ATARI

ATARI® BASIC

QUICK REFERENCE GUIDE

by Gilbert Held

175 YEARS OF

WILEY QUICK REFERENCE GUIDES

REFERENCE GUIDE NOTATIONS AND FORMAT CONVENTIONS

A standard scheme for presenting the general format of BASIC language statements is employed in this reference guide. The capitalization, punctuation and other conventions are listed below:

[ ] Brackets indicate that the enclosed items are optional. Brackets do not appear in the actual statements.

{ } Braces indicate that a choice of one of the enclosed items is to be made. Braces do not appear in the actual statements.

. . . Ellipses indicate that the preceding item may be repeated. Ellipses do not appear in the actual statements.

Italics Italics indicate generic terms. The programmer must supply the actual value or wording required. See Generic Terms and Abbreviations.

Line number A line number is implied for all BASIC language statements.

Punctuation All punctuation characters, including commas, semicolons, colons, quotation marks and parentheses, must appear as indicated.

UPPER CASE Upper case letters and words must appear exactly as indicated.

GENERIC TERMS AND ABBREVIATIONS

cmdno—The number that stands for the particular command to be performed.

constant—Actual value BASIC uses during program execution. May be string or numeric.

constant$—String constant only.

device—Numeric value associated with logical device assigned during an open command.

dummy—Dummy argument or zero.

error number—numerical code associated with an error condition.

exp$_1$ exp$_2$—Expressions that are evaluated.

expr—Numeric or string expression.

expr$—Any valid string constant, variable or expression.

exprnm—Any numeric constant, variable or expression.

exprnm%—Any integer numeric constant, variable or expression.

filenum—The file number assigned to a file by the OPEN statement.

filespec—File specification of a program or data files to be loaded. See File Naming Convention.

format—Format applicable to all BASIC versions.

format$_D$—Format applicable to Disk Operating System (DOS).

line—A BASIC statement line number.

line$_i$—The ***i***th line of a BASIC program.

memadr—Memory address.

sub—Subscript.

x coord—x coordinate screen point.

var—Numeric or string variable.

varnm—Numeric variable.

var$—String variable.

y coord—y coordinate screen point.

BASIC OPERATORS

OPERATION	OPERATOR	SAMPLE
ARITHMETIC		
Exponentiation	^	A ^ B
Negation	–	–A
Multiplication	*	A * B
Division	/	A / B
Addition	+	A + B
Subtraction	–	A – B
RELATIONAL		
Equal	=	A = B
Not Equal	< >	A < > B
Less than	<	A < B
Greater than	>	A > B
Less than or equal	< =	A < = B
Greater than or equal	> =	A > = B

BASIC OPERATORS (cont.)

OPERATION	OPERATOR	SAMPLE
LOGICAL		
Logical complement	NOT	A NOT B
Conjunction	AND	A AND B
Disjunction	OR	A OR B

FILE NAMING CONVENTION

filespec—File specification is a string expression.

format: device: filename

where: device is a device name from the following list:

K Keyboard. Input only.
P Line Printer. Output only.
C Cassette. Input/Output only.
D Disk, if only one. Input/Output.
D***n*** Disk 1–4, if multiple. Input/Output.
E Screen editor. Input/Output.
S TV monitor. Input/Output.
R RS232 interface, modem or plotter. Input/Output.

NOTE: Colon (:) must be included between device and file name.

filename—is composed of two parts: *name.extension*. The name portion may be up to 8 characters in length, consisting of characters or letters. It must begin with an alphabetic character. The extension is optional. May be up to 3 characters; for example:

BAS = Basic Program
DAT = Data File
BIN = Binary File
LOD = Load File
007 = Seventh Version

SYSTEM COMMANDS

BYE (B.)—Exits from BASIC and returns to the resident operating system (DOS) or console processor (MEMO PAD).

format: BYE

CLOAD—Loads a tokenized program from cassette into RAM.

format: CLOAD

CONT—Resumes program execution after it has been stopped by pressing the break key or by encountering a STOP or END command.

format: CONT

CSAVE (CS.)—Outputs program in tokenized form from RAM to cassette.

format: CSAVE

DOS—Displays DOS menu.

format$_D$: DOS

DOS—Displays DOS menu.

format$_D$: DOS

ENTER (E.)—Inputs data or program in pure (untokenized) form.

