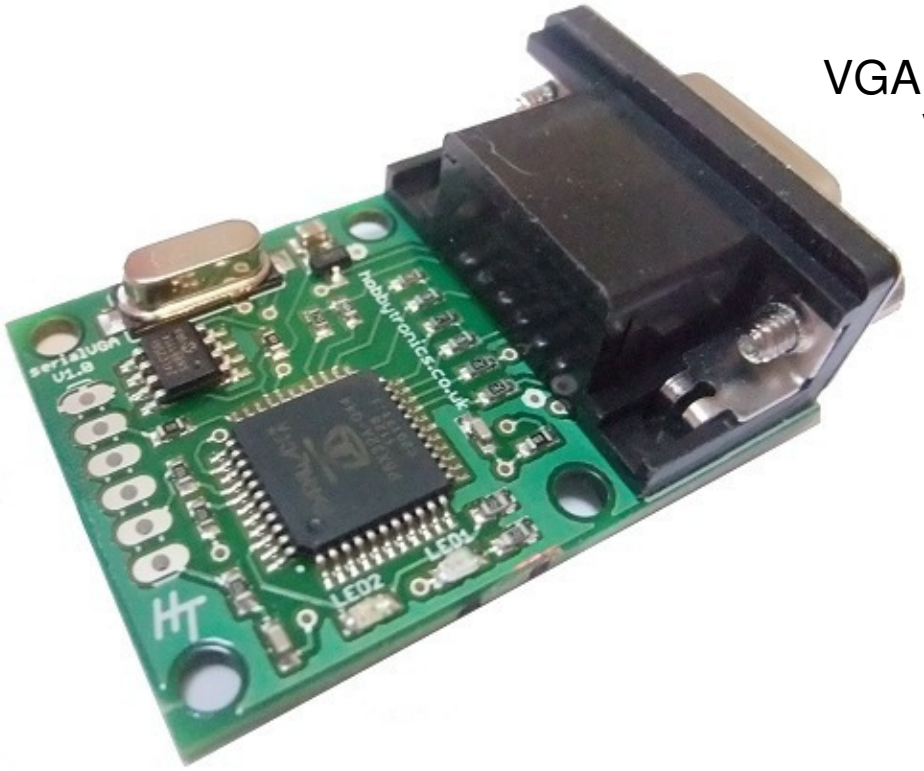
SerialVGA Board

Data Sheet



VGA driver board
Version 1.01

Introduction

The serialVGA board drives a VGA Monitor at 800 x 600 pixel resolution to allow you to display 100 characters by 50 rows of text. All control commands and text to be displayed are sent via a standard serial TTL connection at a selectable baud rate.

Additionally, you can create up to 9 individual "windows" in which to display information independently. Each window wraps and scrolls automatically, has optional titles and borders and allows placement of text at a fixed position within each window.

Through one simple serial TTL connection you can create a complex multi-window display to show a large amount of information.

Main Features

- Drive a VGA Monitor at 800x600 pixel resolution @ 75Hz
- 64 foreground / background colours
- Displays 100 characters x 50 lines
- Fully controlled via simple Serial TTL commands
- Baud rates 2400, 4800, 9600, 19200, 38400, 57600, 115200
- Use one full size window or create up to 9 mini windows each with optional borders and title
- Automatic text wrapping and scrolling within each window
- Set position of text within a window
- Can be used as a Parallax Propeller development board.

Board Overview

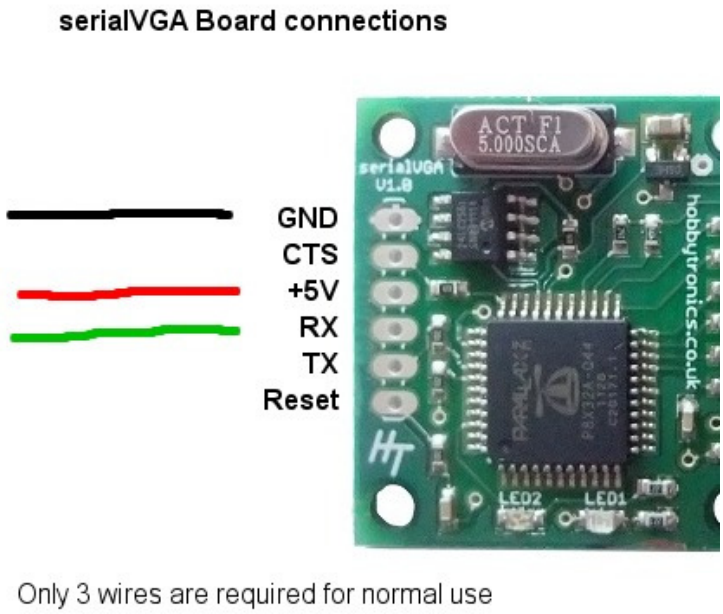
The serialVGA board uses a Parallax Propeller chip to create the VGA signals at the rate required by a monitor. A 24LC256 eeprom stores the software. The board is easily software updatable with newer versions of our software and can also be used as a Propeller development board if you are interested in creating your own Propeller VGA programs.

