Simple BASIC 仕様

分類	Simple BASIC	Simple Embedded BASIC	命令	書式	説明

分岐	○	○	if	if 論理式1 then
				　命令1
				{else if 論理式2 then
				　命令2}
				{else if 論理式3 then
				　命令3}
				{else
				　命令4}
				end if
					複数の行で構成する場合のif文である。
					論理式が真の場合thenに続く命令を、論理式が偽の場合elseに対応する命令を実行する。
					命令は複数行記述可能で、その中にif文を含んでも良い。
					
					【関係演算子】
					・等しい　　=
					・等しくない　<>
					・大なり　　>
					・小なり　　<
					・以上　　>=
					・以下　　<=
					
					【論理演算子】
					・かつ　　&&
					・または　　||

	○	○		if 論理式 then 命令1 {else 命令2}	1行で構成する場合のif文である。 論理式が真のときthenに続く命令を、論理式が偽の場合elseに続く命令を実行する。 命令１、命令２はコロン（：）で区切って複数の文を含むことができるが、その中にifを含めることは不可とする。
	○		gosubM	gosubM 配列,要素番号変数,式１,分岐先１｛・・・,式ｎ,分岐先ｎ｝	配列に含まれる数値に応じたサブルーチンを実行する。 配列を最初から最後の要素まで、配列の値と式の値が一致する場合、分岐先の行番号のサブルーチンに分岐する動作を繰り返す。 分岐する際に、処理中の配列のインデックスを要素番号変数に格納する。 サブルーチンからはreturn命令で復帰する。 例） 　gosubM sts,idx,4,4000,5,5000,6,6000 　stsの値：0,0,4,0,5,0,5,6,1 の時 　最初に idx=2に設定し、行番号4000から始まるサブルーチンを実行する。 　次に　idx=4に設定し、行番号5000から始まるサブルーチンを実行する。 　次に　idx=6に設定し、行番号5000から始まるサブルーチンを実行する。 　次に　idx=7に設定し、行番号6000から始まるサブルーチンを実行する。 その後、命令を終了し、次のインストラクションを実行する。
	○		gosubS	gosubS 配列,要素番号変数,式１,分岐先１｛・・・,式ｎ,分岐先ｎ｝	gosubMと分岐する条件は同じだが、gosubSは最初に条件と合致した１回だけサブルーチンを実行する。
	○		gosubC	gosubC 式,分岐先0｛・・・,分岐先ｎ｝	式の値に従って分岐する。 式の値は０以上の連続した整数でなければならず、変数が取りうる値の数だけ、分岐先を指定しなければならない。 例えば、値が0～2の時、分岐先は３つ指定しなければならず、0の時最初の分岐先、1の時２番めの分岐先、2の時3番めの分岐先に分岐する。 例） gosubC sts,1000,2000,3000 sts=0の時　1000から始まるサブルーチンを実行 sts=1の時　2000から始まるサブルーチンを実行 sts=2の時　1000から始まるサブルーチンを実行
	○	○	for	for 変数 = 初期値 to 評価値 {step 増分} 　命令1 　｛exit for| continue for} 　{命令2} next	変数が範囲の間、命令を繰り返し実行する。 ①　変数に初期値を代入する ②　評価値と変数を比べ、初期値～評価値の範囲に入っていれば、繰り返しを止めてnextの次の行から実行する。 ③　命令を実行する ④　変数に増分を加える　②～④を繰り返す。 exit forを実行した場合は、繰り返しを止めてnextの次の行から実行する。 continue forを実行した場合は、continue for以降の命令（continue for～nextまでの命令）を実行せずに、forに戻る。
	○	○	while	while 条件 　命令 　｛exit while| continue while} 　{命令2} end while	条件が真の間、while～end whileの間を繰り返す。 条件に0でない値を指定すると無限ループとなる。 exit whileを実行した場合は、繰り返しを止めてend whileの次の行から実行する。 continue whileを実行した場合は、continue while以降の命令（continue while～end whileまでの命令）を実行せずに、whileに戻る。
	○	○	goto	goto 行番号	行番号に分岐する。
	○	○	gosub	gosub 行番号 サブルーチン retrun	行番号から始まるサブルーチンを実行する。 サブルーチンは、returnで終了する。 サブルーチンの実行が終わると、gosubの次の命令から実行する。
操作	○	○	new	new	プログラムを消去する。変数も開放する。
	○	○	run	run	プログラムを実行する。
	○	○	list	list {行番号}	指定の行を表示する。省略時は、プログラムのすべての行を表示する。
	○	○	clear	clear	変数を開放する。

