XAPP928 (v1.1) April 19, 2007

Digital Display Panel Reference Design

Reference for Spartan-3E Display Development Kit HW-SPAR3E-DISP-DK-UNI-G

Summary

Digital displays are a fast-growing market comprising LCD, plasma, and rear projection television technologies as well as smaller displays for mobile handsets and automobiles, in addition to many other applications. Digital image processing enhances the overall viewing aesthetics of the displayed image and can differentiate your product.

Xilinx has developed a reference design IP core based on the Xilinx Spartan™-3E Display Development Board and intended for display panel applications to assist in developing products for this market. The display solution FPGA consists of a DVI Input interface, color temperature correction, precise gamma correction, an image dithering engine, and Low-Voltage Differential Signaling (LVDS) Transmit (TX) or DVI TX output interface (see Figure 1).

This document describes the Spartan-3E Display Development Board. It also provides details on the DIP switch settings and detailed resource counts for each of the IP blocks.

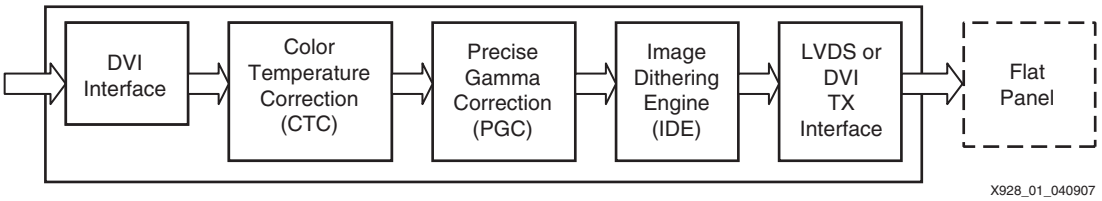

Figure 1: **Xilinx Display Panel IP Reference Design Flow**

IP Block Summary

Table 1 defines the key IP blocks in the Spartan-3E Display Development Board.

Table 1: **IP Block Descriptions**

Block Name	Block Description
DVI Receiver	Accepts input video in DVI format
Color Temperature Correction (CTC)	Incoming RGB values of the entire frame are corrected to the user set color temperature
Precise Gamma Correction (PGC)	Accepts the pixel stream, processes it as per the required gamma value, and sends the modified pixel stream out to the next processing module
Image Dithering Engine (IDE)	Receives the 3x10-bit gamma corrected pixel stream from the precise gamma correction module and dithers it to 3x8-bit pixel stream, without losing the video quality
Display Interface	Allows the DVI transmitter to directly drive LCD modules with LVDS interfaces

User Switch and Push Button Settings

The switch settings in this document refer to the User Set DIP Switch block on the Spartan-3E Display Development Board (see Figure 2).

