Using the Amigo: A Summary of Color BASIC 2.2L Commands

Getting Started

Note	Most commands in Color BASIC can be included in a program line (to be executed when the program is run) or entered at the READY prompt and executed immediately. Program lines in Color BASIC must be numbered (1 to 65535) and will be executed in numerical order. Color BASIC is a tinyBASIC for the Propeller chip, with 32-bit integer arithmetic and only 26 variables, A through Z, case insensitive. It does not directly support strings or arrays and has no data statement, but it is powerful enough in its own right to be an excellent tool for parents and grandparents to mentor their beginning programmers.
RUN	Executes the program currently stored in program memory. Can also be used as **RUN** “name.bas” to **LOAD** and **RUN** a program in one step.
<Esc>	Halts the program currently running.
LIST	Copies the current program to the screen. Each program line must begin with a number, and program lines will be listed and executed in numerical order. To delete a program line, type just its number at the flashing prompt and press <Enter>.
DIR	Lists a directory of files on the SD card. Only shows those files in the root directory.
SAVE “name.bas”	Saves the program currently in program memory to the SD card, with the name and extension inside the quotes.
LOAD “name.bas”	Copies the program “name.bas” from the SD card to program memory.
DELETE “name.bas”	Deletes the file “name.bas” from the SD card.
NEW	Clears program memory; sets the 26 Color BASIC variables to 0.
HOLD	Saves the program currently in program memory to the SD card, with the name of DESKTOP.TMP. Color BASIC and the editor are separate programs and pass this DESKTOP.TMP file back and forth to open and close the editor.
REBOOT	Reboots the Amigo, just like pressing the reset button or interrupting power. If there is a file named DESKTOP.TMP in the root directory of the SD card, loads and runs that file. Otherwise boots up to the READY Color BASIC prompt.

<table>
<tr><th colspan="2">Using the Amigo: A Summary of Color BASIC 2.2L Commands</th></tr>
<tr><td><F1></td><td>Opens the Color BASIC editor, which allows use of the arrow, Home, End, Insert, and Delete keys to edit programs. Returns to Color BASIC and runs the program when <F1> is pressed again. Programs BASIC.BIN and BASEDIT.BIN must be present on the SD card for the editor to function.</td></tr>
<tr><td colspan="2"><b>Screen Commands</b></td></tr>
<tr><td><b>CLS</b></td><td>Clears the screen with the current background color; moves cursor to top left corner.</td></tr>
<tr><td><b>COLOR</b> f,b</td><td>Sets the foreground and background colors for subsequent screen commands. Color BASIC has 64 colors (0-63).</td></tr>
<tr><td><b>LOCATE</b> c,r</td><td>Moves the cursor to the specified text column (0-49) and row (0-36).</td></tr>
<tr><td><b>PRINT</b></td><td>Prints the value of a variable, or a string enclosed in quotes, at the current cursor location, in the current foreground and background colors. Multiple variables and strings can be included in one PRINT statement, separated by commas and/or semicolons.</td></tr>
<tr><td><b>DISPLAY</b> n</td><td>Displays the glyph currently assigned to the character with ASCII value n, at the current cursor location. Multiple values can be included in the same DISPLAY command, separated by commas. The Color BASIC character set includes some graphics characters, and the glyph of any printable character can be customized using the REDEFINE command.</td></tr>
<tr><td><b>REDEFINE</b> n, a,b,c,d,e,f,g,h</td><td>Changes the glyph of ASCII character n to the 8x8 pixel map defined by a-h, which are 8-bit integers. Characters on the screen assume their new glyph immediately upon redefinition.</td></tr>
<tr><td><b>PLOT</b> x,y,c</td><td>Places a small square (one quarter of a text cell, or 4x4 pixels) in color c at graphics column x (0-99) and graphics row y (0-74). Only one foreground color is allowed in any text cell, so plotting a square of a different color in the same text cell will change the color of any other squares in that text cell.</td></tr>
<tr><td><b>BOX</b> u,v,x,y,c</td><td>Places a box on the screen in color c with the top left corner at graphics coordinates u,v and the bottom right corner at x,y.</td></tr>
<tr><td><b>LINE</b> u,v,x,y,c</td><td>Draws a line in color c between graphics coordinates u,v and x,y.</td></tr>
<tr><td colspan="2"><b>Keyboard Commands</b></td></tr>
</table>

Using the Amigo: A Summary of Color BASIC 2.2L Commands	
INKEY	Checks the keyboard buffer for a keystroke and returns the assigned value of any key pressed, or 0 otherwise. Usually used to the right of the assignment operator, as in A=INKEY.
INPUT “prompt”; a,b,c	Prints the “prompt”, if any, at the current cursor location, then waits for the user to input the prompted number of integers, separated by commas, then press <Enter>. Generates a syntax error if the number of variables input by the user does not equal the number expected. Note that Color BASIC does not handle strings, and this capability is for integers only.
Program Control Commands	
GOTO xxx	Directs program flow to jump to the instruction in Line xxx.
GOSUB xxx … **RETURN**	Directs program flow to the subroutine at Line xxx, then back to the calling routine on the RETURN command in the subroutine. All 26 variables in Color BASIC are shared, so arguments are not explicitly passed and subroutine variables are not private.