There are two LEDs on the board.

- A blue LED indicates power to the board
- An orange LED flashes briefly when data is received

Serial Connection

The six connection pins on the board are designed to mate with a Sparkfun FTDI Basic 5V board for ease of updating the board with new software versions and for configuring the baud rate.
Any serial TTL connection can be used though and the pinout is as follows



Only 3 wires are required for normal use

For receiving commands and data from a microcontroller only 3 connections are required, 0V, 5V and RX in. This free's up the RX pin on the microcontroller to be used as an input (e.g. from a GPS unit). This enables a microcontroller to read data in, process it and transmit it out to be displayed using only one serial port.

Command Set

Commands are sent to the VGA board using a 2 character command code followed by the command and its parameters in a comma delimited format. A carriage return character (\n) denotes the end of the command string.

The 2-character command code that prefixes all commands is ^[

The following commands are available

Command	Description
w	Create a window
f	Change which window has focus
e	Clear text in current window
p	Set text position in current window
c	Set text cursor on/off for a window
b	Set Baud rate
r	Reboot board
l	Set foreground / background line colours

Window command (w)

Use this command to create a window.

Optionally, windows can have borders and a title. The *window number* is the key to switching between windows for displaying text. Window numbers can only be a value between 1 and 9.

The serialVGA board needs a little time to process this command and a delay of 20ms is suggested before the next command or data is sent. As windows will generally be created once at the start of your program, this will have little impact on the running of your code.

```
^[w,window,left,top,width,height,border,title
```

Parameter	Description	Values
window	window number	1-9
left	left (x) position of start of window	0-99
top	top (y) position of start of window	0-49
width	width of window (including borders)	1-100
height	height of window (including borders)	1-50
border	Border on/off	1 = on, 0 = off
title	Text title (left aligned)	text

Example: Create a window (number 1) starting at the top left of the screen, the full width of the screen (100 chars) and 30 lines high. It has a border and title text.

```
^[w,1,0,0,100,30,1,Example window title
```

Window Focus (f)

This command is used to change the focus (i.e. where text is to be displayed) to a specific window. Window is the window number of a previously configured window.

```
^[f,window
```

Example: Change focus to window number 1

```
^[f,1
```

Clear (erase) window text (e)

This command clears the text area in the current window. It does not effect the borders or title area.

```
^[e
```

Example: Clear window 2 (set focus to window 2 first, if not current window)

```
^[f,2
^[e
```

Position (p)

Set the text cursor position within a window where text will start to be displayed. By default this is 0,0 when a window is created. The window position follows any text sent to the window, so if “hello” is sent the window position is now 5,0. If “hello\n” was sent instead, the window position would be 0,1.

The position command allows the cursor position to be set to anywhere within the window. This is very useful in displaying results in a fixed position.

You should ensure the window focus has been set if needed.

```
^[p,left,top
```

Example: Set position to beginning of second line

```
^[p,0,1
```

Cursor On/Off (c)

Turn on a text cursor for a given window

The text cursor is a flashing underscore that marks the current cursor position where text will be displayed. It is most useful when using a window as an input window (see "Input Windows" below)

```
^[c,window,on_off
```

where window is the window number, on_off is 1 for on, 0 for off. The default when a window is created is off

Example: Turn cursor on for window 3

```
^[c,3,1
```

Baud Rate (b)

Use this to change the baud rate that the serialVGA board communicates at. A reboot is required for the baud rate to take effect.

```
^[b,baud_rate
```

The default baud rate is 9600.

The board rate is stored in eeprom and will remain set at the value entered until it is either changed or a software upgrade is performed. A software upgrade will reset the baud rate back to 9600.

Baud rates available are

- 2400
- 4800
- 9600
- 19200
- 38400
- 57600
- 115200

Example: Set baud rate to 19200

```
^[b,19200
```

Reboot (r)

Use this command to reboot the board. A reboot clears all windows and data which will need to be setup again. A reboot requires 2 seconds to allow the board to boot up before sending any commands or data to it.

```
^[r
```

Line Colours (l)

Foreground and background colours can be set for each individual line of the display and can be changed at any time. It is not currently possible to have different colours within a line, i.e. two windows displayed side by side cannot have different colours.

If you don't set any colours, the default is green text on a black background.

