and names the diskette ACCOUNTS with the password IRS.

```
FORMAT <ENTER>
```

prompts you for the drive to use before it begins to format. Since no options are used, the diskette will have the name TRSDOS, the password PASSWORD and all the other option defaults.

```
FORMAT 2 {DIR=Ø1,ALT=Ø5,SIZ=36Ø} <ENTER>
```

formats the diskette in Drive 2, puts the primary directory on cylinder 1, the alternate directory on cylinder 2 and sets the number of directory records to 36Ø.

WHEN TO FORMAT

To prepare a new disk
Before you can use a new diskette, you must format it (unless you use the BACKUP command with a floppy diskette). After formatting, record the disk name, date of creation and password in a safe place. This helps you estimate how long a disk has been in use, and prevents your forgetting the master password. (For this application, always use the FULL verify option.)

To erase all data from a disk
To "start over" with a disk, you can format it. This erases all old information on the disk and puts the system information back on it.

To lock out flawed areas
After prolonged use, flaws may develop on a diskette. If you reformat the disk, it locks out these flawed sectors while leaving the good sectors available for data storage. Use the FULL verify option for this application.

FORMS

FORMS {format options}
FORMS {switch options}

Sets up the printer parameters.

The {format options} are:

P=n sets n as number of lines per page. n can be any number between Ø and 255. If omitted, it is 66.

L=n sets n as the maximum number of lines to print on a page before issuing an automatic top of form. n can be Ø to 255 and, if omitted, L=6Ø. The number of lines must be less than or equal to the page length. If either page length or lines

is Ø, both must be Ø. If L=Ø, TRSDOS-16 doesn't issue an automatic top of form.

W=n sets n as the maximum number of characters to print on a line before issuing an automatic carriage return. n can be any number between Ø and 255. If omitted, W=132.
If W=Ø, TRSDOS-16 doesn't issue any automatic carriage returns.

C=h sets the output to h, a one-byte control code in hexadecimal, to the printer. It is sent on completion of FORMS command.

The default parameters are P=66, L=6Ø, W=132 and C=Ø. If you want to use the default parameters, you don't need to issue the FORMS command.

To determine the parameters to set for:

page size	multiply form length in inches by the number of lines per inch.
lines per page	determimes the number of blank lines at the bottom of every page. If page length equals lines, then every line on the page is printed. lines can't be greater than page length. width sets the maximum number of printable characters per line. If a line is greater than width, then TRSDOS-16 automatically breaks the line at the maximum length and continues printing at the next print line.
control codes	are required on some printers, e.g. to set up for double space character, etc. The TRSDOS-16 sends the specified code to the printer or print file during execution of the FORMS command.

The {switch options} are:

X sends all data to printer or printer file without any translation (transparent mode).

D ignores all printer output ("dummy" mode).

N returns to normal (nontransparent, non-dummy) mode. This is the default mode.

A outputs line feed after carriage return (auto line feed mode) even if transparent mode is in effect. Updates count by carriage returns, not by line feeds.

T sends top of form character to printer.
Q cancels auto line feed mode.
S switches to serial Channel B printer driver. (You must do SETCOM before any printing can be done.)
R returns to parallel printer driver.

See the PRCTRL Supervisor Call in the Technical Information Section for information on transparent, dummy modes, etc.

Examples

```
FORMS <ENTER>
```

resets all FORMS parameters to the default values.

```
FORMS P=51, L=46, W=92 <ENTER>
```

sets page length to 51, printed lines per page to 46 and characters per line to 92.

```
FORMS D <ENTER>
```

invokes the dummy mode. This means TRSDOS-16 will ignore all printer commands.

FREE

FREE drive {option}

Returns a list of the disk's free sectors.

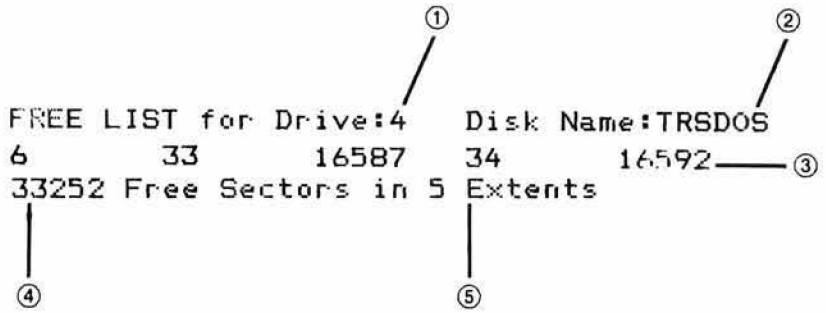
drive is optional; if omitted, it defaults to the primary drive.

The option is:
PRT prints the listing on the Printer. If omitted, TRSDOS-16 automatically displays it on the video display.

This information is useful to optimize file access time. If you use a disk extensively (files updated, killed, extended, etc.), the files often become fragmented. This means that the file may be put in different parts of the

disk's memory -- extents. When this happens, the access time is considerably increased because the disk read/write mechanism must move back and forth across the disk to read or write to a file.

FREE helps you determine the extent that your disk files are fragmented. Once you determine this and you decide that you'd like to re-organize a particular file to allow faster access, you can COPY or MOVE it to a "clean" diskette.



1 Drive Number
2 Disk Name
3 Number of FREE sectors in each extent
4 Total number of free sectors
5 Number of FREE extents (an extent is an area on the disk)

Figure 3. FREE List

Example

FREE <ENTER>

displays the amount of free space on the disk in the primary drive.

FREE {PRT} <ENTER>

lists the amount of free space for the primary drive to the printer. Because no drive specification is included in this example, you must use the braces, { }

FREE 2 PRT <ENTER>

lists the free space for Drive 2 to the printer.

HELP

HELP command

Displays the syntax of a TRSDOS-16 command. command is optional; if you omit it or type an unrecognized command, TRSDOS-16 displays the TRSDOS-II and TRSDOS-16 commands and general subjects for which HELP is available.

Example

HELP MOVE <ENTER>

displays the syntax for the MOVE command.

HELP SYNTAX <ENTER>

returns an explanation of the format of the HELP messages.

KILL

KILL filespec

Deletes filespec from the directory and frees the space allocated to it.

Before it deletes the file, TRSDOS-16 displays the complete filespec and prompts you with the options "Delete? (Y/N/Q)..", i.e, Yes, No, or Quit.

Example

KILL DATAFILE/OLD <ENTER>

deletes DATAFILE/OLD from the directory and frees all space allocated to it.

LIB

LIB

Displays a listing of all system commands.

Example

LIB <ENTER>

LIST

LIST filespec {options}

Lists the contents of filespec. This listing shows the hexadecimal contents and the ASCII characters corresponding to each value. For values outside the range of hexadecimal 2Ø to 7F, TRSDOS-16 displays a period.

The options are:

- **PRT** lists filespec to the printer. If omitted, the listing automatically goes to the screen.
- **SLOW** tells TRSDOS-16 to pause after each record. If omitted, the listing is continuous.
- **R=n** sets the starting record to n. The range for n is 1 to 65,535. If omitted, 1 is used. (See TECHNICAL INFORMATION for details.)
- **A** tells TRSDOS-16 to list only the ASCII characters.

Examples

```
LIST DATA/BAS <ENTER>
```

lists the contents of DATA/BAS.

```
LIST TEXTFILE/1 SLOW <ENTER>
```

lists the contents of TEXTFILE/1, pausing after each record.

```
LIST TEXTFILE/1 R=1ØØ, A <ENTER>
```

lists TEXTFILE/1 starting with the 1ØØth record in it. Only ASCII characters are displayed.

```
LIST PROGRAM/CMD PRT <ENTER>
```

lists PROGRAM/CMD to the printer.

LOAD

LOAD filespec

Loads a machine language program named filespec and then returns to the TRSDOS-16 mode.

Example

```
LOAD MARKET/OBJ <ENTER>
```

loads the machine language program file named MARKET/OBJ into memory.

MOUNT

MOUNT

Tells TRSDOS-16 that you have inserted different diskettes in the floppy disk drives.

Whenever you swap diskettes, you must tell TRSDOS-16 that you have done so -- both before removing the old diskettes and after inserting the new diskettes. (Be sure not to remove a diskette when a file is open.)

See the DISMOUNT system command for instructions on removing diskettes.

Example

```
MOUNT <ENTER>
```

informs TRSDOS-16 that you have inserted different diskettes in the floppy drives.

TRSDOS-16 will acknowledge the MOUNT by displaying:

```
INIT DONE
```

MOVE

MOVE source TO destination {options}

Copies single or multiple user files to the destination disk.

source can be a filespec, a wildcard (* or !), or simply a drive number. However, you cannot move password protected files.

When you specify only drive as the source, TRSDOS-16 moves all user files on the disk in that drive to the destination drive.

destination is the drive number of the disk where you want your file(s) moved. Your MOVEd files will retain the same name as the source filespec.

options are:

ABS instructs TRSDOS-16 to overwrite any existing files on the destination disk that have the same name.

PROMPT tells TRSDOS-16 to display each file before moving it and to give you a set of options for

that file. The PROMPT options are: Y/N/S/Q (Yes--Copy;No--Don't Copy;Stop prompts and proceed with all copies;Quit this command--no more copies.) If PROMPT is omitted, TRSDOS-16 moves all files that match the wildcard specification.

ALL tells TRSDOS-16 to move all user files. This parameter is required when you move all files (except the system files) on a disk.

MOVE is useful when you want to copy all of your TRSDOS-16 files from floppy diskette to hard disk, or vice versa. Because it moves only the user files (those files which you create), you need to use the COPY command to move your system files (your Radio Shack files, such as the Editor, or the Assembler).

For example, type:

```
MOVE Ø TO 4 {ALL} <ENTER>
```

to move all your user files from the diskette in Drive Ø to hard disk.

Then, use the COPY command to copy each of your system files to your hard disk.

Examples

```
MOVE DAT/FLE:1 TO 3 <ENTER>
```

moves the file DAT/FLE on the diskette in Drive 1 to the diskette in Drive 3, keeping the filename DAT/FLE.

```
MOVE */PAY TO 2 <ENTER>
```

copies all user files with the extension /PAY on the disk in the primary drive to the diskette in Drive 2

```
MOVE ! TO 3 <ENTER>
```

moves all user files, with and without extensions, on the primary drive to the diskette in Drive 3.

MSG

MSG "message"

Prints message on the screen. You must enclose the message in quotes.

This command is especially useful in a DO-file. (See DO for more information.)

Example

MSG "THIS PROGRAM REQUIRES ONE DATA DISKETTE" <ENTER>

prints THIS PROGRAM REQUIRES ONE DATA DISKETTE on the screen.

PATCH

PATCH filespec (options)

Allows you to make minor corrections in any disk file, provided that:

1. You know the existing contents and location of the data you want to change.
2. You want to replace one string of code or data with another string of the same length.
3. The file is a fixed-length record (FLR) file.

filespec indicates the file you want to change. If it is a system file, no password is necessary. If it is a protected user file, you must include the password

The options are:

A=aaaa indicates the starting address of the data to be changed. This is where the data resides in memory when the program is loaded. aaaa is a four-digit hexadecimal value without the X'' notation. (The A option will not work on 68ØØØ programs. Use the R and B options instead.)

F=findstring indicates the string that is currently in the patch area.

C=changestring indicates what data will replace findstring. changestring must contain the same number of bytes as findstring

R=record tells which record contains the data to be changed, and is a decimal number from 1 to 65536.

B=starting byte specifies the position of the first byte to be changed. It is a decimal number from 1 to 256.

You can use PATCH to make minor changes to your own machine-language programs; you won't have to change the source code, reassemble it, and recreate the file. You can also use it to make minor replacement changes in data files.

PATCH also allows you to implement any modification to TRSDOS-16 that may be supplied by Radio Shack. This way, you do not have to wait for a later release of the operating system.

NOTE: If you press <BREAK> during a PATCH operation, before any changes have been made in the file, PATCH will close the file and return to TRSDOS-16. The file will be unchanged. Once the PATCH process begins, <BREAK> will have no effect.

USING PATCH ON A TRSDOS-16 SYSTEM FILE

When Radio Shack releases a modification to TRSDOS-16, you will receive a printout of the exact PATCH commands that you must enter to perform the change.

To implement such a change, follow these steps:

1. Make a backup copy of the diskette to be patched.
2. Insert the TRSDOS-16 diskette to be changed into one of the drives. (Make sure the diskette is "non-write-protected.)
3. In the TRSDOS-16 Ready mode, type in the specified PATCH command.
4. After the PATCH is complete, test the diskette in Drive Ø to see that it is operating as a TRSDOS-16 system diskette. You will have to reset the Computer.

USING PATCH ON A Z-8Ø PROGRAM FILE

In this context, "program files" refers strictly to those files stored with the "P" attribute. Use the DIR command to find out the attributes of a file. BASIC programs have the "D", not the "P", attribute. (See instrucitons for changing data files.) Program files are created with DUMP.

If you want to change four bytes in a machine-language program file, you must first determine where the four-byte sequence resides in RAM when the program is loaded. Next make sure that your replacement string is the same length as that of the original string. For example, you might write down the information as follows:

Files to be changed: VDREAD
Start address: H'528Ø'
Sequence of code to be changed: H'CD2C25E5'
Replacement code: H'ØØØØØØC9'

Then you could use the following command:

```
PATCH VDREAD A=528Ø,F=CD2C25E5,C=ØØØØØØC9
```

USING PATCH ON 68ØØØ PROGRAM FILES AND DATA FILES (INCLUDING BASIC AND COBOL PROGRAMS)

If you want to patch a 68ØØØ program file or if you have a file stored with the "D" attribute, you must specify the patch area in terms of the logical record which contains the data, and the starting byte of the data record. (The TRSDOS-16 LIST command gives this information.)

For example, if you want to change a 12-byte sequence in a file called NAMEFILE, use the LIST command to find the location of the sequence. If it is in Record 128, starting at byte 14, write down the information like this:

File to be change: NAMEFILE
Record number: 128
Starting byte: 14
Sequence of text to be changed: "JOHN'S DINER"
Replacement text: "JACK'S PLACE"

Then use the following command to patch a data file:

```
PATCH NAMEFILE R=128,B=14,F="JOHN'S DINER",C="JACK'S PLACE"
```

For data files, notice that either string can include a single-quote, as long as the string is surrounded by double-quotes. If you want to include a double-quote inside either string, you would have to enclose that string in single-quotes.

Use this command to patch a 68ØØØ program file:

```
PATCH BUDGET/PRO R=24,B=8,F=FDCBØØ,C=C3812Ø
```

NOTE: The string you change must be entirely contained inside the specified record. If it spans two records, you will have to perform the patch operation twice, once for each record.

ERROR CONDITIONS

If a TRSDOS-16 error occurs during the patch operation, you will receive the appropriate error message, and the patch will be terminated without changing the file.

PATCH can also produce the following messages

PATCH STRING TOO LONG -- ABORT This occurs when you patch a data file and the patch string spans two records. You need to perform the patch in two steps, one for each record that contains a part of the string to be changed.

FILE CONTAINS VARIABLE-LENGTH RECORDS -- ABORT You can only patch fixed length record files.

STRING NOT FOUND The findstring was not found at the patch location you specified. Before patching a file, you must know the exact patch location and the existing contents of that location

ADDRESS OUT OF PROGRAM-LOAD RANGE -- ABORT This occurs when you attempt to patch a program file and some or all of the patch string is outside the RAM area where the program resides when it is loaded. Check the A=aaaa parameter. Also be sure that the findstring and changestring are not longer than you intend them to be.

PAUSE

PAUSE prompt message

Prints prompt message and then waits for the operator to press <ENTER>. (The prompt message is optional.)

Like MSG, this command is especially useful in a DO-file. (See DO for more information.)

Example

```
PAUSE INSERT DISKETTE "SALESRPT" INTO DRIVE 2 <ENTER>
```

prints PAUSE INSERT DISKETTE "SALESRPT" INTO DRIVE 2, Press any key to continue, then pauses until the operator presses a key.

PRINT

PRINT filespec (options)

Prints out contents of filespec, omitting the record numbers and hexadecimal codes (LIST does that). filespec must be a text file.

The options are:

A causes TRSDOS-16 to treat the first byte in each record as a FORMS control character. The meaning of the character in the first byte is:

- "1" do a form feed before printing (top of form).
- "b" do a carriage return before printing (single space)
- "Ø" perform two carriage returns before printing (double space).
- "+" perform a carriage return without a line feed advance. If current printer can do a carriage return without a line feed, this control code causes the characters following to be over printed on the current line.

V causes TRSDOS-16 to output the filespec to the video display, as well as to the printer.

NOTE: Use the A option only when filespec contains the control codes listed.

Example

```
PRINT PROGRAM/TXT V <ENTER>
```

outputs PROGRAM/TXT to the video display, as well as to the printer.

PROT

PROT drive {options}

Changes password protection of the disk in the drive on a large scale.

The options can be any of the following:

OLD=password specifies the disk's current master password. You cannot use any of the remaining options without specifying this.

NEW=password assigns the TRSDOS-16 disk the new master password.

LOCK tells TRSDOS-16 to assign all user files the latest master password. Both update and access words are then set to this password. (See ATTRIB for information on access and update passwords.)

UNLOCK tells TRSDOS-16 to remove passwords from all user files.

A disk's master password is initially assigned during the format or backup process. The Model 16 systems disk is supplied with the master password PASSWORD.

Example

```
PROT 1 OLD=PASSWORD, NEW=H2Ø <ENTER>
```

changes the master password of the diskette in Drive 1 from PASSWORD to H2O.

```
PROT Ø OLD=H2O, UNLOCK <ENTER>
```

removes passwords from every user file on the diskette in Drive Ø.

```
PROT Ø OLD=H2O, NEW=ELEPHANT, LOCK <ENTER>
```

changes the master password from H2O to ELEPHANT and assigns the new password to every user file.

PURGE

PURGE drive {options}

Quickly deletes files from the disk in the specified drive number.

drive is optional. If omitted, the primary drive is used.

The options are:

SYS	System files (program and data)
PROG	User machine-language program files.
DATA	User data files.
ALL	All files, user and system.

If the options are omitted, TRSDOS-16 only allows you to PURGE data files.

Once you enter the PURGE command, TRSDOS-16 prompts you for the disk's password. Type in up to eight characters and press <ENTER>. (All disks distributed by Radio Shack use the password PASSWORD.)

The system then displays the file names, one at a time, prompting you to kill one file at a time, keep the file, or quit the operation.

Example

```
PURGE 1 <ENTER>
```

allows you to delete data files from Drive 1

PURGE 2 {PROG} <ENTER>

allows you to delete user machine-language program files from the diskette in Drive 2.

RENAME

RENAME filespec1 TO filespec2

Renames filespec1 to filespec2.

RENAME alters only the filename and extension, not the contents or the physical positon of the file on the disk. The file's password also remains the same. (See ATTRIB to change the password.)

Example

RENAME DATA/FLE TO DATFILØ1 <ENTER>

renames DATA/FLE to DATFILØ1

RESET

RESET

Resets/restarts TRSDOS-16.

This command is almost the same as using the RESET switch. The RESET command closes all open files if you are at the TRSDOS-16 Ready prompt.

Example

RESET <ENTER>

RESTORE

RESTORE source TO destination (options)

Recovers any files stored on floppy diskettes that were saved with the SAVE command. Because SAVE stores files in a special format, RESTORE is the only way to return these files to the hard disk drive.

source specifies a floppy diskette and is one of the following:

- drive specifies a drive number between Ø-3.
- filespec:drive where filespec is a TRSDOS file specification and drive specifies a drive number between Ø-3.
- wildcard:drive where wildcard is a standard TRSDOS-16 wildcard and drive specifies a drive number between Ø and 3.

destination is optional, but may be one of the following:

- drive specifies a drive number between Ø-7, but may not be the same as source.
- filespec:drive if {options} is {IND}. If {DIR} is specified in source, destination cannot be specified.

If omitted, destination is first available hard disk drive.

The options and their meanings are:

- **ABS** tells TRSDOS-16 to retrieve the specified file(s). If used, TRSDOS-16 overwrites the already existing file with the same name.
- **DIR** If VOLUME Ø is in source drive, TRSDOS-16 will display the DATASET directory and identifier; if VOLUME Ø is not a source drive, TRSDOS-16 will display only the DATASET identifier.
- **IND** (indirect) tells TRSDOS-16 to use the contents of the destination file as a list of destination filespecs that meet the requirements stated above.
- **KILL** deletes the specified destination file before it is opened for RESTOREing.
- **PROMPT** asks for verification of each file for RESTOREing. Press <Y> (yes), <N> (no), <Q> (quit

restoring), or <S> (stop prompt).

PRT can only be used with the DIR option. Prints the DIRectory listing on the line printer

SYS specifies that all System files will be retrieved. This includes System (language) and Applications programs. If used with **DIR, SYS** will list the directory of System files.

ALL tells TRSDOS-16 to restore all files. (ALL won't transfer system files, use SYS.) If you use drive as source, you must use ALL.

RESTORE reads information from a DATASET created by SAVE. If a VOLUME of this DATASET is entered out of sequence, TRSDOS-16 informs you of the mistake. The System also informs you if a VOLUME from a different DATASET is accidentally entered during a RESTORE. (See SAVE for explanation of DATASET and VOLUME.)

When you're RESTOREing files in a DATASET, TRSDOS-16 prompts you with:

Mount NEXT Diskette in Drive n -- Press ANY Key to continue.

which instructs you to enter the next VOLUME of the DATASET.

Example

RESTORE Ø {ALL} <ENTER>

Restores all SAVEd user files on Drive Ø to the first available hard disk drive.

RESTORE !:2 TO 4 <ENTER>

recovers files from the floppy diskette in Drive 2 and puts them on hard disk Drive 4.

RESTORE 1 PROGRAMS {IND} <ENTER>

where PROGRAMS is an INDirect file containing the files:

```
MAILIST/PRG:4
MAILDAT/TXT:4
CHANGES/TXT:4
```

recovers the files from the floppy diskette in Drive 1, to the filespescs defined in PROGRAMS on Hard Disk Drive 4. Note that "TO" is optional

```
RESTORE */SRC:Ø 4 <ENTER>
```

Restores all user files SAVEd with the extension /SRC on Drive Ø to Hard Disk Drive 4 using the same file names.

SAVE

SAVE source TO destination {options}

Creates a serial file-by-file backup of source onto destination. Normally, you'll want to use the SAVE command to backup your hard disk files onto floppy diskette. This backup will be a compact form which consumes approximately half the space that it would be on a standard floppy diskette.

This gives you a floppy diskette copy of your files that can easily be carried to another hard disk system. It can also be used to create a "safe" copy of important files.

The only way to retrieve a file in this "compact" format is with the RESTORE command. Any attempt to access a SAVEd diskette using a TRSDOS-16 command will cause the System to appear "locked-up" for a short period of time while TRSDOS-16 attempts to read the SAVEd diskette.

source can be one of the following:

- drive specifies a drive number between Ø-7, but may not be the same as destination (ALL must be specified).
- filespec:drive if {options} is {IND}.
- wildcard:drive is a TRSDOS-16 wildcard and includes drive number (Ø-7).

destination specifies a floppy diskette and is one of the following:

- drive specifies a drive number between Ø-3.

options and their meanings are:

ABS tells SAVE not to prompt for destination diskette status. It formats the destination diskette if it isn't already in SAVE format.

DC value date compares the creation date of each specified source file against the date entered and SAVEs the file if all other criteria are met.

DM value date uses the last modification date in the manner specified above.

IND (indirect) tells SAVE to use the contents of the source file as a list of source filespecs that meet the requirements stated above.

PROMPT asks for a file verification before SAVEing. You may respond with <Y> (yes), <N> (no), Q (quit) or <S> (stop prompting and continue).

ALL tells TRSDOS-16 to save all files. (ALL won't transfer system files, use SYS.) If you use drive as source, you must use ALL.

SYS allows you to SAVE language and application programs.

Note: value is <, >, or = where < (less than) and > (greater than) mean less than or equal to and greater than or equal to.
date must be in the form: MMDDYY

When the ABS option is used with SAVE, TRSDOS-16 will write over any diskette. If ABS is not used, you will be prompted first.

SAVEing Multiple Diskettes

Since the hard disk drive is a larger storage system than the floppy diskette, it is sometimes necessary for SAVE to store information on more than one diskette. In these cases, SAVE prompts for the insertion of a new diskette.

There are two terms relative to SAVE which you need to be familiar with:

DATASET A set of one or more diskettes created by SAVE.

VOLUME An individual diskette that is a member of a DATASET.

TRSDOS-16 numbers the VOLUMEs sequentially from Ø. Each DATASET contains a unique identifier so each SAVE VOLUME is identified by its serial position in a specific DATASET.

This prevents the accidental mixing of DATASETS within each other.

If a SAVEd file requires more than one floppy diskette, the DATASET identifier enables you to keep track of diskettes in the same VOLUME. For instance, DATASET identifier 84 4E 56 may include VOLUMEs Ø, 1, and 2.