format: ENTER *filespec*

LIST (L.)—Lists the program currently in memory on the monitor screen or on the device specified by *filespec*.

format: LIST [*filespec*] [,(from)*line*] [,(to)*line*]

LOAD (LO.)—Loads a program in tokenized form from the device specified into memory.

format: LOAD *filespec*

NEW—Deletes old program and allows user to enter new program.

format: NEW

RUN—Begins execution of a program currently in memory or loads and executes a specified file. Initializes variables and undimensions arrays and strings.

format: RUN [*filespec*]

SAVE (S.)—Saves a BASIC program in tokenized form on the device specified.

format: SAVE *filespec*

BASIC LANGUAGE STATEMENTS

AUDIO STATEMENTS

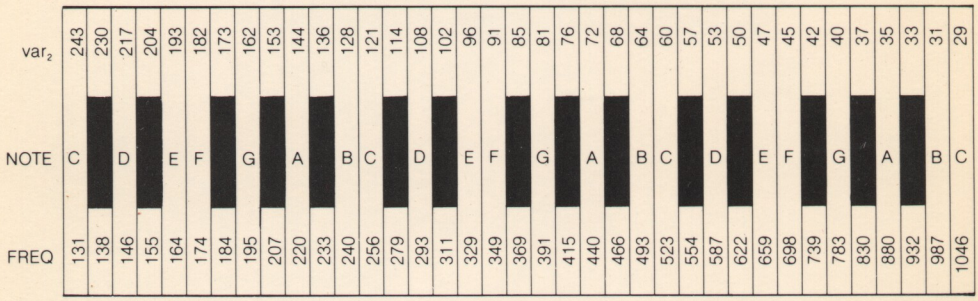
SOUND (SO.)—Plays a tone through the TV speaker until the next SOUND statement is encountered. Parameters control voice registers for pitch, distortion and volume.

format: SOUND var_1 var_2 var_3 var_4

SOUND PARAMETERS

PARAMETER	CONTROL	RANGE	REMARKS
var_1	VOICE REGISTER	Ø–3	Each voice requires separate sound statement.
var_2	SOUND PITCH	Ø–255	The larger the number, the lower the pitch. See chart.
var_3	DISTORTION	Ø–14	Varies:10 = pure tone, 12 = buzzer.
var_4	VOLUME CONTROL	1–15	The larger the number, the greater the volume.

PITCH VALUES (var_2) FOR THE MUSICAL SCALE



BRANCHING

GOSUB (GOS.)—Causes a branch to the subroutine starting at the specified *line*.

format: GOSUB *line*

GOTO (G.)—Causes an unconditional branch to a specified *line*.

format: GOTO *line*

IF - THEN—Used to cause a conditional branch or to execute other statements on the same line (only if first expression is true).

format:

IF *expr* { THEN EXPR [:*expr*]. . . / THEN *line* / GOTO line }

ON-GOSUB—Causes a conditional subroutine call depending upon the current or computed value of the expressions.

format: ON *exprnm* GOSUB *line* [, *line*]. . .

ON-GOTO—Causes a conditional branch based upon the current or computed value of the expression.

format: ON *exprnm GOTO line* [, *line*]. . .

POP—Removes the loop variable from the GOSUB or FOR/NEXT stack. Use when exiting a loop in other than normal RETURN or NEXT manner.

format: POP

RETURN (RET.)—Causes a branch from a subroutine to the statement immediately following the calling GOSUB statement.

format: RETURN

ERROR DETERMINATION AND CONTROL

TRAP (T.)—Takes control of a program in the event of an INPUT error and directs execution to a specific *line*. Without a TRAP the program halts and displays an error code on the screen.

format: TRAP *line*
to reset: TRAP 40000
to examine:
error number = PEEK (195)
line = 256 * PEEK (187) + PEEK (186)

PROCESSING STATEMENTS

CLR—The opposite of DIM. Un-dimensions all strings and matrices.

format: CLR

COM—Same as DIM.

DATA (D.)—Creates a list of numeric and string constants to be assigned to variables through the use of a READ statement.

format: DATA *const* [, *const*]. . .

DIM—Reserves space in memory for arrays, matrices and strings, all of which must be dimensioned prior to use.

format: DIM *var*(*sub*) [, *var*(*sub*)]. . .

