

Radio Shack
**PORTABLE
REFERENCE
GUIDE**

TRS-80® Model 4P



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

Contents

Start-Up	3
TRSDOS	
TRSDOS Commands and Utilities ...	5
Drivers, Filters, and Devices	17
Error Messages	19
BASIC	
BASIC Statements and Functions ...	23
Control Keys	37
Operators	37
Edit Commands	39
Special Characters	39
Error Messages	41
Reserved BASIC Words and Internal Codes	43

Notes:

Start-Up

Make sure all floppy disk drives are empty and all equipment is off.

Model 4 Mode Start-Up

1. Turn on all peripheral equipment (such as a printer), except the hard disk.
2. Hard disk users: Turn on the primary hard disk drive.
3. Turn on the computer.
4. When an error message appears on the screen, insert a TRSDOS 6 diskette into Drive 0. Close the drive latch.
5. Press **RESET**.
6. When TRSDOS prompts you for the date, enter it in the mm/dd/yy format.
7. The following system prompt will appear on your screen:
TRSDOS Ready
Now you can type in a TRSDOS command.
8. To start disk BASIC, type:
BASIC **(ENTER)**
and you will see the BASIC prompt:
Ready
Now you can type in a BASIC command.

Model III Mode Start-Up

1. Turn on all peripheral equipment (such as a printer), except the hard disk.

2. Hard disk users: Turn on the primary hard disk drive.
3. Turn on the computer.
4. Wait for an error message to appear. Then follow a. or b. below.
 - a. If the TRSDOS 1.3 or LDOS 5.1 diskette you wish to use has the file called "MODEL/III" on it, insert the diskette into Drive 0 and close the drive latch. Then press **RESET**. (See the *Introduction to Your Disk System* manual for information on how to copy the MODEL/III file from your TRSDOS 6 system diskette to a TRSDOS 1.3 or LDOS 5.1 diskette.)
 - b. If the TRSDOS 1.3 or LDOS 5.1 diskette you wish to use does not have the MODEL/III file on it, then insert any diskette that does contain this file into Drive 0 and close the drive latch. Press **RESET**. Press **(F3)** and **(D)** right after you press **RESET**. When you see the prompt to switch diskettes, remove the disk in Drive 0 and insert the Model III diskette you want to use. Then press **(ENTER)**. (If you want to use Model III ROM BASIC, press **(BREAK)** instead of **(ENTER)**.)
5. When you are prompted for the date, enter it in the mm/dd/yy format.

Start-Up

6. If you are prompted for the time, enter it in the hh:mm:ss format.
7. The operating system prompt will appear on your screen. Now you can type in a system command.
8. To start disk BASIC, type:
BASIC (ENTER)
if you are using TRSDOS, or type:
LBASIC (ENTER)
if you are using LDOS.
You will see the BASIC prompt:
Ready
Now you can type in a BASIC command.

TRSDOS Commands and Utilities

Information that is upper-case should be typed in exactly as is.

Information that is lower-case represents a value that you supply. Information within brackets [] is optional.

APPEND *source* [TO] *destination*
[(*parameters*)]
Adds the contents of *source* onto the end of a disk file.

ECHO echoes characters to screen

when appending a device to a file

STRIP backsquares destination file one byte before appending

APPEND *K1 TO WEST/DAT:0
(ECHO)

ATTRIB *filespec* (*parameters*)
Changes the protection of a file.

USER = "password" sets user password

OWNER = "password" sets owner password

PROT = *level* sets protection level for user:

EXEC
Execute only

READ
Read and execute

UPDATE
Update, read, execute

WRITE
Write, update, read, execute

RENAME
Rename, write, update, read, execute

REMOVE
Remove, rename, write, update, read, execute (total access except changing attributes with **ATTRIB**)

FULL
Total access

VIS specifies *filespec* as visible

INV specifies *filespec* as invisible
ATTRIB CUSTFILE/DAT:1
(USER=" ", OWNER="BOSS",
PROT=READ,VIS)

ATTRIB [*drive*] (*disk parameters*)
Changes the protection of files on a drive.

LOCK sets user and owner passwords of unprotected/visible files to disk master password

UNLOCK removes user and owner passwords of visible files whose passwords match disk master password

MPW = "password" states disk's current master password

Commands and Utilities

NAME[= "disk name"] specifies new disk name
PW[= "password"] sets new disk master password
ATTRIB :! <NAME="DATA" , PW="SECRET" ;MPW="BOSS")

AUTO [parameters] [*] [command line]

Stores a command line for automatic execution each time TRSDOS starts up. (AUTO by itself deletes the current AUTO command line.)

* disables the **(BREAK)** key during boot; disables **(ENTER)** override of AUTO

parameters:

:drive specifies drive to store AUTO command line on
?:drive1 displays AUTO command stored on drive
-(drive) executes AUTO command stored on drive
AUTO BASIC
AUTO *D0 INIT/JCL:1

MPW= "password" states source disk master password
SYS backs up system and visible files
INV backs up invisible and visible files
MOD backs up files modified since last backup
QUERY = YES prompts for each file
OLD backs up only files that already exist on destination disk
NEW backs up only files that do not already exist on destination disk
X backs up with no system disk in Drive 0
DATE = "M1/D1/Y1-M2/D2/Y2"
backs up files modified on or between two dates
= "M1/D1/Y1" backs up files modified on date
= "M1/D1/Y1" backs up files modified on or before date
= "M1/D1/Y1-" backs up files modified on or after date
BACKUP :0 :1
BACKUP (MOD,QUERY,MPW="SECRET")

BACKUP [partspec] [:source drive] [TO] [:destination drive]

[(parameters)]
Duplicates all or some of the files on a diskette.

BOOT [keys]

Resets the system.

(CLEAR) prevents any sysgened parameters from being set; system is left in default state

Commands and Utilities

UNTER disables breakable AUTO command

D enters DEBUG; no syegened configuration or AUTO command is performed
BOOT

BUILD filespec [(parameters)]

Creates an input file for JCL, KSM, and other TRSDOS commands.

HEX accepts data in hexadecimal format

APPEND appends data to end of filespec

BUILD MYPROGA/FIX:0

BUILD DISPLAY/BLD (HEX)

COMM devspec [(parameters)]

Lets two computers communicate via a device.

XLATES = X'abbb' translates a character being sent

XLATER = X'aabb' translates a character being received

XON = X'cc' changes the XON code

XOFF = X'cc' changes the XOFF code

aa is a character to translate from
bb is a character aa is translated to
cc is the new value of XON or XOFF

NULL = OFF prevents any nulls from being received

Application Keys:

CLEAR(1) Keyboard Device (*KI)

CLEAR(2) Display Device (*DU)

CLEAR(3) Printer Device (*PR)

CLEAR(4) Communications Line Device (*CL)

CLEAR(5) Data Send Device (*FS)

CLEAR(6) Data Receive Device (*FR)

Action Keys:

CLEAR(7) Writes contents of *FR to disk (QTD).