There are 64 colours available for both foreground and background and they are expressed in a base 4 RGB format. So each colour value is made up of 3 numeric digits 0-3.

```
^[l,start_row,end_row,foreground,background
```

Rows start at 0 and end at 49

Here are some example colours.

000	Black
333	White
001	Dark Blue
010	Dark Green
100	Dark Red
011	Cyan
101	Purple

Example: Set lines 0-2 to medium white foreground, purple background.

```
^[l,0,2,222,101
```

Reverse Text Colours

Text can be made to display in reverse colours (based on current line colours) by setting bit 7 of a character to 1. See example program below:

Terminal Input Windows

The serialVGA board requires only the RX receive serial connection (TX from microcontroller). This leaves the RX on the microcontroller free to use. If a keyboard were attached to the RX input (see USB Host board), the microcontroller could use this for command entry and send the received keystrokes back out through the TX pin to the serialVGA board. You effectively have a terminal.

It is useful when using a terminal window to turn on the cursor. If a backspace character (08) received from the keyboard is sent to the serialVGA board, the board will erase the previous character and move the cursor position back one.

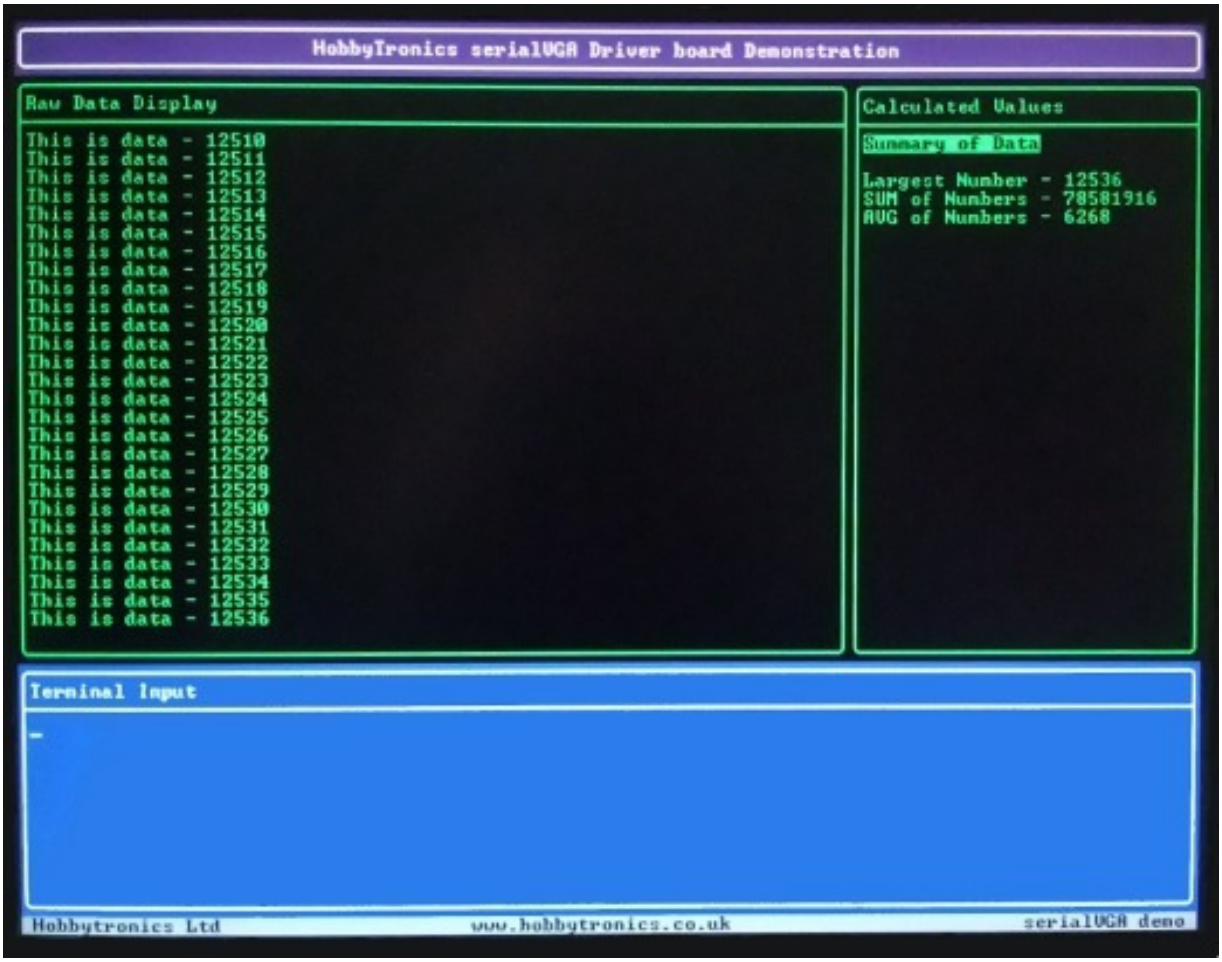
Arduino Example Program

The following Arduino sketch creates a number of windows on the display.

