Rough Draft

Comparison of "Starting Forth Lexicon" to available "PropForth 5.0 Lexicon"

Version 0.1
11/12/12
Created by G. Herzog, aka Loopy Byteloose

SF Lex	PF Lex	Both	
		<	
	<=		Trivial – Can use > instead
	<>		Trivial – Can use 0= instead
		<#	
		=	
		>	
	>=		Trivial – Can use < instead
		-	
		;	
		:	
!			PropForth defines three items: C!, L!, and W! For fetch from main memory.
		?	
		/	
		.	
."			
		'	
"			Reference omission? Maybe Provided in PropForth
(			PropForth uses {
)			PropForth uses }
[			Reference omission? Maybe Provided in PropForth
[']			
		]	Seems odd that PropForth has this, but not its mate.
	{		See)
	}		See (
@			PropForth defines three items: C@, L@, and W@ For store at main memory.
		*	
		*/	
	\		
		#	
		#>	
+!			
		0<	
	0<>		Trivial – Can work around in other ways with testing 0=
		0=	
		0>	

	0>=		Trivial – Can use 0<
	0branch		
	1		Why?
	-1		
1-			PropForth Quick References lists as included
1+			PropForth Quick References lists as included
	11/02/12		
	11/04/12		
	1lock		
	1unlock		
	2		
	-2		
2-			PropForth Quick References lists as included
2!			Would require better definition in PropForth 2C!, 2L!, or 2W!
2/			PropForth Quick References Lists as included - shift right arith
2@			Would require better definition in PropForth 2C@, 2L@, or 2W@
		2*	This is a shift function, shift left arithmetically
2+			PropForth Quick References lists as included
2constant			Would require better definition in PropForth – what length?
		2drop	
		2dup	
	2lock		
2over			
	2>r		
2swap			
	2unlock		
2variable			Would require better definition in PropForth – what length?
	3drop		
	4		
	-4		
	4*		
abort"			Reference omission? Maybe Provided in PropForth
		abs	
		accept	
	alignl		
	alignw		
		allot	
		and	
	andC!		
	andn		
	andnC!		
	asmlabel		
	_asmpfa>nfa		
		base	
begin			Reference omission? Maybe Provided in PropForth
	between		
	_bf		

	bl		Possible PropForth equivalent of “blank”
blank			
blk			File and OS dependent
block			File and OS dependent
	bounds		
	branch		
	build?		
	build_BootKernel		
	build_BootOpt		
	build_DevKernel		
	build_fsrd		
	build_fswr		
	.byte		
	c		
c,			
c!			
	c"		
c@			
	#C		
	C!		
	C@		
	C@++		
	$C_a_0branch		
	$C_a_2>r		
	$C_a_branch		
	$C_a_doconl		
	$C_a_doconw		
	$C_a_dovarl		
	$C_a_dovarw		
	$C_a_exit		
	$C_a_litl		
	$C_a_litw		
	$C_a_(+loop)		
	$C_a_(loop)		
	$C_a_lxasm		
	$C_a_next		
	cappend		
	cappendn		
	$C_a__xasm2>1		
	$C_a__xasm2>1IMM		
	ccopy		
	ccreate		
	cds		
	$C_fMask		
	checkdict		
	$C_IP		
	clearkeys		
	clkfreq		

		cmove
cmove>		
	_cnip	
	cnt	
	cog?	
	COG!	
	COG@	
	cogcds	
	.cogch	
	cogdump	
	coghere	
	cogid	
	cogio	
	cogiochan	
	cognchan	
	cognumpad	
	cogpad	
	cogreset	
	cogstate	
	cogstop	
	cogx	
compare		
	compile?	
	>con	
	.con	
	.conbyte	
	.concr	
	.concstr	
	.conemit	
	.conlong	
	.const?	
		constant
	.conwait	
	.conword	
count		
	cq	
		cr
	.cr	
		create
	(createbegin)	
	(createend)	
	$C_resetDreg	
	$C_rsPtr	
	$C_rsTop	
	$C_stPtr	
	.cstr	
	cstr=	
	$C_stTop	

While both have “create”, PropForth “create” is not the same in function.

	$C_stTOS	
	$C_varEnd	
d<		
d=		
d-		
d+		
		decimal
	delms	
	dictend	
	_dictsearch	
	dirA	
	_dl	
dmax		
dmin		
do		
	doconl	
	doconw	
does>		
	doloop	
	dothen	
	dovarl	
	dovarw	
	dq	
d.r		
		drop
du<		
		dump
	(dumpb)	
	(dumpe)	
	(dumpm)	
		dup
?dup		
	EC@	
	_ecs	
	edump	
	_eeread	
	eereadpage	
	_eestart	
	_eestop	
	_eewrite	
	eewritepage	
else		
		emit
empty-buffers		File and OS dependent
erase		
	ERR	
	EW!	
	EW@	

	exec	
		execute
	execword	
		exit
	_femit?	
	femit?	
		fill
	find	
	_finit	
	fisnumber	
	_fkey?	
	fkey?	
	fl	
	(fl)	
	fl_in	
	fl_lock	
	(flout)	
fm/mod		
	_fnf	
	fnumber	
		forget
	(forget)	
	forthentry	
	_forthpfa>nfA	
	free	
	freedict	
	fsbot	
	fsclear	
	fsdrop	
	_fsfind	
	fsfree	
	_fsfree	
	_fsk	
	_fslast	
	fsload	
	_fsload	
	fsls	
	_fsnext	
	_fsp	
	_fspa	
	fsps	
	_fsrd	
	fsread	
	_fsread	
	fstart	
	fstop	
	_fswr	
	fswrite	

