

# TRS-80<sup>®</sup>

## Color Computer 2

### MICRO- COMPUTER SYSTEM



### Start-Up

1. Turn the television set ON.
2. Select channel 3 or 4.
3. Set the Antenna Switch to "COMPUTER."
4. Turn on any accessory equipment (e.g., a printer).
5. If you're using a Program Pak<sup>™</sup>, insert it now, before turning on the Computer.
6. Turn the Computer ON.
7. If you're not using a Program Pak<sup>™</sup>, the Color BASIC or Extended Color BASIC start-up message will appear on the TV, followed by: OK

The Computer is now ready to use.

**Note:** Information pertaining to Extended Color BASIC *only* is shaded like this paragraph. Non-shaded information pertains to both Extended and non-Extended Color BASIC.

## Radio Shack<sup>®</sup>

The biggest name in little computers<sup>™</sup>

# Video Control Codes

Dec	Hex	PRINT CHR\$ (code)
8	08	Backspaces and erases current character.
13	0D	Line feed with carriage return.
32	20	Space

# Operators

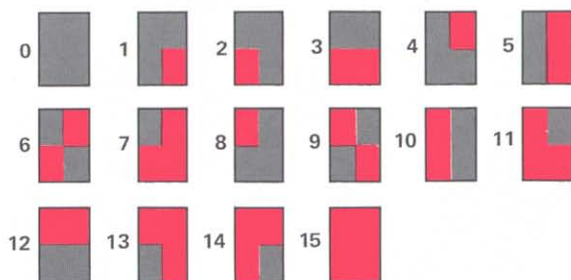
Each operator or group of operators is precedent over the group below it.

↑	Exponentiation
-, +	Unary negative, positive
*, /	Multiplication, division
+, -	Addition and concatenation, subtraction
<, >, =, <=, >=, <>	Relational tests
NOT	
AND	
OR	

# Graphic Character Codes

Given the *color* (1-8) and the *pattern* (0-15), this formula will generate the correct code:

$$\text{code} = 128 + 16 * (\text{color} - 1) + \text{pattern}$$



For example, to print *pattern* 9 in blue (*code* 3), type:

$$C = 128 + 16 * (3 - 1) + 9$$

? CHR\$ (C)

# TRS-80® C

# Functions

Argument ranges are indicated by special symbols:

*numeric*:  $(-10^{38}, +10^{38})$

*x*: (0-255)

*y*: (0-191)

*location*: (0-65535)

*code*: (0-255)

*str*: string argument

*var*: variable name

**ABS (*numeric*)** Computes absolute value.

`Y=ABS(5)`

**ASC (*str*)** Returns ASCII code of first character of specified string.

`A=ASC(T$)`

**ATN (*numeric*)** Returns arctangent in radians.

`Y=ATN(X/3)`

**CHR\$ (*code*)** Returns character for ASCII, control, or graphics code.

`?CHR$(191)`

`P$=CHR$(T)`

**COS (*numeric*)** Returns cosine of an angle given in radians.

`Y=COS(7)`

**EOF (*f*)** Returns FALSE (0) if there is more data; TRUE(-1) if end of file has been read. For cassette,  $f = -1$ ; for keyboard,  $f = 0$ .

`EOF=(-1)`

`EOF(0)`

**EXP (*numeric*)** Returns natural exponential of number ( $e^{\text{number}}$ ).

`Y=EXP(7)`

**FIX (*numeric*)** Returns truncated (whole number) value.

`Y=FIX(7.6)`

**HEX\$ (*numeric*)** Computes hexadecimal value.

`PRINT HEX$(30)`

`Y=HEX$(X/16)`

**INKEY\$** Checks the keyboard and returns the key being pressed (if any).

`A$=INKEY$`

**INT (*numeric*)** Converts a number to an integer.

`X=INT(5.2)`

# OLOR BASI

**JOYSTK (j)** Returns the horizontal or vertical coordinate (j) of the left or right joystick:

- 0 = horizontal, left joystick
- 1 = vertical, left joystick
- 2 = horizontal, right joystick
- 3 = vertical, right joystick

M=JOYSTK(0)

H=JOYSTK(K)

**LEFT\$(str,length)** Returns left portion (length characters) of string.

P\$=LEFT\$(M\$,7)

**LEN(str)** Returns the length of a string.

X=LEN(SEN\$)

**LOG(numeric)** Returns natural logarithm.

Y=LOG(353)

**MEM** Finds the amount of free memory.

PRINT MEM

**MID\$(str,pos,length)** Returns a substring of another string starting at pos. If length is omitted, the entire string right of position is returned.