	○		sound	sound チャネルNo,文字列	サウンドチャネルに音を設定する チャネルNo：１～３ 文字列：音を指定する ・音程　　ドレミファソラシ→ CDEFGAB 半音上げるときは#をつける C#→ド# 　　オクターブ 音程の後に0～9をつけて表現する 数字が大きいほど高い音を表す ・テンポ　Tの後に16分音符が１分間に入る数を指定する ・音の長さ　Lの後に音符の種類を指定する。L4であれば4部音符。 　　付点はL4H のように後ろにHをつける。 ・音量Ｖ　Vの後に0～255の数字をつけて音の大きさを表す。 　　数字が大きいほど音が大きい。0は消音。 音量と音の長さは、一旦指定するとその後に続く音程に引き継がれる。 休符 ピリオード.で表す。 区切り文字 記号と記号の間の_は無視。記号とパラメータの間の_はNG（L_8などはNG）。 例）音量200でT=640で4分音符のドレミファソラシドを表現する文字列 ： 　T640V200L4C4D4E4F4G4A4B4C5.
	○		play	play チャネルNo{,チャネルNo,チャネルNo}	指定したサウンドチャネルを演奏する。
	○	○	sleep	sleep 時間	指定した時間待機する。 時間　待機する時間をmsで指定する。（0～60000ms) 　　ただし、待機時間に正確さはない。 　　20msより小さい数字を指定しても最低20msはsleepする可能性がある。 　　0msを指定する場合はsleepを呼ばない方が良い。
	○		write	write アドレス,文字列	フラッシュメモリーに指定した文字列を書き込む。文字列の１文字を格納するのにアドレス１つを消費する。 アドレス：書き込む先頭アドレス（0～8191) 文字列：書き込む文字列
	○		strToA	strToA 文字列,開始インデックス,式,幅1,配列1{,幅2,配列2,・・・,幅n,配列n}	文字列の開始インデックスから幅で指定した文字数分を10進数と仮定して指定した配列の式で指定したインデックスに代入する。 配列を複数並べて書くと、続けて指定した幅の文字数分を10進数と仮定して代入する。 幅は０～65535の10進数表現のみが可能である。（変数や式の利用不可）
関数	○		soundXstatus()	soundXstatus()	サウンドチャネルの出力状態を取得する。　出力中の時１、出力していない時0を返す X：１～３
	○		keyX()	keyX()	スイッチの状態を返す。スイッチ押下中：１　スイッチを押していない時：０ Xは、A,B,U,D,L,Rの６つがある。
		○	in()	in(端子番号)	端子番号の値を取得する。 High(1)、Low（０）を返す。
		○	inAn()	in(端子番号)	端子番号の値を取得する。 端子の電圧0～５Vを0～1023に変換して返す。
	○	○	format()	format(書式文字列,値)	数値型の変数の値を書式文字列に従って文字列に変換する。 例：format(“%05D”,10) は、00010に変換される。 　format(“%-5D”,60)は、左に３つのスペースを付けた60に変換される。 書式文字列　%[フラグ][フィールド幅][変換文字] ・フラグ　　マイナスを指定すると右詰になる ・フィールド幅　変換文字にDを指定した場合、1～10が有効範囲 　　変換文字にXを指定した場合、1～8が有効範囲 　　フィールド幅の先頭の数字が0の場合、フラグの指定に関係なく右詰めとする。 　　（足りない桁は0で埋める) 　　フィールド幅の先頭の数字が0でない、かつ、右詰めの場合、右詰とする。 　　（足りない桁はスペースで埋める） 　　フィールド幅の先頭の数字が0でない、かつ、左詰めの場合、 左詰めとする。 　　（足りない桁はスペースで埋める） 　　数値の桁数または文字列の文字数がフィールド幅を超える場合、 　　左からフィールド幅で切断する ・変換文字　　D：10進数に変換する 　　X：16進数に変換する
	○		rand()	rand()	0～255の乱数を取得する
	○	○	strlen()	strlen(文字列)	文字列の文字数を取得する
	○	○	chr()	chr(文字コード)	指定した文字コードを１文字の文字列に変換する。
	○	○	val()	val(文字列)	指定した文字列を10進数と仮定して数値に変換する。
	○	○	mid()	mid(文字列,先頭文字No,終了文字No)	指定した文字列の一部を取得する。 先頭文字No：取得する先頭の文字の番号を指定する。 　　番号は0から始まり、左から１文字目を0番で表す。 　　先頭文字Noで指定した文字は取得する文字列に含む。 終了文字No：取得する終端の文字の番号を指定する。 　　番号は0から始まり、左から１文字目を0番で表す。 　　終了文字Noで指定した文字は取得する文字列に含まない。 例：・先頭文字Noと終了文字Noに同じ値を指定すると、空の文字列を取得する。 　・文字列全体を取得するときには、先頭文字Noに０、終了文字Noに文字列の文字数を指定する。
	○		read()	read(アドレス{, 最大文字数})	フラッシュメモリーの指定したアドレスから文字列を読み出す。 アドレス：読み出す先頭アドレス（0～8191) 最大文字：読み出す最大文字数を指定する。省略時はWriteで書き込んだ文字数を読み出す。 　　Writeしていないアドレスを読み出すと暴走する可能性があるので、そういう時に使う。
	○		mod()	mod(a,b)	aをbで割った余りを返す
	○		abs()	abs(a)	aの絶対値を返す
	○		sin()	sin(a)	sinの値を10000倍した値を返す。 aには角度（0～360)を10倍した値を指定する。