CLEAR(8) Displays menu of Function Keys.

CLEAR(9) Specifies what file to use for sending or receiving data (use after pressing **CLEAR**(5) or **CLEAR**(6)).

CLEAR(10) Closes a file opened by **CLEAR**(9).

CLEAR(11) Turns ON a device.

CLEAR(12) Turns OFF a device.

CLEAR(SHIFT)(13) Duplex control.

Follow with **CLEAR**(13) for half-duplex, or **CLEAR**(14) for full-duplex.

CLEAR(SHIFT)(15) Controls character echo.

CLEAR(SHIFT)(16) Controls linefeed echo.

CLEAR(SHIFT)(17) Controls accepting of linefeed.

CLEAR(SHIFT)(18) Clears a file.

CLEAR(SHIFT)(19) Appends new data to a file.

Commands and Utilities

- (CLEAR)(SHIFT)(C)** Displays control characters.
- (CLEAR)(SHIFT)(D)** Clears screen.
- (CLEAR)(SHIFT)(T)** Turns on 8-bit mode (when followed by **(CLEAR)(C)**).
- (CLEAR)(SHIFT)(O)** Allows entry of a TRSDOS library command.
- (CLEAR)(SHIFT)(E)** Controls handshaking.
- (CLEAR)(SHIFT)(=)** Exits to TRSDOS Ready.

CONV [*partspec*]:*source drive* [*destination drive*] [(*parameters*)]
Converts files from a TRSDOS 1.3 (Model III) diskette onto a TRSDOS Version 6 formatted disk.

- VIS** converts visible files
- INV** converts invisible files
- SYS** converts system files
- NEW** converts only files that do not already exist on destination disk
- OLD** converts only files that already exist on destination disk
- QUERY = NO** turns off prompting for each file.
- DIR** displays short directory of TRSDOS 1.3 diskette
- CONV :1 :0 (VIS:0=NO)**

COPY *source* [TO] *destination* [(*parameters*)]
Copies data from one file or device to another.

- LRL = *nnn*** specifies logical record length for *destination*
- CLONE = NO** specifies that *destination* is not to have duplicate directory entry of *source*
- ECHO** echoes characters copied from a devspec to screen
- X** allows a single drive copy from a non-system disk
- COPY TEST/DAT TO :1**
COPY *K1 TO *PR (ECHO)

CREATE *filespec* [(*parameters*)]
Creates a file and reserves space on the disk for future use.

- LRL = *number*** assigns logical record length of file
- REC = *number*** assigns number of fixed-length records in file
- SIZE = *number*** allocates disk space to file (in K)
- CREATE MAIL/DAT:3**
CREATE INVENT/DAT
(**SIZE=20**)

DATE [*mm/dd/yy*]
Sets or displays the current date.

DATE 08/09/82 **DATE**

DEBUG [(*switch*) [,] (*parameter*)]
Sets up the debug monitor for testing and debugging machine-language programs.

switch:

ON turns on DEBUG
 OFF turns off DEBUG

parameter:

EXT specifies the extended
 debugger
 DEBUG DEBUG (OFF)

DEVICE [(parameters)]

Displays the current status of each
 drive and the options in use.

D=NO suppresses drive portion of
 display
 B=YES displays the logical device
 portion of display
 S=NO suppresses options status
 of display
 P=YES prints display on printer
 DEVICE DEVICE (D=YES)

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

Lists the directory of a drive or file.

ALL displays all directory
 information for specified files
 INV displays non-system invisible
 and visible files
 MOD displays files modified since
 last backup
 NON enables non-stop display
 mode
 PRT prints directory on printer
 SYS displays system and visible
 files

DATE displays files with today's
 date

DATE="M1/D1/Y1-M2/D2/Y2"
 displays files modified on or
 between the two dates

- "M1/D1/Y1" displays files
 modified on date

= "M1/D1/Y1" displays files
 modified on or before date

= "M1/D1/Y1." displays files
 modified on or after date

SORT=NO prevents alphabetical
 sorting of directory entries

DIR :1

DIR {DATE="10/01/81.-" }

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

Compiles and executes a DO file.

control characters:

\$ compiles DO file without
 executing it
 = executes DO file without
 compiling it
 + reruns the last compiled DO
 command

parameters:

@label executes filespec starting at
 @label

Commands and Utilities

parm1 = value passes optional value to *filespec* during execution
allows DO command line to be longer than 79 characters
DO DRIVE/JCL
DO=DRIVE/JCL

DUMP *filespec* (*parameters*)

Copies an area of memory to a disk file.

START = address starts dump at *address*

END = address ends dump at *address*

TRA = address sets transfer *address*

ASCII specifies dump is to an ASCII file

ETX = value sets ASCII file end of text character to *value*

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

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

Filters data to or from a device, using a filter program.

```
FILTER *PR USING *DU
```

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

Formats a blank or old disk for use.

ABS overwrites existing data without prompting

NAME = "disk name" assigns name to disk

MPW = "password" assigns master password to disk

SDEN specifies single density disk

DDEN specifies double density disk

CYL = number specifies number of cylinders for disk

STEP = number specifies step rate

QUERY = NO turns off prompts

WAIT = number specifies WAIT value

FORMAT

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

FORMS (*parameters*)

Sets up printer options.

DEFAULT returns all options to their start-up values

ADDLF issues linefeed after every carriage return

CHARS = number sets number of characters per printed line

FFHARD issues a form feed (Top of Form) character instead of a series of linefeeds

INDENT = number sets number of spaces to indent lines that are longer than CHARS

Commands and Utilities

LINES = *number* sets number of lines printed on each page
MARGIN = *number* sets left margin
PAGE = *number* sets physical page length in lines
QUERY prompts for each parameter
TAB specifies that tab characters are to be translated into the appropriate number of spaces
XLATE = 'X*aabb*' specifies a one-character translation
aa is the character to be translated
bb is the character *aa* is translated to
FORMS (**MARGIN** = 10 ,
CHARS = 80 , **INDENT** = 6)

FREE [*drive*] [(*parameter*)]
Lists free space and number of files on each disk; if *drive* is specified, displays space map of that disk.
PRT sends output to printer
FREE **FREE** ;0 (**PRT**)

LIB
Displays library commands.
LIB

LINK *devspec1* [**TO**] *devspec2*
Links two logical devices.
LINK *DO *PR

LIST (*filespec* [(*parameters*)])
Lists contents of a file to the display or printer.
ASCII8 displays graphic and special characters along with text
NUM *numbers* lines in ASCII text files
HEX specifies hexadecimal output format
TAB = *number* specifies that tab stops are to be *number* of spaces apart for ASCII text files (default = 8)
PRT directs output to printer
LINE = *number* sets starting line number
REC = *number* sets starting record number
LRL = *number* formats output using specified logical record length when in hex mode
LIST TESTFILE:0
LIST MONITOR/CMD
(**PRT** , **HEX**)

LOAD [(*parameter*)] *filespec*
Loads a program file into memory.
X loads a file from a non-system disk
LOAD STATUS/CMD
LOAD (X) PROGRAM/CTM

Commands and Utilities

MEMORY [(parameters)]

Reserves a portion of memory, sets or displays current HIGH\$ and LOW\$, modifies a memory address, or jumps to a specified memory location.

