1

HANDY REFERENCE CARD
valFORTH 1.1
T.M

Stack inputs and outputs are shown; top of stack on right.
This card follows usage of the Forth Interest Group
(S.F. Bay Area); usage aligned with the Forth 78
International Standard.
For more info: Forth Interest Group
P.O. Box 1105
San Carlos, CA 94070.

Operand Key:	n,n1,...	16-bit signed numbers
	d,d1,...	32-bit signed numbers
	u	16-bit unsigned number
	addr	address
	b	8-bit byte
	c	7-bit ascii character value
	f	boolean flag
	fp	floating point number
	$	string

Stack Manipulation

DUP	(n -- n n)	Duplicate top of stack.
DROP	(n --)	Throw away top of stack.
SWAP	(n1 n2 -- n2 n1)	Reverse top two stack items.
OVER	(n1 n2 -- n1 n2 n1)	Make copy of second item on top.
ROT	(n1 n2 n3 -- n2 n3 n1)	Rotate third item to top.
<ROT	(n1 n2 n3 -- n3 n1 n2)	Rotate top item to third.
-DUP	(n -- n ?)	Duplicate only if non-zero.
>R	(n --)	Move top item to "return stack" for temporary storage (use caution).
R>	(-- n)	Retrieve item from return stack.
R	(-- n)	Copy top of return stack onto stack.

Number Bases

DECIMAL	(--)	Set decimal base.
HEX	(--)	Set hexadecimal base.
BASE	(-- addr)	System variable containing number base.

Arithmetic and Logical

+	(n1 n2 -- sum)	Add.
D+	(d1 d2 -- sum)	Add double-precision numbers.
-	(n1 n2 -- diff)	Subtract (n1-n2).
*	(n1 n2 -- prod)	Multiply.
/	(n1 n2 -- quot)	Divide (n1/n2).
MOD	(n1 n2 -- rem)	Modulo (i.e. remainder from division).
/MOD	(n1 n2 -- rem quot)	Divide, giving remainder and quotient.
*/MOD	(n1 n2 n3 -- rem quot)	Multiply, then divide (n1*n2/n3), with double-precision intermediate.
*/	(n1 n2 n3 -- quot)	Like */MOD, but give quotient only.
MAX	(n1 n2 -- max)	Maximum.
MIN	(n1 n2 -- min)	Minimum.
ABS	(n -- absolute)	Absolute value.
DABS	(d -- absolute)	Absolute value of double-precision number.
MINUS	(n -- -n)	Change sign.
DMINUS	(d -- -d)	Change sign of double-precision number.
AND	(n1 n2 -- and)	Logical AND (bitwise).
OR	(n1 n2 -- or)	Logical OR (bitwise).
XOR	(n1 n2 -- xor)	Logical exclusive OR (bitwise).
NOT	(n -- f)	True if top number zero (i.e. reverses truth value).

Comparison

<	(n1 n2 -- f)	True if n1 less than n2.
>	(n1 n2 -- f)	True if n1 greater than n2.
<=	(n1 n2 -- f)	True if n1 less than or equal to n2.
>=	(n1 n2 -- f)	True if n1 greater than or equal to n2.
=	(n1 n2 -- f)	True if top two numbers are equal.
<>	(n1 n2 -- f)	True if n1 does not equal n2.
0<	(n -- f)	True if top number negative.
0>	(n -- f)	True if top number positive.
0=	(n -- f)	True if top number zero (i.e. reverses truth value.
0#	(n -- f)	True if n does not equal zero.

Memory

@	(addr -- n)	Replace word address by contents.
!	(n addr --)	Store second word at address on top.
C@	(addr -- b)	Fetch one byte only.
C!	(b addr --)	Store one byte only.
?	(addr --)	Print contents of address.
C?	(addr --)	Print byte at address.
U?	(addr --)	Print unsigned contents of address.
+!	(n addr --)	Add second number on stack to contents of address on top.
CMOVE	(from to u --)	Move u bytes in memory from head to head.
<CMOVE	(from to u --)	Move u bytes in memory from tail to tail.
FILL	(addr u b --)	Fill u bytes in memory with b, beginning at address.
ERASE	(addr u --)	Fill u bytes in memory with zeroes, beginning at address.
BLANKS	(addr u --)	Fill u bytes in memory with blanks, beginning at address.