END—Stops program execution, closes files and turns off sound(s). Program can be restarted with CONT. May be used several times in program.

format: END

FOR (F.)—Initiates a loop that repeats execution of all instructions until the NEXT statement is reached. Exits when *varnm* equals $exprnm_2$.

format: FOR *varnm* = $exprnm_1$, TO $exprnm_2$ [STEP $exprnm_3$]

LET—Assigns a value to the specified variable. LET may be omitted without penalty.

format: [LET] *var* = *exprnm*

NEXT (N.)—Terminates a loop initiated by a FOR statement. All loops execute at least once.

format: NEXT [*varnm*] [,*varnm*] . . .

PADDLE—Returns the position of paddle game controller (*n*). Return value ranges from 1 to 228, increasing as knob rotates counterclockwise.

example: IF PADDLE (*varnm*) =

constant THEN { GOTO / *exprnm* }

PTRIG—Returns the status of the trigger button on game controller (*n*). Return value equals zero when pressed.

example: IF PTRIG (*varnm*) =

constant THEN { GOTO / *exprnm* }

READ—Reads the next item in the DATA list and assigns it to the specified variable.

format: READ $varnm_1$ [,$varnm_2$] . . .

REM (R.)—Permits remarks to be inserted in a program. Not executed.
or **(.sp)**

format: REM *remark*

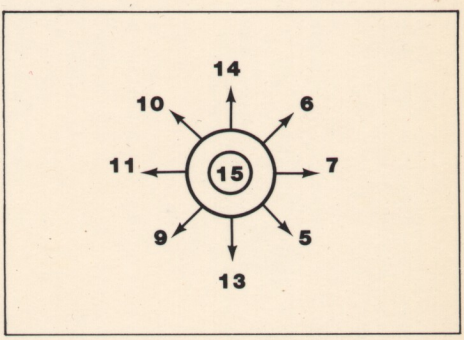
RESTORE (RES.)—Permits DATA statements to be reread from a specific line.

format: RESTORE [*line*]

STICK—Returns the position (see diagram) of a joy stick controller (*n*).

example: IF STICK (*varnm*) =

constant THEN { GOTO / *exprnm* }



STICK RETURN VALUES

STRIG—Returns 0 if stick trigger pressed; 1 if not pressed.

example: IF STRIG (*varnm*) =

constant THEN { GOTO / *exprnm* }

STOP—Stops program execution but does not close files or turn off sound.

format: STOP

INPUT/OUTPUT CONTROL

CLOSE (CL.)—Closes files that were previously opened with a specified file number.

format: CLOSE *#filenum*

GET—Inputs a single byte of data from a specified device.

format: GET *#filenum*, *exprnm*

INPUT (I.)—Requests input from the keyboard during program execution. Execution continues after RETURN key is pressed.

format: INPUT { *expr$* / *exprnm* } [,] . . .

LPRINT (LP.)—Prints data on the line printer. Requires no open or close statements.

format: LPRINT { *expr$* / *exprnm* }

[{ ; / , } *expr* . . .]

NOTE (NO.)—Used in conjunction with POINT when writing DOS files. First variable specifies sector number and second specifies byte within sector.

format$_D$: NOTE *device*, *varnm*, *varnm*

OPEN (O.)—Opens a specified file for input or output operation.

format: OPEN *filespec*

POINT (P.)—Used when reading DOS files into RAM. First variable specifies sector number and second specifies byte within sector to begin reading. Provides pseudo random access.

format$_D$: POINT device, *varnm*, *varnm*

PRINT (PR.)—Displays output on the TV monitor or any other specified output device.
or **(?)**

format: PRINT [*#filenum*]

{ *expr$* / *exprnm* } [; / ,] . . .

PUT—Outputs a single byte of data to a specified device.

format: PUT *#filenum*, *exprnm*

STATUS (ST.)—Calls the status routine for a specified device. The status (error message) is stored in the specified variable.

format: STATUS *#filenum*, *varnm*

XIO (X.)—General purpose I/O statement that executes any input/output operation depending on the value of *cmdno*.

format: XIO *cmdno*, *#filenum*, $exprnm_1$, $exprnm_2$, *filespec*

XIO COMMAND TABLE

cmdno	OPERATION	cmdno	OPERATION
3	Open	18	Fill
5	Get Record	32	Rename File
7	Get Characters	33	Delete File
9	Put Record	35	Lock File
11	Put Characters	36	Unlock File
12	Close File	37	Point
13	Status Request	38	Note
17	Draw Line	254	Format

exprnm$_1$ and *exprnm*$_2$ depend on device. Generally dummy.