When you are SAVEing files that require more than one Volume, TRSDOS-II prompts with:

```
Insert NEXT Blank Diskette on Drive n --
Press ANY Key to Continue.
```

When you do this, TRSDOS-16 then prompts with:

```
The Diskette Presently on Drive n
will be referred to as "VOLUME 1"
```

When all files have been SAVEd, TRSDOS-16 then prompts:

```
Insert "VOLUME Ø" on Drive 1 --
Press ANY Key to Continue
```

When you re-insert VOLUME Ø, TRSDOS-16 then writes it housekeeping information to this diskette. This allows it to record the number of volumes in the DATASET, etc. for use when it RESTOREs the SAVEd files.

Examples

There are a variety of ways to use SAVE. The simplest of these is:

```
SAVE !:4 TO 2 <ENTER>
```

This simply copies all the files on hard disk Drive 4 in a compact form onto the diskette in Drive 2.

WILDCARDING

Wildcards also offer a simplified method of saving files (these can be several files, or an entire disk). For example:

```
SAVE */CBL:4 TO Ø <ENTER>
```

SAVEs all files with the extension /CBL from Drive 4 to the diskette in Drive Ø.

USING THE INDIRECT OPTION

The INDirect option allows you to save groups of files by creating an INDirect file (a file consisting of one or more filespecs -- similar to a DO-file). The only way to do this is to create a BUILD file under the TRSDOS-II operating system. (For complete details on BUILD, see your Model II Owner's manual.)

Reset your computer to start-up under TRSDOS-II and when TRSDOS-II Ready is displayed, type:

```
BUILD PROGRAMS <ENTER>
```

TRSDOS-II then will prompt you to enter the command line. To do so, type in the names of the files you wish to store. For example, type:

```
ORDERS:5 <ENTER>
REPORTS/*:6 <ENTER>
```

and press <BREAK> to return to TRSDOS-II Ready.

You are now ready to SAVE your INDirect file from hard disk to the specially formatted floppy diskette. Type:

```
SAVE PROGRAMS:4 TO Ø (IND) <ENTER>
```

Both ORDERS and REPORTS are now found in the file named PROGRAMS on the floppy diskette in Drive Ø.

NOTE: Because the INDirect option allows you to SAVE multiple files from more than one hard disk, there is a chance that you could SAVE more than one file with the same name. The SAVE and RESTORE DIRectory does not specify drive numbers for files, therefore you could possibly some of the duplicate filenames.

For example, if you created an INDirect file consisting of these files:

```
*/FOR:4
*/CBL:4
*/FOR:5
```

there is the chance that there are duplicate filenames on drives 4 and 5. Therefore, before using the INDirect option, we suggest that you examine all the files to be SAVEd. If there are duplicate names, RENAME those files before SAVEing.

USING THE DC AND DM OPTION

Another way to SAVE files is to do so in respect to their creation or modification date. For example, if your directory showed these creation and update dates for your files:

Filename	Created	Updated
MENU/PRG	6/1/81	9/2/81
PRGONE/PRG	6/1/81	8/16/81
PRGTWO/PRG	6/1/81	7/3Ø/81
PRGTHR/PRG	6/1/81	6/16/81
PAYROLL/DAT	9/15/81	1Ø/15/81
CHECKS/DAT	9/15/81	1Ø/15/81
TEST/PRG	1Ø/29/81	1Ø/29/81

```
SAVE */PRG:5 TO Ø.
```

The most efficient way to SAVE these files would be by comparing the file creation date to a specified date. For example, all of the first four files were created on June 1, 1981 (6/1/81). Therefore, type:

```
SAVE */*:5 TO Ø {DC=Ø6Ø181} <ENTER>
```

and the first four files would be SAVEd to the floppy diskette in Drive Ø.

In the same sense, the first four files were modified (updated) on or before September 2, 1981 (9/2/81). Therefore, type:

```
SAVE */PRG:5 TO Ø {DM<Ø91581} <ENTER>
```

and all files modified before the specified date would be SAVEd.

SETCOM

SETCOM {options}

Sets up the A or B channels (on the back panel) for communicating with a remote device, via a modem or hardwire connection.

(If you are not a machine language programmer and want to communicate with a remote device, you need to buy a communications program. The manual that comes with it will explain how to use it. See your Radio Shack store for information.)

SETCOM without any options tells TRSDOS-16 to display the status of both serial channels.

The options are:

A=OFF turns off the A channel's RS-232 Communication settings.

B=OFF turns off the B channel's RS-232 Communication settings.

A=(baud rate,word length,parity,stop bits) sets the A Channel for RS-232 communication.

B=(baud rate,word length,parity,stop bits) sets the B Channel for RS-232 communication.

The RS-232 settings can be the following:

baud rate	11Ø, 15Ø 3ØØ, 6ØØ, 12ØØ, 24ØØ, 48ØØ, 96ØØ. If not specified, 3ØØ is used. (Some programs will not run correctly at speeds higher than 24ØØ baud.)
word length	5, 6, 7, 8. If not specified, 7 is used.
parity	E for even, O for Odd, N for none. If not specified, even is used.
stop bits	1, 2. If not specified, 1 is used.

Every option but the last must be followed by a comma. The options are positional; e.g., the third item in an option list must always specify parity. To use a default value, omit the option and insert only the comma.

To change the settings on a currently active channel, first turn the channel OFF. If the channel is already off when you try to turn if off, you'll get an error message.

Before executing this command, connect the remote device to the A or B channel.

Then, after executing it, you can begin sending and receiving data, using one of these TRSDOS-16 Supervisor Calls. (See the Technical Information Section for details.)

ARCV	Channel A receive
ATX	Channel A transmit
BRCV	Channel B receive
BTX	Channel B transmit
ACTL	Channel A control
BCTL	Channel B control

These system routines are only available when the respective channel has been initialized. See Technical Information for details.

Example

```
SETCOM A=( ) <ENTER>
```

sets up channel A for serial communicaitons, using all the default parameters. System function calls 96 and 97 are available for serial I/O. The status of channel B is unchanged.

```
SETCOM B=(6ØØ, 8,  , 2), A=OFF <ENTER>
```

sets up channel B:

baud rate	6ØØ
word length	8 bits
parity	Even (default)
stop bits	2

and turns channel A OFF.

```
SETCOM A=(12ØØ, 8, Ø,), B=( , , , 2) <ENTER>
```

sets up channels A and B.

	Channel A	Channel B
baud rate	12ØØ	3ØØ (default)
word length	8	7 (default)
parity	Odd	Even (default)
stop bits	1(default)	2

SETCOM <ENTER>

displays the status of both channels.

SETCOM A=OFF, A=( ) <ENTER>

resets channel A to default parameters.

SIZE

SIZE

Returns the amount of user memory currently available.

Example

SIZE <ENTER>

SPOOL

SPOOL {options}

Captures printer output or prints a spool file.
SPOOL increases the efficiency of the system by allowing you to use the system while a print operation is in progress.

The {options} control the spool function. If omitted, the SPOOL status is displayed.

The options are:
ON activates the spooler. You must set this switch

before you can use the other switches.

OFF turns off the spooler and closes the capture- and print-files.

N,F=filespec creates a capture file named filespec

P,F=filespec begins background-printing. filespec is the file to be printed.

K keeps the file after printing it. If omitted, deletes the file after printing it. (TRSDOS-16 won't delete a print-file if the file is closed by a SPOOL S or if a disk error occurs in the print file.)

C=n specifies the number of copies you want. If omitted, one copy is made. n can be any number from 1 to 255.

L=line specifies the line number where printing starts. A line is a sequence of characters terminated by a carriage return. If omitted, printing starts at line one. line may be any number from 1 to 65535.

H halts background-printing but saves the current position for later resumption (R switch).

R,L=line resumes background-printing after a halt (H switch), or displays the current line number if the spooler has not been halted. If L=line is used, printing resumes at the specified line. If omitted, printing resumes at the point where it was stopped.

S stops printing. It closes but doesn't kill the print-file and leaves the capture-file open.

The TRSDOS-16 spooler performs two functions which you can use simultaneously or one at a time:

1. It saves or "captures" the data that normally goes to the printer. The spooler then can either throw away this captured data or save it in a capture-file for later use.

2. It prints data from a disk file while other operations are in progress. That is, you can use the system -- everything except the printer -- while printing the file. While the spool-file is printing, your system captures the real-time printer output for later use.

Example 1

CAPTURE-FILE
In this example, you can run a program that outputs to the printer. Instead of waiting to use your system until the printing is complete, you can capture the program in a disk file to print out later.

To do this, call the capture-file SPOOL1 and type:

```
SPOOL ON <ENTER>
SPOOL N,F=SPOOL1 <ENTER>
```

This saves all printer output in SPOOL1. To stop capturing the printer output in SPOOL1, type:

```
SPOOL OFF <ENTER>
```

Now SPOOL1 is a text file which you can LIST or PRINT in the normal means, but at your convenience.

Example 2

BACKGROUND PRINTING
Here you can print to a file created by the spooler while you simultaneously use the system. Using the SPOOL1 file from the first example, type:

```
SPOOL ON <ENTER>
SPOOL P,F=SPOOL1 <ENTER>
```

TRSDOS-16 begins printing the file as a "background task", i.e., printing takes place only when the system isn't busy with some higher priority operation such as interpreting and executing your keyboard commands. Because this example doesn't include the K or C=copies option, TRSDOS-16 deletes SPOOL1 after it prints it and prints only one copy.

Type:

```
SPOOL OFF <ENTER>
```

after completing the print-file since the spooler doesn't turn itself off.

Example 3

SIMULTANEOUS CAPTURE-FILE AND BACKGROUND PRINTING

To save real-time printer output at the same time as the spooler prints a file, you can use this example.

First you need one capture-file (SPOOL1) and one print-file (SPOOL2). To turn the spooler on and begin capturing printer output in SPOOL1, type:

```
SPOOL ON <ENTER>
SPOOL N,F=SPOOL1 <ENTER>
```

You can now use the computer normally until you're ready to print out SPOOL1. To do so, type:

```
SPOOL  N,F=SPOOL2 <ENTER>
```

This closes SPOOL1 and makes SPOOL2 the new capture-file. To begin printing SPOOL1, type:

```
SPOOL  P,F=SPOOL1 <ENTER>
```

which prints out SPOOL1 and saves any real-time printing in SPOOL2.

If you want to halt the print-file operaton, type:

```
SPOOL H <ENTER>
```

This doesn't affect the capture-file operation. To resume printing, type:

```
SPOOL R <ENTER>
```

T

T

Moves the printer to the next page (top of form). This command is like FORMS with the T option.

If you are currently using spooler and it is capturing, T sends Top-Of-Forms character X'0C' to the spooler capture file.

Example

T <ENTER>

starts a new page after printing a file.

TERMINAL

TERMINAL

allows communication between the Model 16 (or the Enhanced Model II) and another computer running a host program. TERMINAL can only be used for transmission and reception of ASCII text rather than machine-language object code.

Input/Output is through Serial Channel A. In most applications, hookup is through telephone lines via a modem.

TERMINAL has three modes of operation, all described in detail in MODES OF OPERATION:

Menu allows you to select or change options, and even execute TRSDOS-16 system commands.

Interactive terminal transmits your keyboard input and displays incoming data.

Transmit from RAM for high-speed transfer of prepared data. Incoming data is displayed on the screen.

SETTING UP

For communications through telephone lines, you need a modem such as the Telephone Interface II, (26-1171), Modem I (26-1172), or Modem II (26-1173), and the RS-232 Cable, 26-44Ø3.

1. Set up the modem according to its instructions, and connect it to Serial Channel A on the back panel of the computer display console. If Serial Channel B is not connected to another device, install the serial terminator on that channel.

2. Set the modem to originate or answer mode -- whichever is the opposite of the host program with which you will communicate. Set it to full or half duplex, again depending on the requirements of the host program.

3. Turn on the modem and the Model 16/Enhanced Model II computer system.

4. Find out what RS-232C parameters are required by the host program you plan to use:
   Baud Rate
   Word length
   Parity
   Number of stop bits
   Initialize Serial Channel A accordingly (see RUNNING TERMINAL).

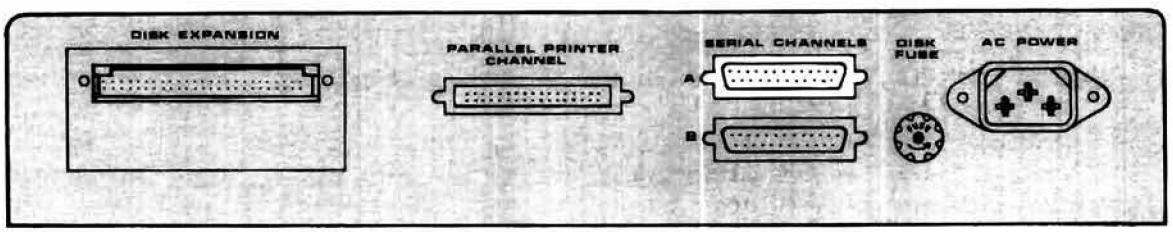


Figure 4. Connect RS-232-C Cable from Modem (or other Serial I/O device) to Serial Channel A on the back panel of the Model 16.

Note: In the examples illustrating sample uses of TERMINAL, underlining indicates what you should see on your display. The information following the underlined text is what you type.

RUNNING TERMINAL

1. From TRSDOS-16 Ready, you can start TERMINAL by typing:

   TERMINAL <ENTER>

   The program starts up in the Menu mode with the prompt:

   -- Enter Menu Selection ..

   Initialize Serial Channel A according to the

requirements (Baud Rate, Word Length, Parity, and Number of Stop Bits) of the host program you will communicate with. Type:

S <ENTER>

When the program prompts you to type in a TRSDOS-16 command, type in the SETCOM command just as you would in the TRSDOS-16 Ready mode. For example:

SETCOM A=(3ØØ,7,N,2) <ENTER>

enables Serial Channel A with 3ØØ baud, seven-bit words, no parity and two stop bits. After executing the command, control will return to TERMINAL's Menu mode.

3. If you plan to use the printer option (P) of TERMINAL (described later), initialize the printer now with the FORMS command. Type:

   S <ENTER>

   and enter the appropriate FORMS command at the prompt.

4. To select another menu command, type in the letter specified in the Menu. (See the Menu Commands Section for a list of the available commands.) To redisplay the entire Menu, type:

   M <ENTER>

MODES OF OPERATION

Menu Mode
is an off-line mode, i.e., you cannot transmit characters to the host program, and if characters are sent to you, they will be lost. This is the only mode where you can select Menu options. You can also enter the Transmit from RAM or Interactive Terminal mode from the Menu mode.

Interactive Terminal Mode
sends the characters you type to the host program and displays incoming characters as they are received. If the host program echoes your transmissions, they also will appear on the display; if not, you can select the echo option to instruct TERMINAL to display your keyboard input.

You can save incoming characters in the RAM buffer (R option) and output them to the printer (P option).

If transmission errors occur, TERMINAL displays a descriptive error message and waits for the error condition to be corrected. When it is, normal I/O resumes in the Interactive Terminal mode.

There are three ways to enter this mode:

1. With the T command from the Menu mode.
2. With the O command -- upon completion of an auto sign-on.
3. After transmission from the RAM buffer.

To return to the Menu mode, press <BREAK>

NOTE: Certain hosts will prompt you to use a break character or sequence to initialize transmissions. Since the <BREAK> key sends the program from the Terminal Mode to the Menu Mode, the Model 16 uses <ESC> for this break character. You can also set your own break character or sequence with the B command.

Transmit from RAM (and Auto Sign-On)
sends the contents of the RAM buffer to the host program, and passes control to the Interactive mode. Auto sign-on (O command) works in the same way as transmit from RAM. (The following applies to both operations.)

Load the RAM buffer with prepared text from a disk file with the G option. (If you are using auto sign-on, your auto sign on message is sent.) You can send the data in the RAM buffer one line at a time when the host program prompts you that it is ready (W option), or you can send it in a continuous stream.

During the transmission, your computer displays incoming text on the screen. If the host program echoes your transmissions, you can verify that the data was accurately sent.

During the transmissions, adjust the delay between characters by repeatedly pressing the <up arrow> (faster) and <down arrow> (slower) keys. If echoed data appears garbled, slow down the transmissions. If not, you might want to speed it up.

If TERMINAL receives a break character or sequence in this mode, it pauses until it receives the next character. If an H'13" is received, TERMINAL will pause until an H'11' is received. (By convention, H'13' is called the DC3 signal and means pause; H'11' is called the DC1 signal and means resume).

Use the X command to enter this mode.

To exit this mode and return to the Menu mode, press <BREAK>.

TERMINAL COMMANDS AND OPTIONS

A Build Auto Sign-On Message -- Allows you to prepare an automatic sign-on to be sent to the host with the O option. The message should contain the responses you use to answer the standard sign-on questions provided when you first call a host.

The message can be up to 6Ø keyboard characters, including control characters. All control characters will be displayed as +, but the true control code will actually be sent. (When you display a message, no control codes will be shown.) To imbed a carriage return (H'ØD') in the message, press <down arrow>.

For example, if the host requires responses to these prompts during sign-on:

User ID?
User Password?
Program Name?

instead of typing the information each time you call the host, you can store the responses in an auto sign-on buffer. To store the following information in the buffer:

STL-314 <H'ØD'
SHOWME <H'ØD>
MENU <H'ØD'>

Type:

--Enter Menu Selection.. A <ENTER>

```
The Current Auto Sign-On is

Change? (Y/N) Y <ENTER>
Enter Auto Sign-On Message (1-6Ø)
STL-314 <down arrow> SHOWME <down arrow> MENU <ENTER>
The Current Auto Sign-On is
     STL-314
     SHOWME
     MENU
-- Enter Menu Selection ..
```

The blank line above Change? (Y/N) Y <ENTER> indicates that the original auto sign-on was blank or contained non-display characters.

B Set/Change Break Character or Sequence -- Allows you to select the incoming code that will be interpreted as a "break" and also lets you define a key to send the same break character or sequence.

You can use any code from Ø to 255 as the break character; you can specify any duration from 1 to 451 milliseconds for the break sequence (this is determined by the host program). For the user-defined break key, you can use any key except <BREAK> or <CTRL> <C>.

The following example shows how to set up H'ØA' as the break character, and <CTRL> <D> as the break key:

```
--Enter Menu Selection B <ENTER>
Break Key is Now 1BHex
Change? (Y/N) Y <ENTER>
Enter New Key (1) <CTRL> <D> <ENTER>
Break Key is Now Ø4 Hex
Type of Break is Now CHR
Change? (Y/N) N <ENTER>
Break Char is Now Ø3 Hex
Enter new CHAR Value in Hex (2) ØA <ENTER>
Break Char is Now ØA Hex
```

C Copy RAM Buffer to Disk -- Creates a disk file copy of the text in the RAM buffer. The new file will have a record length of one.

Use this command to save data received into the RAM buffer in the Interactive Terminal mode. To minimize hookup time, do this after ending the connection to the host program. Or

if the RAM buffer is full, save it in a disk file, then reset it and reopen it for more data.

For example, to save a report you have just received in the Interactive Terminal mode as a disk file named REPORT, type:

```
-- Enter Menu Option.. C <ENTER>
Enter Filespec (1-34)
REPORT <ENTER>
```

The new file will be created (if REPORT already exists, it will be overwritten with the new data), and the RAM buffer contents and status will be unchanged.

To stop the copy process, press <BREAK>. The disk file will be closed and you will be returned to the menu.

D Display RAM Buffer -- displays the contents of the RAM buffer. To pause the display, press <HOLD>. To continue, press <HOLD> again. If the printer option is on when you issue this command, the text will also be output to the printer. To enter the command, type:

```
--Enter Menu Selection.. D <ENTER>
```

To stop the display function, press <BREAK>. You will be returned to the menu.

E Toggle Self Echo Option -- Allows you to display the characters you send via the TERMINAL.

Some hosts echo the text you send. As the host receives each character, it sends it right back to you and what you sent is displayed on the screen. When communicating with this type of host, set your modem to full duplex.

If the host does not echo your text, what you send will not be displayed unless you use the self-echo option. With such hosts, set your mode to half duplex.

To toggle the echo option, simply type E <ENTER>. The new state of the option (ON or OFF) will be displayed and the menu prompt will return.

F Set/Change <F1> and <F2> Keys -- Allows you to program <F1> and <F2> to output any code from Ø-255. This is useful if you use a particular code frequently.

For example, if the host recognizes H'13' (<CTRL> <S>) as pause control and H'11' (<CTRL> <Q>) as resume control, you may want to change these to <F1> and <F2> for convenience sake. To do this, type:

```
-- Enter Menu Selection.. F <ENTER>
F1 Key Will Send a Ø1 Hex Code
Change? (Y/N).. Y <ENTER>
Enter New Char Value in Hex (2) 11 <ENTER>
F1 Key Will Send a 11 Hex Code
F2 Key Will Send a Ø2 Hex Code
Change? (Y/N).. Y <ENTER>
Enter New Char Value in Hex (2).. 13 <ENTER>
F2 Key Will Send a Hex Code
-- Enter Menu Selection ..
```

Now when you type <F1> in the Interactive Terminal Mode, TERMINAL will transmit the resume control H'11'; for <F2>, the pause control H'13'.

G Get Disk File into RAM Buffer -- lets you load text stored in a disk file into the RAM buffer and then send it to the host via the X command (transmit from RAM). The previous contents of the RAM buffer are lost.

The disk file can contain fixed- or variable-length records of any length. However, only ASCII files should be loaded and sent. You can send any programs as long as you saved them in ASCII format.

For example, to send a document stored in the file DOCUMENT/TXT type:

```
--Enter Menu Selection.. G <ENTER>
Enter Filespec (1-34)
DOCUMENT/TXT <ENTER>
```

TERMINAL will load the file and return to the Menu. The RAM buffer will be closed.

If the host program is ready to accept data, you can now send it with the X command. After transmission is complete, TERMINAL will go to the Interactive Terminal mode.

L Toggle Line Feed Option -- tells TERMINAL how to handle an incoming line H'ØA'. When the option is on, all line feeds are ignored; when off, they are not ignored.

This is useful if the host always sends a line feed after a carriage return. Since the TRSDOS-16 display and printer drivers automatically perform a line feed after a carriage return is sent, the incoming line feed is redundant. Therefore, the line feed option should normally be on.

To toggle the line feed option, simply type L <ENTER>. The new state of the option (ON or OFF) will be displayed and the Menu prompt will return.

M Display Menu -- Clears the display and redisplays the Menu. Use this command when you have entered so many commands that all the Menu commands are not visible.

O Enter Terminal Mode with Auto Sign-On -- Starts transmission of the current auto sign-on message. After it sends the message, TERMINAL enters the Interactive Terminal mode.

To stop transmitting the auto sign-on, press <BREAK>. This returns control to the menu.

For details, see "Transmitting from RAM."

Note: Most host programs cannot receive anything until they send the first prompting message. Because of this, you should:

1. When connection is first made, go to the Interactive Terminal mode (T option) and wait for the host to send its first prompt character.
2. Press <BREAK> to return to the menu.
3. Start the auto sign-on (O option).

P Toggle Printer Option -- Turns the printer option ON and OFF. When ON, incoming text is copied to the printer as it is received and displayed. Initialize the printer with the FORMS command before you try to use it. When you use the D command while the P option is on, the RAM buffer text is copied to the printer.

TERMINAL uses a circular buffer for efficient output to the printer. If characters come in too fast, they will not be printed. They will be displayed, though, and saved in RAM if the buffer is open. (Check your printer's specifications for maximum character input rate. At 3ØØ baud, 7-bit characters may come in as fast as 3Ø per second.)

To minimize hookup time, do not use the printer option while on-line with the host. Save the incoming text in RAM and upon completion of the hookup, turn the printer option on and use the D command to get a hard copy of the data.

To toggle the printer option, simply type: P <ENTER>. The new state of the option (ON or OFF) will be displayed and the menu prompt will return.

Q Quit -- returns control to TRSDOS-16. If there is data in the RAM buffer, it is lost; i.e., you cannot restart TERMINAL and recover the date.

R Toggle RAM Buffer Option -- (Interactive Terminal Mode only) lets you save in RAM some or all the data received by "opening" and "closing" the RAM buffer. With this, you can examine the data later with the D command, or save the data in a disk file with the C command.

When you open the RAM buffer, you can either reset it or retain its current contents. If you retain the contents, new incoming text will be loaded after the existing text in the RAM buffer.

To toggle the RAM buffer option, simply type R <ENTER>. The new state of the option (OPEN or CLOSED) will be displayed. If you have just opened the buffer, you will receive the following prompt:

```
RAM Buffer Now Open
Reset RAM Buffer? (Y/N) ..
```

If you type Y <ENTER>, the buffer will be reset and previous contents will be lost. For more information, see "Using the RAM Buffer".

S Perform System Command -- enters the operating system and allows you to enter a TRSDOS-16 system command. After it executes the system command, control returns to TERMINAL's Menu. A few TRSDOS-16 commands and programs automatically return to TRSDOS-16 Ready. If you execute any of these commands while in TERMINAL, control will not be returned to TERMINAL, but to TRSDOS-16 Ready.

T Enter Terminal Mode -- directly enters the Interactive Terminal mode. When in this mode, press <BREAK> to return to the Menu.

V Toggle Video Filter -- filters out data characters which produce undesirable results when output to the display.