Control Structures

DO...LOOP	do: (end+1 start --)	Set up loop, given index range.
I	(-- index)	Place current index value on stack.
I'	(-- index)	Used to retrieve index after a >R.
J	(-- index)	Place index of outer DO-LOOP on stack.
LEAVE	(--)	Terminate loop at next LOOP, +LOOP, or /LOOP.
?EXIT	(--)	LEAVE if ?TERMINAL is true (i.e. pressed).
DO... +LOOP	do: (end+1 start --) +loop: (n --)	Like DO...LOOP, but adds stack value (instead of always '1') to index.
DO... /LOOP	do: (end+1 start --) /loop: (u --)	Like DO... +LOOP, but adds unsigned value to index.
IF...(true) ...ENDIF	if: (f --)	If top of stack true (non-zero), execute. (Note: Forth 78 uses IF...THEN.)
IF...(true) ...ELSE ...(false) ...ENDIF	if: (f --)	Same, but if false, execute ELSE clause. (Note: Forth 78 uses IF...ELSE...THEN.)
BEGIN... UNTIL	until: (f --)	Loop back to BEGIN until true at UNTIL. (Note: Forth 78 uses BEGIN...END.)
BEGIN... WHILE ...REPEAT	while: (f --)	Loop while true at WHILE;REPEAT loops unconditionally to BEGIN. (Note: Forth 78 uses BEGIN...IF ...AGAIN.)

Terminal Input - Output

.	(n --)	Print number.
.R	(n fieldwidth --)	Print number, right-justified in field.
D.	(d --)	Print double-precision number
D.R	(d fieldwidth --)	Print double-precision number, right-justified in field.
CR	(--)	Do a carriage return.
SPACE	(--)	Type one space.
SPACES	(n --)	Type n spaces.
."	(--)	Print message (terminated by ").
DUMP	(addr u --)	Dump u words starting at address.
TYPE	(addr u --)	Type string of u characters starting at address.
COUNT	(addr -- addr+1 u)	Change length-byte string to TYPE form.
?TERMINAL	(-- f)	True if terminal break request present.
KEY	(-- c)	Read key, put ascii value on stack.
EMIT	(c --)	Type ascii value from stack.
EXPECT	(addr n --)	Read n characters (or until carriage return) from input to address.
WORD	(c --)	Read one word from input stream, using given character (usually blank) as delimiter.

Input - Output Formating

NUMBER	(addr -- d)	Convert string at address to double-precision number.
<#	(--)	Start output string.
#	(d -- d)	Convert next digit of double-precision number and add character to output string.
#S	(d -- 0 0)	Convert all significant digits of double-precision number to output string.
SIGN	(n d -- d)	Insert sign of n into output string.
#>	(d -- addr u)	Terminate output string (ready for TYPE).
HOLD	(c --)	Insert ascii character into output string.

Disk Handling

LIST	(screen --)	List a disk screen.
LOAD	(screen --)	Load disk screen (compile or execute).
BLOCK	(block -- addr)	Read disk block to memory address.
B/BUF	(-- n)	System constant giving disk block size in bytes.
BLK	(-- addr)	System variable containing current block number.
SCR	(-- addr)	System variable containing current screen number.
UPDATE	(--)	Mark last buffer accessed as updated.
FLUSH	(--)	Write all updated buffers to disk.
EMPTY-BUFFERS	(--)	Erase all buffers.