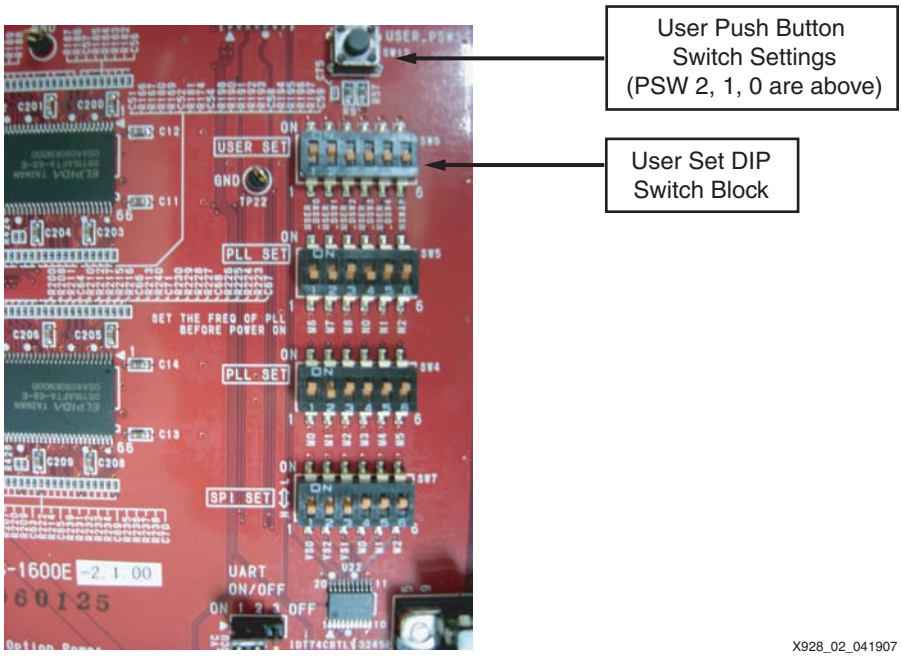

Figure 2: **Display Development Board Switch Settings**

Table 2: **User DIP Switch Settings**

DIP Switch	Function
SW4	DVI RX. Must be set to “ON” in all cases.
SW3, SW2	Precise Gamma Correction (PGC) Control • On,On (equals 00): Gamma 2.2/2.2 (1.00) • On,Off (equals 01): Gamma 2.4/2.2 (1.09) • Off,On (equals 10): Gamma 2.6/2.2 (1.18) • On,On (equals 11): Disable PGC
SW1, SW0	Color Temperature Correction (CTC) Control • On,On (equals 00): Bypass • On,Off (equals 01): 6500K • Off,On (equals 10): 8500K • On,On (equals 11): 10000K

The image dithering engine is modified using the user push button settings:

- “0”: used when the push button is depressed
- “1”: used when the push button is released

DVI Receiver (RX) Interface Block

The DVI RX interface must be used for this reference design. Refer to the *Display Development Board User Guide* to set up the DVI option. Table 3 shows the DIP switch settings to configure the DVI RX interface.

Table 3: **LVDS DIP Switch Settings for LVDS/DVI RX Selection**

DIP Switch	Port Name	Configuration
DIP_SWITCH_0 [User Switch SW4]	DIPS_RX_SELECT	Must be set to “ON” in all cases

CTC IP Block

Introduction to Color Temperature

White light can be described by color temperature. To determine the color temperature of a light source, its output is compared with a theoretical “black-body radiator” at a certain temperature in Kelvin. Specifically, 5000K to 5500K is seen in typical daylight, 2000K is red/orange, and 15000K is bluish. Different light sources and different display technologies show differing color temperatures. For example, as the sun crosses the sky, it may appear to be red, orange, white, or blue, depending on its position.

In digital displays, white color is realized by a superposition of the R, G, and B color emitted from the R, G, and B cells of the specific display. RGB data can be transformed to fit the CIE x-y color space, the color space where many calculations are performed in this algorithm.

The term “White Point” is loosely defined as color temperature. On a CIE chromaticity diagram (Figure 3), a white point at 5500K is near the point x = y = 0.33.

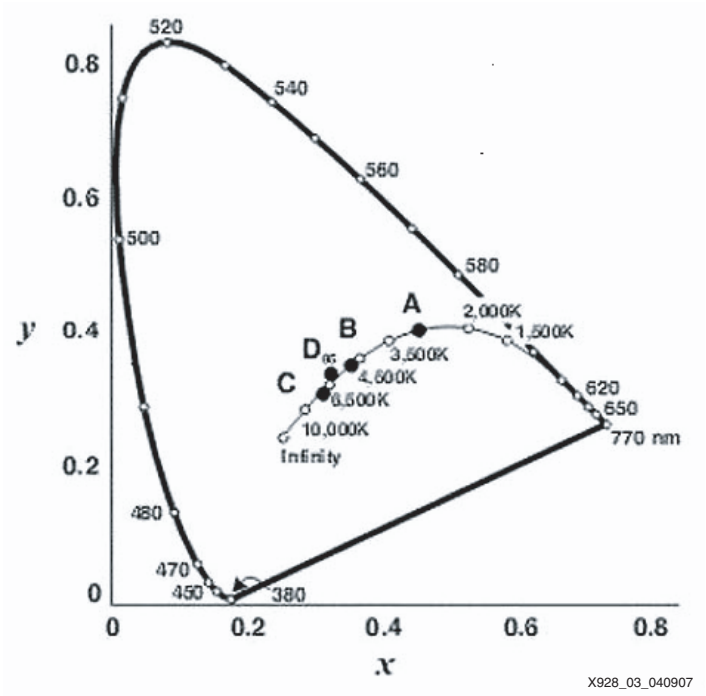

Figure 3: **Color Temperature on CIE Chromaticity Diagram**

For example, plasma display panels (PDPs) show a low color temperature, especially compared to a conventional cathode ray tube (CRT) display. The low color temperature of the PDP is caused by an inherent low blue luminance. Furthermore, the end customer cannot arbitrarily vary the color temperature once the PDP cell structure and the related driving scheme are fixed. A digital display output such as a PDP must be color corrected to achieve better image quality.

CTC Algorithm

In this algorithm implementation, the white point or color temperature of the frame is changed to the desired color temperature of that particular frame. First, a CIE reference white point is

selected in x and y format from a user-selected temperature input. For every incoming frame, a white point is estimated. The input frame of the 8- or 10-bit RGB data is converted into x-y format using a color-mixing program.