GRAPHICS AND VIDEO CONTROL

COLOR (C.)—Used in text mode Ø–2 in conjunction with PLOT to generate a character on the screen.

format: COLOR *exprnm*

COLOR (C.)—Used in graphics mode 3–8 in conjunction with SETCOLOR to determine the color of each PIXEL.

format: COLOR *exprnm*

COLUMNS

SCREEN FORMATS

GRAPHICS MODE	MODE TYPE	PIXEL SIZE	COLUMNS	ROWS (SPLIT)	ROWS (FULL)*	NUMBER COLORS	BYTES (SPLIT) REQUIRED	BYTES (FULL) REQUIRED
0	Text	1W; 1H	40	—	24	1½	992	—
1	Text	2W, 1H	20	20	24	5	674	672
2	Text	2W, 2H	20	10	12	5	424	420
3	Graphics	1W, 1H	40	20	24	4	434	432
4	Graphics	1/2W, 1/2H	80	40	48	2	694	696
5	Graphics	1/2W, 1/2H	80	40	48	4	1174	1176
6	Graphics	1/2W, 1/2H	160	80	96	2	2174	2184
7	Graphics	1/4W, 1/4H	160	80	96	4	4190	4200
8	Graphics	1/8W, 1/8H	320	160	192	1½	8112	8138
9**	Graphics	1/2W, 1/8H	80	—	192	1	8112	8138
10**	Graphics	1/2W, 1/8H	80	—	192	9	8112	8138
11**	Graphics	1/2W, 1/8H	80	—	192	16	8112	8138

*Add + 16 to mode number for full screen (no text window).

**Requires GTIA chip.

DRAWTO (DR.)—Draws a straight line between a PLOTed point and a specified point.

format: DRAWTO *xcoord*, *ycoord*

GRAPHICS (GR.)—Specifies which of the twelve graphics modes is to be used. GR.O clears screen.

format: GRAPHICS { *constant* / *varnm* }

LOCATE (LOC.)—Positions the invisible graphics cursor to specified location in graphics window and retrieves data stored there.

format: LOCATE *xcoord*, *ycoord*, *var*

PLOT (PL.)—Causes a single point to be plotted at the specified location.

format: PLOT *xcoord*, *ycoord*

POSITION (POS.)—Sets the cursor to the specified screen position.

format: POSITION *xcoord*, *ycoord*

SETCOLOR (SE.)—Selects a particular color hue and luminance and places it in the specified color register.

format: SETCOLR *exprnm*$_1$, *exprnm*$_2$, *exprnm*$_3$

where:

exprnm$_1$ = color register 0–4, depending on mode.
exprnm$_2$ = color hue 0–15 (see HUE table).
exprnm$_3$ = color luminance 0–14, the higher the brighter.

 ISBN 0 471-87041-2

HUE TABLE

exprnm$_2$	COLOR HUE	*exprnm*$_2$	COLOR HUE
0	Gray	8	Blue
1	Gold	9	Light Blue
2	Orange	10	Turquoise
3	Red-Orange	11	Green-Blue
4	Pink	12	Green
5	Purple	13	Yellow-Green
6	Purple-Blue	14	Orange-Green
7	Blue	15	Light Orange

GRAPHICS MODE AND COLOR COMBINATIONS

TEXT MODE	COLOR VARIABLE	SETCOLOR REGISTER	DEFAULT HUE	DEFAULT LUMINANCE	DERIVED COLOR
0&	Actual	1	9	8	Light Blue
Text	Text	2	9	4	Dark Blue
Window	Character	4	0	0	Black
1&2		0	2	8	Orange
		1	12	10	Light Green
		2	9	4	Dark Blue
		3	4	6	Red
		4	0	0	Black
GRAPHIC MODE	**COLOR VARIABLE**	**SETCOLOR REGISTER**	**DEFAULT HUE**	**DEFAULT LUMINANCE**	**DERIVED COLOR**
3,5,7	1	0	2	8	Orange
	2	1	12	10	Light Green
	3	2	9	4	Dark Blue
	0	4	0	0	Black
4&6	1	0	2	8	Orange
	0	4	0	0	Black
8	1	1	9	8	Light Blue
	0	2	9	4	Dark Blue
	0	4	0	0	Black

XIO (X.)—A special application of the general XIO statement used to "paint" colors between plotted points and lines. Must first POKE 765 with color variable number.
format: XIO 18, #6,12, *dummy*, "S:"

BASIC FUNCTIONS

FUNCTION FORMAT AND DESCRIPTION

ABS (*exprnm*)—Returns the absolute value of the expression.

