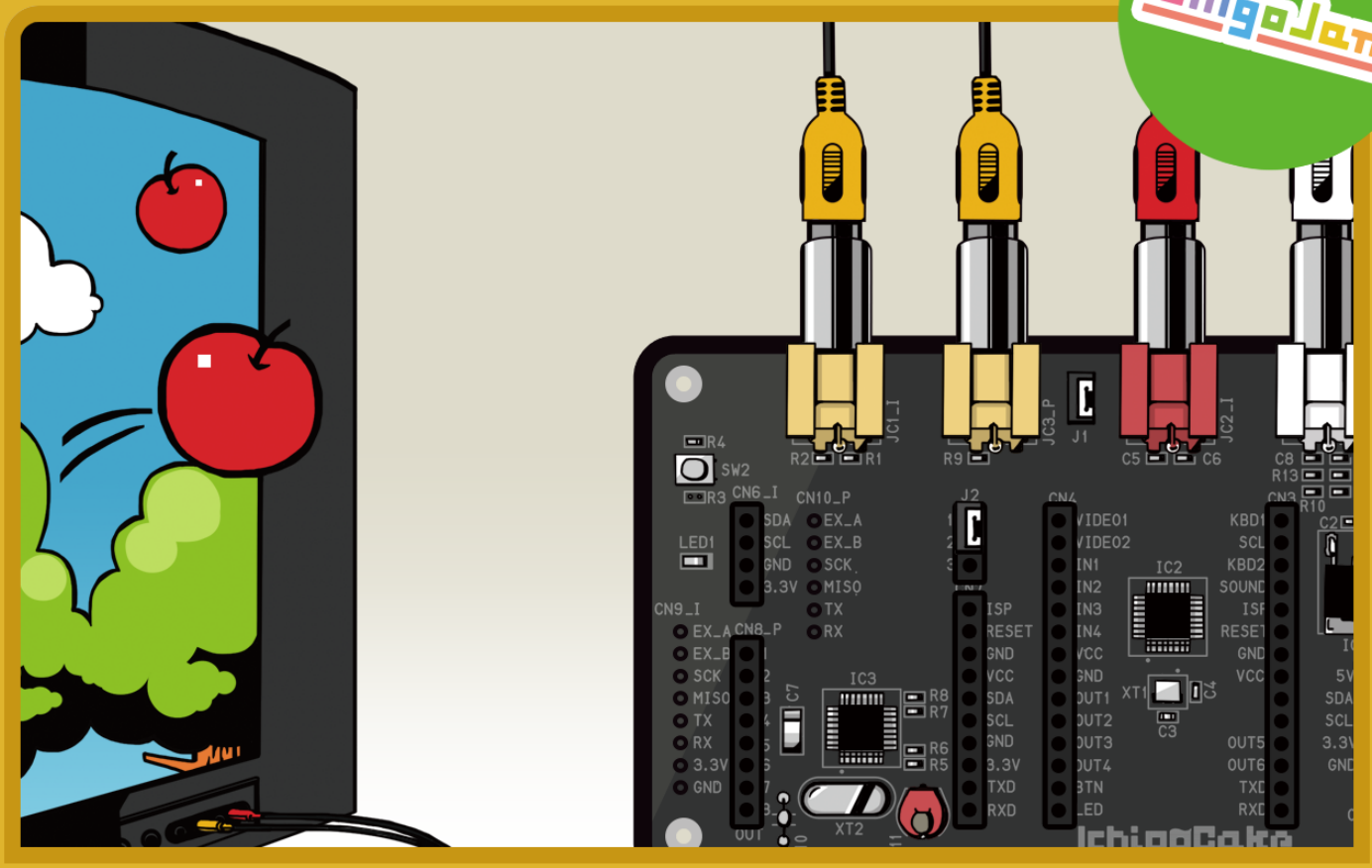
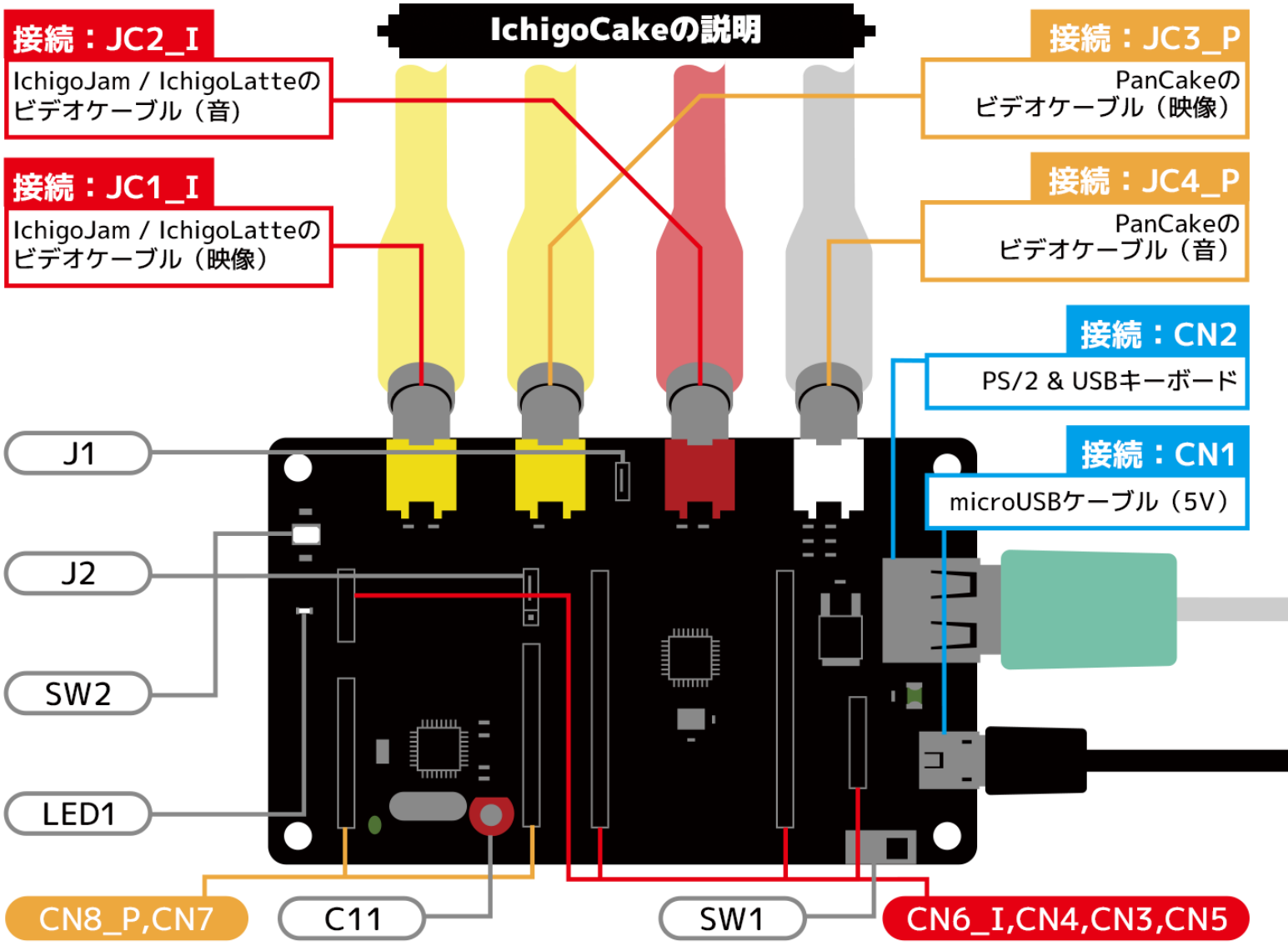
IchigoCake

