IchigoJam BASIC reference ver 1.2 (extracted)

basic commands

command	description	example
LED num	light on the LED when n equals 1, light off when n equals 0	LED 1
WAIT num{,num2}	wait n frames (60frame = 1sec) (if num2 eqauls 0 low power consumption mode, if num1 is minus short wait mode -261 same as WAIT1)	WAIT 60
:	series the commands	WAIT 60:LED 1
RUN	execute program in memory [F5]	RUN
LIST {linenum1{,linenum2}}	show the program in memory [F4] (line num1: show the line, if minus to the line / line num2: show to the line, if 0 to the end / ESC to stop)	LIST 10,300
GOTO linenum	change the execution line (it's OK as using variables)	GOTO 10
END	end this program	END
IF num {THEN} command1 {ELSE command2}	if num does not equals 0 execute command1, else execute command2 (you can ommit THEN / ELSE)	IF BTN() END
BTN({num})	return 1 if you push the botton, else 0 (num:0(embeded button)/UP/DOWN/RIGHT/LEFT/SPACE, 0:no num)	LED BTN()
NEW	delete all program in the memory	NEW
PRINT {num or strings}	write the letter or nunmber to the screen (strings must surround ["], connect pars with [;]) Abbreviation:?	PRINT "HI!"
LOCATE x,y	set the position to write (if y equals -1 no write mode) Abbreviation:LC	LOCATE 3,3
CLS	clear the screen	CLS
RND(num)	return the random number 0 to num - 1	PRINT RND(6)
SAVE {num}	save the program (num:0-3, 100-227:optional EEPROM, if omit using number)	SAVE 1
LOAD {num}	load the program (num:0-3, 100-227:optional EEPROM, if omit using number)	LOAD
FILES {num1{,num2}}	show the file list from num1 to num2 (all if num1 equals 0, ESC to stop)	FILES
BEEP {num1{,num2}}	sound the BEEP, num1 is period(1-255), num2:length(1/60sec) (you can omit num1 and num2) *to connect the sounder on SOUND(EX2)-GND	BEEP
PLAY {mml}	play the music specified mml as MML(Music Macro Language) just PLAY to stop the music *to connect the sounder on SOUND(EX2)-GND	PLAY "$CDE2CDE2"
x + y	return x plus y	PRINT 1+1
x - y	return x minus y	PRINT 2-1
x * y	return x times y	PRINT 7*8
x / y	return integer of x divide y	PRINT 9/3
x % y	return reminder of x divide y	PRINT 10%3
(num)	return calculate the number in priority	PRINT 1+(1*2)
LET var,num	set the number to 1 letter of alphabet as named memory(variable) (series put to the array) Abbreviation:var=num	LET A,1
INPUT strings,var	set the number to var from keyboard input (you can omit strings and comma)	INPUT "ANS?",A
TICK()	return the time count from CLT(count up in 1/60sec)	PRINT TICK()
CLT	clear the time count	CLT
INKEY()	return from keyboard or UART (0:no input, #100:0 input from UART)	PRINT INKEY()
CHR$(num)	In PRINT, return the letter string specified the num (you can set series with comma)	PRINT CHR$(65)
ASC("string")	return the letter code from string	PRINT ASC("A")
SCROLL num	scroll the screen (0/UP:up、1/RIGHT:right、2/DOWN:down、3/LEFT:left)	SCROLL 2
SCR({x,y})	return the letter code located x, y on the screen (if omit x and y, using current position) Alias:VPEEK	PRINT SCR(0,0)
x = y	return 1 if x equals y else 0 (Alias:==)	IF A=B LED 1
x <> y	return 1 if x does not equal y else 0 (Alias:!=)	IF A<>B LED 1
x <= y	return 1 if x <= y, else 0	IF A<=B LED 1
x < y	return 1 if x < y else 0	IF A<B LED 1
x >= y	return 1 if x >= y else 0	IF A>=B LED 1
x > y	return 1 if x < y else 0	IF A>B LED 1
x AND y	return 1 if x and y else 0 (Alias:&&)	IF A=1 AND B=1 LED 1
x OR y	return 1 if x or y else 0 (Alias:||)	IF A=1 OR B=1 LED 1
NOT x	return 1 if x equals 0 else 0 (Alias:!)	IF NOT A=1 LED 1
REM	not execute after this command (comment) Abbreviation:'	REM START
FOR var=num1 TO num2 {STEP	set num1 to var, execute the loop to the NEXT until var reach num2 by	FOR I=0 TO 10:?I