Next, the incoming frame temperature and the reference temperature are checked to see if they are the same. A recursive calculation controls the delta temperature value and selects the reference white point to give an exact color temperature correction. If the color temperature is the same (within the delta), processing is not done. If the color temperature is different, the frame is changed to the desired color temperature.

A ratio method is used to modify the color temperature. The CIE temperature to incoming frame temperature ratio is then applied to individual RGB data of the input frame to set the desired color temperature. At this point, correlated color temperature conversion is then achieved.

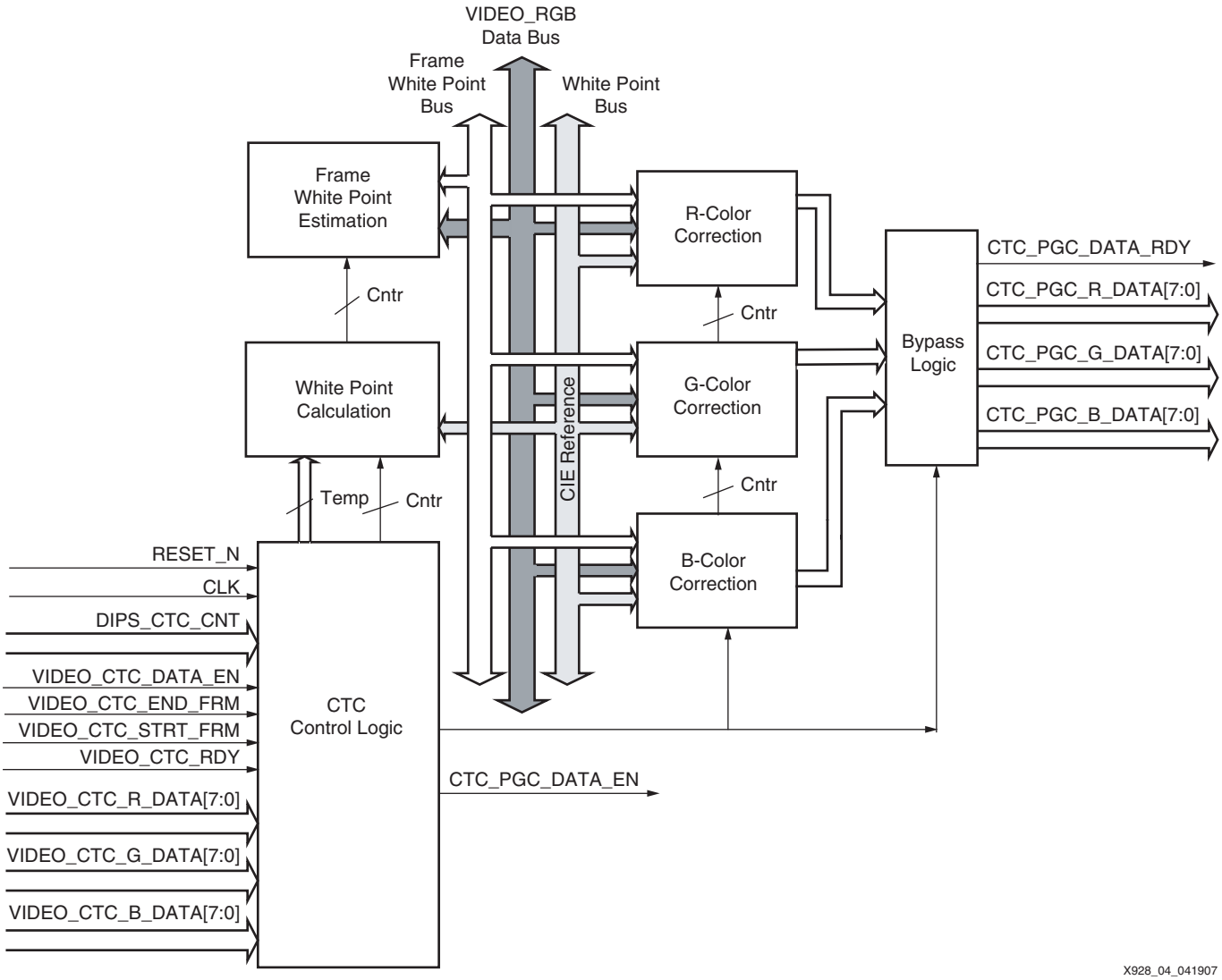

Figure 4: **CTC IP Block Diagram**

Table 4 defines the signals in the CTC module.