for ゲームクリエイター



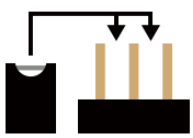
PCN

IchigoCakeの説明



J1	IchigoJam / IchigoLatteのTXと PanCakeのRXを接続します。(※1) ※1 ジャンパーピンを外すとPanCakeは使えません。
J2	ビデオ信号を切り替えます。
SW2	Jam　：BTN()で値[0/1]が取得できます。 Latte：BTN()で値[0/1]が取得できます。
LED1	Jam　：LED1で点灯、LED0で消灯します。 Latte：led(1)で点灯、led(0)で消灯します。
CN8_P,CN7	PanCakeのピンソケット
CN6_I,CN4,CN3,CN5	IchigoJam / IchigoLatteの ピンソケット
C11	PanCakeの映像調整に使います。(※2) ※2 起動時に画面がカラーにならない場合、 内部を精密ドライバーなどで回して調整してください。
SW1	電源をON / OFFします。

J2について：



ジャンパーピンを挿す位置によって、
IchigoJam/IchigoLatteとPanCakeの映像出力を
操作できます。

[ジャンパーピンを挿す位置:1-2]

JC1-IからIchigoJam/IchigoLatte、
JC3-PからPanCakeの映像を出力します。
JC1-I、JC3-Pを
1台のモニタで差し替えるか、
2台のモニタにそれぞれ接続してください。

[ジャンパーピンを挿す位置:2-3]

JC1-IからIchigoJam/IchigoLatte、
PanCakeの両方の映像を出力します。
1台のモニタに接続し、下記コマンドを
プログラムの最初と最後の行に入れる
などして切り替えてください。

- IchigoJam→PanCake
  VIDEO 0:?"PC VIDEO 01"
- PanCake→IchigoJam
  VIDEO 1:?"PC VIDEO 00"

- IchigoLatte→PanCake
  video(0);uart("PC VIDEO 01\n");
- PanCake→IchigoLatte
  video(1);uart("PC VIDEO 00\n");

IchigoJam BASIC reference

キーボード操作

操作	解説
キー	文字を入力する
Shift / シフト	キーと共に押し記号や小文字などを入力する
カタカナ	アルファベットとカタカナ（ローマ字入力）を切り替える（CTRL+SHIFT / コントロール＋シフトでも可）
Enter / エンター	コマンドを実行する（プログラム変更時もその行でEnterキー）
Shift+Enter / シフト＋エンター	行を分割する
ESC / エスケープ	プログラムの実行、リスト表示、ファイル一覧表示を止める
カーソルキー	カーソルキーを移動する
Backspace / バックスペース	カーソルの前の文字を消す
Delete / デリート	カーソルにある文字を消す
ALT / オルト	0-9/A-Kと合わせて押すことで拡張文字入力（SHIFT押しながらで切り替え）、'['と合わせて押して'_'、']'と合わせて押して'\'や'￥'の入力
Home End / ホーム エンド	カーソルを行頭へ移動、カーソルを行末へ移動
Page Up Page Down / ページアップ ページダウン	カーソルを画面上へ移動、カーソルを画面下へ移動
Caps / キャップス	大文字と小文字を切り替える
Insert / インサート	キーボードの上書きモード/挿入モードを切り替える（CTRL+ALTでも可能）
ファンクションキー	F1:画面クリア、F2:LOAD、F3:SAVE、F4:LIST、F5:RUN、F6:?FREE()、F7:OUT0、F8:VIDEO1、F9:FILES、F10:SWITCH
ボタン	押しながら起動でFILE0を自動実行する

初級コマンド

コマンド	解説	例
LED 数 / エルイーディー	数が1なら光り、0なら消える	LED 1
WAIT 数1{,数2} / ウェイト	数1の数値フレーム分待つ 60で約１秒、省略可の数2指定で低電力化、数1のマイナス指定で走査線分で待つ(-261でWAIT1と同等)	WAIT 60
: / コロン	コマンドを連結する	WAIT 60:LED 1
行番号 コマンド	プログラムとしてコマンドを記録する（1～32767まで、標準表示は16384まで）	10 LED1
行番号	指定した行番号のプログラムを消す	10
RUN / ラン	プログラムを実行する [F5]	RUN
LIST {行番号1{,行番号2}} / リスト	プログラムを表示する [F4] （行番号1で1行表示、行番号1がマイナスでその行まで表示、行番号2指定でその行まで表示、行番号2が0の時終わりまで表示、ESCで途中停止）	LIST 10,300
GOTO 行番号 / ゴートゥー	指定した行番号へ飛ぶ（式も指定可能）	GOTO 10
END / エンド	プログラムを終了する	END
IF 数 {THEN} 次1 {ELSE 次2} / イフ・ゼン・エルス	数が0でなければ次1を実行し、0であれば次2を実行する（THEN,ELSE以降は省略可）	IF BTN() END
BTN({数}) / ボタン	ボタンが押されていれば1、そうでないとき0を返す（数：0(付属ボタン)/UP/DOWN/RIGHT/LEFT/SPACE/X(88)、省略で0、-1でビットパターンで返す）	LED BTN()
NEW / ニュー	プログラムを全部消す	NEW
PRINT {数や文字列} / プリント	文字を表示する （文字列は"で囲む、";" で連結できる） 省略形：？	PRINT "HI!"
LOCATE 数1,数2{,数3} / ロケート	次に文字を書く位置を横、縦の順に指定する（左上が0,0、縦=-1で無表示）。数3が0でなければ指定した場所にカーソルを表示する。省略形：LC	LOCATE 3,3