Defining Words

: xxx	(--)	Begin colon definition of xxx.
;	(--)	End colon definition.
VARIABLE xxx	(n --) xxx: (-- addr)	Create a variable named xxx with initial value n; returns address when executed.
CONSTANT xxx	(n --) xxx: (-- n)	Create a constant named xxx with value n; returns value when executed.
CODE xxx	(--)	Begin definition of assembly-language primitive operative named xxx.
;CODE	(--)	Used to create a new defining word, with execution-time "code routine" for this data type in assembly.
<BUILDS... DOES>	does: (-- addr)	Used to create a new defining word, with execution-time routine for this data type in higher-level Forth.
LABEL xxx	(-- addr)	Creates a header xxx which when executed returns its PFA.

Software and Documentation
©Copyright 1982
Valpar International

HANDY REFERENCE CARD

valFORTH 1.1

T.M

Vocabularies

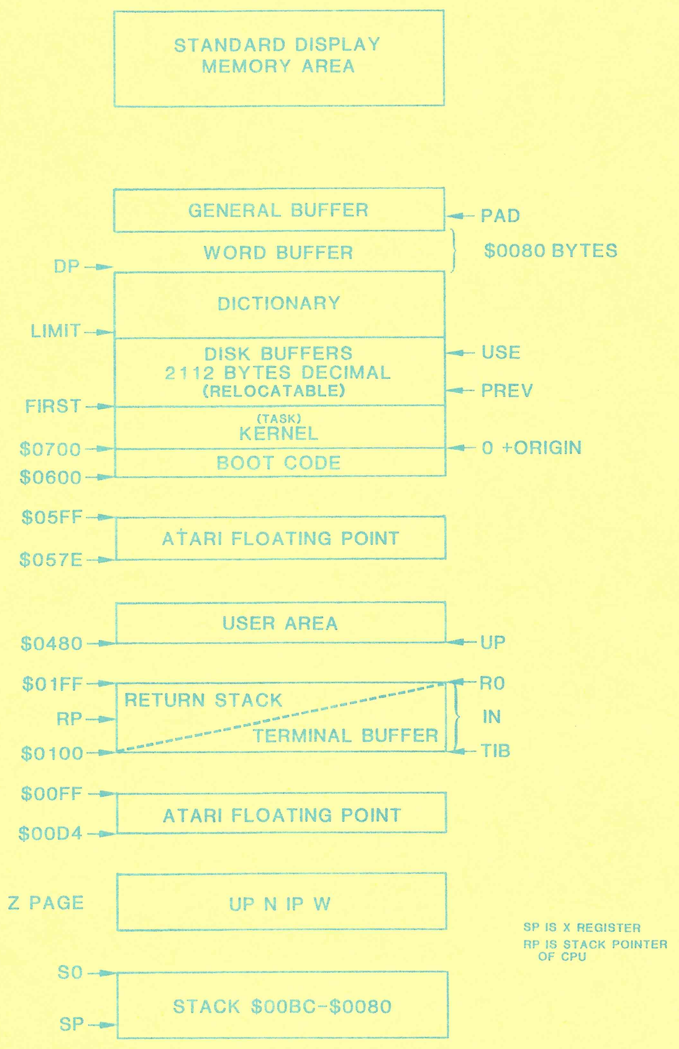
CONTEXT	(-- addr)	Returns address of pointer to context vocabulary (searched first).
CURRENT	(-- addr)	Returns address of pointer to current vocabulary (where new definitions are put).
FORTH	(--)	Main Forth vocabulary (execution of FORTH sets CONTEXT vocabulary).
EDITOR	(--)	Editor vocabulary; sets CONTEXT.
ASSEMBLER	(--)	Assembler vocabulary; sets CONTEXT.
DEFINITIONS	(--)	Sets CURRENT vocabulary to CONTEXT.
VOCABULARY xxx	(--)	Create new vocabulary named xxx.
VLIST	(--)	Print names of all words in CONTEXT vocabulary.