ADR (*var$*)—Returns memory address of a string.

ASC (*expr$*)—Returns numeric value of a single string character.

ATN (*exprnm*)—Returns the arctangent in radians or degrees.

CHR$ (*exprnm*)—Converts an ASCII code to its character equivalent.

CLOG (*exprnm*)—Returns the common (base 10) logarithm of an expression.

COS (*exprnm*)—Returns the cosine of an angle expressed in radians or degrees.

DEG—Tells computer to perform trigonometric functions in degrees instead of radians (default radians).

EXP (*exprnm*)—Returns the base of the natural logarithm (e) raised to the specific power.

FRE (*dummy*)—Returns the number of bytes in memory not used by BASIC.

INT (*exprnm*)—Returns the largest integer which is less than or equal to the *exprnm*.

LEN (*expr$*)—Returns the length of the specified string in bytes.

LOG (*exprnm*)—Returns the natural logarithm of the specified number.

PEEK (*memadr*)—Returns the decimal value of the contents of the specified memory location.

POKE *memadr, exprnm*—Writes the value of the specified expression into the indicated memory address.

RAD—Tells the computer to perform trigonometric functions in radians instead of degrees (default).

RND (*dummy*)—Returns a random number between 0 and 1.

SGN (*exprnm*)—Returns +1 if *exprnm* is positive, 0 if zero, −1 if negative.

SIN (*exprnm*)—Returns the sine of an angle expressed in radians or degrees.

SQR (*exprnm*)—Returns the square root of *exprnm*.

STR$ (*exprnm*)—Returns the string value of the numeric expression.

USR (*exprnm* [, *exprnm*] . . .)—Executes a machine language subroutine.

VAL (*exprnm$*)—Returns the equivalent numeric value of a string.

KEY COMBINATIONS

ESC CTRL CLEAR—Clears the screen during the execution of a program with PRINT.

SHIFT SET-CLR TAB—Sets a tab. Default 7, 15, 23, 31, 39.

CTRL SET-CLR TAB—Clears a tab.

SHIFT INSERT—Inserts a line.

CTRL INSERT—Inserts a character.

SHIFT DELETE—Deletes a line.

CTRL DELETE—Deletes a character.

CTRL CLEAR—Clears screen and homes cursor.

SHIFT CAPS/LOWR—Restores characters to upper case. Lower case set without shift.

CTRL CURSOR—Moves cursor one physical line.

CTRL-1—Temporarily stops and restarts display without breaking program.

CTRL-3—Use with screen editor and disk to dump file.

BASIC ERROR MESSAGES

NUMBER	MESSAGE	NUMBER	MESSAGE
2	Memory Insufficient	134	Bad IOCB Number
3	Value Error	135	IOCB Read Only Device
4	Too Many Variables	136	EOF not expected
5	String Length Error	137	Truncated Record
6	Out of Data	138	Device Timeout
7	Number greater than 32767	139	Device NAK
8	Input Statement Error	140	Serial Bus Frame Error
9	Array or String Dim Error	141	Cursor Out of Range
10	Argument Stack Overflow	142	Serial Bus Overrun
11	Floating Point Under/Overflow	143	Serial Bus Checksum
12	Line Not Found	144	Device Error
13	No Matching FOR Statement	145	Read/Write Compare
14	Line Too Long	146	Invalid Function
15	GOSUB or FOR Line Deleted	147	Insufficient RAM
16	Return Error	160	Drive Number Error
17	Syntax Error	161	Too Many OPEN Files
18	Invalid String Error	162	Disk Full
19	Load Program Too Large	163	Unrecoverable I/O Error
20	Device No. Larger Than 7	164	File Number Mismatch
21	Load File Error	165	File Name Error
128	Break Abort	166	Point Data Length Error
129	IOCB Already Open	167	File Locked
130	Nonexistent Device	168	Command Invalid
131	IOCB Write Only Device	169	Directory Full
132	Invalid Command	170	File Not Found
133	Device or File Not Open	171	Point Invalid