CLEAR = *value* fills memory from hex 2600 to HIGH\$ with *value*

HIGH = *address* sets *address* as HIGH\$

LOW = *address* sets *address* as LOW\$

ADD = *address* displays the word at *address* and specifies the address of WORD and BYTE

WORD = *word* changes contents of ADD and ADD + 1 to *word*

BYTE = *byte* changes contents of ADD to *byte*

GO = *address* transfers control to *address*

MEMORY

MEMORY (ADD=X'E100')

WORD=X'3E0A')

PATCH *filespec* (patch commands)

Changes the contents of a disk file.

address = *value* specifies patch by memory load location. Changes contents of memory beginning at *address* to *value*.

Record, byte = *value* specifies direct modify patch. *record* specifies record to be changed (in hex); *byte* specifies first byte to be changed (in hex).

Frecord, byte = *value* checks location specified by the D patch command to make sure it currently contains the data specified by *Frecord, byte*. Also used with REMOVE parameter (see below) to remove a patch and replace it with the original data.

Code specifies library mode patch. *code* is binary coded location in the format *nn* where change begins. *value* is a series of hex bytes in the format *nn nn nn...*, or a string of ASCII characters in the format '*string*'.

PATCH MONITOR/CMD

```
(X'E100'=C3 66 00 C0 03  
40)
```

PATCH *filespec1* USING *filespec2*

[(parameters)]

Makes changes contained in *filespec2* to *filespec1*.

YANK removes the patch specified by *filespec2* from *filespec1*. *filespec2* contains code in the *address* format.

REMOVE removes the patch specified by *filespec1* from *filespec2*. *filespec2* contains code in the *Drecord*, *byte* format.

PATCH BACKUP/CH0:0 USING
SPECIAL/FIX

PURGE [*partspec*]:*drive*
[(*parameters*)]
Deletes files.

QUERY = NO removes files without prompting

MPW = "password" states disk master password

INV removes invisible and visible files

SYS removes system and visible files

DATE = "M1/D1/Y1-M2/D2/Y2"
removes files modified on or between two dates
- "M1/D1/Y1" removes files modified on date
= "M1/D1/Y1" removes files modified on or before date
= "M1/D1/Y1-" removes files modified on or after date

PURGE :0 (MPW="SECRET")
PURGE /BAS:1 (Q=M)

REMOVE *filespec* [*filespec...*]
Deletes files from the directory.

REMOVE ALPHA/DAT:0

REMOVE *devspec* [*devspec...*]
Removes devices from the device table.

REMOVE *LU

RENAME *filespec1* TO *filespec2*

RENAME *devspec1* TO *devspec2*
Changes the name of a file or device.

RENAME TEXT/DAT:0 TO
OLD/DAT
RENAME *UD TO +TX

REPAIR :*drive*
Updates system information on disks which were formatted under Model I TRSDOS so they can be read by TRSDOS Version 6. The disks may not be readable on a Model I after you use this command, so it is best to use REPAIR on a backup copy.

REPAIR :1

Commands and Utilities

RESET *devspec*

RESET *filespec*

Returns a device to its original start-up condition. Closes an open file.

```
RESET *PR
RESET PRINTER/DAT
```

ROUTE *devspec1* [TO] *devspec2*

ROUTE *devspec* [TO] *filespec* [(REWIND)]

ROUTE *devspec1* (NIL)

Routes a device to another device, to a disk file, or to nothing (NIL).

```
ROUTE *PR *DO
ROUTE *PR TO PRINTER/DAT
```

[RUN] [(X)] *filespec* {*command text*}

Loads and executes a program. *command text* is optional values the program may require.

X executes a program from a non-system disk in a single drive

```
RUN CONTROL/CMD
CONTROL/CMD
```

SET *devspec* [TO] *driver* *filespec* [*parameters*]

Assigns a driver program to a device. *parameters* are optional values the driver program may require.

```
SET *SP TO SERIAL/DRV
```

SET *phantom devspec* [TO] *filter* *filespec* [USING] [*parameters*]

Assigns a filter program to a phantom device. *parameters* are optional values the filter program may require.

```
SET *LC TO TRAP/FLT
```

SETCOM [(*parameters*)]

Sets up HS-232C communications or display status.

```
DEFAULT returns all parameters to their start-up values
BAUD=number sets the BAUD rate to number
WORD=number sets the word length to number
STOP=number sets the stop bits to number
PARITY=switch sets the parity switch to YES or NO. If YES, then you can also use EVEN or ODD.
QUERY prompts for each parameter
BREAK=value sets the logical BREAK to value
EVEN sets parity to even if PARITY=YES
ODD sets parity to odd if PARITY=YES
SETCOM (BAUD=300,WORD=8,STOP=1,PARITY=NO)
```


SETKI [(*parameters*)]

Sets keyboard repeat values. (SETKI by itself displays current values.)

DEFAULT returns the parameters to their start-up values

RATE = *number* sets the repeat rate

WAIT = *number* sets initial delay between the time a key is first pressed and the first repeat of that key

QUERY prompts for new RATE and WAIT values

SETKI (WAIT=15)

SPOOL *devspec* [TO] [*diskspec*] (*parameters*)

Establishes an output buffer for a device.

NO turns off spooler and resets *devspec*

MEM = *number* specifies amount of memory buffer (in K) for spooler

BANK = *number* selects one of three 32K memory banks to use as spool buffer (0, 1, or 2)

DISK = *number* specifies amount of disk buffer (in K) for spooler

PAUSE temporarily suspends output to *devspec*

RESUME restarts output to *devspec* after a PAUSE

CLEAR clears the spool buffer

SPOOL *PR TO TEXTFILE:0
(MEM=5,DISK=15)

SPOOL *PR (NO)

SYSGEN [(*switch*)] [,] [DRIVE = *drive*]]

Stores current system options in a file (CONFIG/SYS) on *drive*. If *switch* is NO, the configuration file is removed.