Miscellaneous and System

(	(--)	Begin comment, terminated by right paren on same line; space after (.
FORGET xxx	(--)	Forget all definitions back to and including xxx.
ABORT	(--)	Error termination of operation.
'xxx	(-- addr)	Find the address of xxx in the dictionary; if used in definition, compile address.
HERE	(-- addr)	Returns address of next unused byte in the dictionary.
PAD	(-- addr)	Returns address of scratch area (usually 128 bytes beyond HERE).
IN	(-- addr)	System variable containing offset into input buffer. Used, e.g., by WORD.
SP@	(-- addr)	Returns address of top stack item.
ALLOT	(n --)	Leave a gap of n bytes in the dictionary.
,	(n --)	Compile a number into the dictionary.

valFORTH Memory Map

T.M.



Atari is a trademark of Atari, Inc., a division of Warner Communications.

Software and Documentation
© Copyright 1982
Valpar International

②

HANDY REFERENCE CARD valFORTH 1.1

T.M.

Graphics and Color

SETCOLOR	(n1 n2 n3 --)	Color register n1 (0...3 and 4 for background) is set to hue n2 (0 to 15) and luminance n3 (0-14, even).
SE.	(n1 n2 n3 --)	Alias for SETCOLOR.
GR.	(n --)	Identical to GR. in BASIC. Adding 16 will suppress split display. Adding 32 will suppress display preclear. In addition, this GR. will not disturb player/missiles.
POS.	(x y --)	Same as BASIC POSITION or POS. Positions the invisible cursor if in a split display mode, and the text cursor if in 0 GR. .
POSIT	(x y --)	Positions and updates the cursor, similar to PLOT, but without changing display data.
PLOT	(x y --)	Same as BASIC PLOT. PLOTs point of color in register specified by last COLOR command, at point x y.
DRAWTO	(x y --)	Same as BASIC DRAWTO. Draws line from last PLOT'ted, DRAWTO'ed or POSIT'ed point to x y, using color in register specified by last COLOR command.
DR.	(x y --)	Alias for DRAWTO.
FIL	(b --)	Fills area between last PLOT'ted, DRAWTO'ed or POSIT'ed point to last position set by POS., using the color in register b.
G"	(--)	Used in the form G" ccccc". Sends text cccc to text area in non-0 Graphics mode, starting at current cursor position, in color of register specified by last COLOR command prior to cccc being output.
GTYPE	(addr count --)	Starting at addr, output count characters to text area in non-0 Graphics mode, starting at current cursor position, in color of register specified by last COLOR command.
LOC.	(x y -- b)	Positions the cursor at x y and fetches the data from display at that position. Like BASIC LOCATE and LOC. .
(G")	(--)	Run-time code compiled in by G".
POS@	(-- x y)	Leaves the x and y coordinates of the cursor on the stack.
CPUT	(b --)	Outputs the data b to the current cursor position.
CGET	(-- b)	Fetches the data b from the current cursor position.
>SCD	(c1 -- c2)	Converts c1 from ATASCII to its display screen code, c2. Example: ASCII A >SCD 88 @ C! will put an "A" into the upper left corner of the display.
SCD>	(c1 -- c2)	Converts c1 from display screen code to ATASCII c2. See >SCD.
>BSCD	(addr1 addr2 count --)	Moves count bytes from addr1 to addr2, translating from ATASCII to display screen code on the way.
BSCD>	(addr1 addr2 count --)	Moves count bytes from addr1 to addr2, translating from display screen code to ATASCII on the way.
COLOR	(b --)	Saves the value b in the variable CLRBYT.
CLRBYT	(-- addr)	Variable that holds data from last COLOR command.

GREY	--	0	PINK	--	4	BLUE	--	8	GREEN	--	12
GOLD	--	1	LVNDR	--	5	LTBLUE	--	9	YLWGRN	--	13
ORNG	--	2	BLPRPL	--	6	TURQ	--	10	ORNGRN	--	14
RDORNG	--	3	PRPLBL	--	7	GRNBL	--	11	LTORNG	--	15

(CONSTANTs)

SOUND	(chan freq dist vol --)	Sets up the sound channel "chan" as indicated. Channel: 0-3 Frequence: 0-255, 0 is highest pitch. Distortion: 0-14, evens only. Volume: 0-15. Suggested mnemonic: CatFish Don't Vote
SO.	(chan freq dist vol --)	Alias of SOUND.
FILTER!	(n --)	Stores n in the audio control register and into the valFORTH shadow register, AUDCTL. Use AUDCTL when doing bit manipulation, then do FILTER!.
AUDCTL	(-- addr)	A variable containing the last value sent to the audio control register by FILTER!.
XSND	(n --)	Silences channel n.
XSND4	(--)	Silences all channels.