CLS / クリア スクリーン	画面を全部消す	CLS
RND(数) / ランダム	0から数未満の正数をランダムに返す	PRINT RND(6)
SAVE {数} / セーブ	プログラムを保存する (0～5の6つ、省略で前回使用した数) ボタンを押した状態で起動すると0番を読み込み自動実行	SAVE 1
LOAD {数} / ロード	プログラムを読み出す (0～5の6つ、省略で前回使用した数)	LOAD
FILES {数1{,数2}} / ファイルズ	数1(省略可)～数2のプログラム一覧を表示する（EEPROM内ファイル表示に対応、0指定ですべて表示、ESCで途中停止）	FILES
BEEP {数1{,数2}} / ビープ	BEEPを鳴らす 周期(1-255)と長さ(1/60秒単位)は省略可 ※SOUND(EX2)-GNDに圧電サウンダーなどの接続必要	BEEP
PLAY {MML} / プレイ	MMLで記述した音楽を再生する MML省略で停止 ※SOUND(EX2)-GNDに圧電サウンダーなどの接続必要（次項のMML参照）	PLAY "$CDE2CDE2"
TEMPO 数 / テンポ	再生中の音楽のテンポを変更する	TEMPO 1200
数 + 数	足し算する	PRINT 1+1
数 - 数	引き算する	PRINT 2-1
数 * 数	掛け算する	PRINT 7*8
数 / 数	割り算する（小数点以下は切り捨て）	PRINT 9/3
数 % 数	割り算した余りを返す	PRINT 10%3
(数)	カッコ内を優先して計算する	PRINT 1+(1*2)
LET 変数,数 / レット	アルファベット１文字を変数として数の値を入れる（配列に連続代入可能 LET[0],1,2） 省略形：変数=数	LET A,1
INPUT {文字列,}変数 / インプット	キーボードやUARTからの入力で数値を変数にいれる（文字列とコンマは省略可）	INPUT "ANS?",A
TICK() / ティック	CLTからの時間を返す（1/60秒で1進む） *数に1指定で1/(60*261)秒で1進む時間	PRINT TICK()
CLT / クリア ティック	時間をリセットする	CLT
INKEY() / インキー	キーボードやUARTから１文字入力する（入力がない時は0、UARTから0が入力された時は#100）	PRINT INKEY()
定数	LEFT=28, RIGHT=29, UP=30, DOWN=31, SPACE=32	IF INKEY()=SPACE LED1
CHR$(数) / キャラ	文字コードに対応する文字を返す（コンマ区切りで連続表記可）	PRINT CHR$(65)
ASC("文字") / アスキー	文字に対する文字コードを返す	PRINT ASC("A")
SCROLL 数 / スクロール	指定した方向に1キャラクター分スクロールする（0/UP:上、1/RIGHT:右、2/DOWN:下、3/LEFT:左）	SCROLL 2
SCR({数,数}) / スクリーン	画面上の指定した位置に書かれた文字コードを返す（指定なしで現在位置） 別名：VPEEK	PRINT SCR(0,0)
VPEEK({数,数}) / ブイ・ピーク	画面上の指定した位置に書かれた文字コードを返す（指定なしで現在位置）	PRINT VPEEK(0,0)
数 = 数	比較して等しい時に1、それ以外で0を返す (==でも可)	IF A=B LED 1
数 <> 数	比較して等しくない時に1、それ以外で0を返す (!=でも可)	IF A<>B LED 1
数 <= 数	比較して以下の時に1、それ以外で0を返す	IF A<=B LED 1
数 < 数	比較して未満の時に1、それ以外で0を返す	IF A<B LED 1
数 >= 数	比較して以上の時に1、それ以外で0を返す	IF A>=B LED 1
数 > 数	比較してより大きい時に1、それ以外で0を返す	IF A>B LED 1
式 AND 式 / アンド	どちらの式も1の時に1、それ以外で0を返す (&&でも可)	IF A=1 AND B=1 LED 1
式 OR 式 / オア	どちらかの式が1の時に1、それ以外で0を返す (||でも可)	IF A=1 OR B=1 LED 1
NOT 式 / ノット	式が0の時に1、それ以外で0を返す (!でも可)	IF NOT A=1 LED 1
REM / リマーク	これ以降の命令を実行しない（コメント機能） 省略形：'	REM START

IchigoJam BASIC reference

初級コマンド

FOR 変数=数1 TO 数2 {STEP 数3} NEXT / フォー・トゥー・ステップ・ネクスト	変数に数1をいれ、数2になるまで数3ずつ増やしながらNEXTまでをくりかえす （STEPは省略可、6段まで）	FOR I=0 TO 10:?I:NEXT
POS({数}) / ポジション	カーソル位置を返す（数、省略時または0:X+Y*幅、1:X座標、2:Y座標） *ver1.4以上	?POS(0),POS(1)
DRAW 数1,数2{,数3,数4}{,数5} / ドロー	数1,数2の座標から数3,数4の座標へ線を引く（座標は最大63x47）、数5に0指定で線を消し、2指定で反転する、省略時または1指定で線を引く *ver1.4以上	DRAW 1,5,10,15
POINT(数1,数2) / ポイント	数1,数2の座標にDRAWで描かれた点または文字があるときに1、そうでないとき0を返す *ver1.4以上	?POINT(1,5)
OUT 数1{,数2} / アウト	外部出力OUT1-11に0または1を出力する 数2を省略でまとめて出力できる（数2に-1指定でINへ切り替え、-2指定でプルアップ付きINへ切り替え ※IN3は除く）	OUT 1,1
IN({数}) / イン	IN0-10から入力する（0または1） 数を省略してまとめて入力できる （IN0,1,4,9はプルアップ、IN5-8,10-11はOUTで切り替え時使用可能、IN0,9はボタン）	LET A,IN(1)
ANA({数}) / アナログ	外部入力の電圧(0V-3.3V)を0-1023の数値で返す(2:IN2、5-8:IN5-8(OUT1-4)、0,9:BTN、省略で0)	?ANA()
PWM 数1,数2{,数3} / ピーダブリューエム	外部出力OUT2-5に数2で0.01msec単位で指定するパルスを出力する（0-2000、周期20msec）、数3で周期を指定（省略時2000=20msec、マイナス値指定で周期1/480）	PWM 2,100