When a character such as ESC(H'1B) is output, it causes Terminal to clear the screen and home the cursor. With the video filter option, you can prevent this by "filtering" these characters from the display. If the RAM buffer is open, they will be saved in RAM, regardless of the state of this option.

The codes (given in hexadecimal) which this option filters are:

Ø1,Ø2,Ø3,Ø4,Ø5,Ø6,Ø7,ØB,ØC,ØE,ØF,
1Ø,11,12,13,14,15,16,1E,1F

If Terminal receives any of these characters while the video filter is on, it will display a "±" in its place.

W Set/Change Prompt Wait Character -- allows you to toggle this feature on or off and set a special character as the prompt-wait character to cue the terminal to continue the transmission.

This allows you to use the high speed transmit from RAM mode, even when the host program can accept only one line at time. (It does not affect operation in the Interactive terminal mode.)

Normally, the host program sends a prompt such as a question mark when it is ready for the next line. (A line is defined as a string of characters terminated by a carriage return H'ØD'.) In the Interactive Terminal mode, you simply wait until this prompt is displayed; the prompt wait feature makes TERMINAL do the same thing while in the transmit from RAM mode.

You can define the prompt wait character as any keyboard character from H'2Ø' to H'7F'.

Leave the prompt wait feature off when the host program is simply storing characters as received and is not sending a ready-for-next-line prompt. TERMINAL will transmit text from RAM in a continuous stream.

NOTE: When you start the transmit from RAM (X option) or auto sign-on (O option), the first line is sent immediately, without waiting for a prompt. Each subsequent line is then sent after the prompt is received.

To turn the prompt wait feature off, press <HOLD> when the program asks for a new character.

X Transmit RAM Buffer and Enter Terminal Mode -- enters the Transmit from RAM mode where it sends the current contents of the RAM buffer to the host program. When the entire buffer has been sent, TERMINAL goes into the Interactive Terminal mode. For details, see "Transmitting from RAM."

To stop transmission from RAM, press <BREAK>. Control returns to the Menu.

USING THE RAM BUFFER

You can use the RAM buffer to store incoming text (R option) and prepared text from a disk file (G option). This, in turn, allows the stored files to be rapidly sent. The RAM buffer helps reduce costly hookup time by letting you perform time-consuming operations -- preparing data or printing it out -- while TERMINAL is off-line.

If the buffer is filled during a load from disk (G command) or while receiving data in the Interactive Terminal mode, a warning message will be displayed and the buffer will be closed. If you are loading a disk file, control returns to the Menu mode and the buffer will be filled with the data that was loaded.

If you are in the Interactive Terminal Mode, normal I/O will continue, except that it will no longer be saved in the buffer.

Saving the RAM Buffer

When the buffer is filled in the Interactive Terminal Mode (or when you suspect it is almost full):

1. Transmit a pause or break control character to the host program.
2. Press <BREAK> to return to the Menu.

3. Use the C command to copy the contents of the RAM buffer to a disk file.
4. Reset the RAM buffer with the R command.
5. Use the T command to return to the Interactive Terminal Mode.

Opening and Closing the RAM Buffer

To save portions of the text during I/O of the Interactive Terminal Mode, use the R command. Prior to receiving the data you want to save:

1. Transmit a pause or break control character to the host program.
   Press <BREAK> to return to the Menu.
3. Use the R command to toggle the RAM buffer status. If it is off, toggle it again to open it. If it is already open, you have the option of resetting it or leaving it as is. To add new data onto the end of old, do not reset it. To delete old data, reset it.
4. Enter the T command to return to the Interactive Terminal Mode.
5. Direct the host program to resume transmission. The data will now be saved in the RAM buffer as it is received.

Saving the Options You Have Selected

You can save these options in a customized version of TERMINAL:

- Prompt wait and definition of prompting character
- Definition of break character or sequence from host program and assignment of a break key on your computer
- <F1> and <F2> characters
- Line feed option
- Printer option
- Self-echo option
- Video filter option
- Auto sign-on option
- Speed of transmit from RAM and auto sign-on (as set by the "up"and "down" keys). Once you find out the maximum rate of transmission the host program can handle, you can set that as the default rate.

After you select the options for your customized version, use the DUMP command to create a new program file. Terminal resides from H'3ØØØ'. Give this customized program a name other than TERMINAL -- and leave the TERMINAL program in its original configuration.

For example, to call your customized version MINE, type:

```
--Enter Menu Selection.. S  <ENTER
Enter TRSDOS Command (1-79)
DUMP MINE START=3ØØØ, END=3FFF   <ENTER>
```

Now you have a customized version of TERMINAL that starts up when you type:

```
MINE <ENTER>
```

SAMPLE USES

To Send a Program

If you intend to send a program via TERMINAL, you must first store it in an ASCII-format disk file. When you have done this, set up the modem and initialize Serial Channel A as explained previously. See your modem manual for the appropriate procedure for getting on-line.

For example, to send a disk file named "SORTDATA" on Drive 1, get on-line and load the TERMINAL program. Enter the Interactive Terminal mode by typing:

```
Enter Menu Selection.. T <ENTER>
```

Go through the necessary sign-on and when you want to send the program, press <BREAK> to return to the Menu. (If you want to use the prompt wait option, select it now.) Then type:

```
--Enter Menu Selection.. G <ENTER>
Enter Filespec (1-34)
SORTDATA:1  <ENTER>
```

TERMINAL will now load the program into RAM. Make sure the host is ready to receive the program, then type:

```
--Enter Menu Selection.. X  <ENTER>
```

The Terminal will now send the program to the host. Press <BREAK> if you want to stop the transmission for any reason. This returns control to the Menu. Otherwise, upon completion of the program transmission, control will go into the Interactive Terminal mode.

To Receive a Program

If the host program you are communicating with is ready to send you an ASCII-formatted program, you must first go to the Menu and type:

```
--Enter Menu Selection.. R <ENTER>
```

If the buffer is now closed, repeat this command and TERMINAL will display the message:

```
RAM Buffer Now Open
Reset RAM Buffer (Y/N) Y  <ENTER>
```

This opens and clears the buffer. Now return to the Interactive Terminal mode (T option) and tell the host to send the program.

After you receive the entire program, press <BREAK> to return to the Menu; then type:

```
-- Enter Menu Selection.. C  <ENTER>
Enter Filespec (1-34)
NEWPROG  <ENTER>
```

This copies the program in RAM into a disk file named NEWPROG.

ERROR CONDITIONS

In the Interactive Terminal Mode, Transmit from RAM Mode, or during Auto Sign-on, TERMINAL may detect errors related to the serial transmission. In such cases, it will display an error message in reverse video and, if possible, will continue normal I/O.

The error messages that may occur while you are in the Interactive Terminal Mode are:

P Parity error. The received character will be after the P.

O Over-run. At least one character has been received but not picked up by TERMINAL. This occurs if you are in the Menu mode while the host program is sending characters.

F Framing error. The received character will be displayed after the F. Check your SETCOM parameters to see that they match the requirements host program.

The following errors can occur in any mode except the Menu:

DATA CARRIER LOST Check the telephone/modem connection.

DATA CARRIER RESTORED TERMINAL will pause until the carrier is restored. If TERMINAL was transmitting from RAM or sending an auto sign-on, it will start over at the beginning of the text when data carrier is restored.

BREAK SEQUENCE RECEIVED If the host program sends a break sequence, or sends TERMINAL'S own break character, this message will displayed; if TERMINAL is in the transmit from RAM or auto sign-on mode, it will pause until the next character is received from the host.

TIME

TIME

Displays the date-time string. It works the same way as the DATE system command. The format for the date-time string is:

```
WED MAR 25, 1981 84 -- 16.24.34
```

for Wednesday, March 25, 1981, the 84th day of the year, 4:24:34 p.m.

Example

```
TIME <ENTER>
```

VERIFY

VERIFY {options}

Sets the verify function ON or OFF. When ON, TRSDOS-16 checks data after each write operation.

The options are:

ON tells TRSDOS-16 to check all data, as it writes it

OFF turns VERIFY OFF

If you do not specify an option, TRSDOS-16 returns the current status of VERIFY.

When VERIFY is on, TRSDOS-16 reads after each write operation to ensure that the data is readable. If the data is not readable, TRSDOS-16 retries and, if data is still not readable, returns an error message.

If you need to increase the speed of TRSDOS-16, turn VERIFY OFF.

NOTE: TRSDOS always verifies directory writes. The VERIFY function checks only user writes (writing data into a file).

Examples

VERIFY ON <ENTER>

turns the verify function ON.

VERIFY OFF <ENTER>

turns the verify function OFF.

VERIFY <ENTER>

displays the status of the verify switch.

VERSION

VERSION

Displays the version number of the operating system currently in use.

Example

```
VERSION <ENTER>
```

returns -- Version: 4.1 -- indicating the major version level "4", minor version level "1".

TECHNICAL SYSTEM INFORMATION

MEMORY REQUIREMENTS

TRSDOS-16 loads into memory at startup and occupies approximately 48K of memory.

At this time TRSDOS-16 also processes your configuration command file (see **APPENDIX B** for information on the Configuration Command File).

The configuration file tells TRSDOS-16 what files to load into memory for your use. This might include RUNCOBOL, and the Debugger. TRSDOS-16 loads these files into the memory area reserved for TRSDOS-16.

Any memory following the configured files is available to you, the user.

Note: The Editor, Assembler, and Linker load into user memory, not the Configurator memory.

RELATIVE ADDRESSING

TRSDOS-16 loads into memory in a way that is "invisible" to the user. This is TRSDOS-16's way of protecting itself from being modified by other programs.

TRSDOS-16 internally keeps track of two addresses:

- the BASE address - this is the bottom of user memory. (Memory below this address is reserved for TRSDOS-16 and files designated by the configured file.
- the BOUNDS address - this is the top address of memory. (H'1FFFF in a 128K Model 16, H'3FFFF in a 256K Model 16, H'5FFFF in a 384K Model 16, and H'7FFFF in a 512K Model 16.)

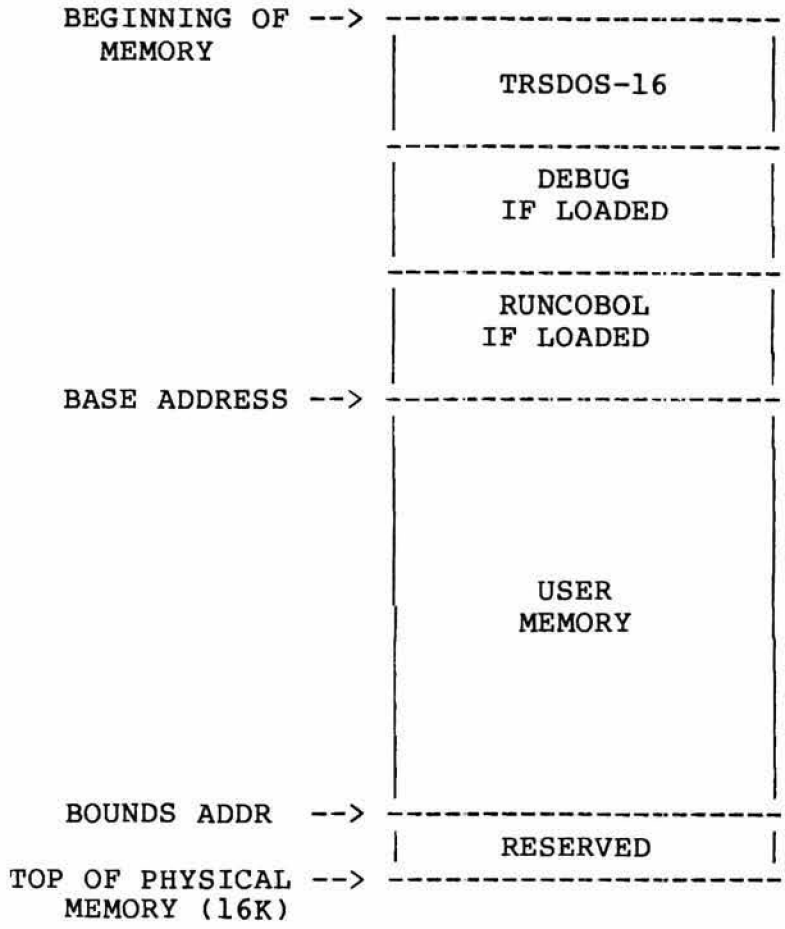
In short:

BOUNDS Address - BASE Address = User Memory

TRSDOS-16 uses the BASE Address to find the physical loading address of your program. When you write a machine-language program, you assign it a "relative loading" address, usually zero. TRSDOS-16 then adds the BASE address to this relative address to determine the physical address where it will load your program:

BASE Address + PROGRAM RELATIVE ADDRESS = PHYSICAL LOAD ADDRESS

TRSDOS-16 uses the BOUNDS address to find the amount of available user memory. This is how the Operating System determines if your program will fall out of the "bounds" of user memory.



MEMORY CHART

DISK ORGANIZATION

FLOPPY DISKETTE

TRSDOS-16 can use either single-sided (double density) or double-sided (double density) diskettes. However, if you are using an Enhanced Model II, you can use only single-sided diskettes because of the type of drives you have.

TRSDOS-16 will automatically format either single or double-sided diskettes according to the type of diskette.

Each side of the double-sided diskette contains 77 tracks, numbered Ø - 76. Therefore, the double-sided diskette can be thought of as one diskette with 154 tracks.

The single-sided diskette has 77 tracks on one side only.

Each track is made up of 32 sectors, numbered 1-32. Each sector contains 256 bytes, except for track Ø (one side only on the double-sided diskette). This track Ø is reserved for System use and is formatted single-density. It contains 26 sectors and 128 bytes per sector.

The total capacity of a double-sided diskette is:

(153 * 32 * 256) + (1 * 26 * 128) = 1,256,7Ø4 bytes

The total capacity of a single-sided diskette is:

(76 * 32 * 256) + (1 * 26 * 128) = 625,92Ø

HARD DISK

The TRS-8Ø Hard Disk Drive is organized into 256 "cylinders". Each cylinder is made up of four tracks that have the same radius on each of the four surfaces (4 * 256= 1Ø24 total tracks per hard disk).

Each track contains 34 sectors, numbered 1-34. Each sector contains 256 bytes.

The total capacity of a hard disk is:

(1Ø24 * 34 * 256) = 8,912,896 bytes

DISK SPACE AVAILABLE TO USER

TRSDOS-16 occupies approximately 31 tracks on the floppy diskette and approximately 32 tracks on the hard disk.

Track Ø is reserved by the System on both the floppy and hard disk. The hard disk also reserves track 1 for diagnostics.

The directory and alternate directory on both the floppy diskettes and hard disk reserve a certain number of tracks, determined by the size of the directories (the size is set by the FORMAT command).

To determine the number of tracks these directories will consume, use this formula:

Directory Space Formula	Description
# of filenames / 4 (round result up to next integer value)	to determine number of sectors used by the primary directory
# of sectors / 32 (floppy) # of sectors / 34 (hard) (round result up to next integer value)	to determine number of tracks used by the primary directory
# of tracks * 2	to determine the total number of tracks used for the primary and alternate directories (the alternate directory is simply a copy of the primary directory)

Note: TRSDOS-16 stores 4 filenames per sector in the directory.

For example, if you use this Format command:

```
FORMAT :1 {PW=PASSWORD,SIZE=25Ø}
```

the directories will consume a total of four tracks:

25Ø filenames / 4	=	62.5
round up	=	63
63 sectors / 32 (floppy)	=	1.96
round up	=	2 tracks for each directory
2 * 2	=	4 total tracks used for the primary and alternate directories.

UNIT OF ALLOCATION

All allocation of disk space is made by single sectors. This means that the smallest non-empty TRSDOS-16 file will consist of one sector.

Single-Sided

Diskette	Tracks	Sectors	Bytes
1	76	2,432	622,592
---	1	32	8,192
---	---	1	256

Double-Sided

Diskette	Cylinders	Tracks	Sectors	Bytes
1	76	154	4,896	1,253,376
---	1	2	64	16,384
---	---	1	32	8,192
---	---	---	1	256

Note: Track Ø on the floppy diskette (only one side for double-sided diskettes) is reserved for System use and is not available for user storage. It is formatted single density with 26 sectors that contain 128 bytes each.

Hard Disk	Cylinders	Tracks	Sectors	Bytes
1	256	1Ø24	34,816	8,912,896
---	1	4	136	34,816
---	---	1	34	8,7Ø4
---	---	---	1	256

DISK FILES

METHODS OF FILE ALLOCATION

TRSDOS-16 provides two ways to allocate disk space for files: Dynamic Allocation and Pre-Allocation.

DYNAMIC ALLOCATION

With dynamic allocation, TRSDOS-16 allocates sectors only at the time of write. For example, when a file is first opened for output, no space is allocated. The first allocation of space is done at the first write. Additional space is added as required by subsequent writes.

With dynamically allocated files, unused sectors are de-allocated (recovered) when the file is closed.

Dynamic allocation is the method TRSDOS-16 uses, unless you execute the CREATE system command.

PRE-ALLOCATION

With pre-allocation, the file is allocated a specified number of sectors when it is created. Pre-allocated files can only be created by the system command CREATE.

TRSDOS-16 will dynamically extend (enlarge) a pre-allocated file as needed.

However, TRSDOS-16 will not de-allocate unused sectors when a pre-allocated file is closed. To reduce the size of a pre-allocated file you must copy it to a dynamically allocated file. The COPY system command does this automatically when the destination is a dynamically-allocated file.

RECORD LENGTH

TRSDOS-16 transfers data to and from disks one sector at a time; i.e., in 256-byte blocks. These are the System's "physical" records.

User records or "logical" records are the buffers of data you wish to transfer to or from a file. These can be from 1 to 256 bytes long.

The Operating System will automatically "block" your logical records into physical records which will be transferred to disk, and "de-block" the physical records into logical records which are used by your program.

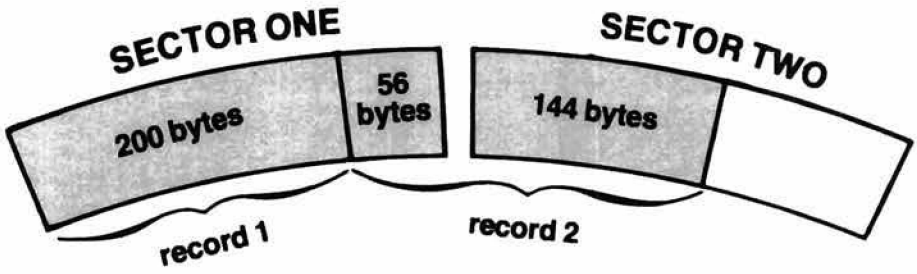
Therefore, your **only** concern during file-access is with logical records. You never need to worry about physical records, sectors, tracks, etc. This is to your benefit, since physical record lengths and features may change in later TRSDOS versions or with other peripheral devices, while the concept of logical records will not.

From this point on, the term "record" refers to a "logical record".

SPANNING

If the record length is not an even divisor of 256, the records will automatically be spanned across sectors.

For example, if the record length is 2ØØ, Sectors 1 and 2 will look like this:



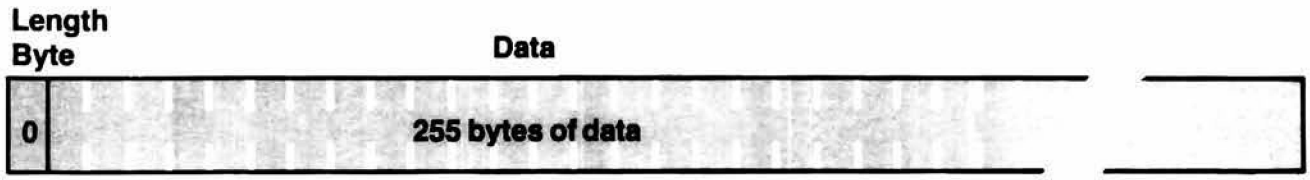
FIXED-LENGTH AND VARIABLE LENGTH RECORDS

TRSDOS-16 files can have either fixed-length or variable-length records. Files with fixed-length records will be referred to as FLRs; files with variable length records, VLRs.

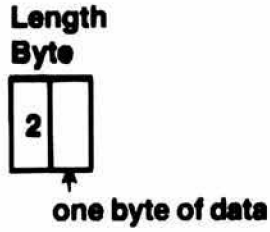
Record length in an FLR file is set when the file is opened for the first time. This length can be any value from 1 to 256 bytes. Once set, the record length in an FLR cannot be changed unless the file is being over-written with new data.

The record-lengths in a VLR file can vary. For example, the first record in a file might have a length of 32; the second, 17; the third, 25Ø; etc.

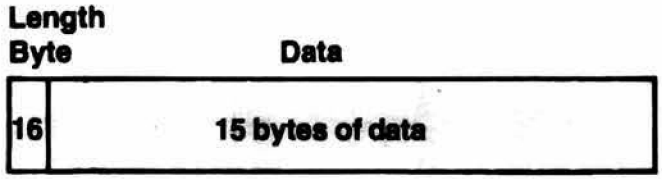
The record length in a VLR file is specified in a one-byte length-field at the beginning of each record. The record-length byte indicates the entire length of the record, **including** the length-byte. This can be any value from Ø to 255. A value of 1 can be used to mean an empty record (e.g., a blank line in an ASCII text file).



A length-byte value of 2 indicates that the record contains 1 byte of data:



A length-byte value of 16 indicates that the record contains 15 bytes of data:



RECORD NUMBERS

Records are numbered from Ø (beginning of file) to 16,777,214.

A disk file can also contain up to 16,777,216 bytes of storage, however, your storage medium may not be capable of storing this much information.

To determine the number of records a file will hold, use the following formula:

16777216 / logical record length = number of records

For example:

16777216 / 38 = 441,5Ø5 records

Example

If a three megabyte file (possible with hard disk) is opened with a record length of 3 bytes, it would hold approximately 1,ØØØ,ØØØ records.

RECORD PROCESSING CAPABILITIES

TRSDOS-16 allows both direct and sequential file access.

Direct access -- sometimes called "random access", allows you to process records in any sequence you specify.

Sequential access allows you to process records in sequence: Record N, N+1, N+2,.... With sequential access, you do not specify a record number; instead, TRSDOS-16 accesses the record following the last record processed, starting with record Ø.

FLR files may be opened as either direct access or sequential access.

VLR files can only be opened as sequential access. You cannot position to a specific record in the file, since the varying record lengths make it impossible to calculate the position of the VLRs.

The direct access SVC's are DIRRD (Direct-Read) and DIRWR (Direct-Write). Direct access SVC's always access the record you specify.

The sequential access SVC's are READNX (Read-Next) and WRITNX (Write-Next). Sequential access SVC's always access the record following the last record processed. (When the file is first opened, sequential processing starts with record Ø, however DIRWR and DIRRD can be used to position to the EOF, end of file.)

EXAMPLES WITH FIXED LENGTH FILES

Assume you have a Fixed Length Record file currently open. Here are some typical sequences you can accomplish via the file processing routines.

1. Read and/or write records in the file -- in any order
This is done using DIRRD and DIRWR SVC's. You could read record 5, write at end of file, read record 3, write record 3, etc.

2. Sequential Read (or Write) beginning anywhere in the file.
First you would do a direct-read to the record where you want to start reading or writing. After that, you would do sequential reads or writes until done (READNX and WRITNX).

3. Sequential Write starting at end of file.
First do a direct-write to the end of the file. Then do sequential writes until done.

4. Determine the number of records in a file.
First do a direct-read to end of file, then use the LOCATE routine to get the current record number, which now equals number of records +1.

EXAMPLES WITH VARIABLE LENGTH RECORDS

Here are examples of ways to read or write VLR files

1. Start reading or writing sequentially at first record
Open the file and start reading or writing sequentially until done.

2. Sequential Write starting at end of file
First do a direct-write to the end of the file. Then do sequential writes until done.

Note: Whenever you write to a VLR, the end of the file is **automatically** reset to the last record you write.

Also, you cannot update a VLR file directly. You must read in the file, update it and output the updated information to a new VLR file.

SUPERVISOR CALLS

Supervisor Calls (SVCs) are operating system routines available to any user program. These routines alter certain system functions and conditions, provide file access, and perform I/O to the keyboard, video display, and printer.

The available TRSDOS-16 SVCs are:

KEYBOARD SVCs
KBCHAR
KBINIT
KBLINE
VIDKEY

VIDEO SVCs
CURSOR
VDCHAR
VDINIT
VDLINE

PRINTER SVCs
PRCHAR
PRCTRL
PRINIT
PRLINE

SYSTEM CONTROL SVCs	
CLRXIT	HLDKEY
DATE	JP2DOS
DOSCMD	SETBRK
ERRMSG	SETTRP
ERROR	VERSION

MISCELLANEOUS SVCs	
DEBUG	MOUNT
DISMOUNT	MOVBUF
EXECUTE	RESET

FILE ACCESS SVCs	
CLOSE	LOCATE
CLOSEF	OPEN
DIRRD	OPENDO
DIRRW	READNX
DUMP	RENAME
KILL	UNLOCK
LOAD	WRITNX

COMMUNICATIONS SVCs	
ACTL	BCTL
ARCV	BRCV
ATX	BTX
	RS232C

Each SVC has a function code which you use to call it. These codes range from Ø to 512.