IF expression
THEN command
	Checks to see whether the expression is TRUE (non-zero), and if it is executes the command(s) following THEN. Color BASIC does not include ELSE or ENDIF constructs. Multiple commands can be concatenated after the THEN by separating them with a colon.

PAUSE nnnn	Pauses program execution for nnnn milliseconds.
REM	Allows insertion of a textual remark after the REM for internal documentation. May be used on its own program line, or (generally, but not always) at the end of program line containing other commands.
END	Terminates program execution and returns to the flashing cursor prompt to await user command. END is not required, but is useful to separate the main body of a program from following subroutines.
STOP	Like END, terminates the program and returns to the flashing cursor. Sometimes used during program development, or to indicate an abnormal program halt.
Color BASIC Operators	

Using the Amigo: A Summary of Color BASIC 2.2L Commands

Note	Operators in Color BASIC generally take the form a <operator> b, where a and b are 32-bit integers or expressions that reduce to integers. Expressions are evaluated from left to right, parentheses first, according to this order of precedence (highest first): 9 - **SHL**, **SHR**, **ROL**, **ROR**, **SAR**, **REV** (Bitwise shifts) 8 - **&** (Bitwise logical AND) 7 - **|** (Bitwise logical OR) 6 - * , **/** , **//** (Integer multiplication, division, and remainder) 5 - **+** , **-** (Integer addition and subtraction) 4 - **>**, **<**, **=**, **>=**, **<=**, **<>** (Comparison operators) 3 - **NOT** (Logical NOT) 2 - **AND** (Logical AND) 1 - **OR** (Logical OR)
Bitwise Operators	Color BASIC bitwise operators include bitwise logical and (**&**), or (**|**), shift left (**SHL**), shift right (**SHR**), rotate left (**ROL**), rotate right (**ROR**), arithmetic shift right (**SAR**), and bit reverse (**REV**). **SAR** preserves the sign bit (Bit 31 keeps its original value) and discards Bit 0 each shift. **REV** reverses the order of the specified number of least significant bits and sets the rest to zero. Bitwise operations return a 32-bit integer.
Arithmetic Operators	Color BASIC supports 32-bit integer arithmetic, including operators for addition (**+**), subtraction (**-**), multiplication (*), integer division (**/**), and modulo or remainder (**//**). Arithmetic operations return a 32-bit integer value.
Comparison Operators	Comparison operators include greater than (**>**), less than (**<**), equals (**=**), greater than or equal to (**>=**), less than or equal to (**<=**), and not equal to (**<>**). Comparison operations return either -1 (the comparison is TRUE) or 0 (the comparison is FALSE).
Logical Operators	Color BASIC logical operators include **NOT**, **AND**, and **OR**. Logical operations also return either a -1 or a 0 value.

Audio Commands and Parameters

NOTEON nn	Sends a continuing tone of value nn to the audio connector; useful values seem to range from 0 to 100. A powered speaker works best with the Amigo, but earbuds will do in a pinch.
NOTEOFF	Stops any tone currently playing on the audio connector.
PLUCK nn	Sends a note (not a continuing tone) to the audio channel. PLUCK 40 is close to middle C, and consecutive integers will play the notes of adjacent white and black keys on a piano keyboard.

Using the Amigo: A Summary of Color BASIC 2.2L Commands	
Wii Classic Controller Commands	
JOY	Returns the value of any button pressed on a Wii® Classic Controller (not the nunchuck) attached to the Amigo, or zero if no button is pressed. If two or more buttons are pressed at the same time, JOY returns the sum of their assigned values. The **a** and **b** buttons always override any other buttons, with **a** always taking priority over **b**. Color BASIC does not recognize the joysticks on the Wii controller, just the buttons, and the controller must be connected to the Amigo before the system is started. Use on the right side of the assignment operator, as in A=JOY.
File System Commands	
OPEN "filename", mode	Opens a file on the SD card in the desired mode (either R, W, or A — for Read, Write, or Append). Files opened in Read mode will not be modified by any program actions. Files opened in Write mode may have any or all of their content modified. Files opened in Append mode may be modified only by adding content at the end of their current content.
WRITE a,b,c	Stores the values of the specified variables into a file opened for writing or appending. You should not include other Color BASIC commands (including REM) on the same program line as WRITE.
READ a,b,c	Places data from a file opened for reading into the specified variables. The number of variables specified must not exceed the number of values in that line of the data file. (Otherwise Color BASIC returns a syntax error.) Note that the the numerical values to be retrieved by READ from the file cannot be negative; Color BASIC will interpret this as an End of File condition. Also note that you should not include other Color BASIC commands (including REM) on the same program line as READ.
FILE	Reads or writes the next byte in the currently open file on the SD card. VARIABLE = FILE reads a byte from a file open in the Read mode. FILE = <expression> writes a byte to a file in the Write or Append mode.
CLOSE	Closes the file currently open, if any. Color BASIC can have only one file open at any given time.
SRAM Commands	
PEEK nnnnn	Reads data from SRAM. Returns the 8-bit byte value stored at SRAM address nnnnn (0 to 32767). Often used on the right of the assignment operator, as in x = PEEK nnnnn.