SYSGEN (YES,DRIVE=4)

SYSGEN (NO)

SYSTEM (*parameters*)

Selects certain options of your TRSDOS system. In the following SYSTEM commands, *switch* is YES or NO.

SYSTEM (ALIVE! = *switch*)

Displays a moving character when task processor is running.

SYSTEM (BLINK = *switch*)

SYSTEM (BLINK = *number*)

SYSTEM (BLINK!,*parameter*)

Control the cursor character.

number specifies ASCII value in decimal for cursor character

parameter is LARGE or SMALL.

If *parameter* is omitted, the cursor returns to its start-up character and size.

SYSTEM (BREAK! = *switch*)

Enables or disables BREAK key.

Commands and Utilities

SYSTEM (BSTEP = *number*)

Sets default bootstrap step rate used with FORMAT utility.

SYSTEM (DATEI = *switch*)

Turns on or off the start-up date prompt.

SYSTEM (DRIVE = *drive, parameters*)

Sets the following parameters for *drive*:

CYL = *number* sets default number of cylinders used with FORMAT utility (35 to 96)

DELAY = NO/YES sets DELAY time for floppy disks

DISABLE removes access to *drive*

ENABLE allows access to drive that has been disabled

STEP = *number* sets step rate for *drive*

DRIVER = "*filespec*" configures hard drive

WPI = *switch* sets software write protect

SYSTEM (FAST)

Switches system to fast clock rate.

SYSTEM (GRAPHIC)

Specifies that printer can reproduce TRS-80 graphic characters during screen prints.

SYSTEM (RESTOREI = *switch*)

Determines whether all drives are to be restored when the system is reset.

SYSTEM (SLOW)

Switches system to slow clock rate.

SYSTEM (SYSRES = *number*)

Adds TRSDOS system overlays into high memory. *number* is 1-5 or 9-12.

SYSTEM (SYSTEM = *drive*)

Assigns *drive* as system drive.

SYSTEM (TIMEI = *switch*)

Turns on or off the start-up time prompt.

SYSTEM (TRACEI = *switch*)

Continuously displays contents of Program Counter.

SYSTEM (TYPEI = *switch*)

Turns on or off the type-ahead feature.

TIME [*hh:mm:ss*] [(*parameter*)]

Sets the time or displays current time.

CLOCK = YES turns clock display on

CLOCK = NO turns clock display off if it was on

TIME 12:29:34

VERIFY [(*switch*)]

Sets VERIFY function on or off.

VERIFY (YES)

VERIFY (NO)

Drivers, Filters, and Devices

CLICK/FLT

SET *devspec* [TO] CLICK/FLT
FILTER *K1 *devspec*
Establishes the key-click filter.

COM/DVR

SET *CL [TO] COM/DVR
Prepares the Communications Line
(*CL) for use.

FORMS/FLT

SET *FF [TO] FORMS/FLT
Prepares the Printer Filter (*FF)
for use.

JOBLOG

ROUTE *JL [TO] *filespec*
ROUTE *JL [TO] *devspec*
Establishes the Joblog device (*JL),
which sends certain information to a
file or device.
ROUTE *JL TO LISTER/JBL
ROUTE *JL TO *PR

KSM/FLT

SET *devspec* KSM [USING] *filespec*
[(*parameter*)]
FILTER *K1 *devspec*
Establishes the Keystroke Multiply
filter, which allows you to assign a
sequence of characters to one key.
Parameter:

ENTER = *value* specifies the
character TRSDOS recognizes as
an ENTER character in a KSM file.
SET *DU KSM/FLT USING
ROUTINE/KSM
FILTER *K1 *DU

MEMDISK/DCT

SYSTEM (DRIVE = *drive*, DRIVER =
"MEMDISK")
Adds to the system a pseudo floppy
drive which keeps its files in memory.
SYSTEM (DRIVE = 2, DRIVER =
"MEMDISK")

Notes:

Error Messages

Number	Message	Explanation/Action
7 X'07'	Attempted to read locked/deleted data record	Check for error in program
6 X'06'	Attempted to read system data record	Check for error in program
5 X'05'	Data record not found during read	Try again; use another disk; reformat old disk
13 X'0D'	Data record not found during write	Try again; use another disk
39 X'27'	Device in use	Reset device in use before REMOVING it
8 X'08'	Device not available	Check device specification; make sure peripheral is ready
30 X'1E'	Directory full — can't extend file	Copy files to new disk
17 X'11'	Directory read error	Try another drive or disk
18 X'12'	Directory write error	Try another disk
27 X'1B'	Disk space full	Write file to a disk with more available space
28 X'1C'	End of file encountered	Check for error in program
63 X'3F'	Extended error	Error code is in HL register
25 X'19'	File access denied	Use correct password; use no password for unprotected file
41 X'29'	File already open	Use RESET to close the file

Error Messages

24 X'18'	File not in directory	Check spelling of filespec
38 X'26'	File not open	Open file before access
20 X'14'	GAT read error	Try another drive
21 X'15'	GAT write error	Try another drive or disk
22 X'16'	HIT read error	Try another drive
23 X'17'	HIT write error	Try another drive or disk
37 X'25'	Illegal access attempted to protected file	OWNER password is required for the requested access
32 X'20'	Illegal drive number	Drive is not in system or not ready for access
19 X'13'	Illegal file name	Use proper filespec syntax
16 X'10'	Illegal logical file number	Check for error in program
34 X'22'	Load file format error	Attempt was made to load a non-program file
3 X'03'	Lost data during read	Try another drive or disk
11 X'0B'	Lost data during write	Try another drive or disk
42 X'2A'	LRL open fault	COPY file to another file that has the specified LRL

Error Messages

33 X'21'	No device space available	REMOVE non-system devices to provide more space
26 X'1A'	No directory space available	Use a different disk or REMOVE unwanted files
0 X'00'	No error	Check for error in program
1 X'01'	Parity error during header read	Try another drive or disk
9 X'09'	Parity error during header write	Try another drive or disk
4 X'04'	Parity error during read	Try another drive or disk
12 X'0C'	Parity error during write	Try another drive or disk
31 X'1F'	Program not found	Check spelling of filespec; check for proper disk in drive
40 X'28'	Protected system device	System devices cannot be REMOVED
29 X'1D'	Record number out of range	Provide correct record number or try another copy of the file
2 X'02'	Seek error during read	Set step rate with SYSTEM command or try another drive or disk
10 X'0A'	Seek error during write	Set step rate with SYSTEM command or try another drive or disk
—	Unknown error code	Check for error in program

Error Messages

14 X'0E'	Write fault on disk drive	Try another disk or drive
15 X'0E'	Write protected disk	Remove write-protect tab or write enable disk using SYSTEM command

BASIC Statements and Functions

Terms:

integer:

a whole number from -32768 to 32767

string:

a sequence of characters which is to be taken verbatim

dummy number or dummy string:

a number or string used in an expression to meet syntactic requirements, but whose value is insignificant.