SVC BLOCK

To use an SVC, you must assign it an area in memory called a BLOCK. You use this BLOCK to pass parameters to and from the SVC.

The SVC block is a maximum of 16 words long. Each word consists of two bytes.

These words are used to store parameters which you want to pass to or from the SVC. They are addressed by their byte-offset number, an even number ranging from Ø to 32. Some SVC's will use only one byte of a word. In this case the byte will be located in the low order 8-bits (or byte-offset + 1). The high order 8-bits must contain zeroes.

Certain parameters, such as memory addresses, will use two consecutive words of storage (called a long word).

You can store an SVC BLOCK anywhere in memory; however, there must be a minimum of 32 bytes available after the starting address of the block, although the block may occupy less than last 32 bytes.

Figure 5 is a sample SVC BLOCK. (DIRWR SVC -- write a record out to disk.)

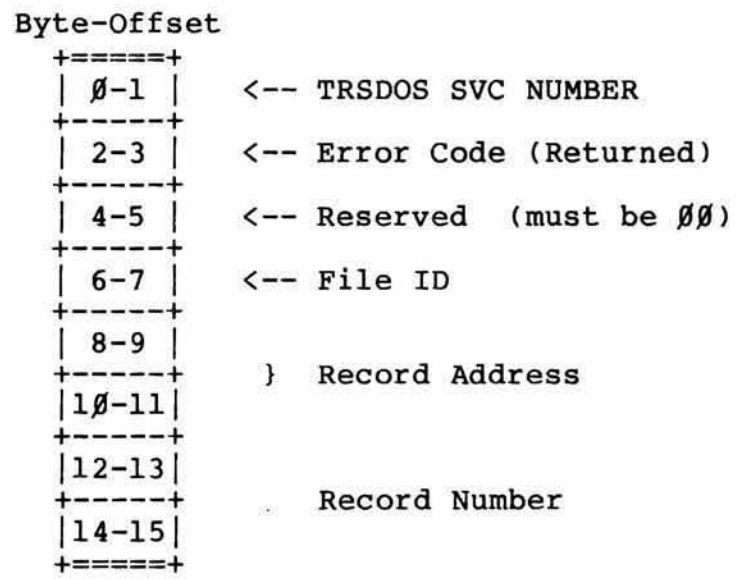


Figure 5. Sample SVC BLOCK

In this illustration, each box represents a word. The numbers within the boxes are the byte-offset numbers for addressing the SVC BLOCK.

The first number (the even number) is used to reference the offset. For example, byte-offset Ø-1 contains the SVC number, but is addressed as offset Ø.

Byte-offsets 16 through 31 are not used for this SVC, and need not be present.

The first three words of every SVC contain the same information:

Ø-1 TRSDOS Function Code Number (for example, 35)
2-3 On Error, returns the Error Code
4-5 Reserved -- Must contain zero

If an area within a block is marked RESERVED, you must set it to zero or a parameter error will occur.

CALLING PROCEDURE

To call a TRSDOS-16 SVC:

1. Load the address of the desired SVC BLOCK into register AØ.

2. Execute a BRK #Ø instruction

3. Upon return from an SVC, you must specifically TEST for for an error condition if you want to see if one exists.

During the execution of an SVC, the SVC processor does not alter any registers, nor does it alter any status bits upon return.

Figure 6 is part of a sample program for the DIRWR SVC. We'll look at it closely to see one of the ways to load a SVC BLOCK for execution.

This sample assumes that the File Identification Number was previously stored in register A1, Record Address in register A2, and Record Number in register A3.

```
WRITE     LDA       .AØ,SVC BLOCK              * line 1
          MOVW      @AØ,#DIRWR SVC NUMBER      * line 2
          STW       A1,6@AØ                      line 3
          STL       A2,8@AØ                      line 4
          STL       A3,12@AØ                     line 5
          BRK       #Ø                           line 6
          TESTW     2@AØ                         line 7
          BNE       ERROR                        line 8


SVC BLOCK
     RDATAB         32,Ø                         line 9
DIRWR SVC NUMBER
          EQUW      44                           line 1Ø
```

Figure 6. Sample Program

In this program, line 1 loads the address of the SVC BLOCK into register AØ.

Line 2 moves the function code number to byte-offset Ø and line 3 stores the file identification number in byte-offset 6.

Lines 4 and 5 store the Record Address and the Record Number in byte-offsets 8 and 12, respectively. Each is a long word.

Line 6 executes the SVC.

Line 7 tests for an error returned at offset 2.

If there is an error, the return is non-zero and line 8 either branches to an ERROR handling subroutine elsewhere in the program or executes the next line of the program.

Line 9 defines the SVC BLOCK to 32 bytes of zeroes.

And line 1Ø equates the SVC NUMBER to 44.

PROGRAMMING WITH USER INTERRUPTS

TRSDOS-16 allows user-programmed interrupts as described under SETBRK and SETTRP. When the interrupt is received (that is, when the <BREAK> key is pressed or a system trap is taken), control transfers to your interrupt handling routine.

When SETBRK terminates, or when an intercepted interrupt occurs, the following takes place:

1. The current User PC and status registers are pushed on the USER stack.

2. A branch is taken to the user-specified address.

Before doing any processing, you should save any registers you plan to alter. When processing is complete, execute an RTR instruction (return with restore) to return to the interrupted program.

A fatal error occurs if TRSDOS-16 cannot push three words onto the user stack, i.e., if the stack pointer contains a H'4 or less.

NOTE: Interrupt handlers entered this way are also subject to interrupts.

CONVERSION OF RELOCATABLE TO REAL ADDRESSES

If a byte-offset of an SVC 'points' to a buffer in user memory, the following restrictions apply:

All buffers are checked against their maximum possible sizes. It is not permissible to have a 256-byte buffer beginning 25 bytes from the end of memory. The check is made in this way:

1. DATE SVC buffer requires 26 bytes
2. ERRMSG SVC buffer requires 8Ø bytes
3. OPEN SVC parameter list requires 5 bytes

4. Any call passing a filespec (KILL, RENAME, etc.) requires a 34-byte buffer
5. Any Disk I/O call passing a buffer address is checked for a 256-byte buffer
6. Any miscellaneous I/O calls (PRLINE, VDLINE, etc. passing a fixed length buffer is checked for 256 bytes.

ACTL **Function Code 1ØØ**
Control Channel

Performs control functions on serial channel A and then returns its status.

TRSDOS-16 sets up A/BRCV, A/BTX and A/BCTL when you initialize channel A/B with RS232C. If you call any of these routines while the channel is not initialized (active), you will get an error return code of 1 (no function code exists).

Entry Conditions

Byte-offset	Ø-1	1ØØ
	6-7	option switch

Exit Conditions

Byte-offset	2-3	error code
	8-9	communications status code

Valid option switch settings are:

option switch	Meaning
Ø	Get status of serial channel into the Status Value
1	Get received buffer count into the Status Value
2	Turn on Request to Send (RTS) signal
3	Turn off RTS
4	Start transmission of a BREAK sequence
5	Stop transmission of a BREAK sequence
6	Clears receive buffer
7	Reset SIO Error condition

The 8-bit grouping of flags returned in the communications status code are:

Bit	Meaning
Ø	Clear to Send not present
1	Unused
2	Transmitter busy
3	Modem data carrier not present
4	Parity error in byte now being received
5	Data overflow due to a byte now being received
6	Framing error in a byte now being received
7	Break Sequence now being received

Example

```
            .
ACTL        LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#ACTL SVC NUMBER
            MOVW        6@AØ,#OPTION SWITCH
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
             .
             .
SVC BLOCK
            RDATAB      32,Ø
ACTL SVC NUMBER
            EQUW        1ØØ
OPTION SWITCH
            EQUW        Ø
```

ARCV **Function Code 96**
Channel A Receive

Returns a single character from serial channel A.

TRSDOS-16 sets up A/BRCV, A/BTX and A/BCTL when you initialize channel A/B with RS232C. If you call any of these routines while the channel is not initialized (active), you will get an error return code of 1 (no function code exists).

INPUT BUFFER

Each channel (A and B) has its own internal 16-character receive buffer to reduce overruns when receiving data at high speeds. This buffer is a First-In, First-Out buffer (FIFO) and is established when the channel is initialized.

When a character is received by the ARCV and BRCV SVC's, it is stored in this buffer along with its status when it was received.

Each time a new character is received into the buffer the character at the top of the buffer (oldest character) is pushed into the character returned byte-offset and its status (stored in the buffer when it was received) is pushed into the communications status code byte-offset.

An overrun occurs only when the 17th character is received into the already-full buffer. The 16th character is replaced with the 17th. When the character that caused the overrun is retrieved into the character returned byte-offset, an overrun will be indicated in communication status code.

If there are no characters "waiting" in the buffer, the communications status code will reflect the current status of the serial interface.

Entry Conditions

Byte-offset	Ø-1	96

Exit Conditions

Byte-offset		
	2-3	error code
	6-7	character returned (if any)
	8-9	communications status code
	1Ø-11	character received status Ø = character received non-zero = character not received

The 8-bit grouping of flags returned in the communications status code are:

Bit	Meaning
Ø	Clear to Send not present
1	Unused
2	Transmitter busy
3	Modem data carrier not present
4	Parity error in byte now being received
.	Data overflow due to a byte now being received
.	Framing error in a byte now being received
.	Break Sequence now being received

Example

```
ARCV        LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#ARCV SVC NUMBER
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR

SVC BLOCK
            RDATAB      32,Ø
ARCV SVC NUMBER
            EQUW        96
```

ATX **Function Code 97**
Channel A Transmit

Outputs a single character to serial channel A.

TRSDOS-16 sets up A/BRCV, A/BTX and A/BCTL when you initialize channel A/B with RS232C. If you call any of these routines while the channel is not initialized (active), you will get an error return code of 1 (no function code exists).

Data bytes will be transmitted even if no carrier is present. You must check the communication status code and character transmitted status for error conditions.

Entry Conditions

Byte-offset	Ø-1	97
	6-7	character to be sent

Exit Conditions

Byte-offset	2-3	error code
	8-9	communications status code
	1Ø-11	character transmitted status Ø = character transmitted non-zero = character not transmitted

The 8-bit grouping of flags returned in the communications status code are:

Bit	Meaning
Ø	Clear to Send not present
1	Unused
2	Transmitter busy
3	Modem data carrier not present
4	Parity error in byte now being received *
5	Data overflow due to a byte now being received *
6	Framing error in a byte now being received
7	Break Sequence now being received *

Bits 7 are only used when the character was not sent

Example

```
            .
            .
ATX         LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#ATX SVC NUMBER
            MOVW        6@AØ,#CHAR TO SEND
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
ATX SVC NUMBER
            EQUW        97
CHAR TO SEND
            TEXT        'G
```

BCTL **Function Code 1Ø1**
Control Channel B

Performs control functions on serial channel B and then returns its status.

Entry Conditions

Byte-offset	Ø-1	1Ø1
	6-7	option switch

Exit Conditions

Byte-offset	2-3	error code
	8-9	comunication status code

Valid option switch settings are:

Option Switch	Meaning
Ø	Get status of serial channel into the Status Value
1	Get received buffer count into the Status Value
2	Turn on Request to Send (RTS) signal
3	Turn off RTS
4	Start transmission of a BREAK sequence
5	Stop transmission of a BREAK sequence
6	Clears receive buffer
7	Reset SIO Error condition

The 8-bit grouping of flags returned in the communications status code are:

Bit	Meaning when set
Ø	Clear to Send not present
1	Unused
2	Transmitter busy
3	Modem data carrier not present

Bit	Meaning when set
4	Parity error in byte now being received
5	Data overflow due to a byte now being received
6	Framing error in a byte now being received
7	Break Sequence now being received

Example

```
            .
            .
BCTL        LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#BCTL SVC NUMBER
            MOVW        6@AØ,#OPTION SWITCH
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
BCTL SVC NUMBER
            EQUW        1Ø1
OPTION SWITCH
            EQUW        Ø
```

BRCV **Function Code 98**
Channel B Receive

Returns a single character from serial channel B.

TRSDOS-16 sets up A/BRCV, A/BTX and A/BCTL when you initialize channel A/B with RS232C. If you call any of these routines while the channel is not initialized (active), you will get an error return code of 1 (no function code exists).

INPUT BUFFER

Each channel (A and B) has its own internal 16-character receive buffer to reduce overruns when receiving data at high speeds. This buffer is a First-In, First-Out buffer (FIFO) and is established when the channel is initialized.

When a character is received by the ARCV and BRCV SVC's, it is stored in this buffer along with its status when it was received.

Each time a new character is received into the buffer the character at the top of the buffer (oldest character) is pushed into the character returned byte-offset and its status (stored in the buffer when it was received) is pushed into the communications status code byte-offset.

An overrun occurs only when the 17th character is received into the already-full buffer. The 16th character is replaced with the 17th. When the character that caused the overrun is retrieved into the character returned byte-offset, an overrun will be indicated in communication status code.

If there are no characters "waiting" in the buffer, the communications status code will reflect the current status of the serial interface.

Entry Conditions

Byte-offset	Ø-1	98

Exit Conditions

Byte-offset		
	2-3	error code
	6-7	character returned (if any)
	8-9	communications status code
	1Ø-1.	character received status
		Ø = character received
		non-zero = character not received

The 8-bit grouping of flags returned in the communication status code are:

Bit	Meaning
Ø	Clear to Send not present
1	Unused
2	Transmitter busy
3	Modem data carrier not present
4	Parity error in byte now being received
5	Data overflow due to a byte now being received
6	Framing error in a byte now being received
7	Break Sequence now being received

Example

```
            .
            .
BRCV        LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#BRCV SVC NUMBER
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
             .
             .
SVC BLOCK
            RDATAB      32,Ø
BRCV SVC NUMBER
            EQUW        98
```

BTX **Function Code 99**
Channel B Transmit

Outputs a single character to serial channel B.

TRSDOS-16 sets up A/BRCV, A/BTX and A/BCTL when you initialize channel A/B with RS232C. If you call any of these routines while the channel is not initialized (active), you will get an error return code of 1 (no function code exists).

Data bytes will be transmitted even if no carrier is present. You must check the communication status code and character transmitted status for error conditions.

Entry Conditions

Byte-offset	Ø-1	99
	6-7	character to be sent

Exit Conditions

Byte-offset	2-3	error code
	8-9	communications status code
	1Ø-11	character transmitted status Ø = character transmitted non-zero = character not transmitted

The 8-bit grouping of flags returned in the communications status code are:

Bit	Meaning
Ø	Clear to Send not present
1	Unused
2	Transmitter busy
3	Modem data carrier not present
4	Parity error in byte now being received *
5	Data overflow due to a byte now being received *
6	Framing error in a byte now being received *
7	Break Sequence now being received *

* Bits 4 - 7 are only used when the character was not sent

Example

```
BTX         LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#BTX SVC NUMBER
            MOVW        6@AØ,#CHAR TO SEND
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
              .
              .
SVC BLOCK
            RDATAB      32,Ø
BTX SVC NUMBER
            EQUW        99
CHAR TO SEND
            TEXT        'G
```

CLOSE **Function Code 42**
Close Disk Files

Terminates output to a specified file. This SVC first writes all unsaved data to disk, then updates the directory.

Entry Conditions

Byte-offset	Ø-1	42
	6-7	file identification number

Exit Conditions

Byte-offset	2-3	error code

The file identification number is a unique number assigned to each file when it is opened.

Example

In this example the file identification number was previously stored in register A1.

```
          .
CLOSE     LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#CLOSE SVC NUMBER
          STW       .A1,6@AØ
          BRK       #Ø
          TESTW     2@AØ
          BNE       ERROR
          .
          .
SVC BLOCK
     RDATAB         32,Ø
CLOSE SVC NUMBER
     EQUW           42
```

CLOSEF **Function Code 133**
Close Files

Closes all open files, except for a currently executing DO-file, SPOOL file, or a file opened by OPENDO.

Entry Conditions

Byte-offset	Ø-1	133

Exit Conditions

Byte-offset	2-3	error code

Example

```
            .
            .
CLOSEF      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#CLOSEF SVC NUMBER
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
CLOSEF SVC NUMBER
            EQUW        265
```

CLRXIT **Function Code 257**
Clear User Memory and Jump to TRSDOS-16

Clears (writes binary zeroes) to user memory, then gives control to TRSDOS-16 Ready. If used with a DO-File, control proceeds to the next command in the DO-File.

Entry Conditions

Byte-offset	Ø-1	257

Exit Conditions

Byte-offset	2-3	error code

Example

```
          .
          .
CLRXIT    LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#CLRXIT SVC NUMBER
          BRK       #Ø
          .
          .
SVC BLOCK
     RDATAB         32,Ø
CLRXIT SVC NUMBER
     EQUW           257
```

CURSOR **Function Code 1Ø**
Position Cursor

Positions the cursor to specified screen coordinates. This routine treats ROW and COLUMN as Modulo 24 and Modulo 8Ø, respectively.

Entry Conditions

Byte-offset	Ø-1	1Ø
	6-7	row to position cursor
	8-9	column to position cursor

Exit Conditions

Byte-offset	2-3	error code

Example

```
            .
            .
CURSOR      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#CURSOR SVC NUMBER
            MOVW        6@AØ,#ROW POSITION
            MOVW        8@AØ,#COLUMN POSITION
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
CURSOR SVC NUMBER
            EQUW        1Ø
ROW POSITION
            EQUW        12
COLUMN POSITION
            EQUW        4Ø
```

DATE **Function Code 45**
Return Date String

Returns the time and date as a 26-byte ASCII string with eight fields.

CONTENTS OF TIME/DATE STRING (SAMPLE)

S A T	A P R	2 8	1 9 7 9	1 1 8	1 3 . 2 Ø . 4 2	4	5
NAME OF DAY	MON.	DAY OF MON.	YEAR	DAY OF YEAR	TIME	MON. #	DAY OF WEEK

For example:

SATAPR28197911813.2Ø.42Ø45

represents the data "Saturday, April 28, 1979, the 118th day of the year, 13:2Ø:42 hours, the fourth month of the year, the fifth day of the week".

Monday is considered day Ø. The date calculations are based on the Julian Calendar.

Entry Conditions

Byte-offset	Ø-1	45
	6-9	address of 26-byte buffer

Exit Conditions

Byte-offset	2-3	error code

Example

```
          .
          .
DATE      LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#DATE SVC NUMBER
          LDA       .A1,DATE BUFFER
          STL       .A1,6@AØ
          BRK       #Ø
          TESTW     2@AØ
```

```
            BNE         ERROR
            .
            .
            .
SVC BLOCK
     RDATAB             32,Ø
DATE SVC NUMBER
     EQUW               45
DATE BUFFER
     RDATAB             26,Ø
```

DEBUG **Function Code 259**
Load the Debugger

Turns the debugger on and off. This call is valid only if DEBUG was configured. See Appendix B of this section for more information on the Configuration Command File.

Entry Conditions

Byte-offset	Ø-1	259
	6-7	function code

Exit Conditions

Byte-offset	2-3	error code

Valid function codes are:

FUNCTION CODE	FUNCTION
Ø	Turns DEBUG off
1	Turns DEBUG on
2	Enters DEBUG

If you use function code 2, the Debugger will load with the PC register set to the current instruction address plus two.

Example

```
          .
          .
DEBUG     LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#DEBUG SVC NUMBER
          MOVW      6@AØ,#DEBUG ON
          BRK       #Ø
          TESTW     2@AØ
          BNE       ERROR
          .
          .
```

```
SVC BLOCK
          RDATAB     32,Ø
DEBUG SVC NUMBER
          EQUW       259
DEBUG ON
          EQUW       1
```

DIRRD **Function Code 35**
Direct Read

Reads a specified record of an FLR (fixed-length record) file.

If you have a VLR (variable-length record) file, you can use DIRRD to read only the first record (Ø) or the end of file (-1)

Entry Conditions

Byte-offset	Ø-1	35
	6-7	file identification number
	8-11	record buffer address
	12-15	record number Ø = position to beginning of file H'FFFFFFFF = position to end of file
	16	record lock flag

Exit Conditions

Byte-offset	2-3	error code

The file identification number is a unique number assigned to each file when it is opened.

The record buffer address is a 32 bit address that points to the beginning of a 256-byte Record Buffer; that is where the record will be placed after the disk read.

The record number is a 32 bit number specifying the desired record number.

If the record lock flag is non-zero, the specified record remains locked until the user program performs an UNLOCK on the record.

If the record lock flag is zero, the record is not locked.

Example

Before executing this program, store the File Identification Number in register A1 and the Record Number in register A3.

```
            .
            .
READ        LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#DIRRD SVC NUMBER
            STW         .A1,6@AØ
            LDA         .A2,RECORD BUFFER
            STL         .A2,8@AØ
            STL         .A3,12@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
            .
SVC BLOCK
            RDATAB      32,Ø
DIRRD SVC NUMBER
            EQUW        35
RECORD BUFFER
            RDATAB      256,Ø
```

DIRWR **Functon Code 44**
Direct Write

Writes a specified record in an FLR (fixed length record) file.

With a VLR (variable length record) files, you can use DIRWR to write only the first record (Ø) or the end of the file (-1).

Entry Conditions

Byte-offset	Ø-1	44
	6-7	file identification number
	8-11	record buffer address
	12-15	record number
		Ø = position to beginning of file
		H'FFFFFFFF = position to end of file

Exit Conditions

Byte-offset	2-3	error code

The file identification number is a unique number assigned to each file when it is opened.

The record buffer address is a 32-bit address that points to the beginning of a 256-byte Record Buffer; that is where the record will be placed after the disk read.

The record number is a 32-bit number specifying the desired record number.

Example

Before executing this program, store the Identification Number in register A1, Record Address in register A2, and Record Number in register A3.

```
          .
          .
WRITE     LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#DIRWR SVC NUMBER
```

```
            STW         .A1,6@AØ
            LDA         .A2,RECORD BUFFER
            STL         .A2,8@AØ
            STL         .A3,12@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
            .
SVC BLOCK
     RDATAB             32,Ø
DIRWR SVC NUMBER
            EQUW        44
RECORD BUFFER
            RDATAB      256,Ø
```

DISMOUNT **Function Code 139**
Dismount Disk

Logically disconnects a MOUNTed disk device. This command is valid for drives Ø-3 and 5-7 for hard disk users and drives 1-3 for floppy disk users. The system disk may not be DISMOUNTed.

A DISMOUNT/MOUNT sequence is required for each diskette swap. (see Chapter 1)

You should make sure all files are closed before executing a DISMOUNT/MOUNT sequence

Entry Conditions

Byte-offset	Ø-1	139
	6-9	device name

Exit Conditions

Byte-offset	2-3	error code

Valid device names are:

FDØØ = floppy drive Ø
FDØ1 = floppy drive 1
FDØ2 = floppy drive 2
FDØ3 = floppy drive 3
HDØØ = hard drive 4
HDØ1 = hard drive 5
HDØ2 = hard drive 6
HDØ3 = hard drive 7

Example

```
          .
          .
DISMNT    LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#DISMNT SVC NUMBER
          LDA.      .A1,DEVICE NAME
          MOVL      6@AØ,@A1
          BRK       #Ø
          TESTW     2@AØ
          BNE       ERROR
          .
          .
```

```
SVC BLOCK
          RDATAB     32,Ø
DISMNT SVC NUMBER
          EQUW       139
DEVICE NAME
          TEXT       'FDØ1'
```

DOSCMD **Function Code 27Ø**
Execute TRSDOS-16 Command

Passes a command string to the TRSDOS-16 Ready mode for execution. After execution is complete, control returns to TRSDOS-16 Ready.

This command alters the buffer used by the MOVBUF SVC.

When chaining programs, do not use DOSCMD to execute the programs. Use EXECUT instead.

Entry Conditions

Byte-offset	Ø-1	27Ø
	6-7	length of string
	8-11	address of string

Exit Conditions

Byte-offset	2-3	error code

Example

```
            .
            .
DOSCMD      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#DOSCMD SVC NUMBER
            MOVW        6@AØ,#STRING LENGTH
            LDA         .A1,STRING
            STL         .A1,8@AØ
            BRK         #Ø
            .
            .
SVC BLOCK
            RDATAB      32,Ø
DOSCMD SVC NUMBER
            EQUW        27Ø
STRING LENGTH
            EQUW        5
STRING
            TEXT        'DIR 3'
```

DUMP **Function Code 13Ø**
Dump Memory to Disk

Writes a 68ØØØ format program from memory to diskette. The Dump replaces any existing file with the same name.

Entry Conditions

Byte-offset	Ø-1	13Ø
	6-9	filespec address
	1Ø-13	start dump address
	14-17	end dump address
	18-21	relocation address
	22-25	transfer Address

Exit Conditions

Byte-offset	2-3	error code

filespec address specifies the memory location of a standard TRSDOS-16 file specification. The filename is an ASCII string terminated by a carriage return.

start dump address and end dump address are the beginning and ending locations in memory where the program resides.

relocation address is the starting location where you want the file to load. If you don't want to relocate the file, set the relocation address at Ø.

transfer address is the program entry point (the address of the first instruction to execute). If you specify a relocation address, this address offsets the transfer address.

Note: all addresses must be an even value.