上級コマンド

コマンド	解説	例
CLV / クリア バリアブル	変数、配列を全部0にする 別名：CLEAR	CLV
CLK / クリア キー	キーバッファとキーの状態をクリアする	CLK
CLO / クリア アウトプット	入出力ピンを初期状態に戻す	CLO
ABS(数) / アブソリュート	絶対値を返す（マイナスはプラスになる）	?ABS(-2)
[数]	配列（[0]から[357]までの358コの連続した変数として使える） LET[0],1,2,3で連続代入可能	[3]=1
GOSUB 行番号 RETURN / ゴーサブ・リターン	数または式で指定した行番号に飛び、RETURNで戻ってくる 省略形:GSB (30段まで)/RTN	GOSUB 100
DEC$(数{,数}) / デシ	数を文字列にする（２番目の数は桁数、省略可）	?DEC$(99,3)
#16進数	16進数で数を表記する	#FF
HEX$(数{,数}) / ヘックス	数を16進数の文字列にする（２番目の数は桁数、省略可）	?HEX$(255,2)
`2進数	2進数で数を表記する	`1010
BIN$(数{,数}) / バイナリー	数を2進数の文字列にする（２番目の数は桁数、省略可）	?BIN$(255,8)
数 & 数	論理積（ビット演算）	?3&1
数 | 数	論理和（ビット演算）	?3|1
数 ^ 数	排他的論理和（ビット演算）	?A^1
数 >> 数	右シフトする（ビット演算）	?A>>1
数 << 数	左シフトする（ビット演算）	?A<<1
~数	ビット反転（ビット演算）	?~A

COS(数) / コサイン	指定された数を角度の度数としてコサインの値の256倍を返す *ver1.4以上	?COS(90)
SIN(数) / サイン	指定された数を角度の度数としてサインの値の256倍を返す *ver1.4以上	?SIN(90)
STOP / ストップ	プログラムを中断する	STOP
CONT / コンティニュー	実行中の行や、中断した行を再度実行する	CONT
SOUND() / サウンド	音が再生中なら1、そうで無いとき0をを返す	?SOUND()
FREE() / フリー	プログラムの残りメモリ数を返す	?FREE()
VER() / バージョン	IchigoJam BASICのバージョン番号を返す	?VER()
RENUM {数1{,数2}} / リナンバー	プログラムの行番号を数1(省略時は10)から数2(省略時は10)刻みにする。GOTO/GOSUBの飛び先は手で変更必要な場合がある	RENUM
LRUN {数} / ロードラン	プログラムを読み込み後、実行する	LRUN 1
FILE() / ファイル	最後にプログラムを読み込み、書き込み行った数を返す	?FILE()
LINE() / ライン	現在実行中の行番号を返す（非実行時は0）	?LINE()
SRND 数 / エスランド	種を指定して乱数を初期化する	SRND 0
HELP / ヘルプ	メモリマップを表示する	HELP
PEEK(数) / ピーク	メモリ読み出し（キャラクターパターン0-#7FFなど）	?PEEK(#700)
POKE 数,数 / ポーク	メモリへの書き込み（連続書き込み可能 POKE#700,1,2,3）	POKE #700,#FF
COPY 数1,数2,数3 / コピー	メモリコピー 数1のアドレスへ数2のアドレスから数3の長さ分コピー(数3マイナスでコピー方向が逆になる)	COPY #900,0,256
CLP / クリア パターン	キャラクターパターン(#700-#7FF)を初期化する	CLP
"文字列"	文字列の先頭アドレスを返す	A="ABC"
STR$(数1{,数2}) / ストリング	文字列を返す（数2(省略可)で長さ指定）	PRINT STR$(A)
LEN("文字列") / レングス	文字列の長さを返す	PRINT LEN("ABC")
@ラベル / アットマーク	行の先頭に書くとラベルとなり、行番号の代わりとして使える（GOTO @LOOPなど） ※前方一致	@LOOP
VIDEO 数1{,数2} / ビデオ	画面表示非表示を切り替える 0で画面表示を停止し処理高速化 （F8で表示）、省略可能な数2でVIDEO0時CPUクロックを1/数2に変更し省電力化、数1が2の倍数で白黒反転、数1が3以上で拡大モード	VIDEO 0
RESET / リセット	IchigoJamをリセットする	RESET
SLEEP / スリープ	プログラムを休止する (ボタンを押すと起動し、LRUN0を実行する)	SLEEP
UART 数1{,数2} / ユーアート	数1：シリアル出力設定（0:オフ、1:PRINTのみ、2:PRINT/LC/CLS/SCROLL、3:PRINTのみ/改行コード\r\n、+4で入力エコーバック、+8で画面表示OFF、初期値:2）、数2：シリアル受信設定（0:オフ、1:オン、+2:ESC無効、+4:CR変換(13→10) 省略時1）	UART 0
BPS 数1{,数2} / ビーピーエス	シリアル通信速度を変更する(0で初期値の115,200bps、-1:57600bps、-2:38400bps、-100以下指定で指定した数の-100倍bpsに指定、-2304:230400bps)。数2でI2Cの通信速度設定（単位kHz、0:デフォルト400kHz)	BPS 9600
OK {数} / オーケー	OKやエラーメッセージの表示有無を切り替える（数、1:表示、2:非表示、省略で1）	OK 2
I2CR(数1,数2,数3,数4,数5) / アイツーシーリード	I2Cで周辺機器から読み込む I2Cアドレス、コマンド送信アドレス・長さ、受信アドレスと長さ（コマンド送信が1byteの時数3を省略可、コマンド送信が0byteの時数2/数3を省略可）	R=I2CR(#50,#114A,2,#114C,2)
I2CW(数1,数2,数3,数4,数5) / アイツーシーライト	I2Cで周辺機器に書き込む I2Cアドレス、コマンド送信アドレス・長さ、送信アドレスと長さ（数4/数5は省略可、コマンド送信が1byteの時数3を省略可）	R=I2CW(#50,#114A,2,#114C,2)