ABS (number)

Computes the absolute value of *number*.

$Y = ABS(X)$

ASC (string)

Returns the ASCII code for the first character of *string*.

`PRINT ASC("A")`

ATN (number)

Computes the arctangent of *number*; returns the value in radians.

$Y = ATN(X/3)$

AUTO [*line number*] [*increment*]

Automatically generates line numbers every time you press **ENTER**. AUTO begins numbering at *line number* and displays the next line using *increment*.

```
AUTO AUTO 1000 : 100  
AUTO , 5
```

CALL *address* [(*parameter list*)]

Transfers program control to an assembly-language subroutine stored at *address*. The *parameter list* contains the values to be passed to the external subroutine.

CDBL (*number**)

Converts *number* to double precision.

$Y = CDBL(N*3)$

CHAIN [MERGE] [*filespec*] [*line*]

[ALL] [,DELETE *line - line*]

Loads a BASIC program named *filespec*, chains it to a "main" program, and begins running it. The *line* is the number of the first line to be run in the CHAINED program. The ALL option passes every variable in the main program to the CHAINED program. The MERGE option "overlays" the lines of *filespec* with the main program. The DELETE

BASIC Statements and Functions

option erases lines in the overlay so that you can MERGE in a new overlay.

```
CHAIN "SUBPROC/0A5", ALL
```

CHR\$ (code)
Returns the corresponding character of the ASCII or control code.

```
PRINT CHR$(35)
```

CINT (number)
Converts *number* to integer representation.

```
PRINT CINT(17.65)
```

CLEAR [memory location]
[stack space]
Clears the value of all variables and closes all open files. Optionally, it also sets the highest *memory location* for BASIC and the amount of *stack space*.

```
CLEAR CLEAR ,75  
CLEAR ,61000 ,200
```

CLOSE [buffer...]
Closes access to a file. The *buffer* number (the same used to OPEN the file) may be from 1 to 15.

```
CLOSE 1, 2, B  
CLOSE FIRSTX + COUNTX
```

CLS
Clears the screen.

```
CLS
```

COMMON variable....
Passes one or more variables to a CHAINED program.

```
100 COMMON A, B, C,  
    D): G#  
110 CHAIN "PROC3", 10
```

CONT
Resumes execution of a program when it has been stopped by the **BREAK** key or by a STOP or an END statement in the program.

```
CONT
```

COS (number)
Computes the cosine of *number*.

$$Y = \text{COS}(X * .01745329)$$

CSNG (number)
Converts *number* to single precision.

```
CSNG (.1453885509)
```

CVD (8 byte string)
Restores the string value to a numeric value.

```
A# = CVD (GROSSPAY#)
```

BASIC Statements and Functions

CVI (2-byte string)

Restores the string value to a numeric value.

CVS (4-byte string)

Restores the string value to a numeric value.

DATA constant,...

Stores numeric and string constants to be accessed by a READ statement

```
1340 DATA NEW YORK ,  
      CHICAGO , LOS  
      ANGELES ,  
      PHILADELPHIA ,  
      DETROIT  
1350 DATA 2.72 , 3.14159 ,  
      0.0174533 , 57.29579
```

DATE\$

Returns today's date,
PRINT DATE\$

DEFDBL/INT/SNG/STR

```
DEFDBL letter,...  
DEFINT letter,...  
DEFSNG letter,...  
DEFSTR letter,...
```

Defines any variables beginning with the letter(s) as: (DBL) double

precision, (INT) integer, (SNG) single precision, or (STR) string.

```
DEFDBL L-P  
DEFINT I-N, W, Z  
DEFSNG I, Q-T DEFSTR A
```

DEF FN function name

{[argument,...]} = function definition

Defines function name according to function definition. The argument represents those variables in the function definition that are to be replaced when the function is called.

```
DEF FNR=RND*(99)+9
```

DEF USE [digit] = address

Defines the starting address for digit assembly language subroutines.

```
DEF USE3 = &H7D00  
DEF USE = (BASE + 16)
```

DELETE line1 - line2

Deletes from line1 to line2 of a program in memory.

```
DELETE 70  
DELETE 50-110  
DELETE
```

DIM array (dimension(s)), array (dimension(s))...

Sets aside storage for the arrays with the dimensions you specify.

```
DIM AR(100)  
DIM LX(0,25)
```

BASIC Statements and Functions

EDIT *line*

Enters the edit mode so that you can edit *line*.

```
EDIT 100 EDIT 1
```

END

Ends execution of a program.
END

EOF(*buffer*)

Detects the end of a file.

```
IF EOF(1) THEN 1540
```

ERASE *array*...

Erases one or more *arrays* from a program.

```
ERASE C:F
```

ERL

Returns the line number in which an error has occurred

```
PRINT ERL E = ERL
```

ERR

Returns the error code (if an error has occurred).

```
IF ERR = 7 THEN 1000  
ELSE 2000
```

ERR%

Returns a system error number and message.

```
PRINT "THE LATEST TRSDOS  
ERROR IS "ERR%
```

ERROR *code*

Simulates the error associated with *code* during program execution.

```
ERROR 1
```

EXP (*number*)

Calculates the natural exponential of *number*.

```
PRINT EXP(-2)
```

FIELD *buffer*, *length* AS *field name*...

Divides a direct-access *buffer* into one or more fields. Each field is identified by the *field name* and is the *length* you specify.

```
FIELD 3, 12B AS A$,  
12B AS B$
```

FIX (*number*)

Returns the truncated integer of *number*.

```
PRINT FIX(2.6)
```

FOR/NEXT

FOR *variable* = *initial value* TO *final value* [STEP *increment*]

BASIC Statements and Functions

NEXT (*variable*)

Establishes a program loop.

```
20 FOR H=1 TO -10
   STEP -2
30 PRINT H
40 NEXT H
```

FREE (*dummy number*) or (*dummy string*)

Returns the amount of free memory space.

```
PRINT FREE(40)
PRINT FREE("40")
```

GET *buffer* [, *record number*]

Gets a *record* from a direct disk file and places it in a *buffer*.

```
GET 1 GET 1, 25
```

GOSUB *line*

Goes to a subroutine, beginning at the specified *line*.

```
GOSUB 1000
```

GOTO *line*

Goes to the specified *line*.

```
GOTO 100
```

HEX\$ (*number*)

Calculates the hexadecimal value of *number*.

```
PRINT HEX$(30),
HEX$(50), HEX$(90)
```

IF...THEN...ELSE

If *expression* THEN *statement(s)* or *line* [ELSE *statement(s)* or *line*]

Tests a conditional expression and makes a decision regarding program flow.

```
10 IF X > 127 THEN PRINT
   "OUT OF RANGE" : END
20 IF A < B THEN PRINT
   "A < B" ELSE IF A = B
   THEN PRINT "A = B"
   ELSE PRINT "A > B"
```

INKEY\$

Returns a keyboard character.

```
A$ = INKEY$
```

INP(*port*)

Returns the byte read from a *port*. *Port* may be any integer from 0 to 255.

```
A = INP(42)
```

INPUT [*prompt string*]; *variable1*, *variable2*,...