Table 4: **Signal Descriptions of the CTC Module**

Signal	I/O	Detailed Description
RESET_N	I	System reset. This signal is asserted Low to reset all processes inside the CTC module.
CLK	I	System clock. This clock runs at the desired frequency for video operation.
DIPS_CTC_CNT	I	Control from DIP switches (User Switches SW1, SW0). This signal determines the target color temperature.
VIDEO_CTC _STRT_FRM	I	The CTC asserts this signal to indicate the start of the new frame to the LVDS receiver.
VIDEO_CTC _END_FRM	I	The CTC asserts this signal to indicate the end of the present frame to the LVDS receiver.
VIDEO_CTC _DATA_EN	I	The CTC asserts this signal to indicate that valid data is given to the LVDS receiver block.
VIDEO_CTC_RGB_DATA[23:0]	I	The CTC places the input pixel data in RGB format on this 24-bit data bus.
CTC_PGC_DATA_RDY	O	The CTC asserts this signal to indicate valid data is ready for further processing by the PGC block.
CTC_PGC_DATA_EN	O	The CTC asserts this signal to enable data on CTC_PGC_RGB_DATA[23:0] for the PGC receiver block.
CTC_PGC_RGB_DATA[23:0]	O	The CTC places the output pixel data in RGB format on this 24-bit data bus for the PGC.

The value of the input color temperature register indicates a standard temperature value (6500K, 8000K, or 9300K) as the target color temperature. The values are selected by an external DIP switch. Then CIE standard white point values are determined. These respective RGB white point values are used for further operation. By default, color temperature is not changed; it will bypass the input data. After selecting the color temperature at the start of the next frame, reference white point values are updated in the color correction module.

Frame White Point Estimation

To estimate the white point, the following algorithm is used. First, the pixel with the maximum sum of RGB over the entire frame is located. Related RGB values that represent this sum are considered as white points for that particular frame. Then this white point is updated at the start of the next incoming frame.

Color Temperature Correction Modules

In the CTC module, the incoming RGB values of the entire frame are corrected to the new temperature. For correction of these values, the ratio multiplication method of correction is implemented. For example, for the R-value of a pixel the correction in Equation 1 is used.

$$R_{in} \times \frac{R_{wt}}{R_{wf}} = R_{out} \qquad \text{Equation 1}$$

Where:

- R_{in} = Present R value of input pixel
- R_{wt} = CIE white point value for a standard temperature
- R_{wf} = R value white point of the previous frame
- R_{out} = R value of CTC corrected output pixel

The ratio R_{wt}/R_{wf} remains constant over the entire frame interval. The same procedure is also applied to the G and B values. At every clock cycle, pixels are simultaneously manipulated.

The CTC_PGC_DATA_RDY flag is enabled when the color temperature correction module is ready to send data to the next module, the precise gamma correction block.

DIP Switch Settings

Table 5 shows the DIP switch settings for color temperature correction.

Table 5: **DIP Switch Settings for the CTC Module**

DIP Switch	Port Name	Configuration
DIP_SWITCH_TYPE0 DIP_SWITCH_TYPE1	DIPS_CTC_CNT0_IP DIPS_CTC_CNT1_IP	CTC_TYPE1 & CTC_TYPE0 [Switch1, Switch0] • 11: Bypass • 10: 6500K • 01: 8500K • 00: 10000K

Device Utilization for the CTC Module

Table 6 summarizes the resources used in the CTC module. Device utilization is based on the Spartan-3E XC3S1600E FG484 FPGA on the Display Development Board.

Table 6: **Resource Utilization for the CTC Module**

	Used	Available	Utilization
Logic Utilization			
Number of slice flip-flops	5,275	29,504	18%
Number of 4-input LUTs	3,579	29,504	12%
Total Number of 4-Input LUTs	**3,664**	**29,504**	**12%**
Number used as logic	3,579		
Number used as a route-thru	1		
Number used as shift registers	84		
Number of bonded IOBs	58	376	15%
Number of Block RAMs	4	36	11%
Number of MULT18X18s	4	36	11%
Number of GCLKs	1	24	4%
Logic Distribution			
Number of occupied slices	3,423	14,752	23%
Number of slices containing only related logic	3,423	3,423	100%
Number of slices containing unrelated logic	0	3,423	0%

Precise Gamma Correction

Introduction to Gamma

A gamma characteristic is an exponential relationship that approximates the relationship between the encoded luminance in a display system and the desired image brightness. Mathematically, a generic function is: Output = Input Function ^ (Gamma), or the inverse function to precorrect the data before it is displayed. Many displays show a nonlinear relationship between input and brightness output; hence, the need for gamma correction. Correction can also be applied to enhance perceived image quality.

