VGA Core for Altera DE2/DE1 Boards

Preliminary

1 Core Overview

The VGA Core drives the signals required by the VGA digital-to-analog converter (DAC), which in turn outputs signals to a VGA-compatible monitor. The VGA Core supports both pixel and character input. It greatly simplifies the communication with the VGA DAC. Users can integrate it easily with other SOPC components through SOPC Builder.

2 Functional Description

The VGA Core generates the timing signals required for VGA display, including horizontal and vertical synchronization signals. The timing information generated by the VGA Core produces a screen resolution of 640 × 480 pixels at a refresh rate of 60 frames per second. To generate the timing information correctly, a 25 MHz clock needs to be provided to the VGA Core. This clock is referred to as the VGA internal clock. In addition, another 25 MHz clock signal, with 180 degree phase shift to the VGA internal clock, needs to be connected to the VGA DAC on the DE2/DE1 Boards via the `VGA_CLK` pin on the boards. The VGA Core runs at 25 MHz; the slave ports to the VGA Core runs at 50 MHz.

Though the VGA Core only generates images at a resolution of 640 × 480 pixels, smaller resolutions are supported. To achieve these smaller resolutions, the VGA Core replicates pixel information to adjacent pixels to achieve the actual screen resolution of 640 × 480. The VGA Core supports a character mode in addition to the pixel mode.

Figure 1 shows a block diagram of the VGA Core (For details of the DAC, refer to Video DAC Datasheet). The VGA Core uses the SRAM chip on the DE2/DE1 as a pixel frame buffer and uses on-chip memory as a character buffer.

The following sections describe the different modes of the VGA Core.

2.1 VGA Display Modes

The VGA Core can support three display modes: pixel mode, character mode, and character overlay mode. The pixel and character modes have their own Avalon slave interfaces, *avalon_pixel_slave* and *avalon_character_slave*, respectively. These slave interfaces present two flat, contiguous memory spaces as large as is needed to hold the data required by the chosen resolutions. Therefore, the Avalon interfaces behave as simple memory interfaces, except when back buffering is enabled (See Sec 2.2.3 for details). There are no memory-mapped control registers on either of the Avalon interfaces. In the character overlay mode, both pixel and character modes are enabled such that characters are drawn on top of the pixel mode's image.

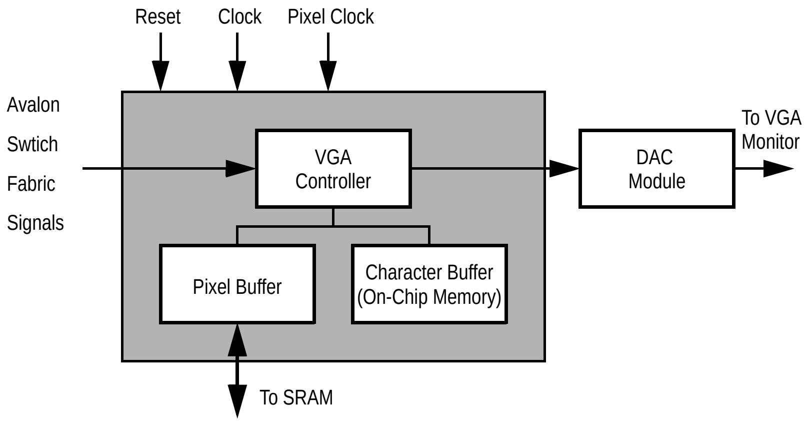

Figure 1. Block diagram for VGA Core

2.2 Pixel Mode

In pixel mode, the user can provide the color of each pixel to the Avalon Interface, *avalon_pixel_slave*, of the VGA Core. The VGA Core uses the SRAM chip on the DE2/DE1 boards as the pixel buffer. This mode is suitable for animation or image display on a VGA monitor. In pixel mode, color and resolution of the VGA Core can be configured in the following ways.

2.2.1 Color Setting

In pixel mode, the VGA Core uses the RGB color model. The following color space settings are available:

- 16-bit color mode. Red and blue have 5-bit color spaces. Green has a 6-bit color space.
- 8-bit color mode for grayscale.

2.2.2 Resolution

The VGA Core supports 640×480 resolution, which means there are $640 \times 480 = 307,200$ pixels. However, due to the size limitation of the frame buffer, a lower resolution may be needed. To support lower resolutions, the VGA Core uses *mega-pixel*s, where a mega-pixel is just a group of pixels that have the same color, to reduce the output resolution. The following “resolutions” on the mega-pixel level are available:

- 640×480. Each mega-pixel is just a pixel.
- 320×240. Each mega-pixel is a 2×2 pixel group.
- 160×120. Each mega-pixel is a 4×4 pixel group.
- 80×60. Each mega-pixel is an 8×8 pixel group.
- 40×30. Each mega-pixel is a 16×16 pixel group.