Inputs data to a program during execution.

```
INPUT YX
```

INPUT# *buffer*, *variable*...

Inputs data from a sequential disk file into one or more *variables*.

```
[INPUT#1, A, B
[INPUT#0, A$, B$, C$
```

BASIC Statements and Functions

INPUT\$ (*number* [, *buffer*])
Inputs a string of *number* characters from either the keyboard or a sequential disk file. The *number* must be a value from 1 to 255.
A\$ = INPUT\$(5)
A\$ = INPUT\$(11,3)

INSTR(*integer*;*string 1*, *string 2*)
Searches for the first occurrence of *string 2* in *string 1* and returns the position at which the match is found.
INSTR(A\$, "12")

INT(*number*)
Converts *number* to integer value.
PRINT INT(79.89)

KILL "*filespec*"
Removes *filespec* from the disk.
KILL "FILE/BAS"
KILL "DATA:2"

LEFT\$(string, integer)
Returns all characters left of position *integer* in the *string*.
PRINT
LEFT\$("BATTLESHIPS", 6)

LEN(string)
Returns the number of characters in *string*.
X = LEN(SENTENCE\$)

[LET] variable = expression
Assigns the value of *expression* to *variable*.
LET A\$ = "A ROSE IS A
ROSE"
LET B1 = 1.23

LINE INPUT [*prompt string*]; *string variable*
Inputs a line from the keyboard
LINE INPUT A\$

LINE INPUT# *buffer variable*
Reads a line of data from a sequential-access file into a *string variable*. The *buffer* is the number used when the file was OPENed.
LINE INPUT# 1, A\$

LIST {*startline*} - {*endline*}
Lists program lines to the display.
LIST 50 LIST 50-85
LIST -227

LLIST [*startline*] - [*endline*]
Lists program lines to the line printer.
LLIST 780 LLIST 50-
LLIST .-

LOAD "*filespec*" [,R]
Loads *filespec*, a BASIC program, into memory. If R is used, the

BASIC Statements and Functions

program is RUN automatically.

```
LOAD "PROG1/BAS:Z"  
LOAD "PROG1/BAS"
```

LOC (buffer)

Returns the current record number.

```
IF LOC(1)>55 THEN END
```

LOF (buffer)

Returns the end-of-file record number.

```
Y = LOF(5)
```

LOG(number)

Computes the natural logarithm of *number*.

```
PRINT LOG(3.14159)  
Z = 10*LOG(PS/P1)
```

LPOS (number)

Returns the position of the line printer's print head within the line printer's buffer.

```
IF LPOS(X)>60 THEN PRINT  
CHR$(13)
```

LPRINT data...

Prints *data* at the printer.

```
LPRINT (A + 2)/3
```

LPRINT USING format: data...

Prints *data* at line printer, using a specified *format*.

```
LPRINT USING "#####.##";  
2.17
```

LSET field name = data

Sets *data* in a direct-access buffer *field name*. The data is left-justified.

```
LSET NM$ = "JIM CRICKET,  
JR."
```

MEM

Returns the amount of memory.

```
PRINT MEM
```

MERGE "filespec"

Loads *filespec*, a BASIC program, and merges it with the program currently in memory.

```
MERGE "PROG2/TXT"
```

MID\$(old string, position[, length]) = replacement string

Replaces a portion of *old string* with *replacement string*.

```
MID$(A$, 3, 4) =  
"12345": PRINT A$
```

MID\$(string, integer[, number])

Returns a substring of the *string*, beginning with the *integer* character. *Number* is the number of characters to include in the substring.

```
MID$(A$, 3, 2)
```

BASIC Statements and Functions

MKD\$(double-precision expression)

Converts *double-precision expression* to a string value and returns the 4-byte string.

MKI\$(integer expression)

Converts *integer expression* to a string value and returns the 8-byte string.

MKS\$(single-precision expression)

Converts *single-precision expression* to a string value and returns the 2-byte string.

NAME old filespec AS new filespec

Renames *old filespec* as *new filespec*.

```
NAME "FILE" AS  
"FILE/OLD"
```

NEW

Erases a program from memory and clears all variables.

```
NEW
```

OCT\$(number)

Computes the octal value of *number*.

```
PRINT OCT$(30),  
OCT$(50), OCT$(99)
```

ON ERROR GOTO line

Goes to a subroutine at the *line* specified by the value of *number*.
ON ERROR GOTO 1500

ON expression GOSUB line...

Goes to a subroutine at the *line* specified by the value of *expression*.
ON L-1 GOSUB 1000, 2000,
3000

ON expression GOTO line...

Goes to the *line* specified by the value of *expression*.
ON X GOTO 100, 200, 210

OPEN mode,buffer,"filespec"

[,*record length*]

Opens a disk file in the specified mode (O for sequential output, I for sequential input, D or R for direct input/output, and E for sequential extend).

```
OPEN "O", 1,  
"CLIENTS/TXT"  
OPEN "D", 5,  
"TESTED/BA5", 64
```

OPTION BASE n

Sets *n* as the minimum value for an array subscript.

```
OPTION BASE 1
```


BASIC Statements and Functions

OUT *port, data byte*

Sends *data byte* to a machine output *port*.

```
OUT 32,100
```

PEEK(*memory location*)

Returns a byte from *memory location*.

```
A = PEEK ($HSA00)
```

POKE *memory location, data byte*

Writes a *data byte* into *memory location*.

```
POKE 15360, 191
```

POS(*number*)

Returns the position of the cursor. *Number* is a dummy argument.

```
PRINT TAB(40) POS(0)
```

PRINT *data...*

Prints numeric or string data on the display.

```
PRINT X, Y PRINT "*"
```

PRINT TAB(*n*)

Moves the cursor to the *n* position on the current line (or on succeeding lines if you specify TAB positions greater than 79).

```
PRINT TAB(5) "TABBED 5":  
TAB(25) "TABBED 25"
```

PRINT USING *format; data item...*

Prints *data items* using the specified *format*.

```
PRINT USING "! "IA#:
```

PRINT @ *location.*

PRINT @ (*row, column*).