	○		cos()	cos(a)	cosの値を10000倍した値を返す。 aには角度（0～360)を10倍した値を指定する。
	○		createBM()	createBM(幅,高さ)	ビットマップを生成し、そのIDを取得する。 幅：横方向のドット数 高さ：縦方向のドット数
	○		createFM()	createFM(幅,高さ)	フレームメモリーを生成し、そのIDを取得する。 幅：横方向のドット数 高さ：縦方向のドット数
	○		isCol()	isCol(sx,sy,sw,sh,dx,dy,dw,dh)	sx,syを左上に持つswの幅,shの高さを持つ単形領域とdx,dyを左上に持つdwの幅,dhの高さを持つ単形領域が重なっているかどうかを調べる。 衝突あり：１ 衝突なし：０ 衝突判定は、sx,sy,sw,shの四角形の内側に,dx,dy,dw,dhの四角形の頂点が１つでも含まれれば、衝突と判定する。
	○		isOut()	isOut(sx,sy,sw,sh,dx,dy,dw,dh)	sx,syを左上に持つswの幅,shの高さを持つ単形領域の中からdx,dyを左上に持つdwの幅,dhの高さを持つ単形領域がはみ出しているかを調べる。 はみ出しあり：１ はみ出しなし：０ はみ出し判定は、sx,sy,sw,shの四角形の内側から,dx,dy,dw,dhの四角形の頂点が１つでも出ている場合、はみ出しと判定する。
	○		isColM()	isColM(sx,sy,sw,sh,dx,dy,dw,dh,sts,val1{,・・・valn})	sx,syを左上に持つswの幅,shの高さを持つ単形領域が重なっているかどうかを調べる。 重なりを調べる対象は、sts[]の値がval１～valnのいずれかであるインデックスと同じ値のインデックスのdx[],dy[],dw[],dh[]で表現される単形領域である。 衝突あり：衝突していた単形領域の配列のインデックス（最初にたまたま見つけたもの） 衝突なし：配列の要素数（最大のインデックス値＋１） 衝突判定は、sx,sy,sw,shの四角形の内側に,dx,dy,dw,dhの四角形の頂点が１つでも含まれれば、衝突と判定する。
コメント	○	○	rem または ‘	rem コメント	rem の後ろにコメントを記入する。省略形としてアポストロフィー（’）を使うことができる 行の先頭のみサポート
宣言	○	○	dim	dim 変数名[要素数]{,変数名[要素数]}	配列を宣言する。　添字は、０～要素数−１まで。
表示	○	○	print	print 式	式の値を表示する
	○		フレームメモリーについて		描画命令を使って、グラフィック描画ができるビットマップ。FMID０の画面以外は、createFM関数で作成する。
	○		cls	cls {bUpdate,FMID}	画面を消去する Bupdate T:表示を更新する　F:フレームメモリのみに描画　省略時はT FMID　描画対象のフレームメモリー　省略時は画面（０）
	○		line	line x1,y1,x2,y2,c{,dm,bUpdate,FMID}	(x1,y1)-(x2,y2)に線を引く c：1or0　1が黒0が白 dm：set,or,xor,not,and Bupdate: T,F FMID　描画対象のフレームメモリー　省略時は画面（０）
	○		box	box x1,y1,x2,y2,c{,dm,bUpdate,FMID}	(x1,y1)-(x2,y2)の四角形(塗りつぶしなし)を描く c：1or0　1が黒0が白 dm：set,or,xor,not,and Bupdate: T,F FMID　描画対象のフレームメモリー　省略時は画面（０）
	○		fill	fill x1,y1,x2,y2,c{,dm,bUpdate,FMID}	(x1,y1)-(x2,y2)の四角形(塗りつぶし)を描く c：1or0　1が黒0が白 dm：set,or,xor,not,and Bupdate: T,F FMID　描画対象のフレームメモリー　省略時は画面（０）
	○		putStr	putStr x,y,文字列,c{,dm,bUpdate,FMID}	x,yを文字の左上の座標として、文字を描く c：1or0　1が黒0が白 dm：set,or,xor,not,and Bupdate: T,F FMID　描画対象のフレームメモリー　省略時は画面（０）
	○		update	update {x,y,width,hight}	FMID０の(x,y)を左上とするwidth,hightの大きさのエリアをLCDに転送する。省略時は全画面を転送する。
	○		get	get x,y,参照FMID,x1,y1,幅,高さ{,dm,bUpdate,操	参照FMIDで指定したフレームメモリーのx1,y1を左上とする幅、高さで指定した広さのイメージを操作FMIDのx,yで指定した場所にフレームメモリーに取得する。 dm：set,or,xor,not,and
	○		put	put x,y,参照FMID または BMID{,dm,bUpdate,操作FMID}	指定した座標に参照FMID または BMIDで指定したイメージを描く。 ※　FMIDとBMIDは同じ値にならないようになっているので、表示してたいイメージのIDを指定すればBASIC内部でどちらを指定したのかを判別する。
	○		setBM	setBM BMID,行,ビットマップデータ	BMIDで指定したビットマップをビットマップデータで初期化する 行　ビットマップデータで初期化するビットマップの行を指定する （0～ビットマップの高さ-1） ビットマップデータ　左のビットから順に１文字を対応させて、 ０と１で構成した文字列でビットマップの値を指定する。
load,save	○	○	save	save C	シリアル通信経由でTeraTermなどの端末にプログラムリストを送る。 シリアル通信経由でsaveC送られた文字列をそのまま送ると、プログラムがロードされる。