IoT.IN() / アイオーティーイン	sakura.ioモジュールから受信した数を一つ読み込む	R=IoT.IN()
IoT.OUT 数 / アイオーティーアウト	sakura.ioモジュールへ数をチャンネル0で即時送信する	IoT.OUT 100
WS.LED 数1{,数2} / ダブリュー・エス・エル・イー・ディー	配列の先頭から緑赤青の順に設定された値でLEDに接続されたWS2812Bを数1の分光らせる。数2を指定するとその数だけ繰り返す。*ver1.4以上	WS.LED 3
SWITCH {数} / スイッチ	画面表示をテレビと液晶とを切り替える（数1 0:テレビ、1:液晶）、数2で液晶の濃さを指定	SWITCH
USR(数,数) / ユーザー	マシン語呼び出し（注意！間違えると高確率でIchigoJamが停止する）	A=USR(#700,0)

演算子優先順位

優先順	演算子	備考
1	()	カッコ
2	- ~ ! NOT	単項演算子（マイナス、ビット反転、論理否定）
3	* / % MOD << >> & ^	掛け算・割り算・除算・ビットシフト・ビットアンド・ビットXOR
4	+ - |	足し算・引き算・ビットオア
5	= != < > <= >=	論理比較
6	AND	論理積
7	OR	論理和

IchigoJam SOUND reference

IchigoCakeのJC1_I（赤）に音声端子を接続すると、PLAY/BEEP/TEMPOで音を操ることができます。MML（ミュージックマクロランゲージ）を使った音楽や効果音でプログラムを豊かにしてみましょう。

MML	説明	例
[音]	音(ドレミファソラシ=CDEFGAB)を鳴らす	PLAY”CDEFG”
[音]n	長さを指定して音を鳴らす(1/2/3/4/8/16/32)	PLAY”C4E2D1”
[音]n.	符点音符、長さが指定の 1.5 倍になる	PLAY”C4.E2.D1.”
[音]+	半音上げる	PLAY”CC+”
[音]-	半音下げる	PLAY”DD-“
Rn	休符（長さ指定、符点にも対応）	PLAY”CRDRE”
Tn	テンポ指定（初期値 120）	PLAY”T60CDE”
Ln	長さ指定しないときの長さ（初期値 4）	PLAY”L16CCC”
On	オクターブで指定（初期値 4）O1C～O6D	PLAY”O1CO5D”
<	オクターブを 1 上げる	PLAY”C<C”
>	オクターブを 1 下げる	PLAY”C>C”
Nn	音の高さ指定(1～255) 長さは L で指定	PLAY”N1N2N4N8”
$	これ以降の MML を繰り返す	PLAY”CDE$GC”
'	音を止める（何も鳴らさない）	PLAY”C'DE”

※　画面出力の水平同期信号を使っているため正確な平均律では鳴りません

IchigoCake BASIC reference

通常のIchigoJam BASIC に加えて、以下のコマンドが追加されています。

初級コマンド

コマンド	解説	例
KBD 数	キーボードのキーマップを切り替える。 （0：US、1：JP）	KBD 1

PacCake用コマンド

コマンド	解説	例
PC.CLEAR cn	画面を消去する。cnには色番号を指定する。	PC.CLEAR #A
PC.LINE x1,y1,x2,y2,cn	線を引く。	PC.LINE 0,0,16,11,1
PC.CIRCLE xc,yc,ra,cn	円を描く。 xc,ycには中心座標を指定する。 raには半径を指定する。	PC.CIRCLE 16,16,16,1
PC.STAMP px,py,tc, "cc...c"	8x8の絵を描く。 px,pyには出力座標を指定する。 tcには透明色を指定する。 cc...cには左上から右方向への色番号を羅列する。	PC.STAMP 0,8,2," 22828282222888222222 22222022220202220222 220222220222022222022"
PC.STAMP1 px,py,cn, “cccccccccccccccc”	8x8の一色絵を描く。 px,pyには出力座標を指定する。 cnには色を指定する。 ccccccccccccccccには左上から右方向への0,1を羅列する。 0は透明になる。	PC.STAMP1 0,8,2, "FF00FF00FF00FF00"
PC.STAMPS px,py,si {,fs{,ra}}	px,pyには出力座標を指定する。 si には組み込みスプライト画像番号を指定する。 fs を指定すると左右反転の制御ができる。ON/OFF[1/0] (Default:0) ra を指定すると角度の制御ができる。 [0:0°,1:-90°,2:180°,3:90°] (Default:0)	PC.STAMPS 36,18,#15 PC.STAMPS 36,18,#15,1,2
PC.IMAGE in	組み込み画像を描く。	PC.IMAGE 2

PC.VIDEO os	ビデオ出力をON/OFF[1/0]する。	PC.VIDEO 0
PC.SSTART in	スプライト処理を開始する。 in には背景に使う組み込み画像番号を指定する。 背景をベタ塗りにする場合は in のHighBitsを1にする。[ex. #12 == 背景赤] スプライト処理を開始するとLINEやIMAGEなどは使えない。 スプライト処理を停止するには in を #FF にする。	PC.SSTART #03
PC.SCREATE sn,si	スプライトを作る。 スプライトは32枚用意されている。 sn はスプライト番号 0～31。 si には組み込みスプライト画像番号を指定する。 スプライトを消すには si を #FF にする。 スプライトは番号が大きい方が上に重なる。	PC.SCREATE 0,#15
PC.SMOVE sn,px,py	スプライトを移動する。 sn は "SCREATE" した番号。 px,pyは座標、スプライトの左上が起点。	PC.SMOVE 0,16,16
PC.SFLIP sn,fs	スプライトを左右反転する。 sn は "SCREATE" した番号。 fs は ON/OFF[1/0]。	PC.SFLIP 0,1
PC.SROTATE sn,ra	スプライトを回転する。 sn は "SCREATE" した番号。 ra は 角度[0:0°,1:-90°,2:180°,3:90°]。	PC.SROTATE 0,1
PC.SUSER sn,tc,"cc...c"	スプライトを自作する。(15個) sn は 番号[#F0～#FE]。 tcには透明色を指定する。 cc...cには左上から右方向への色番号を羅列する。	PC.SUSER #FE,2,"2282 82822228882222228222 22022220202220222222 02222202222022222022"
PC.SOUND o0,s0,o1,s1,o2,s2,o3,s3	4ch同時に音を鳴らす。 o0～o3 はオクターブ(0～4～7)、s0～s3 は音程(0～b)。 音程eはノイズ音。 s0～s3 のHighBits4は音色(0～3)。 音を消すには s0～s3 を #FF にする。	PC.SOUND 4,#00,4,#04, 4,#07,4,#FF PC.SOUND 4,#20,4,#24, 4,#27,4,#FF