Example

```
          .
DUMP      LDA       .AØ,SVC BLOCK
          MOVW      @AØ,DUMP SVC NUMBER
```

```
            LDA         .A1,FILE NAME
            STL         .A1,6@AØ
            MOVL        1Ø@AØ,#START AND TRANS ADDRESS
            MOVL        14@AØ,#END DUMP ADDRESS
            MOVL        18@AØ,#RELOCATION ADDRESS
            MOVL        22@AØ,#START AND TRANS ADDRESS
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR


SVC BLOCK
            RDATAB      32,Ø
DUMP SVC NUMBER
            EQUW        13Ø
FILE NAME
            TEXT        'TEST/SRC'
            DATAB       H'ØD
START AND TRANS ADDRESS
            EQUL        H'6ØØØØ
END DUMP ADDRESS
            EQUL        H'6ØØØØ
RELOCATION ADDRESS
            EQUL        Ø
```

ERRMSG **Function Code 52**
Error Message

In response to a requested error number, this routine returns an 8Ø-byte descriptive error message to a specified buffer area.

Entry Conditions

Byte-offset	Ø-1	52
	6-7	error number
	8-11	message buffer address

Exit Conditions

Byte-offset	2-3	error code

The error number range is Ø to 255, inclusive.

Example

```
            .
            .
ERRMSG      LDW         .A1,2@AØ
            LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#ERRMSG SVC NUMBER
            STW         .A1,6@AØ
            LDA         .A1,MESSAGE BUFFER
            STL         .A1,8@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         JP2DOS
            CALL        VIDEO PRINT LINE ROUTINE
            .
            .
SVC BLOCK
            RDATAB      32,Ø
ERRMSG SVC NUMBER
            EQUW        52
MESSAGE BUFFER
            RDATAB      8Ø,Ø
```

ERROR **Function Code 39**

Display Error Number

Displays the message ERROR followed by the specified error code at the current cursor position.

Entry Conditions

Byte-offset	Ø-1	39
	6-7	error number

Exit Conditions

Byte-offset	2-3	error code

The error number is a Ø - 255 code of the message you want displayed.

Example

```
            .
            .
ERROR       LDW         .A1,2@AØ
            LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#ERROR SVC NUMBER
            STW         .A1,6@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         JP2DOS
            .
            .
SVC BLOCK
            RDATAB      32,Ø
ERROR SVC NUMBER
            EQUW        39
```

EXECUT **Function Code 263**
Execute Program

Begins program execution in user memory.

Entry Conditions

Byte-offset	Ø-1	263
	6-7	ØØ (Reserved)
	8-11	filespec address

Exit Conditions

Byte-offset	2-3	error code

The filespec address is the 32-bit address of a valid TRSDOS-16 filespec terminated by a carriage return.

Example

```
            .
            .
EXECUTE     LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#EXECUTE SVC NUMBER
            MOVW        6@AØ,#ØØ
            LDA         .A1,FILENAME
            STL         .A1,8@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         JP2DOS
            .
            .
SVC BLOCK
            RDATAB      32,Ø
EXECUTE SVC NUMBER
            EQUW        263
FILENAME
            TEXT        'TEST/CMD'
            DATAB       H'ØD
```

HLDKEY **Function Code 29**
Enable Hold Key

Suspends and restarts terminal output whenever you press the <HOLD> key.

This SVC must first be enabled. Then you must periodically call HLDKEY to check if the <HOLD> key has been depressed. This places the program in control of the pause checking.

Note that execution of certain DOSCMDs might alter the HLDKEY status. This will happen if the TRSDOS-16 command issues HLDKEY requests.

Entry Conditions

Byte-offset	Ø-1	29
	6-7	function code

Exit Conditions

Byte-offset	2-3	error code

The HLDKEY function codes are:

Code	Meaning
Ø	Turn off HOLD processor. Pressing <HOLD> generates H'ØØ'.
1	Turn on HOLD processor. <HOLD> key doesn't generate keyboard data, it is intercepted by TRSDOS-16.
2	Check for <HOLD> key. If pressed, wait until pressed again.

Example

```
          .
          .
HLDKEY    LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#HLDKEY SVC NUMBER
          MOVW      6@AØ,#HOLD KEY FUNCTION CODE
```

```
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR

* CHECK TO SEE IF HOLD KEY PRESSED
            LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#HLDKEY SVC NUMBER
            MOVW        6@AØ,#CHECK IF HOLD KEY WAS PRESSED
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
              .
              .
SVC BLOCK
            RDATAB      32,Ø
HDLKEY SVC NUMBER
            EQUW        29
HOLD KEY FUNCTION CODE
            EQUW        1
CHECK IF HOLD KEY PRESSED
            EQUW        2
```

JP2DOS **Function Code 264**
Jump to TRSDOS-16

After closing all open files and performing system housekeeping, returns control to TRSDOS-16 Ready.

If a program under the control of a DO-file executes this routine, control proceeds to the next command in the DO-file.

Entry Conditions

Byte-offset	Ø-1	264
	6-7	ØØ (Reserved)

Exit Conditions

Byte-offset	2-3	error code

Example

```
            .
            .
JP2DOS      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#JP2DOS SVC NUMBER
            BRK         #Ø
             .
             .
SVC BLOCK
            RDATAB      32,Ø
JP2DOS SVC NUMBER
            EQUW        264
```

KBCHAR **Function Code 4**
Keyboard Character

Checks the keyboard for a new character entry.

If characters are present in the key-ahead buffer, the first character in the buffer will be returned in the returned character byte-offset.

The <BREAK> key is masked from the user -- it will never be returned, since it is intercepted by the System. If a SETBRK routine is enabled, control passes to the processing program (see SETBRK) whenever <BREAK> is pressed. Otherwise, control will pass to TRSDOS-16 Ready.

Entry Conditions

Byte-offset	Ø-1	4
	6-7	wait for character flag

Exit Conditions

Byte-offset	2-3	error code
	6-7	character present flag
	8-9	returned character

If you want the SVC to return only when a character is detected, set the wait for character flag to non-zero. A zero in this byte-offset causes the SVC to return, with or without a character, immediately after the keyboard buffer is checked.

A character present flag of non-zero means the routine has returned with a valid ASCII character in the returned character position.

If it returns without a character, (when the SVC is set not to wait for a character) the routine clears the character present flag, but doesn't change the contents of the returned character byte-offset.

Example

This program returns the character most recently pressed and then calls PRINT CHARACTER ROUTINE, which uses either the VDCHAR or PRCHAR routines to print it.

```
            .
            .
KBCHAR      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#KBCHAR SVC NUMBER
            MOVW        6@AØ,#WAIT FOR CHARACTER FLAG
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            CALL        PRINT CHARACTER ROUTINE
            .
            .
SVC BLOCK
            RDATAB      32,Ø
KBCHAR SVC NUMBER
            EQUW        4
WAIT FOR CHARACTER FLAG
            EQUW        1
```

KBINIT **Function Code 1**
Keyboard Initialize

Initializes the keyboard input driver. You might want to call this SVC before starting keyboard input to clear previous keystrokes and the key-ahead buffer. TRSDOS-16 does this automatically at start-up.

Entry Conditions
Byte-offset Ø-1

Exit Conditions
Byte-offset 2-3 error code

Example

```
            .
            .
KBINIT      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#KBINIT SVC NUMBER
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
KBINIT SVC NUMBER
            EQUW        1
```

KBLINE **Function Code 5**
Keyboard Line

Inputs a line from the keyboard to a buffer and echoes the line to the Display, starting at the current cursor position. As it receives and displays each character, the routine advances the cursor to the next position.

When you enter this routine, the input buffer contains a string of periods which it sends to the display. This string is for your convenience to indicate the length of the input field.

The keyboard line ends when you press <ENTER> or when the routine fills the input buffer. When you terminate line input, the routine always sends a carriage return and an erase-to-end-of-screen command to the Display. It stores a carriage return as a character input only if you actually press <ENTER>.

If you type a control code not listed below, the routine places it in the buffer and represents it on the display with the ± symbol.

KEY	HEX CODE	FUNCTION
BACKSPACE	Ø8	Backspaces the cursor and erases a character.
ENTER	ØD	Terminates line. Clears trailing periods on Display
CTRL W	17	Fills remainder of input buffer with blanks, blanks remainder of Display line
CTRL X	18	Fills remainder of input buffer with blanks, blanks to end of Display.
ESC	1B	Reinitializes input function by filling input buffer with periods and restoring cursor to original position.
<-	1C	Backspaces the cursor to allow editing of line. Does not erase characters.
->	1D	Advances the cursor to allow editing of line. Does not erase characters.

Entry Conditions

Byte-offset		
	Ø-1	5
	6-7	maximum number of characters to receive (Ø - 255)
	8-11	input buffer address

Exit Conditions

Byte-offset		
	2-3	error code
	12-13	actual number of characters input
	14-15	terminating character of input H'ØD if line was terminated with a carriage return, Ø if buffer was filled without a carriage return

Example

```
            .
            .
KBLINE      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#KBLINE SVC NUMBER
            MOVW        6@AØ,#MAXIMUM INPUT COUNT
            LDA         .A1,KEYBD BUFFER
            STL         .A1,8@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
KBLINE SVC NUMBER
            EQUW        5
MAXIMUM INPUT COUNT
            EQUW        8Ø
KEYBD BUFFER
            RDATAB      8Ø,Ø
```

KILL **Function Code 41**
Delete File from Directory

Deletes the specified file from the directory. A file must be closed for you to KILL it.

Entry Conditions

Byte-offset	Ø-1	41
	6-9	filespec address

Exit Conditions

Byte-offset	2-3	error code

The filespec address is the 32-bit address of a valid TRSDOS-16 filespec terminated by a carriage return.

Example

```
            .
            .
KILL        LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#KILL SVC NUMBER
            LDA         .A1,FILE NAME
            STL         .A1,6@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
KILL SVC NUMBER
            EQUW        41
FILE NAME
            TEXT        'TEST/SRC'
            DATAB       H'ØD
```

LOAD **Function Code 131**

LOAD's a 68ØØØ program into user memory.

Entry Conditions

Byte-offset	Ø-1	131
	6-9	filespec address
	1Ø-13	low bound address (only if program is position independent, otherwise must be zero)
	14-17	high bound address (only if program is position independent otherwise zero allows all of memory)

Exit Conditions

Byte-offset	2-3	error code
	1Ø-13	start load address
	14-17	address of last byte loaded
	18-21	transfer address

Supply a low bounds address and high bounds address only if your program does NOT have any absolute memory references.

filespec address is the 32-bit address of a valid TRSDOS-16 program filespec terminated by a carriage return.

low bounds address is the logical address you want the program loaded at. Specifying zero means the entire user memory is available. If this number is non-zero then the program MUST be position-independent.

high bounds address is the logical address of the last byte available for the program being loaded. Specifying zero allows entire user memory.

start load address is the location in memory of the first byte loaded.

address of last byte loaded is the highest address of the loaded program.

transfer address is the program entry point (address of first instruction to be executed). If the program is nonexecutable ("load-only"), the byte-offset contains a -1.

Note: all addresses must be an even value.

Example

```
            .
            .
LOAD        LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#LOAD SVC NUMBER
            LDA         .A1,FILE NAME
            STL         .A1,6@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
LOAD SVC NUMBER
            EQUW        131
FILE NAME
            TEXT        'TEST/SRC'
            DATAB       H'ØD
```

LOCATE **Function Code 33**
Locate Record

Returns the number of the current record, i.e., the last record accessed. You can call this routine only with FLR (fixed-length Record) files.

Entry Conditions

Byte-offset	Ø-1	33
	6-7	file identification number

Exit Conditions

Byte-offset	2-3	error code
	8-11	record number

The file identification number is a unique number assigned to each file when it is opened.

The record number is a 32 bit number specifying the desired record number. The record number will be zero if the file was just opened.

Example

Prior to execution of this routine, store the file identification number in register A1.

```
            .
            .
LOCATE      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#LOCATE SVC NUMBER
            STW         .A1,6@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
LOCATE SVC NUMBER
            EQUW        33
```

MOUNT **Function Code 138**
Mount Device

Logically connects a device to the system. The system cannot access an unMOUNTed device. You must mount a diskette prior to any I/O attempts.

A DISMOUNT/MOUNT sequence is required for each floppy diskette swap.

You should make sure all files are closed before executing a DISMOUNT/MOUNT sequence.

Entry Conditions

Byte-offset	Ø-1	138
	6-9	device name

Exit Conditions

Byte-offset	2-3	error code

Valid device names are:

FDØØ = floppy drive Ø
FDØ1 = floppy drive 1
FDØ2 = floppy drive 2
FDØ3 = floppy drive 3
HDØØ = hard drive 4
HDØ1 = hard drive 5
HDØ2 = hard drive 6
HDØ3 = hard drive 7

Example

```
          .
          .
MOUNT     LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#MOUNT SVC NUMBER
          LDA       .A1,DEVICE NAME
          MOVL      6@AØ,@A1
          BRK       #Ø
          TESTW     2@AØ
          BNE       ERROR
          .
          .
```

```
SVC BLOCK
          RDATAB     32,Ø
MOUNT SVC NUMBER
          EQUW       138
DEVICE NAME
          TEXT       'FDØ1'
```

MOVBUF **Function Code 267**
Move Buffer

Either retrieves or stores an 8Ø-byte buffer outside the user's memory. This buffer serves two purposes:

1. To pass parameters when chaining two machine language programs.

   (When chaining programs, do not use DOSCMD to execute the programs. Use EXECUT instead.)

2. To retrieve a TRSDOS-16 command line. When TRSDOS-16 starts a user program, it stores the command line that invoked the job in this area.

   (When TRSDOS-16 stores a command line it terminates it with a carriage-return character. However, when MOVBUF retrieves a command line 8Ø characters long, it does not retrieve the carriage return, implied 81st character.)

There are two ways to alter this buffer:

1. A MOVBUF request from memory to buffer overwrites anything in the buffer. Note that 8Ø bytes (5Ø Hex bytes) are always moved.

2. A DOSCMD request does an implied move into the buffer before executing the request. Anything previously stored in the buffer is lost.

Entry Conditions

Byte-offset	Ø-1	267
	6-9	user buffer address
	1Ø-11	switch
		Ø = Retrieve
		non-zero = Store

Exit Conditions

Byte-offset	2-3	error code

Example

```
            .
            .
MOVBUF      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#MOVBUF SVC NUMBER
            LDA         .A1,BUFFER
            STL         .A1,6@AØ
            MOVW        1Ø@AØ,#FETCH SWITCH
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
MOVBUF SVC NUMBER
            EQUW        267
BUFFER
            RDATAB      8Ø,Ø
FETCH SWITCH
            EQUW        Ø
```

OPEN **Function Code 4Ø**
Open File

Creates new files and opens existing files.

Once a given file has been opened, OPEN assigns a unique file identification number to the file. This number is used in all other file processing SVC's to designate that specific file.

Entry Conditions

Byte-offset	Ø-1	4Ø
	6-9	filespec address
	1Ø-13	parameter list address

Exit Conditions

Byte-offset	2-3	error code
	14-15	file identification number

The filespec address is the 32-bit address of a valid TRSDOS-16 filespec terminated by a carriage return.

The parameter list address is the address of a 5-bytes field (Ø-4) which contains:

Byte	Contents
ØØ	access code "R" Read access to the file "W" Read/Write (data files) "P" Read/Write access to Z8Ø program files
Ø1	record length in bytes
Ø2	file type "F" fixed "V" variable
Ø3	creation code (Ø-3 -- see table)
Ø4	user attribute byte

The creation code specifies the way TRSDOS-16 opens the file:

Code	Function
Ø	Open the file only if it already exists. Do not create a new file. The record length and end of file pointers are not reset. Exclusive access only.
1	Creates a new file. Does not open an existing file (returns an error if a file of the same name already exists on the specified drive). Resets the record length and end of file. Exclusive access only.
2	Open a new file. If a file of the same name already exists, overwrite it. Resets record length and end of file. Exclusive access only.
3	Opens an existing file for shared access. The file MUST already exist (same as mode Ø).

The user attribute byte allows you to give your own identification number to certain types of data files

You can use Ø or any number from 32 - 255 for this value.

TRSDOS-16 will not examine this user attribute; it is solely for your convenience.

You can assign this user attribute only to files you open with a creation code of 1 or 2. Files opened with a creation code of Ø or 3 will retain the file's previously assigned user attribute. All files created with the CREATE command will have a user attribute value of zero.

The file identification number is a unique number assigned to each file when it is opened.

Example

```
            .
            .
OPEN        LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#OPEN SVC NUMBER
            LDA         .Al,FILE NAME
            STL         .Al,6@AØ
            LDA         .Al,PARAM LIST
            MOVW        @Al,#WRITE ACCESS
            MOVW        1@Al,#RECORD LENGTH
            MOVW        2@Al,#FIXED FILE
            MOVW        3@Al,#OPEN ONLY IF EXISTS
            MOVW        4@al,#USER ATTRIB
            STL         .Al,1Ø@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
OPEN SVC NUMBER
            EQUW        4Ø
FILE NAME
            TEXT        'TEST/SRC'
            DATAB       H'ØD
PARAM LIST
            RDATAB      5,Ø
WRITE ACCESS
            EQUB        'P
RECORD LENTGH
            EQUB        8Ø
FIXED FILE
            EQUB        'F
OPEN ONLY IF EXISTS
            EQUB        Ø
USER ATTRIB
            EQUB        66
```

OPENDO **Function Code 14Ø**
Open DO File

Creates a special file or opens an existing one.

OPENDO opens a special file that will not be closed by returning to TRSDOS-16 Ready or executing the CLOSEF SVC.

OPENDO is useful for chaining programs.

Once a given file has been opened, OPENDO assigns a unique file identification number to the file. This number is used in all other file processing SVC's to designate that specific file.

Only one file may be opened with OPENDO at a time. Therefore, this call cannot be made if a DO file is active.

Entry Conditions

Byte-offset	Ø-1	14Ø
	6-9	filespec address
	1Ø-13	parameter list address

Exit Conditions

Byte-offset	2-3	error code
	14-15	file identification number

The filespec address is the 32-bit address of a valid TRSDOS-16 filespec terminated by a carriage return.

The parameter list address is the address of a 5-bytes field (Ø-4) which contains:

Byte	Contents
ØØ	access code
	"R" Read access to the file
	"W" Read/Write (data files)
	"P" Read/Write access to Z8Ø program files

Byte	Contents
Ø1	record length in bytes
Ø2	file type "F" fixed "V" variable
Ø3	creation code (Ø-3 -- see table)
Ø4	user attribute byte

The creation code specifies the way TRSDOS-16 opens the file:

Code	Function
Ø	Open the file only if it already exists. Do not create a new file. The record length and end of file pointers are not reset. Exclusive access only.
1	Creates a new file. Does not open an existing file (returns an error if a file of the same name already exists on the specified drive). Resets the record length and end of file. Exclusive access only.
2	Open a new file. If a file of the same name already exists, overwrite it. Resets record length and end of file. Exclusive access only.
3	Opens an existing file for shared access. The file MUST already exist (same as mode Ø).

The user attribute byte allows you to give your own identification number to certain types of data files.

You can use Ø or any number from 32 - 255 for this value.

TRSDOS-16 will not examine this user attribute; it is solely for your convenience.

You can assign this user attribute only to files you open with a creation code of 1 or 2. Files opened with a

creation code of Ø or 3 will retain the file's previously assigned user attribute. All files created with the CREATE command will have a user attribute value of zero.

The file identification number is a unique number assigned to each file when it is opened.

Example

```
            .
OPENDO      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#OPENDO SVC NUMBER
            LDA         .Al,FILE NAME
            STL         .Al,6@AØ
            LDA         .Al,PARAM LIST
            MOVW        @Al,#WRITE ACCESS
            MOVW        1@Al,#RECORD LENGTH
            MOVW        2@Al,#FIXED FILE
            MOVW        3@Al,#OPEN ONLY IF EXISTS
            MOVW        4@al,#USER ATTRIB
            STL         .Al,1Ø@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR


SVC BLOCK
            RDATAB      32,Ø
OPENDO SVC NUMBER
            EQUW        14Ø
FILE NAME
            TEXT        'TEST/SRC'
            DATAB       H'ØD
PARAM LIST
            RDATAB      5,Ø
WRITE ACCESS
            EQUB        'P
RECORD LENTGH
            EQUB        8Ø
FIXED FILE
            EQUB        'F
OPEN ONLY IF EXISTS
            EQUB        Ø
USER ATTRIB
            EQUB        66
```

PRCHAR **Function Code 18**
Print Character

Sends one character to the Printer's buffer.

Note: Most printers don't print until their buffer is filled or until they receive a carriage return. (See your printer's manual for information.)

Normally, TRSDOS-16 intercepts certain codes and does not send them directly to the printer. There are several ways to override some or all of these character translations. See PRINIT and PRCTRL SVCs.

While the serial printer option is selected, allowing printer output to Channel B, two INPUT characters are recognized from Channel B. These will affect all serial printer output operations:

ASCII Name	Hex Code	Result
DC3, "CTRL-S"	13	Pause Printing
DC1, "CTRL-Q"	11	Resume Printing

INTERCEPTED CODES

ASCII NAME	HEX CODE	RESULT TO PRINTER
Tab	Ø9	From one to eight spaces are sent to provide a tab function.
Vertical Tab	ØB	Same as form feed below.
Form Feed	ØC	TRSDOS-16 sends enough carriage returns or line feeds to the printer to advance the paper to the next top of form.

ASCII NAME	HEX CODE	RESULT TO PRINTER
Carriage Return	ØD	When the current line is empty, (no characters printed since the last carriage return or line feed), TRSDOS-16 translates this as a line feed to allow correct operation of Radio Shack printers. In the auto line feed mode, H'ØA' is sent after every H'ØD'.
Special	8D	TRSDOS-16 sends a carriage return to the printer. In the auto line feed mode, using this code allows you to send a carriage return without a line feed.

Entry Conditions

Byte-offset	Ø-1	18
	6-7	character to output

Exit Conditions

Byte-offset	2-3	error code

Example

Before executing this program, load register A1 with the character from byte-offset 8 of the KBCHAR SVC routine.

```
            .
            .
PRCHAR      LDW         .A1,8@AØ
            LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#PRCHAR SVC NUMBER
            STW         .A1,6@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
PRCHAR SVC NUMBER
            EQUW        18
```

PRCTRL **Function Code 95**
Control Printer Operation

Provides you with various printer options and lets you check the status of printer-related functions.

If you are using the spooler's capture function, the captured data is sent directly to the capture-file -- regardless of what control settings are selected. If you are using the spooler's background printing function, the printed data is interpreted according to the currently selected control settings. See SPOOL for details.

You can operate with two printers, by switching between the serial and parallel output modes. One printer can use the automatic control features of TRSDOS (auto form feed, etc.), while the other printer is controlled by the program. This is necessary since **only one set of control counters is maintained.**

Entry Conditions

Byte-offset	Ø-1	95
	6-7	option code (See Table 2)
	8-9	option value (applicable only with option codes 3 and 4)

Exit Conditions

Byte-offset	2-3	error code
	1Ø-11	physical page length
	12-13	logical page length
	14-15	logical line length
	16-17	number of characters printed on present line
	18-19	number of lines on present page

The option code is a 16-bit value used to select the print control option you wish.

The option value is used by option codes 3 and 4 which allow you to reset the current line count or the current character count. Put the new values in this byte-offset.

Available option codes are:

CODE	OPTION
Ø	Get printer status only (see Exit Conditions)
1	Select serial printer driver (you must initialize channel B first)
2	Select parallel printer driver
3	Reset current line count to the value contained in option value *
4	Reset current character count on current line to value contained in option value *
5	Begin transparent mode *
6	End transparent mode *
7	Begin dummy mode *
8	End dummy mode *
9	Begin auto line-feed after carriage return
1Ø	End auto line-feed after carriage return

* EXPLANATION OF OPTIONS

EXPLANATION OF OPTIONS

Serial/Parallel Printer Option

While the serial printer option is selected, allowing printer output to Channel B, two INPUT characters are recognized from Channel B. These will affect all serial printer output operations:

ASCII Name	Hex Code	Result
DC3, "CTRL-S"	13	Pause Printing
DC1, "CTRL-Q"	11	Resume Printing

Line count/Character count

TRSDOS-16 maintains a single line counter and a single character counter.

Transparent Mode

The transparent mode overrides all data translation. All data bytes go directly to the printer; TRSDOS-16 does not examine the contents or update the line and character counts.

Normally, TRSDOS-16 "intercepts" certain control characters and interprets them. In this way, TRSDOS-16 provides printer-related features which may not be available from the printer

For example, tabs (H'Ø9') are intercepted so that TRSDOS-16 can send the appropriate number of spaces to provide the tab function. Form feeds (H'ØC' or H'ØB') are intercepted so that TRSDOS-16 may advance the paper to the top of the next form.

Special settings of the printer initialization values can override individual code translation. See SVC PRINIT for details.

Dummy Output Mode

The dummy mode "throws away" all printer output and returns with a "good" (Z flag set) return code. During dummy mode operation, the line count and character count remain unchanged.

Auto Line Feed

Normally, the TRSDOS-16 printer driver doesn't output line feeds after carriage returns because most Radio Shack printers do an automatic line-feed after every carriage return.