2.2.3 Back Buffer

Back buffering is a technique used to reduce or remove visible artifacts from the drawing process. Two image buffers are used and, at any one time, a frame in one buffer is actively displayed by the VGA monitor, while a frame in another buffer is drawn. When drawing is completed, the roles of the two frames are switched, so that the previously shown frame is modified, and the previously drawn frame is shown.

In pixel mode, bit b_{31} of the data is used to change the frame buffer. The user should set b_{31} of the last pixel in a frame to 1 so that the VGA Core switches the frame in the background buffer and the frame that is displayed to the screen. Note that switching the back buffer takes time because the VGA Core must wait until the displayed frame has been completely drawn, which can take up to $\frac{1}{60}$ second. The user can read the highest memory location of the *avalon_pixel_slave* to check if the frame buffers have been switched. If the least significant bit (LSB) is 1, the image buffers are still in the process of being switched. Pixels cannot be read or written while the buffers are being switched, and trying to do so results in undefined behavior.

2.3 Character Mode

In character mode, the unit of operation is characters. A device can send ASCII character codes to the VGA Core's Avalon interface *avalon_char_slave*. The VGA Core handles the conversion of characters to pixels.

Upon initialization or reset, the VGA core sets all the characters to "space", so no characters will get displayed. This "clear screen" operation can take up to 5000 clock cycles to finish, which, in general, will not cause any problems.

After initialization, the user can clear the screen by writing 1 to the highest memory location of the *avalon_char_slave*. The VGA core sets the LSB of the word to 1 if it is in the process of clearing, or to 0 if the clear screen operation is done. During the clearing process, no data can be read or written to the character slave, and trying to do so results in undefined behavior.

2.3.1 Color Setting

In character mode, the VGA Core also uses the RGB color model, but fewer color space settings are supported than are supported in pixel mode:

- 9-bit color mode. Each color (red, green, and blue) has a 3-bit color space.
- 8-bit color mode. This mode stores colors in a look-up table in the VGA Core and uses the 8 bits as an index to the look-up table. Four bits are for the foreground color and four bits are for the background color. Also, another bit is used to indicate whether the background color is transparent.
- 4-bit color mode. This mode stores colors in a look-up table in the VGA Core and uses the 4 bits as an index to the look-up table.
- 1-bit color mode. Characters are drawn in white with transparent background.

2.3.2 Resolution/Character Size

In character mode, the resolution is defined by the number of characters per line and the number of lines per screen. All the characters have to be the same size. Two sizes are supported:

- 8 × 8. Each character occupies an 8 × 8 VGA pixel group. Therefore, 80 characters can be displayed per line and the VGA Core can support up to 60 lines.
- 16 × 16. Each character occupies a 16 × 16 VGA pixel group. Therefore, 40 characters can be displayed per line and the VGA Core can support up to 30 lines.

2.4 Character Overlay Mode

In character overlay mode, both characters and pixels get displayed, with the character layer on top of the pixel layer. The background of characters are transparent, except in 8-bit color mode, where the transparency is optional on a per-character basis.

2.5 Address Format

Coordinates are memory-mapped in the VGA Core's addressable space. All data for the VGA Core is word (32-bit) addressable. The supported addressing mode is called the X-Y mode.

- X-Y mode — the address contains x and y coordinates. The coordinate system is shown in Figure 2.

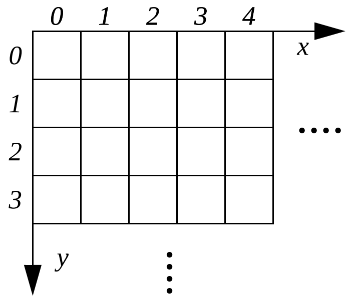

Figure 2. Video coordinate system

The format of the address is shown in Figure 3.

31			1 0
Not used	Y coordinate (n bits)	X coordinate (m bits)	00

Figure 3. X-Y address format

The values of m and n are related to the VGA Core's resolution as follows:

$$m = ceil(log_2 X)$$

$$n = ceil(log_2 Y)$$

where X and Y are the resolution in the X, Y direction, respectively.

For example, for 80 × 60 resolution we have

$$ceil(log_2 80) = 7$$

$$ceil(log_2 60) = 6$$

So for this resolution the address format is shown in Figure 4.

31 15	14 9	8 2	1 0
Not used	Y coordinate	X coordinate	00

Figure 4. X-Y address format for 80 × 60 resolution

2.6 Data Format

2.6.1 Pixel Slave's Data Format

- 16-bit color mode — Red and blue have 5-bit color space; green has 6-bit color space. The data format is shown in Figure 5.