PC.SOUND1 cn,on,sn	1chのみ音を鳴らす。	PC.SOUND1 0,4,#07
PC.MSCORE ch, pn,tt,”mm”	MMLをサウンドchに登録する。 ch: チャンネル(0～3) pn: 1:即再生、0:後で再生 tt: テンポ(0～F)+音色 [ex. #30 == テンポ3、音色0] mm: MML(MAX:バイナリ変換後64byte)	PC.MSCORE 0,1,#30, ”$CDEFGAB>C”
PC.MPLAY ss{,ch}	音楽再生をSTART/STOP[1/0]する。 “PC.MSCORE" または ”PC.MLOAD" コマンドで4ch分のMMLを先に登録して置き、一気に再生可能。 ch(0～3)を指定すると、そのチャンネルだけ制御できる。	PC.MPLAY 1 PC.MPLAY 1,0
PC.MLOAD ch,mn	組み込みMMLをサウンドchに登録する。 mn に組み込みMML番号(0～3)を指定する。 サウンドchを初期状態にしたい場合は mn を #FF にする。	PC.MLOAD 0,1
PC.RESET	PanCakeを初期状態に戻す。 スプライトや音などがすべて初期状態に戻る。	PC.RESET
PC.OUT pf	OUTポートに出力する。 pfにはONにするポート番号を指定する。	PC.OUT `10100000
PC.BPS rrrr	通信速度を設定する。(Default:115,200) rrrrには速度を指定する。 (0で初期値の115,200bps、-1:57600bps、-2:38400bps)	PC.BPS 9600 PC.BPS -1
PC.WBUF on	WバッファモードをON/OFF[1/0]する。 ON時に再びONすることでバッファを入れ替えることができる。 例 PC.WBUF 1	PC.WBUF 1

PanCake REFERENCE

通常のPanCakeのように、uartでコマンドを送ることもできます。

●Text-Command

- ASCII文字列でコマンドを投げてください。
- 各命令は固定長、大文字、最後にLFが必要、解析可能な命令長は MAX:96(byte) です。

コマンド	解説	例
PANCAKE CLEAR cn	画面を消去します。cnには色番号を16進数で指定します。	PANCAKE CLEAR 0A
PANCAKE LINE x1 y1 x2 y2 cn	線を引きます。	PANCAKE LINE 00 00 10 0A 01
PANCAKE CIRCLE xc yc ra cn	円を描きます。 xc,ycには中心座標を指定します。raには半径を指定します。	PANCAKE CIRCLE 10 10 10 01
PANCAKE STAMP px py tc cc...c	8x8の絵を描きます。px,pyには出力座標を指定します。tcには透明色を指定します。 cc...cには左上から右方向への色番号を羅列します。	PANCAKE CIRCLE 00 08 02 2282828222288822...
PANCAKE STAMP1 px py cn cccccccccccccccc	8x8の一色絵を描きます。px,pyには出力座標を指定します。cnには色を指定します。 ccccccccccccccccには左上から右方向への0,1を羅列します。0は透明になります。	PANCAKE STAMP1 00 08 02 FF00FF00FF00FF00
PANCAKE STAMPS px py si [fs] [ra]	px,pyには出力座標を指定します。si には組み込みスプライト画像番号を指定します。 fs を指定すると左右反転の制御ができます。ON/OFF[01/00] (Default:00) ra を指定すると角度の制御ができます。[00:0°,01:-90°,02:180°,03:90°] (Default:00)	PANCAKE STAMPS 24 12 15 PANCAKE STAMPS 24 12 15 01 02
PANCAKE IMAGE in	組み込み画像を描きます。	PANCAKE IMAGE 02
PANCAKE VIDEO os	ビデオ出力をON/OFF[01,00]します。	PANCAKE VIDEO 00
PANCAKE SPRITE START in	スプライト処理を開始します。in には背景に使う組み込み画像番号を指定します。 背景をベタ塗りにする場合は in のHighBitsを1にします。[ex. 12 == 背景赤] スプライト処理を開始するとLINEやIMAGEなどは使えません。 スプライト処理を停止するには in を FF にします。	PANCAKE SPRITE START 03
PANCAKE SPRITE CREATE sn si	スプライトを作ります。スプライトは16枚用意されています。 sn はスプライト番号 0～15 です。si には組み込みスプライト画像番号を指定します。 スプライトを消すには si を FF にします。スプライトは番号が大きい方が上に重なります。	PANCAKE SPRITE CREATE 00 15
PANCAKE SPRITE MOVE sn px py	スプライトを移動します。 sn は CREATE した番号です。px,pyは座標、スプライトの左上が起点です。	PANCAKE SPRITE MOVE 00 10 10
PANCAKE SPRITE FLIP sn fs	スプライトを左右反転します。sn は CREATE した番号です。fs はON/OFF[01/00]です。	PANCAKE SPRITE FLIP 00 01
PANCAKE SPRITE ROTATE sn ra	スプライトを回転します。 sn は CREATE した番号です。ra は角度[0:0°,1:-90°,2:180°,3:90°]です。	PANCAKE SPRITE ROTATE 00 01
PANCAKE SPRITE USER sn tc cc...c	スプライトを自作します。（2個） sn は番号[FD,FE]です。 tc には透明色を指定します。cc...c には左上から右方向への色番号を羅列します。	PANCAKE SPRITE USER FE 02 2282828222288822...
PANCAKE SOUND o0 s0 o1 s1 o2 s2 o3 s3	4ch同時に音を鳴らします。o0～o3 はオクターブ(0～4～7)、s0～s3 は音程(0～b)です。 音程eはノイズ音です。s0～s3 のHighBits4は音色(0～3)です。 音を消すには s0～s3 を FF にします。	PANCAKE SOUND 04 00 04 04 04 07 04 FF PANCAKE SOUND 04 20 04 24 04 27 04 FF
PANCAKE SOUND1 cn on sn	1chのみ音を鳴らします。	PANCAKE SOUND1 00 04 07
PANCAKE MUSIC SCORE ch pn tt mm	MMLをサウンドchに登録します。ch: チャンネル(00～03)　pn: 01:即再生、00:後で再生 tt: テンポ(0～F)+音色 [ex. 30 == テンポ3、音色0] mm: MML(MAX:バイナリ変換後64byte)	PANCAKE MUSIC SCORE 00 01 30 $CDEFGAB>C
PANCAKE MUSIC PLAY ss [ch]	音楽再生をSTART/STOP[01/00]します。 "MUSIC SCORE"または"MUSIC LOAD"コマンドで4ch分のMMLを先に登録して置き、 一気に再生可能です。ch(00～03)を指定すると、そのチャンネルだけ制御できます。	PANCAKE MUSIC PLAY 01 PANCAKE MUSIC PLAY 01 00
PANCAKE MUSIC LOAD ch mn	組み込みMMLをサウンドchに登録します。mn に組み込みMML番号(00～03)を指定します。 サウンドchを初期状態にしたい場合は mn を FF にします。(from 1.1)	PANCAKE MUSIC LOAD 00 01
PANCAKE RESET	PanCakeを初期状態にもどします。スプライトや音などがすべて初期状態に戻ります。	PANCAKE RESET
PANCAKE OUT pf	OUTポートに出力します。pfにはONにするポート番号を16進数で指定します。	PANCAKE OUT FF
PANCAKE BPS rrrr	通信速度を設定します。（Default:115,200） rrrr には速度を16進数で指定します。0000を指定していると115,200になります。	PANCAKE BPS 2580