If your printer does not perform automatic line-feeds after carriage returns, you may enable the TRSDOS-16 auto line-feed function.

While the function is enabled, TRSDOS-16 outputs a line-feed after each carriage return. This is true in all modes, including the transparent mode.

Note: In the auto line-feed mode, you can send a carriage return with no line-feed by outputting the code H'8D'. If the printer can overprint, this code returns the carriage without advancing the paper.

PRECEDENCE OF OPTIONS

The priority of the available options are (in descending order):

Dummy Mode
Auto Line-Feed after Carriage Return
Transparent Mode
Normal Mode

Example

```
            .
            .
PRCTRL      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#PRCTRL SVC NUMBER
            MOVW        6@AØ,#GET STATUS OPTION
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
             .
             .
SVC BLOCK
            RDATAB      32,Ø
PRCTRL SVC NUMBER
            EQUW        95
GET STATUS OPTION
            EQUW        Ø
```

PRINIT **Function Code 17**
Printer Initialization

Initializes the printer driver.

It does **not** advance the printer paper and does **not** check printer status or availability. It will operate even if the printer is offline.

Entry Conditions

Byte-offset	Ø-1	17
	6-7	physical page length
	8-9	logical page length
	1Ø-11	logical line length

Exit Conditions

Byte-offset	2-3	error code

physical page length is the number of lines that can be printed on one page. It is normally 66.

logical page length is the number of lines you want printed on each page. It must be less than physical page length. After that number of lines has been printed, (logical page length < physical page length) TRSDOS-16 will send a top-of-page request to the printer (form feed).

logical line length is the maximum number of characters to be printed on each line. Once that number has been reached, the line will be printed and a new line is begun (wrap-around).

Notes:

1. If logical page length and physical page length are equal and non-zero, TRSDOS-16 will not do an end-of-page skip. It will translate form feed into the correct number of line feeds necessary to get to the top of the next page.

2. If either logical page length or physical page length are zero, the other **must** also be zero. When both are equal to zero, form feeds (H'ØC) and vertical tabs (H'ØB) are not translated into line feeds, but are sent directly to the printer.

3. If logical line length is zero, then tab characters (H'Ø9) will not be translated into spaces, but sent directly to the printer. TRSDOS-16 will still maintain an internal character count, with tabs counting as one character.

4. On Exit, current character count and line-in-page count are reset to zero.

5. TRSDOS-16 assumes the printer is at top-of-page when this SVC is executed.

6. DUMMY and TRANSPARENT modes are reset to NORMAL. AUTO-LINEFEED and DUAL options are unaffected by this SVC. (See PRCTRL)

7. When the system is powered-up or reset, the following defaults are assumed:

physical page length	=	66
logical page length	=	6Ø
logical line length	=	132

The other options selected during initialization are:

Auto-Line Feed	=	off
Dummy Mode	=	off
Transparent Mode	=	off

Example

```
PRINIT    LDA      .AØ,SVC BLOCK
          MOVW     @AØ,#PRINIT SVC NUMBER
          MOVW     6@AØ,#PHY PAGE LENGTH
          MOVW     8@AØ,#LOG PAGE LENGTH
          MOVW     1Ø@AØ,#LOG LINE LENGTH
```

```
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR

              .
              .
SVC BLOCK
            RDATAB      32,Ø
PRINIT SVC NUMBER
            EQUW        17
PHY PAGE LENGTH
            EQUW        32
LOG PAGE LENGTH
            EQUW        3Ø
LOG LINE LENGTH
            EQUW        45
```

PRLINE **Function Code 19**
Print Line

Sends a line of characters to the printer's buffer. The line can include control characters as well as printable data.

See PRCHAR for a listing of intercepted codes.

Entry Conditions

Byte-offset	Ø-1	19
	6-7	length of line (Ø-255)
	8-9	terminator character to send after last character in buffer
	1Ø-13	address of buffer

Exit Conditions

Byte-offset	2-3	error code

Example

In this example, TRSDOS-16 retrieves the length of line, terminator character, and address of buffer from the KBLINE SVC routine.

```
            .
            .
          LDW       .A1,12@AØ  *Get length of line
          LDW       .A2,14@AØ  *Get terminator character
          LDA       .A3,KEYBD BUFFER   *Get buffer address
PRLINE    LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#PRLINE SVC NUMBER
          STW       .A1,6@AØ
          STW       .A2,8@AØ
          STL       .A3,1Ø@AØ
          BRK       #Ø
          TESTW     2@AØ
          BNE       ERROR
            .
            .
```

```
SVC BLOCK
          RDATAB     32,Ø
PRLINE SVC NUMBER
          EQUW       19
```

READNX **Function Code 34**
Read Next Record

Reads the next record after the current record (the last record accessed). If you have just opened the file, READNX reads the first record.

Entry Conditions

Byte-offset	Ø-1	34
	6-7	file identification number
	8-11	record buffer address
	12-13	record lock flag

Exit Conditions

Byte-offset	2-3	error code

The file identification number is a unique number assigned to each file when it is opened.

The record buffer address is a 32-bit address that points to the beginning of a 256-byte Record Buffer; that is where the record will be placed after the disk read.

If the record lock flag is non-zero, the specified record remains locked until the user program performs an UNLOCK on the record.

If the record lock flag is zero, the record is not locked.

Example

Prior to this routine, store the file identification number in register A1.

```
            .
            .
READNX      LDA       .AØ,SVC BLOCK
            MOVW      @AØ,#READNX SVC NUMBER
            STW       .A1,6@AØ
            LDA       .A2,RECORD BUFFER AREA
            STL       .A2,8@AØ
```

```
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
              .
              .
SVC BLOCK
            RDATAB      32,Ø
READNX SVC NUMBER
            EQUW        34
RECORD BUFFER AREA
            RDATAB      256,Ø
```

RENAME **Function Code 47**
Rename File

Changes the name and/or extension of a file.

This SVC cannot change the password. If the old filespec is password protected, you must specify the password and the new filespec retains the same password.

The new filespec can't refer to an existing file.

Entry Conditions

Byte-offset	Ø-1	47
	6-9	old filespec address
	1Ø-13	new filespec address

Exit Conditions

Byte-offset	2-3	error code

The filespec address is the 32-bit address of a valid TRSDOS-16 filespec terminated by a carriage return.

Example

```
            .
            .
RENAME      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#RENAME SVC NUMBER
            LDA         .A1,OLD NAME
            STL         .A1,6@AØ
            LDA         .A1,NEW NAME
            STL         .A1,1Ø@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
RENAME SVC NUMBER
            EQUW        47
```

```
OLD NAME
          TEXT      'TEST/SRC'
          DATAB     H'ØD
NEW NAME
          TEXT      'PROGRAM/SRC'
          DATAB     H'ØD
```

RESET **Function Code 129**
Reset Memory

It is the same as pressing the RESET switch.

Reset will not close an OPENDO file.

Entry Conditions

Byte-offset		
	Ø-1	256
	6-7	ØØ (Reserved)

Exit Conditions

Byte-offset		
	2-3	error code

Example

```
            .
            .
RESET       LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#RESET SVC NUMBER
            MOVW        6@AØ,#ØØ
            BRK         #Ø
            .
            .
SVC BLOCK
            RDATAB      32,Ø
RESET SVC NUMBER
            EQUW        129
```

RS232C **Function Code 55**
Initialize RS-232-C Channel

Sets up or disables either channel A or B.

This routine sets the standard RS-232C parameters, and defines a pair of supervisor calls for I/O to the specified channel. When you initialize Channel A, SVC's 96, 97, 1ØØ are defined; when you initialize Channel B, SVC's 98, 99, and 1Ø1 are defined. See ARCV, ATX, BRCV, BTX, ACTL and BCTL.

Entry Conditions

Byte-offset	Ø-1	55
	6-7	option value Ø = deactivate non-zero = activate
	8-11	parameter list address

Exit Conditions

Byte-offset	2-3	error code

Set the option value to Ø to deactivate the serial channel. Any non-zero value activates the channel. You must always deactivate the channel (turn it off) before you set the parameters.

The parameter list address is the address of a 6-byte field (Ø-5) which contains:

Byte	Contents
ØØ	Channel code 'A' or 'B' (ASCII)
Ø1	Baud Rate: 1 = 1ØØ baud 2 = 15Ø baud 3 = 3ØØ baud 4 = 6ØØ baud 5 = 12ØØ baud 6 = 24ØØ baud 7 = 48ØØ baud 8 = 96ØØ baud

Byte	Contents
	Some applications do not run well at the higher baud rates.
Ø2	Data Word Length (5 - 8 bits)
Ø3	Parity 'E' (even), 'O' (odd), or 'N' (none)
Ø4	Number of stop bits (1 or 2)
Ø5	End of list marker (binary Ø)

Example

```
            .
            .
RS232C      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#RS232C SVC NUMBER
            LDA         .A1,PARAM LIST
            MOVW        @A1,#CHANNEL A
            MOVW        1@A1,#BAUD RATE
            MOVW        2@A1,#DATA WORD LENGTH
            MOVW        3@A1,#EVEN PARITY
            MOVW        4@A1,#STOP BITS
            MOVW        5@A1,#END LIST
            STL         .A1,8@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
RS232C SVC NUMBER
            EQUW        55
PARAM LIST
            RDATAB      6,Ø
CHANNEL A
            EQUB        'A
BAUD RATE
            EQUB        3
DATA WORD LENGTH
            EQUB        7
EVEN PARITY
            EQUB        'E
```

```
STOP BITS
          EQUB      1
END LIST
          EQUB      ØØ
```

SETBRK **Function Code 269**
Set <BREAK>

Lets you enable the <BREAK> key by defining a <BREAK>-key processing program.

Whenever the <BREAK> key is pressed, your processing program takes over. On entry to the <BREAK> processing program, the return address of the interrupted routine is on the User Stack and can be returned to with an RTR instruction (return with restore).

See 'PROGRAMMING WITH USER INTERRUPTS' for further information.

Entry Conditions

Byte-offset	Ø-1	269
	6-7	switch Ø = Off (normal TRSDOS-16 break processing) 1 = On (user programmed break processing) 2 = Disable (TRSDOS-16 ignores <BREAK> key completely)
	8-11	transfer address Ø = Fetch non-zero = Store

Exit Conditions

Byte-offset	2-3	error code

The transfer address is the address you want to jump to when the <BREAK> key is pressed. It must be even.

Example

```
          .
          .
SETBRK    LDØ       .AØ,SVC BLOCK
          MOVW      @AØ,#SETBRK SVC NUMBER
          MOVW      6@AØ,#SWITCH ON
          MOVL      8@AØ,#TRANSFER ADDRESS
```

```
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
              .
              .
SVC BLOCK
            RDATAB      32,Ø
SETBRK SVC NUMBER
            EQUW        269
SWITCH ON
            EQUW        1
TRANSFER ADDRESS
            EQUL        H'8ØØØ
```

SETTRP **Function Code 266**
Sets a Trap Vector

Allows you to set or remove a trap vector. When a BRK is executed (either through the BRK-BRKV instructions or an error trap) control goes to the address specified by the user. See SETBRK and PROGRAMMING WITH USER INTERRUPTS for further information on interrupts.

Entry Conditions

Byte-offset	Ø-1	266
	6-7	function code
	8-9	vector number
	1Ø-13	vector address

Exit Conditions

Byte-offset	2-3	error code
	14-17	previous vector address

The function codes are:
Ø - Install (SET) trap
1 - Remove trap
2 - Remove all traps (except BRK Ø)

The available vector numbers are:

Vector #	Assignment
1 - 15	Trap vectors 1-15
16	Illegal Instruction
17	Zero Divide
18	CHK Instruction
19	TRAPV Instruction
2Ø	Privilege Violation
21	Line 1Ø1Ø Emulator
22	Line 1111 Emulator
23	Access out of Partition
24	Odd Address Error

Note that BRK Ø is reserved for implementation of supervisor calls.

The Emulators (Vectors 21 and 22) are illegal opcodes which can be trapped. These are opcodes that have an 'A' or 'F' in the first nybble (high order 4 bits). That is Axxx or Fxxx as an opcode.

The vector address is the address to jump to when the trap is executed. It must be even.

Example

```
            .
            .
SETTRP      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#SETTRP SVC NUMBER
            MOVW        6@AØ,#SET TRAP FUNCTION
            MOVW        8@AØ,#TRAP 16
            MOVL        1Ø@AØ,#TRAP 16 ADDRESS
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
SETTRP SVC NUMBER
            EQUW        266
SET TRAP FUNCTION
            EQUW        Ø
TRAP 16
            EQUW        16
TRAP 16 ADDRESS
            EQUL        H'5ØØØ
```

UNLOCK **Function Code 136**
Unlock Record

Unlocks a specified record number, or all records within a specified file, that you previously locked.

See READNX or DIRRD for information on how to lock a record.

Entry Conditions

Byte-offset	Ø-1	136
	6-7	file identification number
	8-9	record number

Exit Conditions

Byte-offset	2-3	error code

The record number contains the logical record number of the file to be UNLOCKed. Specifying a -1 in the record number byte-offset UNLOCKs all the records in the file which are locked by the calling user.

The file identification number is a unique number assigned to each file when it is opened.

Example

Before executing this program, store the file identification number in register A1.

```
          .
          .
UNLOCK    LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#UNLOCK SVC NUMBER
          STW       .A1,6@AØ
          MOVW      8@AØ,#UNLOCK ALL RECORDS
          BRK       #Ø
          TESTW     2@AØ
          BNE       ERROR
          .
          .
          .
```

```
SVC BLOCK
     RDATAB          32,Ø
UNLOCK SVC NUMBER
          EQUW       136
UNLOCK ALL RECORDS
          EQUW       -1
```

VDCHAR **Function Code 8**
Video Character

Outputs a character to the display at the current cursor position.

TRSDOS-16 ignores the control codes not listed in the following chart.

KEY	HEX CODE	FUNCTION
F1	Ø1	Blinking cursor on.
F2	Ø2	Cursor off.
CTRL D	Ø4	Turns on steady cursor.
BACKSPACE	Ø8	Moves cursor back one position and blanks the character at that position.
TAB	Ø9	Advances cursor to next tab position. Tab positions are at 8-character boundaries, 8, 16, 24, 32, ...
CTRL J	ØA	Line feed--cursor moves down to next row, same column position.
CTRL K	ØB	Positions cursor to beginning of previous line.
ENTER	ØD	Moves cursor down to beginning of next line.
CTRL N	ØE	Turns dual routing on.
CTRL O	ØF	Turns dual routing off.
CTRL T	14	Homes cursor (upper left corner)
CTRL W	17	Erases to end of line, cursor doesn't move
CTRL X	18	Erases to end of screen, cursor doesn't move.
CTRL Y	19	Sets Normal Display mode (white on black). Remains Normal until reset by programmer.
CTRL Z	1A	Sets Reverse Display mode (black on white). Remains Reverse until reset by programmer.
ESC	1B	Erases screen and homes cursor (position Ø).
left arrow	1C	Moves cursor back one position.
right arrow	1D	Moves cursor forward one position.
up arrow	1E	Sets 8Ø character line and clears Display.
down arrow	1F	Sets 4Ø character line and clears Display.

Entry Conditions

Byte-offset	Ø-1	8
	6-7	character to output (Ø-127)

Exit Conditions

Byte-offset	2-3	error code

Example

This routine clears the screen.

```
            .
            .
VDCHAR      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#VDCHAR SVC NUMBER
            MOVW        6@AØ,#CLS
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
VDCHAR SVC NUMBER
            EQUW        8
CLS
            EQUW        H'1E
```

VDINIT **Function Code 7**
Video Initialization

You might want to call this routine before starting any I/O to the Video Display. It blanks the screen and homes the cursor (moves the cursor to the upper left corner of the video display).

Entry Conditions

Byte-offset	Ø-1	7
	6-7	character size switch Ø = 4Ø characters/line non-zero = 8Ø characters/line
	8-9	normal/reverse switch Ø = reverse mode (black on green/ black on white) non-zero = normal mode (green on black/white on black)

Exit Conditions

Byte-offset	2-3	error code

Example

```
            .
            .
VDINIT      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#VDINIT SVC NUMBER
            MOVW        6@AØ,#CHAR8Ø
            MOVW        8@AØ,#NORMAL
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
            .
            .
SVC BLOCK
            RDATAB      32,Ø
VCINIT SVC NUMBER
            EQUW        7
```

```
CHAR8Ø
            EQUW    1
NORMAL
            EQUW    1
```

VDLINE **Function Code 9**
Video Line

Writes a specified text buffer to the display, starting at the current cursor position. The text buffer must contain codes less than H'8Ø.

Entry Conditions

Byte-offset	Ø-1	9
	6-7	buffer size (Ø - 255
	8-9	terminator character to be sent after the buffer text
	1Ø-13	buffer address

Exit Conditions

Byte-offset	2-3	error code
	14-15	on error (number of characters not sent)
	16-17	on error (character causing error)

The buffer pointed to by the buffer address should contain ASCII codes in the range Ø to 127.

This routine handles control codes in the buffer in the same manner as in the VDCHAR routine.

Example

This example retrieves the buffer size, terminator character, and buffer Address from the KBLINE SVC routine.

```
            .
            .
            LDW     .A1,12@AØ  *Get length of input line
            LDW     .A2,14@AØ  *Get terminator character
            LDA     .A3,KEYBD BUFFER  *Get address of buffer
VDLINE      LDA     .AØ,SVC BLOCK
            MOVW    @AØ,#VDLINE SVC NUMBER
            STW     .A1,6@AØ
            STW     .A2,8@AØ
            STL     .A3,1Ø@AØ
```

```
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
              .
              .
SVC BLOCK
            RDATAB      32,Ø
VDLINE SVC NUMBER
            EQUW        9
```

VERSION **Function Code 137**
Get Version of Operating System

Determines the version of software currently servicing 68ØØØ requests.

Entry Conditions

Byte-offset	Ø-1	137

Exit Conditions

Byte-offset	2-3	error code
	6-7	major version level
	8-9	minor version level
	1Ø-11	patch level

The major version level is the integral portion of the version number printed during startup.

The minor version level is the fractional portion of this quantity.

The patch level is any level following the version number.

For example, version 2.Øa returns major version level 2, minor version level Ø, and patch level 'a'.

Example

```
VERSION   LDA       .AØ,SVC BLOCK
          MOVW      @AØ,#VERSION SVC NUMBER
          BRK       #Ø
          TESTW     2@AØ
          BNE       ERROR

SVC BLOCK
          RDATAB    32,Ø
VERSION SVC NUMBER
          EQUW      137
```

VIDKEY **Function Code 12**
Video Key

This routine combines the functions of VDLINE and KBLINE.

It writes a buffer of data to the display, starting at the current cursor position, then waits for a line from the keyboard.

Once the text message has been displayed, the cursor will be positioned immediately after the last character displayed. To move it to another position, a control code can be placed at the end of the text buffer.

VIDKEY then uses KBLINE to get a line from the keyboard. Note that before starting the line input, all previously stored keystrokes are cleared.

Refer to KBCHAR and VDCHAR for a list of control codes and other details.

Entry Conditions

Byte-offset	Ø-1	12
	6-7	number of characters to be displayed (Ø-255)
	8-9	length of keyboard input field (Ø-255)
	1Ø-13	address of keyboard input buffer
	14-17	address of display text buffer

Exit Conditions

Byte-offset	2-3	error code
	18-19	on error (number of characters not sent)
	2Ø-21	on error (character causing error)

Example

```
            .
VIDKEY      LDA         .AØ,SVC BLOCK
            MOVW        @AØ,#VIDKEY SVC NUMBER
            MOVW        6@AØ,#DISPLAY COUNT
            MOVW        8@AØ,#INPUT COUNT
            LDA         .Al,KEYBD BUFFER
            STL         .Al,1Ø@AØ
            LDA         .Al,MESSAGE PROMPT
            STL         .Al,14@AØ
            BRK         #Ø
            TESTW       2@AØ
            BNE         ERROR
              .
              .
SVC BLOCK
            RDATAB      32,Ø
VIDKEY NUMBER
            EQUW        12
DISPLAY COUNT
            EQUW        21
MESSAGE PROMPT
            TEXT        'ENTER THE ANSWER HERE'
INPUT COUNT
            EQUW        2Ø
KEYBD BUFFER
            RDATAB      2Ø,Ø
```

WRITNX **Function Code 43**
Write Next Record

Writes the next record after the last record accessed; i.e., this routine writes records sequentially.

If WRITNX is the first access after opening the file, it writes the first record.

Entry Conditions

Byte-offset	Ø-1	43
	6-7	file identification number
	8-11	record buffer address

Exit Conditions

Byte-offset	2-3	error code

The file identification number is a unique number assigned to each file when it is opened.

The record buffer address is a 32-bit address that points to the beginning of a 256-byte Record Buffer; that is where the record will be placed after the disk read.

Example

Store the File Identification Number in register A1 before executing this routine.

```
            .
            .
WRITNX      LDA       .AØ,SVC BLOCK
            MOVW      @AØ,#WRITNX SVC NUMBER
            STW       .A1,6@AØ
            LDA       .A2,RECORD BUFFER AREA
            STL       .A2,8@AØ
            BRK       #Ø
            TESTW     2@AØ
            BNE       ERROR
            .
            .
```

```
SVC BLOCK
          RDATAB    32,Ø
WRITNX SVC NUMBER
          EQUW      43
RECORD BUFFER AREA
          RDATAB    256,Ø
```

APPENDICES

APPENDIX A / Error Messages

There are three kinds of error messages you might get while using your Computer:

- Boot Errors, such as "BOOT ERROR DC." See the BOOT ERRORS TABLE for more information.
- Operating System Errors, such as "ERROR 24" or "FILE NOT FOUND." To get a brief description of a numbered error, type "ERROR" followed by the error number displayed. For example, type:

  ERROR 31 <ENTER>

  and your screen will show:

  PROGRAM NOT FOUND

  For more information, see the SYSTEM ERRORS TABLE.
- Application Program Errors -- see your application program manual.

When an error message is displayed:

- Try the operation several times.
- Look up boot errors and operating system errors in the following tables and take the recommended actions. See your application program manual for explanations of application program errors.
- Try using other diskettes.
- Reset the Computer and try the operation again.
- Check all the power connections.
- Check all interconnections.
- Remove all diskettes from drives, turn off the Computer, wait 15 seconds, and turn it on again.
- If you try all these remedies and still get an error message, contact a Radio Shack Service Center.

Note: If there is more than one thing wrong, the Computer might wait until you correct the first error before displaying the second error message.

RSSC = Radio Shack Service Center

SYSTEM ERRORS TABLE

Error Code	Description	Explanation/Action
6	Attempt to open a file which hasn't been closed.	Close file before re-opening.
28	Attempt to read past end of file.	Record Number specified is past the EOF.
34	Attempt to use a non program file as a program.	File specified for execution is not a program file or an illegal load address was given.
133	Bad CRT number.	For multi-user only.
1	Bad function code on SVC call or no function exists.	Check function code number used on an SVC call.
132	Bad partition number.	For multi-user only.
129	Bad SVC-Block format.	Check format of SVC-block for errors.
	BOOT ERROR	See BOOT ERROR TABLE
4	CRC error during disk I/O (input/output) operation.	Try operation again, using a different diskette. If problem occurs frequently call RSSC.
2	Character not available.	No record or character was available when the SVC was called.
16	DCB is modified and is unusable.	DCB (used in machine-language programming) has been modified since last disk access (while the file was open).

Error Code	Description	Explanation/Action
41	Data lost during disk I/O (input/output). Hardware fault.	Contact a Radio Shack Service Center.
135	Debug Not Configured.	Include Debug at configuration time.
136	Device not available.	Device already assigned For multi-user only.
137	Device Unassigned.	For multi-user only.
17	Directory read error.	Error occured while trying to read directory. Use a different diskette.
26	Directory space full.	Number of filenames exceed the amount set when diskette was formatted.
18	Directory write error.	Error occured while trying to write to the directory. Use a different diskette.
33	Disk space allocation can't be made because of fragmentation of space. (not used on TRSDOS-II)	Use a different diskette or copy files to a clean diskette to reduce fragmentation.
27	Disk space full.	No available space on diskette.
8	Disk drive not ready.	Drive door open or diskette not in drive. On thinline drives, use TRSDOS-II, TRSDOS-16, or patched version of TRSDOS 2.Øb.

Error Code	Description	Explanation/Action
15	Disk is write protected.	Use a diskette with a write-enable tab on it.
5	Disk sector not found.	Try a different diskette.
128	DO-Nesting not allowed.	A 'DO' command was encountered within a DO file.
25	File access denied due to password protection.	Incorrect password given for protection level -- See ATTRIB in the Model 16 Owner's Manual.
11	File already in directory.	Filename already exists as a directory entry. Kill existing file or choose another filename.
24	File not found.	Filename given not found on available diskettes or file is incorrect type for desired operation.
49	Hardware fault during disk I/O (input/output).	Contact a Radio Shack Service Center.
38	I/O (input/output) attempt to an unopen file.	Open file before access.
39	Illegal I/O (input/output) attempt.	On Thinline drives -- use patched version of TRSDOS. Can be caused by an I/O attempt to a differently formatted diskette. Format diskette under current version of TRSDOS or use FCOPY.