Using the Amigo: A Summary of Color BASIC 2.2L Commands	
POKE nnnnn,v	Writes to SRAM. Places the 8-bit byte value v at SRAM address nnnnn**.** If v is less than 0 or greater than 255, only the eight least significant digits will be saved.
CLEAR	Clears SRAM memory (0 to 32767) to zeros. Also occurs when the **REBOOT** command is executed, but not when **NEW** or **RUN** is issued. Since SRAM is not cleared between separate program runs, it can be used to transfer data between programs.
GPIO Commands	
INA [<pin>]	Reads the value of the voltage level (either a logic 1 or 0) on the specified input/output <pin> (0 through 31) of the Propeller chip. Used for interfacing Color BASIC programs to external circuits through the Amigo experimenter's section.
OUTA[<pin>] = n	Changes the specified input/output <pin> of the Propeller chip (0 through 31) to n, where n is either a logic 1 or 0. Used for interfacing programs to external circuits through the Amigo experimenter's section via Propeller pins 0 through 15.
Propeller Hardware Register Access	
CNT	Reads (only) the value of the Propeller system counter register. For the Amigo each tick of the system clock is 0.125 nanoseconds.
Note	The Propeller chip consists of eight identical processors (cogs), and each of these contains two programmable hardware counters. Color BASIC provides access to the three key registers (CTRx, PHSx, and FRQx) on both counters of one cog (Counter A and Counter B). This gives the Amigo control of two state machines that can perform a wide variety of hardware functions without continuing software involvement. See Parallax application note AN-001 on the Propeller chip for details.
CTRA / CTRB	Provides read/write access to the Control Register in cog Counter A or Counter B. CTRMODE (Bits 26..30) selects the counter mode of operation. PLLDIV (Bits 23..25) sets the PLL division factor for PLL modes. APIN (Bits 0..5) and BPIN (Bits 9..14) select the I/O pin(s) used for that mode of operation. (Both Counter A and Counter B have an APIN and a BPIN.)
PHSA / PHS B	Provides read/write access to the accumulator register in Counter A or Counter B.
FRQA / FRQB	Provides read/write access to the value-to-add register in Counter A or Counter B.

Using the Amigo: A Summary of Color BASIC 2.2L Commands	
Other Commands	
BRUN “filename”	Loads a binary file from the SD card to Amigo main memory and runs it. The loaded file completely replaces Color BASIC (which is itself a Propeller binary). Color BASIC is reloaded and run after “filename” terminates. This means you can write games or other programs in Spin for the Amigo Propeller platform and run them without having to re-flash the EEPROM.
BYTE [nnnnn]	Reads or writes an 8-bit value from / to the Amigo main memory location specified, where nnnnn is between 0 and 32767. The command can be used on either side of the “equals sign” assignment operator, as in A=BYTE[1921] (reading from main memory) and BYTE[1921]=65 (writing to it). Caution should be used when writing to main memory because this could corrupt the Color BASIC image and lead to unexpected results. If this happens, just reboot.
Note	The BYTE command can be used to read from or write to specific memory locations associated with the 50x37 Amigo screen display. You can use the formulas below to access or change the content, foreground color, and background color of any character cell on the Amigo display: Cell Content = BYTE [2931 - Y*50 - X] Foreground Color = BYTE [21449 + 2*X +100*Y] SHR 2 Background Color = BYTE [21448 + 2*X + 100*Y] SHR 2 where X is the screen column (0-49) and Y is the screen row (0-36).
LET <variable> = <expression>	Sets a variable equal to the value of an expression, as in LET x=5 or LET A = (X + Y) / 2. The use of LET is optional in Color BASIC.
LONG [nnnnn]	Like BYTE, but with a 32-bit value. The least significant two bits of the address are ignored.
MEM	Returns the amount of unused program memory, in bytes. The Amigo has 4K of program memory with no program loaded.
RND (n)	Returns a random integer between 0 and n - 1.

Using the Amigo: A Summary of Color BASIC 2.2L Commands	
SERIAL r,t,m,b	Initializes the Amigo serial communication channel, where r is the receive Propeller pin (default 31), t is the transmit pin (default 30), m is the mode (default 0), and b is the baud rate (between 300 and 115200). Generally placed at the beginning of a program, as in SERIAL 31,30,0,115200. On the Amigo, Propeller pins 30 and 31 are available via the PropPlug.
RX	Fetches a byte from the previously initialized serial receive pin and passes it to a variable, as in A=RX.
TX <byte>	Pushes a byte to the previously initialized serial transmit pin, as in TX A.
VER	Returns the version of Color BASIC loaded in the EEPROM of the Amigo, used on each boot up. This documentation is for version 2.2L.
WORD [nnnnn]	Like BYTE, but with a 16-bit value. The least significant bit of the address is ignored.
$, **%**	Color BASIC can interpret and process hexadecimal (prefixed with **$**) and binary (prefixed with **%**) values.