num3} / NEXT	step num3 (you can omit STEP, nest limit:6)	
IN({num})	return 1 if when input terminal pin is high else 0 (num:0-11 (IN0/1/4/9 pull up, IN5-8,10-11:if switched, IN0,9:button), you can get all states when you omit num)	LET A,IN(1)
ANA({num})	return the value 0-1023 specified voltage of input terminal (2:IN2, 5-8:IN5-8(OUT1-4), 0,9:BTN, 0:omitted)	?ANA()
OUT num1{,num2}	output num2 to the output port specified num1 (num1:OUT1-11, you can set all states when you omit num2, if num2 equals -1 the output pot switch into the input port)	OUT 1,1
PWM num1,num2{,num3}	output num2(0.01msec) length pulse in num3(if omit 2000) period to the output port specified num1 (num1:OUT2-5, OUT2-4 same period)	PWM 2,100

senior commands

command	description	example
CLV	clear (set to zero) variables and array variables Alias:CLEAR	CLV
CLK	clear key buffer and key status	CLK
CLO	initialize the input and output pins	CLO
ABS(num)	return the absolute value	?ABS(-2)
[num]	array variables (from [0] to [101] 102 series variables) you can set in series using LET[0],1,2,3	[3]=1
GOSUB linenum / RETURN	move to linenum and execute after this command when RETURN Abbreviation:GSB/RTN (nest limit:30)	GOSUB 100
DEC$(num1{,num2})	In PRINT, return strings from num1 with beam specified num2 (you can omit num2)	?DEC$(99,3)
HEX$(num1{,num2})	In PRINT, return hexadecimal strings from num1 with beam specified num2 (you can omit num2)	?HEX$(255,2)
BIN$(num1{,num2})	In PRINT, return binary number strings from num1 with beam spcified num2 (you can omit num2)	?BIN$(255,8)
x & y	return x logical and y (bit calculation)	?3&1
x | y	return x logical or y (bit calculation)	?3|1
x ^ y	return x logical exclusive or y (bit calculation)	?A^1
x >> y	return x shift down y bits (bit calculation)	?A>>1
x << y	return x shift up y bits (bit calculation)	?A<<1
~x	return bit inverted x (bit calculation)	?~A
STOP	stop the program	STOP
CONT	continue the same line or stop line	CONT
SOUND()	return 1 if sound playing else 0	?SOUND()
FREE()	return free memory of program (up to 1024 bytes)	?FREE()
LRUN {num}	LOAD num and RUN	LRUN 1
FILE()	return the number of last using FILE	?FILE()
LINE()	return the line number of last execution	?LINE()
SRND num	initialize the seed of random/td>	SRND 0
PEEK(num)	read 1 byte number from the memory in address specified num2	?PEEK(#700)
POKE num1,num2	write 1 byte num1 specified to the memory in address specified num2	POKE #700,#FF
COPY num1,num2,num3	memory copy from num1 to num2 length specified num3 (if num3 is minus, copy direction is inverted)	COPY #900,0,256
CLP	initialize the character pattern memory (#700-#7FF)	CLP
STR$(num1{,num2})	In PRINT, return strings from address specified num1 (num2:length you can omit)	PRINT STR$(A)
LEN("strings")	return the length of strings	PRINT LEN("ABC")
RESET	reboot the IchigoJam	RESET
SLEEP	sleep the IchigoJam (after push the button, execute and run file 0)	SLEEP
UART num1{,num2}	set UART mode (num1 0:off 1:with PRINT 2:with PRINT/LC/CLS/SCROLL 3:with PRINT and enter code is \r\n, initial value:2) (num2-0:UART recv off 1:on initial value:1)	UART 0
BPS num	set UART speed (0:115,200bps -1:57600bps -2:38400bps or 9600 and so on ... initial value:0)	BPS 9600
I2CR(num1,num2,num3,num4,num5)	read from I2C device num1:I2C address, num2:address of command, num2:length of command, num4:address of return data, num5:length of return data	R=I2CR(#50,#700,2,#702,2)
I2CW(num1,num2,num3,num4,num5)	write to I2C device num1:I2C address, num2:address of command, num2:length of command, num4:address of return data, num5:length of return data	R=I2CW(#50,#700,2,#702,2)
USR(address,num)	call program written in machine language (easy to freeze the IchigoJam)	A=USR(#700,0)