Error Code	Description	Explanation/Action
131	Illegal Address.	SVC block or SVC argument is not within the memory range.
138	Illegal device name.	Device name specified for ASSIGN not valid.
7	Illegal disk change.	The system detected an illegal disk swap.
144	Illegal File Type	File type used is not the type required by the system (VLR or FLR).
134	Illegal operation	For multi-user only.
19	Improper file name (filespec).	Filespec given does not meet TRSDOS standard file specifications.
48	Incorrect command parameter.	Option or argument given in command is incorrect.
9	Invalid data provided by caller.	Data stream to be processed has illegal characters.
5Ø	Invalid Space Descriptor.	Try a different diskette.
1Ø	Maximum of 16 files may be open at once.	Too many files opened at one time.
35	Memory fault during program load.	Program not loaded correctly, possibly because of faulty memory or because a bad load address was given.
12	No drive available for an open.	No on-line drive is: a) write enabled or b) has enough space to create a new file.

Error Code	Description	Explanation/Action
Ø	No error found.	No error occurred.
3Ø	No more extents available (16 maximum). (not used on TRSDOS-II)	Data on diskette too fragmented, copy files to a clean diskette.
46	Not applicable to VLR type files.	Operation performed not valid for VLR files.
2Ø	Not Used.	
21	Not Used.	
22	Not Used.	
23	Not Used.	
13Ø	Odd address.	Address required by SVC block must be even.
37	Open attempt for a file already open.	File specified for open is already open.
3	Parameter error on call.	Parameter incorrect or required parameter (option) missing.
36	Parameter for open is incorrect.	Check OPEN statements or DCB for errors.
31	Program not found.	Program specified not found on available volumes.
44	Printer fault (may be turned OFF).	Check connections, power, ribbon, etc.
45	Printer not available.	Check connections, power, ribbon, etc.

Error Code	Description	Explanation/Action
42	Printer not ready.	Check connections, power, ribbon, etc.
43	Printer out of paper.	Check printer's paper supply.
29	Read attempt outside of file limits.	Use valid record numbers.
47	Required command parameter not found.	Required option or argument missing in command.
4Ø	SEEK error.	Data cannot be read from diskette -- faulty media. Try a different diskette.
14Ø	Undefined	
141	Undefined	
142	Undefined	
32	Unknown drive number (filespec).	Drive number specified not a valid drive number.
51-127	Unknown Error Codes.	
139	User Stack Overflow	Overflow occured in user stack during SETBRK SVC or SETTRP SVC operation.
13	Write attempt to a read only file.	File was opened for read only, not read/write.
14	Write fault on disk I/O (input/output).	Error occurred during a write operation -- try a different diskette. If problem continues - RSSC.
143	SVC Table Overflow	For multi-user only.

BOOT ERROR TABLE

Error Code	Description	Explanation/Action
BOOT ERROR CK	Defective ROM (Checksum Error)	Contact a Radio Shack Service Center
BOOT ERROR CT	Defective CTC Chip	Contact a Radio Shack Service Center
BOOT ERROR DC	1. Defective diskette. 2. Floppy disk expansion unit not on. 3. Defective FDC Chip or Drive.	1. Try a different diskette. 2. Turn on floppy disk expansion unit. 3. Contact a Radio Shack Service Center.
BOOT ERROR DM	Defective DMA Chip	Contact a Radio Shack Service Center
BOOT ERROR DØ	Drive not ready. 1. Improperly inserted diskette. 2. Defective diskette. 3. Defective drive.	1. Insert diskette again and reset the Computer. 2. Try a different diskette. 3. Contact a Radio Shack Service Center.
BOOT ERROR HA	Controller Error. Aborted command: Problem during boot-up of hard disk.	Re-initialize hard disk or contact a Radio Shack Service Center.
BOOT ERROR HC	CRC Error. Invalid data in data field.	Re-initialize hard disk or contact a Radio Shack Service Center.

Error Code	Description	Explanation/Action
BOOT ERROR HD	Controller Error. Busy not reset.	Re-initialize hard disk or contact a Radio Shack Service Center.
BOOT ERROR HI	CRC Error. Invalid data in ID field.	Re-initialize hard disk.
BOOT ERROR HM	Data address mark not found.	Re-initialize hard disk
BOOT ERROR HN	ID not found. No Boot Track.	Re-initialize hard disk
BOOT ERROR HØ	Track Ø Error on hard disk	
	1. Didn't find Track Ø before time-out.	1. Reset the Computer
	2. Secondary hard disk drives not turned on	2. Turn on your secondary hard disk drives
BOOT ERROR HT	Time-out while waiting for READY.	
	1. Hard disk drive not powered up.	1. Follow correct power up procedure: turn on hard disk first.
	2. Hard Disk Drive isn't turned ON and ready within 1Ø seconds after Computer.	2. Reset the Computer
	3. Hard Disk Drive is disconnected.	3. Connect the hard disk drive or operate under floppy disk control

Error Code	Description	Explanation/Action
BOOT ERROR LD	Lost Data during read -- FDC (Floppy Disk Controller) or Drive fault.	Try another TRSDOS diskette or contact a Radio Shack Service Center.
BOOT ERROR MF	Defective RAM in address range H'1ØØØ'-H'7FFF'.	Contact a Radio Shack Service Center.
BOOT ERROR MH	Defective RAM in address range H'8ØØØ'-H'FFFF'. (64K Computers only)	Contact a Radio Shack Service Center.
BOOT ERROR ML	Defective RAM in address range H'ØØØØ'-H'ØFFF'.	RSSC
BOOT ERROR PI	Defective PIO Chip	RSSC
BOOT ERROR RS	The diskette in Drive Ø is not Radio Shack el 16 or el II ating System at.	1. Insert a TRSDOS, TRSDOS-II or TRSDOS-16 formatted diskette into Drive Ø and reset the Computer. 2. Remove diskettes and turn power off. Wait 15 seconds and turn on the system again.
	ror. l data on e or ve diskette.	Try a different diskette.

H'1000' = 4096
H'7FFF' = 32767

H'0000' = 0000
H'0FFF' = 4095

Error Code	Description	Explanation/Action
BOOT ERROR TK	Record not found on bootstrap track. Improperly formatted or defective diskette.	Re-format your diskette or try a different diskette.
BOOT ERROR Z8 (68ØØØ Memory Fault at Page Address=xxxxxxx)	Defective CPU. Memory Fault. xxxxxx is the hex addr of 1K block where fault occured.	RSSC. RSSC
NOT A SYSTEM DISK	Diskette in Drive Ø isn't a TRSDOS, TRSDOS-II or TRSDOS-16 Operating System Diskette	Insert a TRSDOS, TRSDOS-II, or TRSDOS-16 Operating System diskette into Drive Ø

APPENDIX B / The Configuration Command File

Whenever TRSDOS-16 starts up or is reset, it looks for a file named CONFIG16/SYS. This "configuration command file" tells TRSDOS-16 to link in certain extra operating system programs.

CONFIG16/SYS should be present on the primary disk device (Drive Ø or Drive 4). It contains these directives:

```
INCLUDE RUNCOBOL
INCLUDE DEBUG
END
```

which tell TRSDOS-16 to link in the RUNCOBOL program and the DEBUG program.

You may create your own CONFIG16/SYS file, or modify the existing one to meet your needs, by using EDIT16.

SAVING THE EXISTING CONFIG16/SYS FILE

Before creating a new CONFIG16/SYS file, you will probably want to save the existing one by renaming it.

For example:

```
RENAME CONFIG16/SYS:Ø TO DEBCOB/CFG:Ø
```

renames the default configuration file. (The new filename tells you it includes both **DEB**UG and RUN**COB**OL modules.

After renaming the existing CONFIG16/SYS file, you can create a new one.

Since you "saved" the existing file, you can use it again. To do this, rename the present CONFIG16/SYS file (if you want to save it) and then rename DEBCOB/CFG back to CONFIG16/SYS:

```
RENAME DEBCOB/CFG:Ø TO CONFIG16/SYS:Ø
```

TO EDIT OR CREATE CONFIG16/SYS

Use EDIT16 to edit or create a CONFIG16 command file.

1. Type:

   EDIT16 <ENTER>

   and the Editor's Command mode prompt will be displayed:

   C?.....

2. To insert commands into the command file, you must get in the Insert mode, type:

   IN <ENTER>

   The Editor will display the I? prompt, indicating that you are in the Insert mode.

3. You are now ready to insert the names of the programs you want linked to TRSDOS-16.

   Comments may be used. They are indicated by an asterisk (*) in the first column.

   The key word **INCLUDE** tells TRSDOS-16 the name of the program. The syntax for the INCLUDE statement is:

   INCLUDE filename

   The default extension for filespec is **/SYS**; it is optional. Drive numbers, disk ID and Passwords are not permitted.

   Programs are loaded sequentially in memory in the order they are encountered in the CONFIG16/SYS file. The maximum number of programs that may be INCLUDEd is 15.

   The programs must be resident on the primary drive (Drive Ø or Drive 4).

   The list is concluded with an **END** statement.

For example:

```
* This is the Configuration File for DEBUG
INCLUDE DEBUG
END
```

tells TRSDOS-16 to link only the DEBUG program. The first line is a comment and is not executed by TRSDOS-16

4. When you are finished inserting, press <ENTER> to exit the Insert Mode.

5. Save the file with the following command:

```
SA CONFIG16/SYS <ENTER>
```

6. You now have a new CONFIG16 command file that TRSDOS-16 will use when it powers up or resets.

CONFIGURATOR ERROR MESSAGES

When the Configurator lists a line generating an error, it prints an error message directly underneath the line number. Preceding the message, it inserts three asterisks.

In cases of certain syntax or file I/O errors, the Configurator also marks, with a dollar sign ($), where in the line the error occurred.

For example:

```
Ø11 INCLUBE RUNCOBOL
    $
*** Illegal Command
```

shows a syntax error in the spelling of INCLUDE.

There are three catagories of Configurator error messages:

A. Configuration Control File Errors
B. Configuration Command Errors
C. Completion Errors

A. Congifuration Control File Errors

These errors are FATAL. If one of these errors occur, the Configurator could not properly execute the CONFIG16/SYS file. TRSDOS-16 will still be displayed but certain defaults will have occured:

1. No programs have been INCLUDED
2. DEBUG is kept resident (if available)
3. Any memory not occupied by DEBUG and the resident Operating System is available to the user.

Use EDIT16 to correct the error (or create a new configuration file) and reset the system.

Can't Open CONFIG16/SYS: TRSDOS Error Code = nnn

Look up TRSDOS-16 Error Code **nnn** in Appendix B and take appropriate action.

Can't configure system: File CONFIG16/SYS not proper format

The CONFIG16/SYS file is not a VLR type file.

Can't configure system: File CONFIG16/SYS not found

TRSDOS-16 could not find the CONFIG16/SYS file.

I/O Error on File CONFIG16/SYS: TRSDOS Error Code = nnn

Look up TRSDOS-16 Error Code **nnn** in Appendix B and take appropriate action.

B. Configuration Command Errors

These errors occur when a command cannot be processed by the Configurator. If one of these error occurs, the Configurator will continue to process the command lines. However, the desired result of the configuration file may not have been accomplished. For example, an INCLUDE file may have been left out.

Can't INCLUDE program: TRSDOS Error Code = nnn

The Configurator cannot load the program because of an I/O error. Look up the TRSDOS-16 Error Code in Appendix B.

Can't INCLUDE program: Out of Memory

More resident programs were requested than will fit into user memory.

Can't INCLUDE program: Program already configured

This error occurs any time a program is included twice.

Too many INCLUDED programs: this request ignored

This error occurs if more than 15 programs are included. The command line that is flagged is ignored (treated as a comment).

C. Completion Error

***** CONFIGURATION ABORTED *****

This message appears when the configurator could not finish processing the CONFIG16/SYS file because of an I/O error.

ABOUT THE CONFIGURATOR

The Configurator is invoked whenever the 68ØØØ processor is initialized. It performs several important functions:

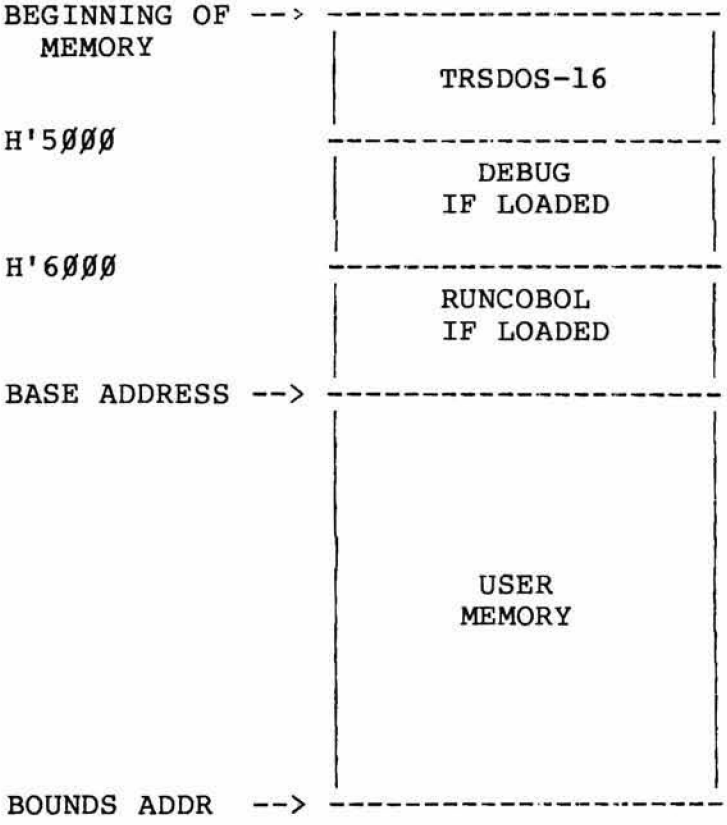
- It determines whether the machine debugger is required. If not, it is eliminated from memory. This gives you an extra 4K of memory.
- It initializes traps and interrupts. This eliminates the need to keep extra code resident in memory.
- It loads in resident programs as specified in the CONFIG16/SYS file.
- It reads the AUTO file and passes it to TRSDOS-16 for execution.

The Configurator is linked in at the end of user memory and occupies 4K of memory. Upon system initialization, it moves itself to the top of physical memory. This is because the resident programs will be loaded at low address, overlaying the original configurator.

Next the Configurator begins to load the resident programs requested in the CONFIG16/SYS file (i.e., DEBUG and RUNCOBOL). It loads these programs sequentially starting at the beginning of user memory and up to the beginning of where the Configurator has relocated itself. This guarantees that after loading is complete, the user has at least 4K of memory available (the size of the Configurator).

After configuration is complete, the Configurator is no longer neccessary and is overwritten.

APPENDIX C ,' Memory Map



MEMORY CHART

User memory begins at H'5ØØØ if the Debugger is not configured and at H'6ØØØ if the Debugger is configured.

APPENDIX D / ASCII Character Codes

Code		Character	
Dec.	Hex.	Keyboard	Video Display
00	00	HOLD	
01	01	F1 CTRL A	Turns on blinking cursor
02	02	F2 CTRL B	Turns off cursor
03	03	BREAK CTRL C	
04	04	CTRL D	Turns on steady cursor
05	05	CTRL E	
06	06	CTRL F	
07	07	CTRL G	
08	08	BACKSPACE CTRL H	Backspaces cursor and erases character
09	09	TAB CTRL I	Advances cursor to next 8-character boundary
10	0A	CTRL J	Line feed
11	0B	CTRL K	Cursor to previous line
12	0C	CTRL L	
13	0D	ENTER CTRL M	Carriage return
14	0E	CTRL N	Dual routing on
15	0F	CTRL O	Dual routing off
16	10	CTRL P	
17	11	CTRL Q	
18	12	CTRL R	
19	13	CTRL S	
20	14	CTRL T	Homes cursor to upper left
21	15	CTRL U	
22	16	CTRL V	
23	17	CTRL W	Erases to end of line
24	18	CTRL X	Erases to end of screen
25	19	CTRL Y	Sets white-on-black mode
26	1A	CTRL Z	Sets black-on-white mode
27	1B	ESC	Clears screen, homes cursor

*BREAK is always intercepted. It will never return a code 3 to the user program.

Code		Character	
Dec.	Hex.	Keyboard	Video Display
28	1C	←	Moves cursor back
29	1D	→	Moves cursor forward
30	1E	↑	Sets 80-character mode and clears Display
31	1F	↓	Sets 40-character mode and clears Display
32	20	SPACE BAR	Ø
33	21	!	!
34	22	"	"
35	23	#	#
36	24	$	$
37	25	%	%
38	26	&	&
39	27	'	'
40	28	(	(
41	29	)	)
42	2A	*	*
43	2B	+	+
44	2C	,	,
45	2D	−	−
46	2E	.	.
47	2F	/	/
48	30	Ø	Ø
49	31	1	1
50	32	2	2
51	33	3	3
52	34	4	4
53	35	5	5
54	36	6	6
55	37	7	7
56	38	8	8
57	39	9	9
58	3A	:	:
59	3B	;	;
60	3C	<	<
61	3D	=	=
62	3E	>	>
63	3F	?	?
64	40	@	@
65	41	A	A
66	42	B	B
67	43	C	C
68	44	D	D
69	45	E	E
70	46	F	F
71	47	G	G

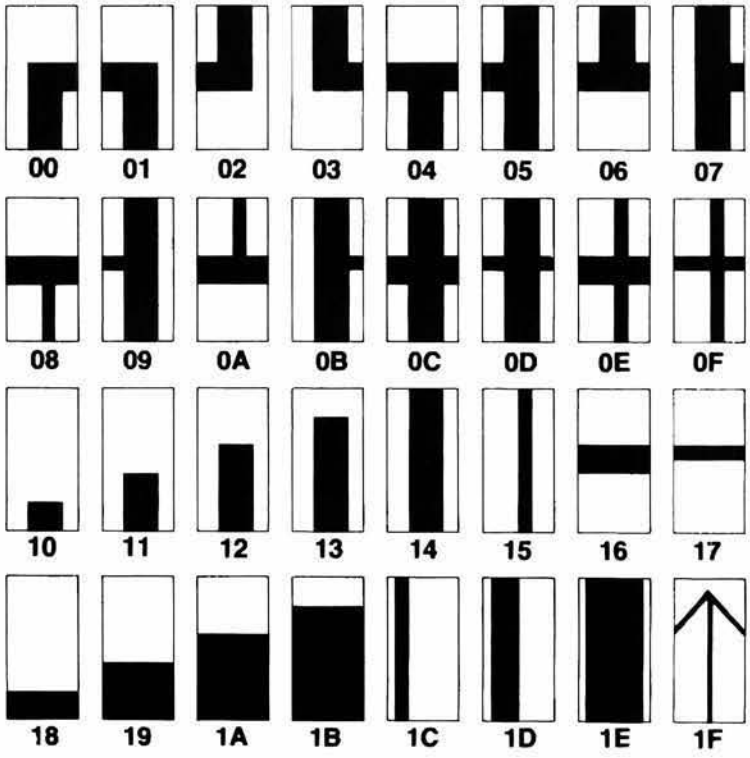
Code		Character	
Dec.	Hex.	Keyboard	Video Display
72	48	H	H
73	49	I	I
74	4A	J	J
75	4B	K	K
76	4C	L	L
77	4D	M	M
78	4E	N	N
79	4F	O	O
80	50	P	P
81	51	Q	Q
82	52	R	R
83	53	S	S
84	54	T	T
85	55	U	U
86	56	V	V
87	57	W	W
88	58	X	X
89	59	Y	Y
90	5A	Z	Z
91	5B	[	[
92	5C	CTRL-9	\
93	5D	]	]
94	5E	^	^
95	5F	_	_
96	60		`
97	61	A	a
98	62	B	b
99	63	C	c
100	64	D	d
101	65	E	e
102	66	F	f
103	67	G	g
104	68	H	h
105	69	I	i
106	6A	J	j
107	6B	K	k
108	6C	L	l
109	6D	M	m
110	6E	N	n
111	6F	O	o
112	70	P	p
113	71	Q	q
114	72	R	r
115	73	S	s
116	74	T	t

Code		Character	
Dec.	Hex.	Keyboard	Video Display
117	75	U	u
118	76	V	v
119	77	W	w
120	78	X	x
121	79	Y	y
122	7A	Z	z
123	7B	{	{
124	7C	CTRL-0	|
125	7D	}	}
126	7E	CTRL-6	~
127	7F		±
128	80		
129	81		
130	82		
131	83		
132	84		
133	85		
134	86		
135	87		
136	88		
137	89		
138	8A		
139	8B		
140	8C		
141	8D		
142	8E		
143	8F		
144	90		
145	91		
146	92		
147	93		
148	94		
149	95		
150	96		
151	97		
152	98		
153	99		
154	9A		
155	9B		
156	9C		
157	9D		
158	9E		
159	9F		↑
160	A0		ß
161	A1		!
162	A2		"

Code		Character	
Dec.	Hex.	Keyboard	Video Display
163	A3		#
164	A4		$
165	A5		%
166	A6		&
167	A7		'
168	A8		(
169	A9		)
170	AA		*
171	AB		+
172	AC		'
173	AD		–
174	AE		.
175	AF		/
176	B0		Ø
177	B1		1
178	B2		2
179	B3		3
180	B4		4
181	B5		5
182	B6		6
183	B7		7
184	B8		8
185	B9		9
186	BA		:
187	BB		;
188	BC		{
189	BD		=
190	BE		}
191	BF		?
192	C0		@
193	C1		A
194	C2		B
195	C3		C
196	C4		D
197	C5		E
198	C6		F
199	C7		G
200	C8		H
201	C9		I
202	CA		J
203	CB		K
204	CC		L
205	CD		M
206	CE		N
207	CF		O
208	D0		P

Code		Character	
Dec.	Hex.	Keyboard	Video Display
209	D1		Q
210	D2		R
211	D3		S
212	D4		T
213	D5		U
214	D6		V
215	D7		W
216	D8		X
217	D9		Y
218	DA		Z
219	DB		[
220	DC		\
221	DD		]
222	DE		^
223	DF		_
224	E0		`
225	E1		a
226	E2		b
227	E3		c
228	E4		d
229	E5		e
230	E6		f
231	E7		g
232	E8		h
233	E9		i
234	EA		j
235	EB		k
236	EC		l
237	ED		m
238	EE		n
239	EF		o
240	F0	Unused	
241	F1	Unused	
242	F2	Unused	
243	F3	Unused	
244	F4	Unused	
245	F5	Unused	
246	F6	Unused	
247	F7	Unused	
248	F8	Unused	
249	F9	Unused	
250	FA	Unused	
251	FB	Unused	
252	FC		Moves cursor left
253	FD		Moves cursor right
254	FE		Moves cursor up
255	FF		Moves cursor down

APPENDIX E / Graphics Codes



APPENDIX F / MODEL 16 & ENHANCED MODEL II SPECIFICATIONS

SPECIFICATIONS

The Radio Shack TRS-8Ø Model 16 and Enhanced Model II are disk-based computer systems with two major components:

1. A Display Console with up to two built-in, double-sided, double-density floppy disk drives (Model 16) or one built-in, single-sided floppy disk drive (Enhanced Model II).
2. A separate keyboard enclosure which can be positioned for maximum operator comfort and efficiency

The operating system software is loaded from a system diskette in Drive Ø or Drive 4 by a built-in ROM "bootstrap" program.

PROCESSORS

Input/Output Processor System:

- Z8Ø-A based with 64K bytes of random access memory
  Independent bus can support all the standard system boards
- Emulation mode allows you to execute programs previously developed for the TRS-8Ø Model II without changing them first.

Computational Processor System:

- 68ØØØ based with either 128K or 256K (384K or 512K bytes on a Model 16) of RAM
  Independent bus can support multiple bus masters

The two processors share the computing load from the application programs (the Z8Ø-A based processor performs input/output tasks while the 68ØØØ based processor performs computational tasks).

VIDEO DISPLAY

LSI Controller Chip:

- Frees the input/output (Z8Ø-A based) processor from much of the overhead required to update and maintain the video display.

Four Modes:

Model 16:

- green on black (normal)
- black on green (reversed)
- 8Ø characters by 24 lines
- 4Ø characters by 24 lines

Enhanced Model II:

- white on black (normal)
- black on white (reversed)
- 8Ø characters by 24 lines
- 4Ø characters by 24 lines

Displayable Characters:

- Full ASCII set
- 32 graphics characters

KEYBOARD

- LSI Controller frees the input/output (Z8Ø-A based) processor from keyboard scan and related tasks
- Located in separate case for convenience
- Connected to Display Console via a built-in cable exiting the bottom front of the Console
- Standard typewriter keys, repeat key and two general-purpose function keys
- Four modes: 1) Unshift; 2) Shift; 3) Caps; 4) Control

FLOPPY DISK DRIVES

Minimum:

- Model 16: One (if system contains a hard disk) or two built-in 8" double-sided floppy disk drives
- Enhanced Model II: One built-in 8" single-sided floppy disk drive

Maximum:

- Model 16: Two built-in and two external 8", double-sided floppy disk drives (Disk Expansion Unit needed for two external drives)
- Enhanced Model II: One built-in and three external 8" single-sided floppy disk drives (Disk Expansion Unit needed for three external drives)

Storage Capacity:

- 1,256,7Ø4 bytes per double-sided diskette (for User Data Capacity, see Operating System Manual.
- 625,92Ø bytes per single-sided diskette (for User Data Capacity, see Operating System Manual)

Diskette Organization:

- 154 tracks per double-sided diskette, 77 tracks per single-sided diskette
- 32 (Ø-31) sectors per track -- will vary with operating system software. See TRSDOS Reference Manual, Technical Information section for more details.
- 256 bytes per sector (except track Ø which has 128 bytes per sector) -- varies with operating system software. See TRSDOS-16 and the TRSDOS/Model II Reference Manuals, Technical Information sections for more details.