Text Output and Disk Preparation

S:	(flag --)	If flag is true, enables handler that sends text to text screen. If false, disables the handler. (See PFLAG in main glossary.)
P:	(flag --)	If flag is true, enables handler that sends text to printer. If false, disables the handler. (See PFLAG in main glossary)
BEEP	(--)	Makes a raucous noise from the keyboard.
ASCII	(c, -- n (executing)) (c, -- (compiling))	Converts next character in input stream to ATASCII code. If executing, leaves on stack. If compiling, compiles as literal.
EJECT	(--)	Causes a form feed on smart printers if the printer handler has been enabled by ON P:. May need adjustment for dumb or nonstandard printers.
LISTS	(start count --)	From start, lists count screens. May be aborted by CONSOLE button at the end of a screen.
PLIST	(scr --)	Lists screen scr to the printer, then restores former printer handler status.
PLISTS	(start cnt --)	From start, lists cnt screens to printer three to a page, then restores former printer handler status. May be aborted by CONSOLE button at the end of a screen.
FORMAT	(--)	With prompts, will format a disk in drive of your choice.

Debugging Utilities

DECOMP	xxx	Does a decompilation of the word xxx if it can be found in the active vocabularies.
CDUMP	(addr n --)	A character dump from addr for at least n characters. (Will always do a multiple of 16.)
#DUMP	(addr n --)	A numerical dump in the current base for at least n characters. (Will always do a multiple of 8.)
(FREE)	(-- n)	Leaves number of bytes between bottom of display list and PAD.
FREE	(--)	Does (FREE) and then prints the stack and "bytes".
H.	(n --)	Prints n in HEX, leaves BASE unchanged.
STACK	(flag --)	If flag is true, turns on visible stack. If flag is false, turns off visible stack.
.S	(... -- ...)	Does a signed, nondestructive stack printout, TOS at right. Also sets visible stack to do signed printout.
U.S	(... -- ...)	Does unsigned, nondestructive stack printout, TOS at right. Also sets visible stack to do unsigned printout.
B?	(--)	Prints the current base, in decimal. Leaves BASE undisturbed.
CFALIT	xxx (-- cfa (executing)) xxx (-- (compiling))	Gets the cfa (code field address) of xxx. If executing, leaves it on the stack; if compiling, compiles it as a literal.