Specifies where printing is to begin.

```
PRINT @ 0, "*"
```

PRINT# *buffer; item 1, item 2,...*

Prints *data items* in a sequential disk file.

```
PRINT#1, A;B
```

PUT *buffer; record*

Puts a *record* in a direct-access file. *Buffer* is the number used to OPEN the file.

```
PUT 1 PUT 1, 25
```

RANDOM

Reseeds the random number generator.

```
RANDOM
```

READ *variable...*

Reads values from a DATA statement and assigns them to *variables*.

```
READ T READ S#, T, U
```

BASIC Statements and Functions

REM

Inserts a remark line into a program and instructs the computer to ignore the rest of the program line.

```
10 REM INPUT  
   SINGLE-PRECISION  
20 INPUT A
```

RENUM [*new line*] [*line*] [*increment*]

Renumbers a program, starting at the specified *line* and numbering it as *new line*. The optional *increment* sets the increment to be used between each line number.

```
RENUM  
RENUM 6000, 5000, 100
```

RESTORE [*line*]

Restores a program's access to previously read DATA statements.

```
RESTORE
```

RESUME [*line*]

RESUME NEXT

Resumes program execution after an error-handling routine has been performed. RESUME *line* causes BASIC to branch to the specified line. RESUME NEXT causes it to branch to the statement following the point at which the error occurred.

```
RESUME RESUME 10  
RESUME NEXT
```

RETURN

Returns control to the line immediately following the most recently executed GOSUB.

```
RETURN
```

RIGHT\$(*string*, *number*)

Returns the last *number* characters of the *string*.

```
PRINT  
  RIGHT$("WATERMELON", 5)
```

RND(*number*)

Generates a pseudorandom *number* between 0 and the *number*. The *number* must be greater than 0 and less than 32768.

```
A = RND(12)   B = RND(45)  
PRINT RND(0)
```

ROW(*number*)

Returns the row position of the cursor. *Number* is a dummy argument.

```
X = ROW(Y)
```

RSET *field name* = *data*

Places *data* in a direct-access buffer *field name*.

```
RSET NM# = "JIM CRICKET,  
JR."
```

BASIC Statements and Functions

RUN *(line)*

RUN *filespec* [,R]

RUN or RUN *line* runs the program that is in memory. RUN *filespec* loads a program from disk, then runs it.

```
RUN "PROGRAM/A"  
RUN "EDITDATA", R
```

SAVE *filespec* [,A] [,P]

Saves a program in a disk under *filespec*. A causes the file to be stored in ASCII format. P causes the file to be stored in an encoded binary format.

```
SAVE "FILE1/  
BAS.JOHNDOE:3"  
SAVE "MATHPAK/TXT", A
```

SGN (*number*)

Determines *number*'s sign. If *number* is positive, SGN returns 1. If it is negative, SGN returns -1. If it is zero, SGN returns 0.

```
Y = SGN(A * B)
```

SIN (*number*)

Computes the sine of *number*; the *number* must be in radians.

```
PRINT SIN(7.96)
```

SOUND *tone*, *duration*

Generates a sound with the specified *tone* (0-7) and *duration* (0-31).

```
SOUND 1, 2
```

SPACES (*number*)

Returns a string of *number* spaces. The *number* must be from 0 to 255.

```
PRINT "DESCRIPTION"  
SPACE$(4)
```

SPC (*number*)

Prints a line of *number* blanks. The *number* must be from 0 to 255.

```
PRINT "HELLO" SPC(15)  
"THERE"
```

SQR (*number*)

Calculates the square root of *number*.

```
PRINT SQR(150.7)
```

STOP

Stops program execution.

```
STOP
```

STR\$(*number*)

Converts *number* into a string. If the *number* is positive, STR\$ places a blank before the string.

STRING\$(*number*, *character*)

Returns a string of the specified *number* of characters. The *number* must be from 0 to 255. The *character* is a string or an ASCII code.

```
0$ = STRING$(25, "X")  
PRINT STRING$(0, 10)
```

BASIC Statements and Functions

SWAP *variable1*, *variable2*

Exchanges the values of two variables.

```
SWAP F1*, F2*
```

SYSTEM [*command*]

Returns to TRSDOS. If you specify a *command*, TRSDOS executes it and returns you to BASIC.

```
SYSTEM SYSTEM "DIR"
```

TAB(*number*)

Spaces to position *number* on the display. The *number* must be from 0 to 255.

```
PRINT A$ TAB(25) B$
```

TAN(*number*)

Computes the tangent of *number*. The *number* must be in radians. If it is in degrees, use **TAN**(*number* * .11745329). The result is always single-precision.

```
PRINT TAN(7.56)
```

TIME\$

Returns the time (in 24-hour format).

```
A$ = TIME$
```

TRON

Turns off the trace function.

```
TRON
```

TRON

Turns on the trace function (to id/low program /low).

```
TRON
```

USR(*digit*) (*expression*)

Calls the user's assembly-language subroutine identified by *digit* and passes the result of *expression*.

```
X = USR(Y)
```

VAL(*string*)

Calculates the numeric value of *string*. VAL terminates its evaluation on the first character that has no meaning in a numeric term.

```
PRINT VAL("100 DOLLARS")
```

VARPTR (*variable*) or (#*buffer*)

Returns the absolute memory address. When used with a *variable*, **VARPTR** returns the address of the first byte of data identified with *variable*. When used with a *buffer*, it returns the address of the file's data buffer.

```
Y = USR1(VARPTR(X))
```

WAIT *port*, *integer1* [, *integer2*]

Suspends program execution until a machine input *port* develops a specified bit pattern.

```
WAIT 32:2
```

WHILE *expression*

:

{loop statements}

WEND

Executes a series of statements in a loop as long as a given condition is true.

```
WHILE...WEND
```

WRITE [*data...*]

Prints *data* on the display.

```
WRITE A,B,C$
```

WRITE# *buffer*, *data...*

Writes *data* to a sequential file.

```
WRITE#1,A$B$
```

Notes:

Control Keys

Command Mode

- ←** or **(CTRL)B** Backspaces the cursor, erasing the preceding character in the line.
- (SPACEBAR)** Enters a blank space and advances the cursor one space.
- (BREAK)** Interrupts line entry and starts over with a new line.
- (CTRL)J** Line feed— Starts a new physical line without ending the current logical line.
- (CAPS)** or **(SHIFT)Q** Switches to either all upper case or all lower case.