Data Transfer Rate:

- 5ØØ,ØØØ bits per second (except track Ø which has 25Ø,ØØØ bps)

Required Media:

- Model 16: Radio Shack Double or Single-sided, 8" Floppy Diskettes
- Enhanced Model II: Radio Shack Single-sided, 8" Floppy Diskettes

Preventative Maintenance Interval:

- Typical usage (3,ØØØ Power-on hours per year): Every 8ØØØ Power-On Hours
- Heavy usage (8,ØØØ Power-on hours per year): Every 5ØØØ Power-On Hours

Diskette Life:
3.5 million passes per track
Usually limited by improper handling. Follow handling recommendations for maximum use.

POWER SUPPLY

Power Requirements:
- 105 - 130 VAC, 60 Hz
- 240 VAC, 50 Hz (Australian)
- 220 VAC, 50 Hz (European)
- Grounded outlet

Maximum Current Drain:
- 2.0 Amps

Typical Current Drain:
- 1.5 Amps

OPERATING TEMPERATURE
- 32 to 110 degrees Fahrenheit
- 0 to 43 degrees Centigrade

PERIPHERAL INTERFACES

Standard:

- Serial port A (RS232-C)
- Serial port B (RS232-C)
- Parallel input/output channel, for connection to TRS-80 standard parallel interface line printers
- Floppy disk input/output channel for connection of a Disk Expansion Unit

Optional:

- Hard Disk Drive Interface
- ARCNET Interface
- Graphic Board

Serial Interface

Two Channels
- Channel A allows asynchronous or synchronous

transmission.

- Channel B allows asynchronous transmission only.
- Both conform to the RS-232-C standard.
- Both use the DB-25 connectors on the back of the Display Console.

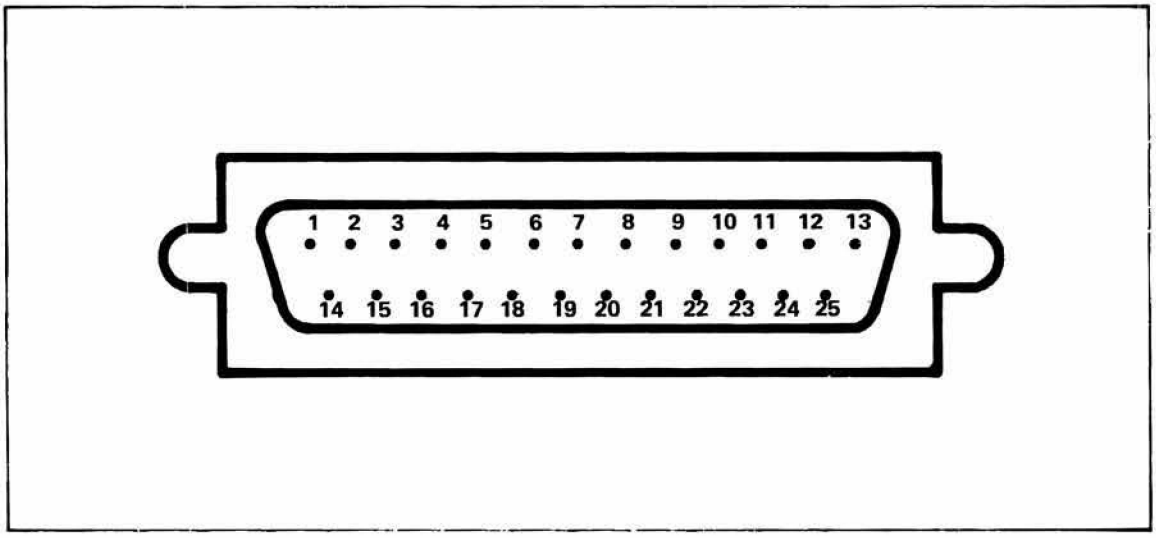
The DB-25 connector pin-outs and signals available are listed below.

CHANNEL A

STANDARD RS-232C SIGNAL	PIN #
I/O TRANSMIT S.E.T.	15
GROUND	1,7
RECEIVED DATA	3
RECEIVER CLOCK	17
TRANSMIT CLOCK	24
DATA SET READY	6
CLEAR-TO-SEND	5
CARRIER DETECT	8
TRANSMIT DATA	2
REQUEST-TO-SEND	4
DATA TERMINAL READY	2Ø

CHANNEL B

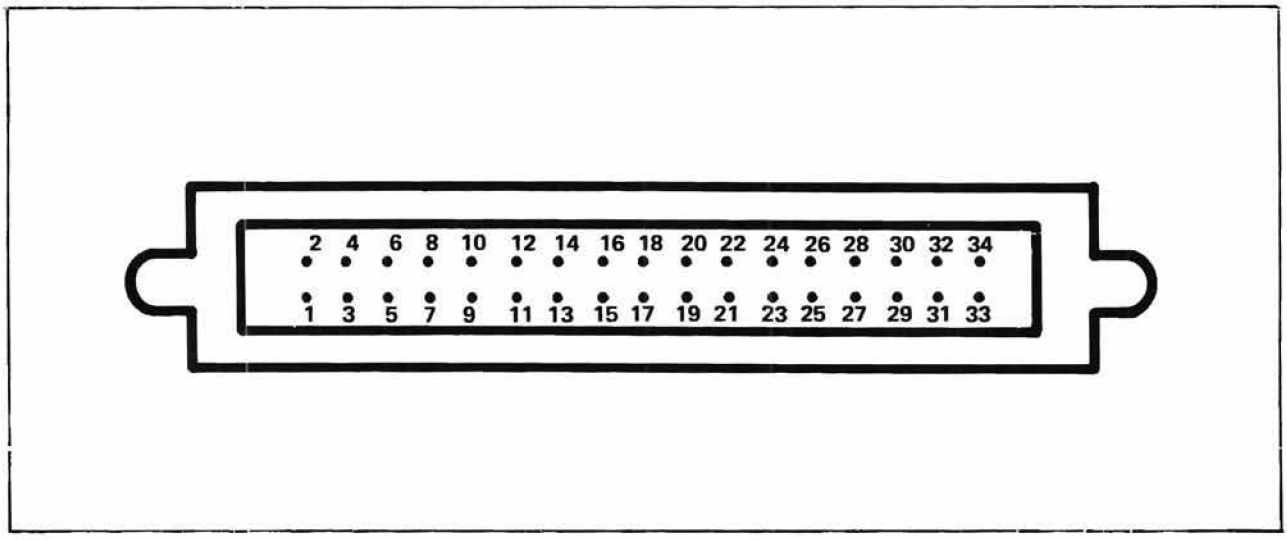
STANDARD RS-232-C SIGNAL	PIN #
GROUND	1,7
RECEIVED DATA	3
RECEIVER XMITTER CLOCK	17
DATA SET READY	6
CLEAR-TO-SEND	5
CARRIER DETECT	8
TRANSMIT DATA	2
REQUEST-TO-SEND	4
DATA TERMINAL READY	2Ø



Parallel Interface

Connection to a line printer via the 34-pin connector on the back panel of the Display Console.
Eight data bits are output in parallel.
Four data bits are input.
All levels are TTL compatible.

The connector pin-outs and signals available are listed on the next page.



SIGNAL	FUNCTION	PIN
STROBE	1 microsecond pulse to clock the data from processor to printer	1
DATA Ø	Bit Ø (lsb) of output data byte	3
DATA 1	Bit 1 of output data byte	5
DATA 2	Bit 2 of output data byte	7
DATA 3	Bit 3 of output data byte	9
DATA 4	Bit 4 of output data byte	11
DATA 5	Bit 5 of output data byte	13
DATA 6	Bit 6 of output data byte	15
DATA 7	Bit 7 (msb) of output data byte	17
ACK*	Input to Computer from Printer, low indicates data byte received	19
BUSY	Input to Computer from Printer, high indicates busy	21
PAPER EMPTY	Input to Computer from Printer, high indicates no paper -- if Printer doesn't provide this, signal is forced low	23
SELECT	Input to Computer from Printer, high indicates device selected	25
PRIME*	Output to Printer to clear buffer and reset printer logic	26
FAULT*	Input to Computer from Printer low indicates fault (paper empty, light detect, deselect, etc.)	28
GROUND	Common signal ground	2,4,6,8,1Ø,12,14, 16,18,2Ø,22,24, 27,31,33
NC	Not connected	29,3Ø,32,34

*These signals are active-low.

APPENDIX G / SVC Quick Reference List

NAME	DESCRIPTION	NO.
ACTL	Control Channel A	1ØØ
ARCV	Channel A Receive	96
ATX	Channel A Transmit	97
BCTL	Control Channel B	1Ø1
BRCV	Channel B Receive	98
BTX	Channel B Transmit	99
CLOSE	Terminates output to specified file.	42
CLOSEF	Terminates output to all open files except OPENDO file	133
CLRXIT	Clear user memory and jump to TRSDOS-16 Ready	257
CURSOR	Position cursor	1Ø
DATE	Returns the real-time (time/date)	45
DEBUG	Load the Debugger	259
DIRRD	Allows a specific record in a FLR file to be read.	35
DIRWR	Allows a specific record in a FLR file to be read.	44
DISMOUNT	Logically disconnect a MOUNTed disk device	139
DOSCMD	Execute TRSDOS-16 command and return to TRSDOS-16 Ready	27Ø
DUMP	Writes a 68ØØØ format program file from 68ØØØ memory.	13Ø
ERRMSG	Returns an 8Ø-byte descriptive error message for the requested error number	52

NAME	DESCRIPTION	NO.
ERROR	Causes the error message refered to by ERROR NUMBER to be printed on the video display.	39
EXECUTE	Execute program.	263
HLDKEY	Enable / Disable HOLD key.	29
JP2DOS	Jump to TRSDOS-16 Ready.	264
KBCHAR	Strobes the keyboard and returns with or without a character.	4
KBINIT	Initializes the keyboard input driver.	1
KBLINE	Inputs a line from the keyboard into a a buffer and echoes the line to the display.	5
KILL	Deletes the specified file from the directory.	41
LOAD	Loads a 68ØØØ-format program into the user memory.	131
LOCATE	Returns the number of the current record. (i.e. the last record accessed)	33
MOUNT	Logically connects a disk device	138
MOVBUF	Retrieves and stores an 8Ø-byte buffer	267
OPEN	Handles both the creation and opening of files.	4Ø
OPENDO	Opens and creates a special file not closed by CLOSEF	14Ø
PRCHAR	Sends one character to the Printer.	18
PRCTRL	Lets you select various printer options.	95
PRINIT	Initializes the printer driver	17

NAME	DESCRIPTION	NO.
PRLINE	Sends a line to the Printer.	19
READNX	Reads the next record after the current record.	34
RENAME	Changes the name and/or extension of a file	47
RESET	Same as pressing the RESET switch	129
RS232C	Initialize RS-232C Channel	55
SETBRK	Enable / Disable the BREAK key	269
SETTRP	Set or remove trap vectors	266
UNLOCK	Unlocks a specified record	136
VDCHAR	This routine outpus a character at the current cursor position.	8
VDINIT	Initialize the Video Driver	7
VDLINE	Writes a buffer of data to the display	9
VERSION	Get version of Operating System	137
VIDKEY	Sends a prompting message to the display and then waits for a line from the keybd.	12
WRITNX	Writes the next record after the last record accessed. (i.e., sequentially)	43

INDEX

	Page
ABS	27,34, 48, 52, 71, 74
Access Password	15
ACTL	79, 122, 129-13Ø
Addresses	
Base Address	1Ø5, 1Ø6,1Ø7, 238
Bounds Address	238
Buffer Address	128, 212
Dump Address	156
High Bound Address	17Ø
Low Bound Address	17Ø
Parameter List Address	177, 199
Start Load Address	17Ø
Transfer Address	17Ø, 2Ø2
ALL	48, 72, 74
Alternate Directory	53
APPEND	25, 29, 51
ARCV	79, 122, 131
ASCII	59, 84, 145
Character Codes	239-244
Text File	116
Assembler	1Ø5
ATTRIB	15, 25, 29-31, 51
ATX	79,122, 133-134
AUTO	12, 25, 31-32
Auto File	237
Auto Line Feed	187, 188
Auto Sign-On	88, 92
Background Printing	82, 185
BACKUP	25, 32, 68, 73
Base Address	1Ø5, 1Ø6,1Ø7, 238
BASIC	65
Baud Rate	79,8Ø, 85,86
BCTL	79, 122, 135-136
BLOCK	123
BOOT16 TRSDOS16/SYS	12
Bounds Address	1Ø5, 1Ø7, 238
Braces { }	27
BRCV	79
BRCV	122, 137
<BREAK>	12, 64,87-9Ø,164, 2Ø2
Break Character/Sequence	87, 88
BTX	79, 122, 139-14Ø
Buffer	114, 127, 155, 175
address	128, 212
BUILD	76
Byte	1Ø8, 112, 123
Byte-Offset	123, 124, 126, 127
Capture-File	82, 83
Character Transmitted Status	133, 139

	Page
CLEAR	25, 33
CLOSE	122, 141
CLOSEF	122,142, 18Ø
CLRXIT	122, 143
CLS	18, 25, 26, 33
COBOL	35
Codes	
Access Code	177, 18Ø
ASCII Character Codes	239-244
Communications Status Code	13Ø, 131, 133, 135, 138, 139
Creation Code	177, 181
Error Code	177
Graphics Code	245
Column	144
Command File	18, 1Ø5
Command Syntax	26-28
Comment	27
Communications Status Code	13Ø, 131, 133, 135, 138, 139
Compilers	29
Completion Errors	234,236
Configuration	1Ø5
Errors	234,235
Command File	12,147, 232
Configurator	
Error Messages	234-236
CONFIG16/SYS	12, 232, 233,237
COPY	25, 27, 34, 35, 51
CREATE	25, 35,36,51, 113
Cylinders	1Ø8
Data Files	64, 65, 66, 69, 9Ø, 91, 97
DATASET	72, 74, 75
DATE	18, 25, 36
DATE SVC	127, 145-146
DB-25 Connector Pin-Outs	25Ø
(DC) Date Created	74, 77
DEBUG	26, 122, 147, 232,234
Debugger	147, 238
DELETE	69
Diagnostics	11Ø
DIR	17-2Ø, 25, 31, 37-39
Direct File Access	119
Directory	14, 58,169
Alternate Directory	11Ø,111
Primary Directories	11Ø,111
Space Formula	11Ø
DIRRD (Direct-Read)	119, 121, 122, 149-15Ø, 2Ø6
DIRWR (Direct-Write)	119, 121,122, 123, 151-152

	Page
Disk	
Disk capacity	
Double-Sided Diskette	1Ø8
Hard Disk	1Ø8, 1Ø9
Single-Sided Diskette	1Ø8
Diskette	6
Diskette Swap	61
Disk Files	14, 18, 19, 48, 57, 63, 81, 82, 113, 118, 141
Disk I/O	128
disk name	16, 38
Disk Organization	1Ø8
Disk Read/Write Mechanism	57
Disk Sectors	56
Disk Space Allocation	112
Double-Sided diskette	112
Hard Disk	112
Single-Sided diskette	112
DISMOUNT	21, 25, 39-4Ø, 61, 122, 153, 173
(DM) Date Modified	74, 77
DO	25, 28, 4Ø-41, 51
Do-file	18, 19, 28, 32, 142, 143, 163
DOSCMD	122, 155
Double-Sided Diskette	32, 52
drive	15
DRIVE	41-46
Drive Settings	
RATE	42, 43, 44
DETECT	42, 43, 44
NODETECT	42, 43, 44
WAIT	42, 43, 44
NOWAIT	42, 43, 44
OFFLINE	42, 45, 46
ONLINE	42, 45, 46
DUAL	19Ø
Dummy Modes	56
Dummy Output Mode	187, 188
DUMP	46-47, 51, 97, 122, 156-157
Dump Address	156
Editor	14, 4Ø, 1Ø5
EDIT16	14, 18, 4Ø, 232, 233
Emulation Mode	246
Emulators	2Ø5
ERRMSG	122, 127, 158
ERROR	122, 126, 159
Error-Code	124
Error Conditions	66, 139
Error Messages	158, 221

	Page
Boot Errors	221, 228
Operating System Errors	221, 222
Application Program Errors	221
EXEC	26, 47-48
EXECUT	122,16Ø
extension (/ext)	14, 7Ø
extents	57
FCOPY	17, 18, 25, 48-5Ø
FCOPY DIR	49
Fields	145
File Access	122
File Allocation	
Dynamic Allocation	113
Pre-Allocation	113
File Fragmentation	56-57
File Identification Number	124, 125-126, 141, 15Ø 172, 177-178, 18Ø, 182 194, 2Ø6, 217
filename	14, 38, 7Ø
FILES	5Ø-51
filespec	14, 16
First-In, First-Out buffer (FIFO)	131, 137
Fixed length Record (FLR)	29, 38, 63, 91, 116, 119, 149, 151, 172
Flags	138, 139
Wait For Character	164
Character Present	164
Returned Character	164
FLOPPY	51-52
Floppy Diskette	71, 73, 1Ø8
Single-Sided	1Ø8
Double-Sided	1Ø8
Double Density	1Ø8
Swap	173
Floppy Disk Drives	41, 42, 43 6Ø, 247-249
Latch	42-45
Push button	42-45
Thinline	42-45
Floppy Disk System	11
FORMAT	52-54, 68
FORMS	54-56, 86, 92
Format Options	54-55
Parameters	55
Switch Options	55-56
FREE	56-57
Function Code	122, 124, 126
Graphics Code	245
Hard Disk	5, 73, 112, 153
Drives	42

	Page
System	11, 73
HELP	58
Hexadecimal Codes	67
Hexadecimal Number	47
HLDKEY	122, 161-162
<HOLD>	161
Illegal Opcodes	2Ø5
Indexed Access Files(ISAM)	29
INDirect (IND)	71-72, 74, 76-77
Input Buffer	131-132, 137-138, 167
Input/Output Processor System	246
Instruction Address	147
Interactive Terminal Mode	84, 87-9Ø
Error Messages	98
P	99
O	99
F	99
DATA CARRIER LOST	99
DATA CARRIER RESTORED	99
DATA SEQUENCE RECEIVED	99
Interrupt Handlers	127
Interrupts	237
JP2DOS	122, 163
KBCHAR	122, 164-165
KBINIT	122, 166
KBLINE	122, 167-168, 215
Key-Ahead Buffer	164, 166
Keyboard	122, 247
Buffer	164
Input Driver	166
KILL	18, 51, 58-59, 122, 128, 169
LIB	13, 59
Linker	1Ø5
LIST	51, 59-6Ø
LOAD	51, 6Ø, 122, 17Ø-171
LOCATE	121, 122, 172
Logical Record Length	118
Long Word	123
Machine Language Object Code	84
Machine Language Programs	6Ø, 64-65, 7Ø, 78
Master Password	68
MC68ØØØ	1
Memory	
Memory Addresses	123
Memory Chart	1Ø7
Memory Map	238
Memory Requirements	1Ø5
Menu Mode	84-86

	Page
Model II Mode	1, 11, 48
Modem	78, 84
Modulo 24	144
MOUNT	21, 6Ø-61, 122, 173-174
MOVBUF	122, 155 175-176
MOVE	18, 51, 61-62
MSG	18
Normal Mode	188
Notations	5
OPEN	51, 122, 177-179
OPENDO	122, 18Ø-182, 198
Operating System	1, 3, 214
Routines	122
Parameters	26, 123, 175
Errors	124
List Address	18Ø
Parity	79-8Ø, 85-86
Partition	2Ø4
password	15, 63, 68, 69
PATCH	63-66
Patch Level	214
PAUSE	67
PC Register Set	147
Peripheral Interfaces	249-252
Physical Load Address	1Ø6
Power Supply	249
PRCHAR	122, 183-184, 192
PRCTRL Supervisor Call	56, 122, 185-186
Pre-Allocation	113
Primary Directory	53
Primary Disk Device	232
Primary Drive	6, 13, 17, 5Ø, 69
PRINIT	122, 189-191
PRINT	67-68
Print-File	82
Printer's Buffer	183, 192
Printer Output	82
PRLINE	122, 128, 192-193
Program Files	64
Programming With User Interrupts	127
PROMPT	48, 71, 74
PROT	68-69
Protection level	3Ø
PRT	37, 5Ø, 56, 59, 71
PURGE	69-7Ø
RAM	65
RAM Buffer	47
Random Access	119
READNX (Read-Next)	119-12Ø, 122, 194-195,

	Page
	2Ø6
Record	114
Address	124-126, 149, 151, 194
Length	38, 114
Logical Records	114
Numbers	118, 124-126, 149-151, 172, 2Ø6, 217
Physical Records	114
Processing	119
Record Lock Flag	149, 194
Relative Addressing	1Ø5-1Ø6
Relocation Address	156
RENAME	51, 7Ø, 77, 122, 128, 196-197
Reserved	1Ø7, 124
RESET	7Ø, 122, 198
RESTORE	71-73
Row	144
RS-232C	78, 122, 129, 131, 199-2Ø1
Cable	84, 85
Parameters	199
RTR Instruction	127
RUNCOBOL	1Ø5,232
SAVE	71-75, 77
DIRectory	76
Sectors	39, 1Ø8-1Ø9, 111-115
Secondary Drives	42,43,46
Sequential File Access	119
Sequential Read	12Ø
Sequential Write	121
Serial Channel	78, 8Ø, 84-86, 129, 131, 133 ,135, 137, 139, 25Ø
SETBRK	122, 127, 164, 2Ø2-2Ø3
SETCOM	78-8Ø, 86
SETTRP	122, 127, 2Ø4-2Ø5
Single-Sided Diskettte	32, 52
68ØØØ	
Operating System	12
Processor	237
Program	13, 65, 156, 17Ø
SIZE	8Ø
Spanning	114
Specifications	246
SPOOL	8Ø-84, 185
SPOOL File	142
Capture Function	185
Stack Pointer	127
Status Bits	125
Status Registers	127
Stop Bits	79-8Ø, 85-86

	Page
Supervisor Calls (SVCs)	119, 122-123, 126-127
BLOCK	123-126
Calling Procedure	125-126
Number	124
Quick Reference List	253-255
Swap Disketttes	21
Switch	
Character Size Switch	21Ø
Normal/Reverse Switch	21Ø
Syntax	58
SYS	37, 49,5Ø, 69, 72, 74
System Commands	12, 29-1Ø3
System Diskette	32
System Files	37
System Trap	127
T	83-84
TERMINAL	84-1Ø1
Error Conditions	98-1Ø1
P	
O	
F	
DATA CARRIER LOST	
DATA CARRIER RESTORED	
BREAK SEQUENCE RECEIVED	
Terms	5
TEST	125
Text Buffer	212
TIME	99
Tracks	1Ø8, 11Ø-112
Transfer Address	156
Transparent Mode	56, 187-188
Trap	2Ø5, 237
Trap Vector	2Ø4
UNLOCK	122, 2Ø6-2Ø7
Update Password	15
Update	38
User	
Attribute	178, 181-182
Attribute Byte	178, 181-182
File	61
Memory	33, 8Ø, 1Ø5, 1Ø7, 127, 143, 17Ø, 238
PC	127
Stack	127
Variable Length Record (VLR)	29, 38, 91, 116, 119, 121, 147, 151
VDCHAR	122, 2Ø8-2Ø9
VDINIT	122, 21Ø-211
VDLINE	122, 128, 212-213, 215
Vectors	2Ø5

	Page
Verification	53
Address	2Ø5
VERIFY	1ØØ
VERSION	1ØØ, 122, 214
Major Version Level	214
Minor Version Level	214
Video Display	122, 247
VIDKEY	122, 215-216
VOLUME	72, 74-75
Wildcarding	75
Wildcards	
Wildcard (*)	17,18,2Ø,5Ø,61
Super Wildcard (!)	17,18,2Ø, 5Ø, 61
Word	123
Word Length	79-8Ø, 85-86
WRITNX (Write-Next)	119-12Ø, 122, 217-218
Z8Ø	1
Z-8Ø Program File	64

RADIO SHACK A DIVISION OF TANDY CORPORATION

U.S.A.: FORT WORTH, TEXAS 76102
CANADA: BARRIE, ONTARIO L4M 4W5

TANDY CORPORATION

AUSTRALIA	BELGIUM	U. K.
280-316 VICTORIA ROAD RYDALMERE, N.S.W. 2116	PARC INDUSTRIEL DE NANINNE 5140 NANINNE	BILSTON ROAD WEDNESBURY WEST MIDLANDS WS10 7JN

8749341-482-SP

PRINTED IN U.S.A.