	○			save 番号	フラッシュROMにプログラムを保存する （サイズ上限：60KB） 番号：1,2
		○		save	フラッシュROMにプログラムを保存する （サイズ上限：16KB）
		○		save A	フラッシュROMにプログラムを保存する。保存したプログラムは、電源ONで自動実行される。 （サイズ上限：16KB） DI0とDO1を接続した状態で電源ONすると自動実行はスキップされる。
	○		load	load 番号	フラッシュROMからプログラムをロードする。 番号：1,2
	○		フラッシュROMについて		128KBのフラッシュメモリーは、以下のように使う。 ・先頭8KBはコモン領域でBASICプログラムから書き換え可能。 ・残りの120KBの前半をloadsave番号1、後半をloadsave番号2が使う。
		○	フラッシュROMについて		16KBのフラッシュメモリー
演算子	○	○	四則演算	+,-,*,/	四則演算をする。文字列の場合は、＋のみ可能
	○	○	関係演算子	'=,<>,<,>,<=,>=	左右を比較し、真、偽を返す　0：偽　0以外：真
	○	○	論理演算子	'&&,||	真偽　を　AND　または　OR　の論理演算をし　真偽を返す
I/O		○	入出力端子について		入出力端子には番号を付けて端子番号で指定する。 BASICの動く機器によって端子番号と機能は異なる。 ○SimpleTestBench 端子名　端子番号　機能 IO0～7　・・・　0～7　全ての端子はデジタル入出力機能を持つ。PWM0,1を有効にすると、No4と7はPWM出力となる。 電源投入時は、全ての端子はデジタル出力になる。（初期値０） AI0～7　・・・　8～15　全ての端子はデジタル・アナログ入力機能を持つ。 In()で読むとデジタル値、InAn()で読むとアナログ値が読める。 アナログ値は10ms間隔で更新する。 D00,DO1　・・・　16,17　デジタル出力として機能する。LEDを直接ドライブできる。(初期値０) DI0　・・・　18　デジタル入力として機能する。
		○	out	out　端子番号,値	指定した端子電圧を値に従って、値が０のとき０V、値が０以外の時５Vに変更する。
		○	setFunc	setFunc 端子番号,機能	指定した端子の機能を切り替える。 機能に０を指定した時、入力になる。 機能の１を指定した時、出力になる。 機能に２を指定した時、PWM出力になる。（PWM出力に一旦出力すると、その後は再起動まで他への変更は不可）初期値は、インターバル1ms、デューティ比０%。
		○	setInterval	setInterval 端子番号,インターバル	PWM出力端子のインターバルを設定する。インターバルはnsで指定する。 インターバルを変更した後は、デューティ比を設定しないとデューティ比が不定になる。
		○	setDuty	setDuty 端子番号,比率	PWM出力端子のデューティ比を設定する。デューティ比(%)を100倍した値 (10000-0）。
デバッグ支援	○		trace	trace レベル｛,機能｝・・・｛,機能｝	トレース情報のComポートへの出力方法を指定する トレースレベル　：off　トレース情報を表示しない。 low　実行とフリーメモリ、エラー時にのみ発生するログ mid　Lowに加え、各ジャンプスタック情報、変数の追加タイミング hi　Midに加え、すべての変数、配列の値 トレース機能　：VG　ValueGettable LOOP　For-Next,While-EndWhile 例）trace mid,VG,GS

※　命令は、基本は小文字とする。
※　{}の中は省略可能である
※　灰色は未実装