Floating Point

FCONSTANT	xxx (fp --) xxx (-- fp)	The character string is assigned the constant value fp. When xxx is executed, fp will be put on the stack.
FVARIABLE	xxx (fp --) xxx: (addr --)	The character string xxx is assigned the initial value fp. When xxx is executed, the addr (two bytes) of the value of xxx will be put on the stack.
FDUP	(fp1 -- fp1 fp1)	Copies the fp number at top-of-stack.
FDROP	(fp --)	Discards the fp number at top-of-stack.)
FOVER	(fp2 fp1 -- fp2 fp1 fp2)	Copies the fp number at 2nd-on-stack to top-of-stack.
FLOATING	xxx (-- fp)	Attempts to convert the following string, xxx, to a fp number.
FP	xxx (-- fp)	Alias for FLOATING.
F@	(addr -- fp)	Fetches the fp number whose address is at top-of-stack.
F!	(fp addr --)	Stores fp into addr. Remember that the operation will take six bytes in memory.
F.	(fp --)	Type out the fp number at top-of-stack. Ignores the current value in BASE and uses base 10.
F?	(addr --)	Fetches a fp number from addr and types it out.
F+	(fp2 fp1 -- fp3)	Replaces the two top-of-stack fp items, fp2 and fp1, with their fp sum, fp3.
F-	(fp2 fp1 -- fp3)	Replaces the two top-of-stack fp items fp2 and fp1, with their difference, fp3=fp2-fp1.
F*	(fp2 fp1 -- fp3)	Replaces the two top-of-stack fp items fp2 and fp1, with their product, fp3.
F/	(fp2 fp1 -- fp3)	Replaces the two top-of-stack fp items fp2 and fp1, with their quotient, fp3=fp2/fp1.
FLOAT	(n -- fp)	Replaces number at top-of-stack with its fp equivalent.
FIX	(fp (non-neg, less than 32767.5) -- n)	Replaces fp number at top-of-stack, constrained as indicated, with its integer equivalent.
LOG	(fp1 -- fp2)	Replaces fp1 with its base e logarithm, fp2. Not defined for fp1 negative.
LOG10	(fp1 -- fp2)	Replaces fp1 with its base 10 decimal logarithm, fp2. Not defined for fp1 negative.
EXP	(fp1 -- fp2)	Replaces fp1 with fp2, which equals e to the power fp1.
EXP10	(fp1 -- fp2)	Replaces fp1 with fp2, which equals 10 to the power fp1.
F0=	(fp -- flag)	If fp is equal to floating-point 0, a true flag is left. Otherwise, a false flag is left.
F=	(fp2 fp1 -- flag)	If fp2 is equal to fp1, a true flag is left. Otherwise, a false flag is left.
F>	(fp2 fp1 -- flag)	If fp2 is greater than fp1, a true flag is left. Otherwise, a false flag is left.
F<	(fp2 fp1 -- flag)	If fp2 is less than fp1, a true flag is left. Otherwise, a false flag is left.
FLITERAL	(fp --)	If compiling, then compile the fp stack value as a fp literal.

Operating System

OPEN	(addr n0 n1 n2 -- n3)	This word opens the device whose name is at addr. The device is opened on channel n0 with AUX1 and AUX2 as n1 and n2 respectively. The device status byte is returned as n3.
CLOSE	(n --)	Closes channel n.
PUT	(b1 n -- b2)	Outputs byte b1 on channel n, returns status byte b2.
GET	(n -- b1 b2)	Gets byte b1 from channel n, returns status byte b2.
GETREC	(addr n1 n2 -- n3)	Inputs record from channel n2 up to length n1. Returns status byte n3.
PUTREC	(addr n1 n2 -- n3)	Outputs n1 characters starting at addr through channel n2. Returns status byte n3.
STATUS	(n -- b)	Returns status byte b from channel n.
DEVSTAT	(n -- b1 b2 b3)	From channel n1 gets device status bytes b1 and b2, and normal status byte b3.
SPECIAL	(b1 b2 b3 b4 b5 b6 b7 b8 -- b9)	Implements the Operating System "Special" command. AUX1 through AUX6 are b1 through b6 respectively, command byte is b7, channel number is b8. Returns status byte b9.
RS232	(--)	Loads the Atari 850 drivers into the dictionary (approx 1.8K).

Software and Documentation
©Copyright 1982
Valpar International

HANDY REFERENCE CARD

valFORTH 1.1 ™

valFORTH 6502 Assembler

ASSEMBLER	(---)	Calls up the assembler vocabulary for subsequent assembly language programming.
CODE xxx	(---)	Enters the new word "xxx" into the dictionary as machine language word and calls up the assembler vocabulary for subsequent assembly language programming.
C;	(---)	Terminates an assembly language definition by performing a security check and setting the CONTEXT vocabulary to the same as the CURRENT vocabulary.
END-CODE	(---)	A commonly used synonym for the word C; above. The word C; is recommended over END-CODE.
SUBROUTINE	xxx (---)	Enters the new word "xxx" into the dictionary as machine language subroutine and calls up the assembler vocabulary for subsequent assembly language programming.
;CODE	(---)	When the assembler is loaded, puts the system into the assembler vocabulary for subsequent assembly language programming. See main glossary for further explanation.