F\$=MID\$(A\$,3)

?MID\$(A\$,3,2)

**PEEK(location)** Returns the contents of specified memory location.

A=PEEK(32076)

**POINT(x,y)** Tests whether specified graphics cell is on or off, x (horizontal) = 0-63; y (vertical) = 0-31. The value returned is -1 if the cell is in a text character; mode; 0 if it is off, or the color code if it is on. See CLS for color codes.

IF POINT(10,10) THEN PRINT "ON" ELSE PRINT "OFF"

**POS(device)** Returns current print position. Device-1 = printer, -2 = display.

PRINT TAB(8) POS(0)

**PPOINT(x,y)** Tests whether specified graphics cell is on or off and returns color code of specified cell.

PPOINT(13,35)

**RIGHT\$(str,length)** Returns right portion of string.

ZIP\$=RIGHT\$(AD\$,5)

**SGN(numeric)** Returns sign of specified numeric expression:

- 1 if argument is negative
- 0 if argument is 0
- +1 if argument is positive

X=SGN(A\*B)

**SIN(numeric)** Returns sine of angle given in radians.

Y=SIN(5)

# C and EXTE

**STRING\$ (length,code or string)** Returns a string of characters (of specified length) specified by ASCII code or by the first character of the string.  
 ? STRING\$(5,"%")  
 ? STRING\$(5,91)

**STR\$ (numeric)** Converts a numeric expression to a string.  
 S\$=STR\$(X)

**SQR (numeric)** Returns the square root of a number.  
 Y=SQR(5+3)

**TAN (numeric)** Returns tangent of angle given in radians.  
 Y=TAN(45.7)





**TIMER** Returns contents or allows setting of timer (0-65535).  
 ? TIMER  
 TIMER=0

**USRn (numeric)** Calls user's machine-language subroutine.  
 X=USR(Y)

**VAL (str)** Converts a string to a number.  
 A=VAL(B\$)

**VARPTR (var)** Returns address of pointer to the specified variable.  
 Y=USR(VARPTR(X) )

## Control Keys

- |  |   |
|--|---|
|                 | Cancels last character typed; moves cursor back one space.                            |
| <b>SHIFT</b>  | Erases current line.  |
| <b>BREAK</b>   | Interrupts anything in progress and returns to command level.                         |
| <b>CLEAR</b>   | Clears the screen.  |
| <b>ENTER</b>   | Signifies end of current line.  |
| <b>SPACEBAR</b>  | Enters a space (blank) character and moves cursor one space forward.                  |
| <b>SHIFT</b>  | Causes currently executing program to pause (press any key to continue).              |
| <b>SHIFT</b>  | All-caps/upper-lowercase keyboard switch. (Lowercase displayed as reversed capitals.) |

ENDED COLO

## Special Characters

- ' Abbreviation for REM.
- \$ Makes variable string type.
- : Separates statements on the same line.
- ? Same as PRINT.
- , PRINT punctuation: spaces over to the next 16-column PRINT zone.
- ; PRINT punctuation: separates items in a PRINT list but does not add spaces when they are output.

## Error Messages

Abbreviation	Explanation
/0	Division by 0
AD	File already OPEN
BS	Subscript out of range
CN	Can't continue
DD	Redimensioned array
DN	Device number error
DS	Direct statement in file
FC	Illegal function call
FD	Bad file data
FM	Bad file mode
ID	Illegal direct
IE	Input past end of file
I/O	Input/Output error
LS	String too long
NF	NEXT without FOR
NO	File not open
OD	Out of data
OM	Out of memory
OS	Out of string space
OV	Overflow
RG	RETURN without GOSUB
SN	Syntax error
ST	String formula too complex
TM	Type mismatch
UL	Undefined line

# FOR BASIC



# Statements

**AUDIO** Connects or disconnects cassette output to TV speaker.

AUDIO ON

AUDIO OFF

**CIRCLE (x,y),r,c,hw,start,end** Draws a circle with center at point (x,y) radius r, specified color c, height/width ratio (hw) of 0-4. Circle can start and end at specified point (0-1).

CIRCLE (128,96),50,4,1,.5,.75

**CLEAR n,h** Reserves n bytes of string storage space. Erases variables. h specifies highest BASIC address.

CLEAR

CLEAR 500

CLEAR 100,14000

**CLOAD** Loads specified program file from cassette. If file name is not specified, first file encountered is loaded. File name must be eight character/spaces or fewer.

CLOAD

CLOAD "PROGRAM"

**CLOADM** Loads machine-language program from cassette. An offset address to add the loading address may be specified.

CLOADM "PROG"

CLOADM

CLOADM "PROG",1000

**CLOSE d** Closes open files.

CLOSE

CLOSE -2

**CLS c** Clears display to specified color c. If color is not specified, green is used.

0-Black

1-Green

2-Yellow

3-Blue

4-Red

CLS

CLS 3

5-Buff

6-Cyan

7-Magenta

8-Orange

**COLOR (foreground, background)** Sets foreground and background color.

COLOR 1,3

**CONT** Continues program execution after pressing **BREAK** or using STOP statement.

CONT

**CSAVE** Saves program on cassette (program name must be eight character/spaces or less). If A is specified, program saved is ASCII format.