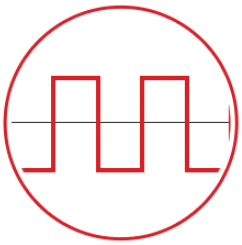
●Binary-Command

- テキストコマンドに対応したバイナリコマンドもあります。詳しくは http://pancake.shizentai.jp/ または FacebookのPanCake-FAN をご参照ください。

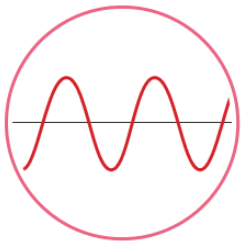
PALETTE

0	黒	0. 0. 0	4	橙	242. 126. 48	8	黄緑	145. 202. 24	c	濃紺	026. 036. 102
1	白	255. 255. 255	5	黄	255. 203. 61	9	緑	28. 77. 55	d	青紫	099. 055. 187
2	赤	227. 27. 41	6	薄茶	255. 222. 169	a	水	67. 175. 215	e	赤紫	178. 063. 171
3	桃	255. 104. 139	7	茶	107. 74. 43	b	青	38. 74. 208	f	灰	204. 204. 204

TIMBRE



0　方形波
Square wave



1　正弦波
Sine wave

2　エイティーズ
80's

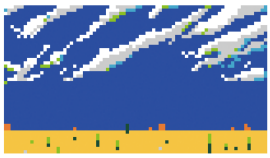
3　ヴァイオリン
Violin

IMAGE

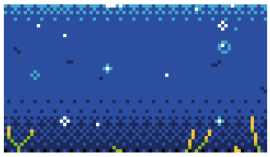
00

01

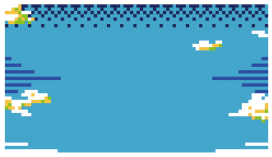
02



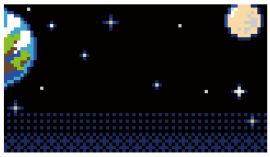
03



04



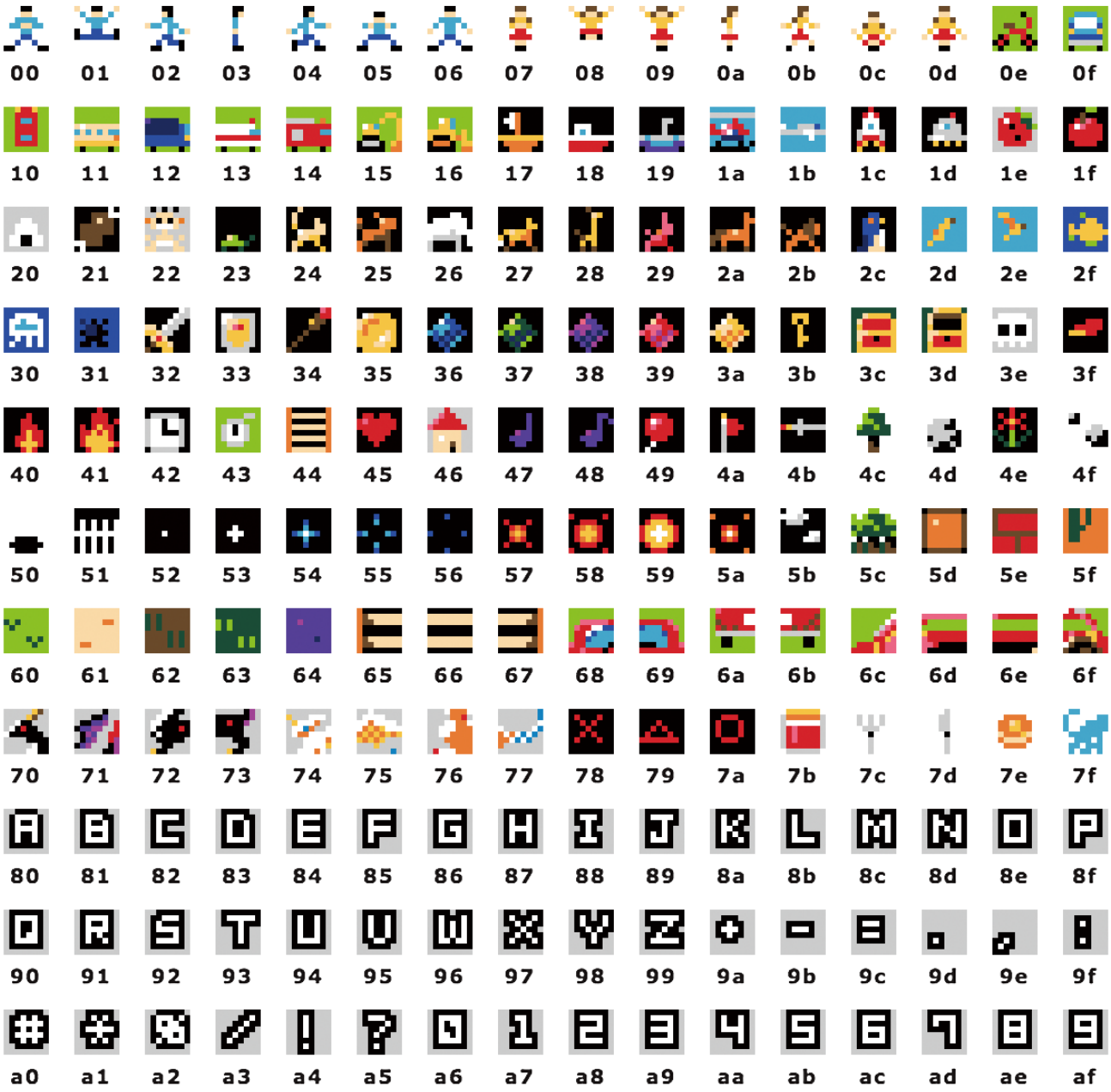
05



06

07

SPRITE


00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f
20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f
30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f
50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f
60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f
70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e 7f
80 81 82 83 84 85 86 87 88 89 8a 8b 8c 8d 8e 8f
90 91 92 93 94 95 96 97 98 99 9a 9b 9c 9d 9e 9f
a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad ae af

通常のPanCakeとの違い

・スプライトを拡張しました
　使用できるスプライト数：32枚
　自作できるスプライト数：15枚

・ダブルバッファリングができます

・別売りのEEPROMを使って、画像と音楽が追加変更できます

通常のPanCakeに加えて、以下のコマンドが追加されています。

●Text-Command
- ASCII文字列でコマンドを投げてください。
- 各命令は固定長、大文字、最後にLFが必要、解析可能な命令長は MAX:96(byte) です。

コマンド	解説	例
PANCAKE WBUF on	Wバッファモードを ON/OFF[01/00]します。 ON時に再びONすることでバッファを入れ替えることができます。 (from 1.2C)	PANCAKE WBUF 01

つくろう！ダンブンゲーム for Iam

```
1 ´DANBUN KURUKURU
10 PC.RESET:WAIT5
20 PC.SSTART #1B
30 PC.SCREATE 0,#22
40 PC.SMOVE 0,36,20
50 R=0:S=0
60 PC.SROTATE 0,R
70 WAIT15
80 IF INKEY()=0 GOTO110
90 IF R=0 S=S+1 ELSE S=S-1
100 ?"SCORE: ";S
110 R=(R+1)%4