Control Structures

IF,	(flag --- addr 2)	Begins a machine language control structure based on the 6502 status flag on top of the stack. Leaves an address and a security check value for the ELSE, or ENDIF, clauses below. "flag" can be EQ , NE , CC , CS , VC , VS , MI , or PL . Command forms: ...flag..IF,..if-true..ENDIF,...all... ...flag..IF,..if-true.. ELSE,..if-false..ENDIF,..all...
ELSE,	(addr 2 --- addr 3)	Used in an IF, clause to allow for execution of code only if IF, clause is false. If the IF, clause is true, this code is bypassed.
ENDIF,	(addr 2/3 ---)	Used to terminate an IF, control structure clause. Additionally, ENDIF, resolves all forward references. See IF, above for command form.
BEGIN,	(--- addr 1)	Begins machine language control structures of the following forms: ...BEGIN,...AGAIN,... ...BEGIN,...flag..UNTIL,... ...BEGIN,...flag..WHILE,..while-true..REPEAT,... where "flag" is one of the 6502 statuses: EQ , NE , CC , CS , VC , VS , MI , and PL .
UNTIL,	(addr 1 flag ---)	Used to terminate a post-testing BEGIN, clause thus allowing for conditional looping of a program segment while "flag" is false.
WHILE,	(addr 1 flag --- addr 4)	Used to begin a pre-testing BEGIN, clause thus allowing for conditional looping of a program segment while "flag" is true.
REPEAT,	(addr 4 ---)	Used to terminate a pre-testing BEGIN,..WHILE, clause. Additionally, REPEAT, resolves all forward addresses of the current WHILE, clause.
AGAIN,	(addr 1 ---)	Used to terminate an unconditional BEGIN, clause. Execution cannot exit this loop unless a JMP, instruction is used.

Parameter Passing (These routines must be jumped to.)

NEXT	(--- addr)	Transfers control to the next FORTH word to be executed. The parameter stack is left unchanged.
PUSH	(--- addr)	Pushes a 16 bit value to the parameter stack whose low byte is found on the 6502 return stack and whose high byte is found in the accumulator.
PUSHOA	(--- addr)	Pushes a 16 bit value to the parameter stack whose low byte is found in the accumulator and whose high byte is zero.
PUT	(--- addr)	Replaces the value currently on top of the parameter stack with the 16 bit value whose low byte is found on the 6502 stack and whose high byte is in the accumulator.
PUTOA	(--- addr)	Replaces the value currently on top of the parameter stack with the 16 bit value whose low byte is in the accumulator and whose high byte is set to zero.
BINARY	(--- addr)	Drops the top value of the parameter stack and then performs a PUT operation described above.
POP and POPTWO	(--- addr)	POP drops one value from the parameter stack. POPTWO drops two values from the parameter stack.
SETUP	(--- addr)	Moves one to four values to the N scratch area in the zero page and drops all values moved from the parameter stack.
N	(--- addr)	Points to a nine-byte scratch area in the zero page beginning at N-1 and going to N+7.
Opcodes	(various --- various)	ADC, AND, ASL, BIT, BRK, CLC, CLD, CLI, CLV, CMP, CPX, CPY, DEC, DEX, DEY, EOR, INC, INX, INY, JSR, JMP, LDA, LDX, LDY, LSR, NOP, ORA, PHA, PHP, PLA, PLP, ROL, ROR, RTI, RTS, SBC, SEC, SED, SEI, STA, STX, TAX, TAY, TSX, TXA, TXS, TYA,

Aliases

NXT,	=	NEXT JMP,	POP2,	=	POPTWO JMP,
PSH,	=	PUSH JMP,	XL,	=	XSAVE LDX,
PUT,	=	PUT JMP,	XS,	=	XSAVE STX,
PSHA,	=	PUSHOA JMP,	THEN,	=	ENDIF,
PUTA,	=	PUTOA JMP,	END,	=	UNTIL,
POP,	=	POP JMP,			

Software and Documentation
©Copyright 1982
Valpar International

HANDY REFERENCE CARD valFORTH™ SOFTWARE SYSTEM EDITOR 1.1 COMMAND SUMMARY

Below is a quick reference list of all the commands which the video editor recognizes.