CSAVE "PROGRAM"

CSAVE "PROGRAM",A

# TRS-80<sup>®</sup> COMMANDS

**CSAVEM** *name, start, end, transfer*      Writes out a machine-language file.  
CSAVEM "X", 4E, 6F, 5F

**DATA**      Stores data in your program. Use READ to assign this data to variables.  
DATA 5, 3, PEARS  
DATA PAPER, PEN

**DEF FN**      Defines numeric function  
DEF FN(X) = X \* 3

**DEFUSR n**      Defines entry point for USR function *n*. *n* = 0-9.  
DEFUSR5 = 45643

**DEL**      Deletes program lines.  
DEL -  
DEL 25  
DEL 25 -  
DEL -25  
DEL 10-25

**DIM**      Dimensions one or more arrays.  
DIM R(65), W(40)  
DIM A\$(8, 25)

**DLOAD**      Loads BASIC program at specified baud.  
baud.  
0 = 300 baud      1 = 1200 baud  
DLOAD "X", 1

**DRAW**      Draws a line beginning at specified starting point of specified length of specified color. Will also draw to scale, draw blank lines, draw non-updated lines, and execute substrings. If starting point is not specified, last DRAW position or (128,96) is used.  
DRAW "BM100, 100;S10;U25;BR25;ND25;XA#;"

**EDIT**      Allows editing of program line.  
*n*C      Changes *n* number of characters.  
*n*D      Deletes *n* number of characters.  
I      Allows insertion of new characters.  
H      Deletes rest of line and allows insert.  
L      Lists current line and continues edit.  
*n*Sc      Searches for *n*th occurrence of character *c*.  
X      Extends line.  
(SHIFT) (↑)      Escape from subcommand.  
*n* (SPACEBAR)      Moves cursor *n* spaces to right.  
*n* (←)      Moves cursor *n* spaces to left.  
EDIT 25 (ENTER)

**END**      Ends program.  
END

**EXEC (address)**      Transfers control to machine-language programs at specified address. If *address* is omitted, control is transferred to address set in last CLOADM.  
EXEC  
EXEC 32453



# LOR BASIC

**FOR..TO** Creates a loop in program which the Computer must repeat from the first number to the last number you specify.

**STEP/** Use STEP to specify how much to increment the number each time through the loop. If you omit STEP, one is used.

FOR X=2 TO 5 : NEXT X

FOR A=1 TO 10 STEP 5 : NEXT A

FOR M=30 TO 10 STEP -5 : NEXT M

**GET (start)-(end), destination, G** Reads the graphic contents of a rectangle into an array for future use by PUT.

GET (5,20)-(3,8),V,G

**GOSUB** Calls a subroutine beginning at specified line number.

GOSUB 500

**GOTO** Jumps to specified line number.

GOTO 10

**IF TEST ELSE action 1, action 2** Performs a test.

If it is true, computer executes action 1. If false, action 2 is

executed.

IF A=5 THEN

**INPUT** Causes Computer to stop and await input from the keyboard.

INPUT X\$

NAME N\$

**INPUT#-1** Inputs data from cassette.

INPUT#-1,A

**INSTR (position, search, target)** Searches for the first occurrence of *target* string in *search* string beginning at *position*. Returns the position at which the match is found.

? INSTR (5,X\$,Y\$)

**LET** Assigns value to variable (optional).

LET A\$ = "JOB A"

**LIST** Lists specified line(s) or entire program on screen.

LIST 50-85

LIST 30

LIST -30

LIST 30-

**LLIST** List specified program line(s) or entire program to printer.

LLIST

LLIST 50-85

LLIST 30

LIST -30

LIST 30-

**LINE (x1,y1)-(x2,y2), PSET or PRESET, BF** Draw a line from (x1,y1) to (x2,y2). If (x1,y1) is omitted, the last end point or (128,96) is used. PSET selects foreground color and PRESET selects background color. ,B draws a box with (x1,y1) and (x2,y2) as the opposing corners. ,BF will fill in the box with foreground color.