- A header window with border and heading text
- A "*Raw Data Display*" window
- A "*Calculated Values*" window
- A "*Terminal Input*" window
- A footer window. The footer has no border or title and is only one line high.

Colours are set for the title window, Terminal window and footer window. The other two windows default to the standard green on black.

Data is sent to the *Raw Data Display* window which will scroll when it gets full. Data in the *Calculated Values* window is positioned and doesn't move. Some text is displayed in reverse.



```
/*
  serialVGA - Test program for communicating with Hobbytronics serialVGA board

  Creates 5 windows and displays data

  Copyright www.hobbytronics.co.uk 2012
*/

unsigned long currentTime;
unsigned long loopTime;
unsigned int  counter=0;
unsigned long number_sum=0;
unsigned long number_avg=0;

// Window Creation strings (without initial escape sequence)
char header[]  = {"w,1,0,0,100,3,1,"};                // no title
char window2[] = {"w,2,0,3,70,32,1,Raw Data Display"};
char window3[] = {"w,3,70,3,30,32,1,Calculated Values"};
char window4[] = {"w,4,0,35,100,14,1,Terminal Input"};
char footer[]  = {"w,5,0,49,100,1,0,"};               // no border, no title
// Line colour strings (without initial escape sequence)
char header_colour[]  = {"l,0,2,222,101"};    // White text on purple lines 0-2
char window4_colour[] = {"l,35,48,222,001"};  // white text on blue lines 35-48
char footer_colour[]  = {"l,49,49,000,111"};  // black text on grey line 49

void vga_command(char *command_str) {
   // Function to easily send command to VGA board
   Serial.print("^[");           // send escape sequence
   Serial.println(command_str);  // send Command string
   // Most commands don't take very long, but need a small delay to complete
   // The Reboot cammand needs 2 seconds
   if(command_str[0]=='r') delay(2000);  // Wait 2 seconds for reboot
   if(command_str[0]=='w') delay(20);    // Small delay for window commands
                                         // 5ms at 9600, 20ms at 115200
   delay(2);                             // Other commands need a tiny delay

}

void vga_inverse(char *text_str) {
   // Function to write inverse characters to VGA board
   // Makes bit 7 of each character a 1
   unsigned int i;
   for(i=0;text_str[i] != '\0';i++) {
      text_str[i]=text_str[i] | 0x80;
   }
   Serial.print(text_str);
}
```

```
void setup() {
  // initialize
  Serial.begin(9600);         // Set baud rate for serialVGA board
  vga_command("r");           // reboot VGA board
  vga_command(header);        // Create header
  Serial.print("                          ");
  Serial.print("HobbyTronics serialVGA Driver board Demonstration");
  vga_command(window2);       // Create Window2
  vga_command(window3);       // Create Window3
  vga_command(window4);       // Create Window4
  vga_command("c,4,1");       // Turn text cursor ON in window 4
  vga_command(footer);        // Create footer
  Serial.print(" Hobbytronics Ltd                    www.hobbytronics.co.uk");
  Serial.print("                           serialVGA demo");
  vga_command(header_colour);       // header window colour
  vga_command(window4_colour);      // window 4 colour
  vga_command(footer_colour);       // footer window colour
  currentTime = millis();
  loopTime = currentTime;
}

void loop() {
  currentTime = millis();
  if(currentTime >= (loopTime + 500)){
    // send every half second
    loopTime = currentTime;               // Updates loopTime

    vga_command("f,2");                   // Set Window2 as focus
    Serial.print("This is data - ");      // print data
    Serial.println(counter, DEC);
    vga_command("f,3");                   // Set Window3 as focus
    vga_command("p,0,0");                 // Set Window3 text position to 0,0
    vga_inverse("Summary of Data");       // print title in reverse
    Serial.print("\r\n\r\n");             // send 2x CR to move down 2 lines
    Serial.print("Largest Number - ");    // print data
    Serial.println(counter, DEC);
    Serial.print("SUM of Numbers - ");    // print data
    Serial.println(number_sum, DEC);
    Serial.print("AVG of Numbers - ");    // print data
    Serial.println(number_avg, DEC);

    counter++;
    number_sum+=counter;
    number_avg=number_sum/counter;

  }
  if (Serial.available() > 0) {
     // Data received into Arduino board, echo'd out to Window 4 (command
window)
       int inByte = Serial.read();
       vga_command("f,4");                // Set Window4 as focus
       Serial.write(inByte);
   }
 }
```