(ENTER) Ends and enters the current logical line.

Execution Mode

- (SHIFT)C** Pauses execution. Press any other key (except **(BREAK)**) to continue.
- (BREAK)** Terminates execution and returns to command mode.
- (ENTER)** Interprets data entered from the keyboard with the **INPUT** statement.

Operators

Each operator or group of operators is precedent over the group below it.

- () (Parentheses)
- ^ (Exponentiation)
- +, - (Unary positive, negative)
- *, \ (Multiplication, division)
- / (Integer division)
- MOD (Modulus)
- +, - (Addition, subtraction)
- >, <, =, <=, >=, <> (Relational tests)
- NOT
- AND
- OR
- XOR
- EQV
- IMP

Notes:

Edit Commands

(A)	Moves the cursor to the beginning of the line and cancels editing changes.	(I)	at the current cursor position. Lets you insert material at the current cursor position.
n(BACKSPACE)	Moves the cursor <i>n</i> spaces to the left. If no <i>n</i> is given, moves cursor one space to the left.	n(C)	Deletes all characters up to the <i>n</i> th occurrence of character <i>c</i> and moves the cursor to that position.
n(D)	Lets you change <i>n</i> characters, beginning at the current cursor position.	(L)	Lists the line.
n(D)	Deletes <i>n</i> characters to the right of the cursor.	(Q)	Quits edit mode and cancels all changes.
(E)	Ends editing and saves all changes.	n(S)	Searches for <i>n</i> th occurrence of character <i>c</i> and moves the cursor to that position.
(ENTER)	Records all changes and exits edit mode.	n(SPACEBAR)	Moves the cursor <i>n</i> spaces to the right.
(SHIFT)(X)	Escapes from an insert subcommand (I, H, or X).	(X)	Displays the rest of the line and lets you add material at the end.
(R)	Deletes the rest of a line and lets you insert material		

Special Characters

' (apostrophe)	Abbreviation for :REM.	.	Indicates current line; use with EDIT and LIST commands.
, (comma)	PRINT punctuation; spaces over to the next 16-column PRINT zone. Also separates parameters in command line.	D	Used in double-precision exponential notation.
: (colon)	PRINT punctuation; separates items in a PRINT list but does not add spaces when they are output	E	Used in single-precision exponential notation.
! (exclamation point)	Separates statements on the same line.	%	Makes variable integer-precision.
		!	Makes variable single-precision.
		#	Makes variable double-precision.
		\$	Makes variable string type.

Notes:

Error Messages

Code	Explanation	Code	Disk Errors Explanation
1	NEXT without FOR	50	FIELD overflow
2	Syntax error	51	Internal error
3	RETURN without GOSUB	52	Bad file number
4	Out of DATA	53	File not found
5	Illegal function call	54	Bad file mode
6	Overflow	55	File already open
7	Out of memory	57	Device IO error
8	Undefined line number	58	File already exists
9	Subscript out of range	61	Disk full
10	Duplicate definition	62	Input past end
11	Division by zero	63	Bad record number
12	Illegal direct	64	Bad file name
13	Type mismatch	66	Direct statement in file
14	Out of string space	67	Too many files
15	String too long	68	Disk write protected
16	String formula too complex	69	File access DENIED
17	Can't continue	70	Command Aborted
18	Undefined user function		
19	No RESUME		
20	RESUME without error		
21	Unprintable error		
22	Missing operand		
23	Line buffer overflow		
26	FOR without NEXT		
29	WHILE without WEND		
30	WEND without WHILE		

Notes:

Reserved BASIC Words and Internal Codes

Keyword	Code	Keyword	Code	Keyword	Code
ABS	65414	EQV	251	LOG	65418
AND	246	ERASE	166	LPOS	65435
ASC	65429	ERL	215	LPRINT	157
ATN	65422	ERR	216	LSET	201
AUTO	171	ERROR	168	MEM	225
CALL	182	ERRBS	223	MERGE	187
CDBL	65438	EXP	65419	MID\$	65411
CHAIN	165	FIELD	192	MKD\$	65458
CHR\$	65430	FX	65439	MKIS	65456
CINT	65436	FN	212	MKS\$	65457
CLEAR	146	FOR	130	MOD	253
CLOSE	196	FRE	65423	NAME	199
CLS	159	GET	193	NEW	148
COMMON	184	GOSUB	141	NEXT	131
CONT	153	GOTO	137	NOT	214
COS	65420	HEX\$	65434	OCT\$	65433
CSNG	65437	IF	139	ON	149
CVD	65452	IMP	252	OPEN	191
CYL	65450	INKEY\$	224	OPTION	186
CYS	65451	INP	65424	OR	249
DATA	132	INPUT	133	OUT	166
DATES	222	INSTR	219	PEEK	65431
DEF	151	INT	66413	POKE	152
DEFDBL	176	KILL	200	POS	65425
DEFINT	174	LEFT\$	65409	PRINT	145
DEFSNG	175	LEN	65426	PUT	194
DEFSTR	173	LET	136	RANDOM	187
DELETE	170	LINE	177	READ	135
DIM	134	LIST	147	REM	143
EDIT	167	LLIST	158	RENUM	172
ELSE	162	LOAD	196	RESTORE	140
END	129	LOC	65454	RESUME	169
EOF	65153	LOF	65455	RETURN	142

Reserved BASIC Words and Internal Codes

Keyword	Code	Keyword	Code	Keyword	Code
RIGHT\$	65410	STRING\$	217	WAIT	160
RND	65416	SWAP	185	WEND	181
ROW	65459	SYSTEM	189	WHILE	180
RSET	202	TAB	200	WRITE	183
RUN	138	TAN	65421	XOR	250
SAVE	203	THEN	208	+	243
SGN	65412	TIMES	228	-	244
SIN	65417	TO	207	*	245
SOUND	205	TROFF	164	/	246
SPACE\$	65432	TRON	163	^	247
SPC	213	USING	218	\	254
SQR	65415	USR	211	.	220
STEP	210	VAL	65428	>	240
STOP	144	VARPTR	221	=	241
STR\$	65427			<	242

Notes:

RADIO SHACK, A DIVISION OF TANDY CORPORATION

U.S.A.: FORT WORTH, TEXAS 76102
CANADA: BARRIE, ONTARIO L4M 4W5

TANDY CORPORATION

AUSTRALIA

81 KURRAJONG ROAD
MOUNT BRUXT, N.S.W. 2770

BELGIUM

PAPC INDUSTRIEL DE NANINNE
5140 NANINNE

U.K.

BILSTON ROAD WEDNESBURY
WEST MIDLANDS W510 7JN