120 WAIT1:GOTO60
```

■遊び方
ダンブンが立っているときを狙って
タイミングよくキーボードを押そう！

■改造のヒント
WAIT15 を WAIT 10*RND(10) にしてみよう。
ダンブンのまわる速度がランダムになるよ！

つくろう！ダンブンゲーム for I.a.m

1

```
1 ´DAI GYAKUSO
10 PC.RESET:WAIT5
20 PC.MSCORE 0,0,#30,
"$>ECECECECEDCDC<G>FC"
30 PC.MSCORE 1,0,#32,"$<CRCRCRCR
<BRBRBRBRARARARARARBRBRBRBR"
40 PC.MSCORE 2,0,#33,
"$>>GRGRGRGRGFEFECAG"
50 PC.MPLAY 1
60 PC.SSTART 2
70 PC.SCREATE 0,#10
80 PC.SCREATE 1,#0F
90 PC.SCREATE 2,#A6
100 PC.SMOVE 2,0,0
110 PC.SCREATE 3,#A6
120 PC.SMOVE 3,8,0
130 A=0:X=RND(2):Y=0:Z=2:T=0:S=0
140 K=INKEY
150 IF K=28 A=0:PC.MSCORE 2,1,
#02,"GAGA"
160 IF K=29 A=1:PC.MSCORE 2,1,
#02,"GAGA"
170 PC.SMOVE 0,25+23*A,36
180 PC.SMOVE 1,25+23*X,Y
190 IF (A=X)*(27<Y) GOTO500
```

2

```
200 IF 45<Y X=RND(2):Y=0
210 Y=Y+Z:T=T+1:S=T/10
220 PC.SCREATE 2,S/10+#A6
230 PC.SCREATE 3,S%10+#A6
240 WAIT2:GOTO140
500 PC.MPLAY 0
510 PC.MSCORE 2,1,#30,"<ERE~"
```

■遊び方
← → キーで車を動かして
対向車をよけよう！

■改造のヒント
Z=2 を Z=4 にしてみよう。
対向車のスピードが上がるよ！

1

```
100 'LINGO WO SACCHAN
110 X=36:F=0:B=X+7:M=RND(72):
N=0:R=0:S=0:A=1
120 PC.RESET
130 PC.MSCORE 0,0,#81,"$C>C<EG"
140 PC.MSCORE 1,0,#80,
"RRRRR$RRRRRRRRCCDE~~~~EFED~~~~
DEDC~~~~"
150 PC.MPLAY 1
160 PC.SSTART 1
170 PC.SCREATE 0,#0B
180 PC.SCREATE 2,#5D
190 PC.SCREATE 1,#1F
200 PC.SCREATE 3,#A6
210 PC.SCREATE 4,#A6
220 PC.SMOVE 3,0,0
230 PC.SMOVE 4,8,0
240 PC.SMOVE 0,X,35
250 PC.SFLIP 0,F
260 PC.SMOVE 2,B,35
270 PC.SFLIP 2,F
280 PC.SMOVE 1,M,N
290 PC.SROTATE 1,R
300 PC.SCREATE 3,#A6+S/10
310 PC.SCREATE 4,#A6+S%10
```

2

```
320 K=INKEY()
330 IF K=28 X=X-4:F=1:B=X-7
340 IF K=29 X=X+4:F=0:B=X+7
350 R=(R+1)%4:N=N+RND(A+1)
360 IF (35<N)*(B-4<M)*(M<B+4)
S=S+1:PC.MSCORE 3,1,#22,">>CE>C"
370 IF (35<N)*(X-4<M)*(M<X+4)
GOTO 400
380 IF 35<N M=RND(72):N=0:
A=(S+10)/10
390 GOTO 240
400 PC.MPLAY 0
410 PC.MSCORE 3,1,#40,
"<F~ED~<B>C~~~"
```

■遊び方
← → キーでさっちゃんを動かしてりんごを
カゴに入れよう！りんごがさっちゃんに当たると
ゲームオーバーだよ。

■改造のヒント
PC.SCREATE 1,#1F を
PC.SCREATE 1,#20 にしてみよう。
りんごがおにぎりになるよ！