31 16	15 11	10 5	4 0
Not used	R	G	B

Figure 5. Pixel slave's data format for 16-bit color mode

- 8-bit color mode for gray scale — the data format is shown in Figure 6.

31 8	7 0
Not used	Grayscale

Figure 6. Pixel slave's data format for 8-bit color mode

2.6.2 Character Slave's Data Format

Below shows the data format for the character slave under different modes.

- 9-bit color mode — The data format is shown in Figure 7. The R, G, B values specify the foreground color of a character.

31 17	16 14	13 11	10 8	7	6 0
Not used	R	G	B	0	ASCII

Figure 7. Character slave's data format for 9-bit color mode

- 8-bit color mode — The data format is shown in Figure 8. The T field indicates the transparency of the background color, with T = 0 for non-transparent background color and T = 1 for transparent background color. The FG and BG fields are foreground and background color indexes, respectively.

31 17	16	15 12	11 8	7	6 0
Not used	T	BG	FG	0	ASCII

Figure 8. Character slave's data format for 8-bit color mode

- 4-bit color mode — The data format is shown in Figure 9. The FG field is foreground color index.

31 12	11 8	7	6 0
Not used	FG	0	ASCII

Figure 9. Character slave's data format for 4-bit color mode

- 1-bit color mode — the character color is white and the background is transparent. The data format is shown in Figure 10. There is no bit in the data format used for color, since pixels are either white, if the pixel is part of the character being displayed, or transparent, if the pixel is not part of the character.

31 7	6 0
Not used	ASCII

Figure 10. Character slave's data format for 1-bit color mode

3 Instantiating the Core in SOPC Builder

Designers use the VGA Core's configuration wizard in the SOPC Builder to specify the desired features. The following sections describe the available options in the configuration wizard.

3.1 Mode Tab

The options in this tab control the operating modes of the VGA Core. The following options are available:

Pixel Mode — Enables the pixel mode, which is explained in Section 2.2.

Character Mode — Enable the character mode, which is explained in Section 2.3.

Character Overlay Mode — Enable the character overlay mode, which is explained in 2.4.

3.2 Pixel Mode Setting

Options in this tab control the resolution and color setting of the VGA Core, as well as the use of the back buffer.

3.3 Character Mode Setting

Options in this tab control the character size and color setting of the VGA Core.

3.4 Clock Setting

After the VGA Core is instantiated in the SOPC Builder, it requires three clock signals to be connected as follows:

- VGA — 25 MHz. This clock signal is used inside the Core and has a 180 degree phase shift with the `VGA_CLK` signal that outputs to the VGA DAC.
- avalon_pixel_slave Port — 50 MHz.
- avalon_character_slave Port — 50 MHz.

Another Altera University Program IP Core named *Development Board External Interface* can create the necessary signals automatically. Refer to that component's documentation on how to properly instantiate and connect it to a system.

4 Software Programming Model

4.1 Programming with the VGA Core

The VGA Core is packaged with C-language functions accessible through the SOPC Builder-generated software development kit (SDK) libraries. These functions implement basic operations that users need for the VGA Core. When using the Altera Debug Client, these functions are automatically provided for use in a C-language application program. They are presented in Section 4.2. To use the functions, the C code must include the statement:

```
#include "alt_up_vga.h"
```

4.2 VGA Functions

4.2.1 int alt_up_vga_draw_pixel (alt_u32 *pixel*, unsigned *x*, unsigned *y*)

Draw a pixel in the location specified by *(x, y)* on the VGA monitor.

Parameters:

pixel – the pixel

x – the *x* coordinate

y – the *y* coordinate

Returns:

0 for success, -1 for error (such as out of bounds)

4.2.2 int alt_up_vga_draw_char (alt_u8 *ch*, unsigned *x*, unsigned *y*)

Draw a character to the location specified by *(x, y)* on the VGA monitor.

Parameters:

ch – the character to draw

x – the *x* coordinate

y – the *y* coordinate

Returns:

0 for success, -1 for error (such as out of bounds)

5 RGB Color Table

Table 1 shows the RGB composition of some commonly-used colors. Please note that RGB values here have a range of 0 to 255. While using the VGA Core, users may need to adjust the value range according to different modes.

Table 1. RGB Color Table

Color Name	R	G	B	Color
Red	255	0	0	🟥
Green	0	255	0	🟩
Blue	0	0	255	🟦
Yellow	255	255	0	🟨
Cyan	0	255	255	🟦
Magenta	255	0	255	🟪

■