LINE (5,3)-(6,6),PSET

# and EXTEN

**LINE INPUT** Input line from keyboard.

LINE INPUT ANSWER" ;X\$

**MID\$ (oldstr, position, length)** Replaces a portion of one string with another string.

MID\$ (A\$, 14, 2) = "KS"

**MOTOR** Turns cassette ON or OFF.

MOTOR ON

MOTOR OFF

**NEW** Erases everything in memory.

NEW

**ON...GOSUB** Multi-way branch to call specified subroutines.

ON Y GOSUB 50, 100

**ON...GOTO** Multi-way branch to specified line

ON X GOTO 190, 200

**OPEN m, #d, f** Opens file (f) at: Screen (0);

Cassette (-1); Printer (-2). For input (1), output (2).

OPEN "O", -1, "DATA"

**PAINT (x,y), c, b** Paints graphic screen starting at point (x,y) with specified color c and stopping at border (b), of specified color.

PAINT (10, 10), 2, 4

**PCLEAR n** Reserves n number of 1.5 K graphics memory pages.

PCLEAR 8

**PCLS c** Clears screen with specified color c. If color code is omitted, current background color is used. (See CLS for color codes.)

PCLS 3

**PCOPY** Copy graphics from source page to destination page.

PCOPY 5 TO 6

**PLAY** Plays music of specified note (A-G or 1-12), octave (O), volume (V), note-length (L), tempo (T), pause (P), and any number of repetitions. Also sharps (# or +) and flats (-).

PLAY "L1 ; A# ; P8 ; V10 ; T3 ; L2 ; B- ; 9 ; XA# ;"

**PMODE mode, start-page** Selects resolution and first memory page.

PMODE 4, 1

**POKE (location, value)** Puts value (0-255) into specified memory location.

POKE 15872, 255

**PRESET** Reset a point to background color.

PRESET (5, 6)

**PRINT** Prints specified message or number on TV screen.

PRINT "HI"

# DED COLOR

**PRINT#-1** Writes data to cassette.

PRINT A\$

? A\$

PRINT#-1,A

**PRINT#-2** Prints an item or list of items on the printer.

PRINT#-2,CAP\$

**PRINT TAB** Moves the cursor to specified column position.

PRINT TAB(5)"NAME"

**PRINT USING** Prints numbers in specified format.

# Formats numbers.

PRINT USING "####";62.2

, Decimal point.

PRINT USING "##,##";58.6

, Displays comma to left of every third character.

PRINT USING "####,.";44.0

++ Fills leading spaces with asterisks.

PRINT USING "####,##";33.3

\$ Places \$ ahead of number.

PRINT USING "\$###,##";33.3

\$\$ Floating dollar sign.

PRINT USING "\$\$###,##";11.544

\*\*\$ Floating dollar sign.

PRINT USING "\*\*\$###,##";8.333

+ In first position, causes sign to be printed. In last position, causes sign to be printed after the number.

PRINT USING "+###,##";-216

↑↑↑↑

Exponential format.

PRINT USING "##,##↑↑↑↑";546

- Minus sign after negative numbers.

PRINT USING "##,##-";-534.7

! Returns first string character.

PRINT USING "!";"YELLOW"

%spaces% String field; length of field is number of spaces plus 2.

PRINT USING "% %";"BLUE"

**PRINT @ location** Prints specified message at specified text screen location.

PRINT @ 256,"HI"

PRINT @ 256,A\$

**PSET (x,y,c)** Sets a specified point (x,y) to specified color c. If c is omitted, foreground is used.

PSET(5,6,3)

**PUT (start)-(end), source, action** Stores graphics from source onto start/end rectangle on the screen. (Array rectangle size must match GET rectangle size.)

PUT(3,2)-(5,6),V,PSET

**READ** Reads the next item in DATA line and assigns it to specified variable.

READ A\$

READ C,B

# R BASIC

**REM** Allows insertion of comment in program line. Everything after REM is ignored by Computer.

REM THIS IS IGNORED  
10 ? X\$ : REM IGNORE

**RENUM *newline, startline, increment*** Allows program line renumbering.

RENUM 1000,5,100

**RESET (x,y)** Resets a point.

RESET (14,15)

**RESTORE** Sets the Computer's pointer back to first item on the first DATA line.

RESTORE

**RETURN** Returns the Computer from subroutine to the BASIC word following GOSUB.

RETURN

**RUN** Executes a program.

RUN

**SCREEN *screen-type, color-set*** Selects either graphics (1) or text (0) screen and color-set (0 or 1).

SCREEN 1,1

**SET (x,y,c)** Sets a dot at specified text screen location to specified color.

SET (14,13,3)

**SKIPF** Skips to next program on cassette tape, or to end of specified program.

SKIPF "PROGRAM"

**SOUND *tone, duration*** Sounds specified tone for specified duration.

SOUND 128,3

**STOP** Stops execution of a program.

STOP

**TROFF** Turns off program tracer.

TROFF

**TRON** Turns on program tracer.

TRON

**Radio  
Shack®**

Printed in the U.S.A.  
8759244