	_ft		
	$H_cogdatA		
	$H_cq		
	$H_dq		
	$H_entry		
		here	
	herelal		
	herewal		
		hex	
hold			
	hubopf		
	hubopr		
		i	
	I;		
	I."		
	Ibegin		
	ibound		
	Ido		
	Ielse		
	_if		
	[if		
if			Reference omission? Maybe Provided in PropForth
	[ifdeF		
	[ifndeF		
	IiF		
	Iloop		
	I+loop		
		immediate	
		>in	
	ina		
include			File and OS dependent
	init_coghere		
	initcon		
	interpret		
	interpretpad		
	invert		
	io		
	io>cogchan		
	ioconn		
	(ioconn)		
	iodis		
	(iodis)		
	iolink		
	(iolink)		
	iounlink		
	(iounlink)		
	isdigit		
	isnamechar		

	isnumber		
	isunumber		
	lthen		
	lthens		
	iuntil		
		j	
		key	
	l		
	L!		
	L@		
	lasterr		
	lasti?		
	lastnfa		
	_lc		
		leave	
	_lf		
list			File and OS dependent
literal			See PropForth “litl” and “litw”
	litl		See “literal”
	litw		See “literal”
load			File and OS dependent
	lock		
	lock?		
	_lockarray		
	lockdict		
	.long		
loop			Reference omission? Maybe Provided in PropForth
	(+loop)		
	(loop)		
loop+			Reference omission? Maybe Provided in PropForth
	lshift		Logical left shift
	l>w		
	lxasm		
	>m		
m*			
m*/			
m+			
marker			A “Starting Forth” alternative for 'forget”
	_maskin		
	_maskouthi		
	_maskoutlo		
		max	
	memend		
		min	
	_mmcs		
mod			
		/mod	
		***/mod**	

move			
	name=		
	namecopy		
	namelen		
	namemax		
	_nd		
		negate	
	nextword		
	nfa>lfa		
	nfa>next		
	nfa>pfA		
	nfcog		
	(nfcog)		
	nip		
	npfx		
	number		
>number			
	numpad		
	numpadsize		
octal			Trivial – rather obsolete, can use Base to create if requred
	onb001		
	onboot		
	onre001		
	onreset		
		or	
	orC!		
	orlnfA		
	>out		
	outA		
		over	
	_p?		
	_p+	**pad**	
	padbl		
	pad>in		
	pad>out		
	padsize		
page			Useful to some users as it blanks the terminal screen when cluttered
	par		
	parse		
	parsebl		
	parsenw		
	parseword		
	pfa?		
	pfa>nfa		
	pinhi		
	pinin		
	pinlo		

	pinout		
	_pna		
postpone			
	prop		
	(prop)		
	propid		
	px		
	px?		
	_qp		
quit			
		>r	
		r>	
r@			
	reboot		
repeat			
	reset		
	rev		
	revb		
	rnd		
	rndtF		
		rot	
	rot2		
	rs?		
	RS!		
	RS@		
	rshift		
		#s	
	saveforth		Important PropForth word – saves to EEPROM an image of the dictionary in 32K ram
	$S_baud		
	sc		
	$S_cdsz		
	_sclh		
	_scli		
	_scll		
	_sclo		
	$S_con		
scr			
	_sda?		
	_sdah		
	_sdai		
	_sdal		
	_sdao		
	serflags?		
	serial		
	_serial		
	sersendbreak		
	sersetflags		
	seti		

		sign
	skipbl	
sm/rem		
	_sp	
sp@		
sp0		
		space
		spaces
	$S_rxpin	
	st?	
	ST!	
	ST@	
?stack		
	state	
	.str	
	.strname	
	$S_txpin	swap
	t0	
	t1	
	tbuF	
then		
tib		
#tib		
	tochar	
	todigit	
-trailing		
	tuck	
type		
u<		
	u>=	
	u/	
		u.
	u*	
	u*/	
	_udF	
		um*
		um/mod
	u/mod	
	u*/mod	
	unlock	
	unlockall	
until		
	unumber	
update		File and OS dependent
u.r		
use		File and OS dependent
		variable

	version		
	(version)		
	w		
	W!		
	W@		
	W+!		
	waitcnt		
	waitpeq		
	waitpne		
	_wc1		
	wconstant		
	_wf		
while			
	_wkeyto		
	w>l		
	wlastnfA		
word			PropForth “words” provides similar function
	.word		
	words		See “word”
	_words		
	wvariable		
	_xasm1>1		
	_xasm2>0		
	_xasm2>1		
	_xasm2>1IMM		
	_xasm2>flag		
	_xasm2>flagIMM		
	_xis		
	xisnumber		
	_xnu		
	xnumber		
	xor		