Entering the Edit Mode (executed outside of the edit mode)

V	(scr# ---)	* Enter the edit mode and view the specified screen.
L	(---)	* Re-view the current screen.
WHERE	(---)	* Enter the edit mode and position the cursor over the word that caused a compilation error.
LOCATE cccc	(---)	Enter the edit mode and position the cursor over the word "cccc" where it is defined.
LOCATOR	(ON/OFF ---)	When ON, allows all words compiled until the next OFF to be locatable using the LOCATE command above.
#BUFS	(#lines --)	Sets the length (in lines) of the storage buffer. The default is five.

Cursor Movement (issued within the edit mode)

ctrl	↑	* Move cursor up one line, wrapping to the bottom line if moved off the top.
ctrl	↓	* Move cursor down one line, wrapping to the top line if moved off the bottom.
ctrl	←	* Move cursor left one character, wrapping to the right edge if moved off the left.
ctrl	→	* Move cursor right one character, wrapping to the left edge if moved off the right.
RETURN		Position the cursor at the beginning of the next line.
TAB		Advance to next tabular column.

Editing Commands (issued within the edit mode)

ctrl	INS	Insert one blank at cursor location, losing the last character on the line.
ctrl	DEL	Delete character under cursor, closing the line.
shift	INS	* Insert blank line above current line, losing the last line on the screen.
shift	DEL	* Delete current cursor line, closing the screen.
ctrl	I	Toggle insert-mode/replace-mode. (see full description of ctrl-I).
BACKS		* Delete last character typed, if on the same line as the cursor.
ctrl	H	Erase to end of line (Hack).

Buffer Management (issued within the edit mode)

ctrl	T	Delete current cursor line sending it to the edit buffer for later use.
ctrl	F	Take the current buffer line and insert it above the current cursor line.
ctrl	K	Copy current cursor line sending it to the edit buffer for later use.
ctrl	U	Take the current* buffer line and copy it to the current cursor line.
ctrl	R	Roll the buffer making the topmost buffer line current.
ctrl	B	Roll the buffer backwards making the fourth buffer line on the screen current.
ctrl	C	Clear the current* buffer line and performs a ctrl-B.

*Note: The current buffer line is bottommost on the video display.

Changing Screens (issued within the edit mode)

ctrl	P	Display the previous screen saving all changes made to the current screen.
ctrl	N	Display the next screen saving all changes made to the current screen.
ctrl	S	* Save the changes made to the current screen and end the edit session.
ctrl	Q	* Quit the edit session forgetting all changes made to the current screen.

Special Keys (issued within the edit mode)

ESC		* Do not interpret the next key typed as any of the commands above. Send it directly to the screen instead.
ctrl	A	Put the arrow "-->" ("next screen") in the lower-right-hand corner of the screen unless it is already there, in which case remove it.
ctrl	J	Split the current line into two lines at the point where the cursor is.
ctrl	O	Corrects any major editing blunders.

Screen Management (executed outside of the edit mode)

FLUSH	(--)	* Save any updated FORTH screens to disk.
EMPTY-BUFFERS	(--)	* Forget any changes made to any screens not yet FLUSHed to disk.
COPY	(from to --)	* Copies screen #from to screen #to.
CLEAR	(scr# --)	* Blank fills specified screen.
CLEARS	(scr# #screens --)	Blank fills the specified number of screens starting with screen scr#.
SMOVE	(from to #screens --)	Duplicate the specified number of screens Starting with screen number "from".

* EDITOR 1.0 COMMAND

Software and Documentation
©Copyright 1982
Valpar International