Uncorrected images or the incorrect gamma can cause poor contrast, poor color balance, and an improper overall light level. In addition, it is difficult to correct these image deficiencies with other color adjustments. Therefore, it is important to first encode the proper gamma for all images.

For example, in the case in Figure 5, the linear encoded (uncorrected) input signal V_s shows a large jump in perceived brightness from 0.1 to 0.4 and a much smaller increase from 8 to 10. Basically, the gamma function is applied to the input to achieve a linear output intensity for each input step for this display, as is desired with output I.

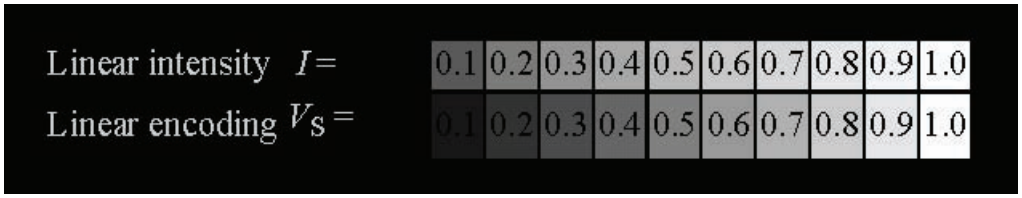

Figure 5: **Uncorrected (V_s) and a Gamma-Corrected Intensity (*I*)**

Gamma Correction Implementation

A gamma curve of a 10-bit width was chosen to increase the accuracy of the output because the 10-bit output can be approximated to the nearest integer with more degrees of freedom. The 8-bit and 10-bit gamma curves are shown in Figure 6 and Figure 7, respectively, for comparison. The curves show the gamma-corrected luminance value with respect to the pixel input. The gray curve has the 8- or 10-bit resolution and the black curve is the desired response.

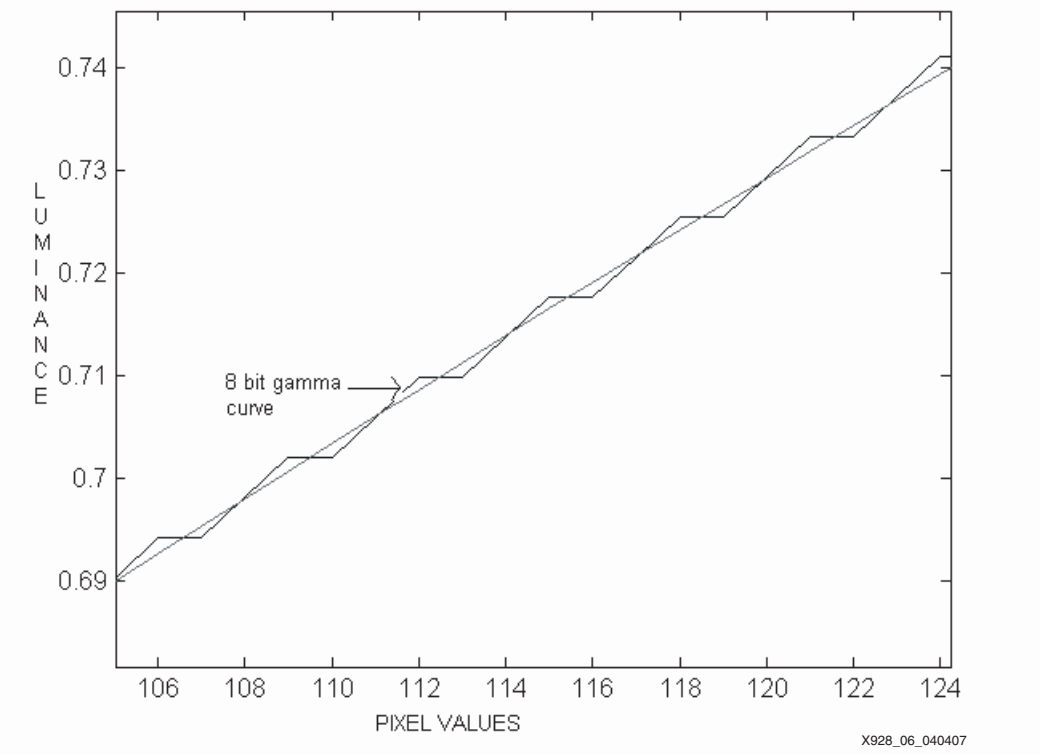

Figure 6: **8-bit Gamma Curve versus a Desired Linear Response**

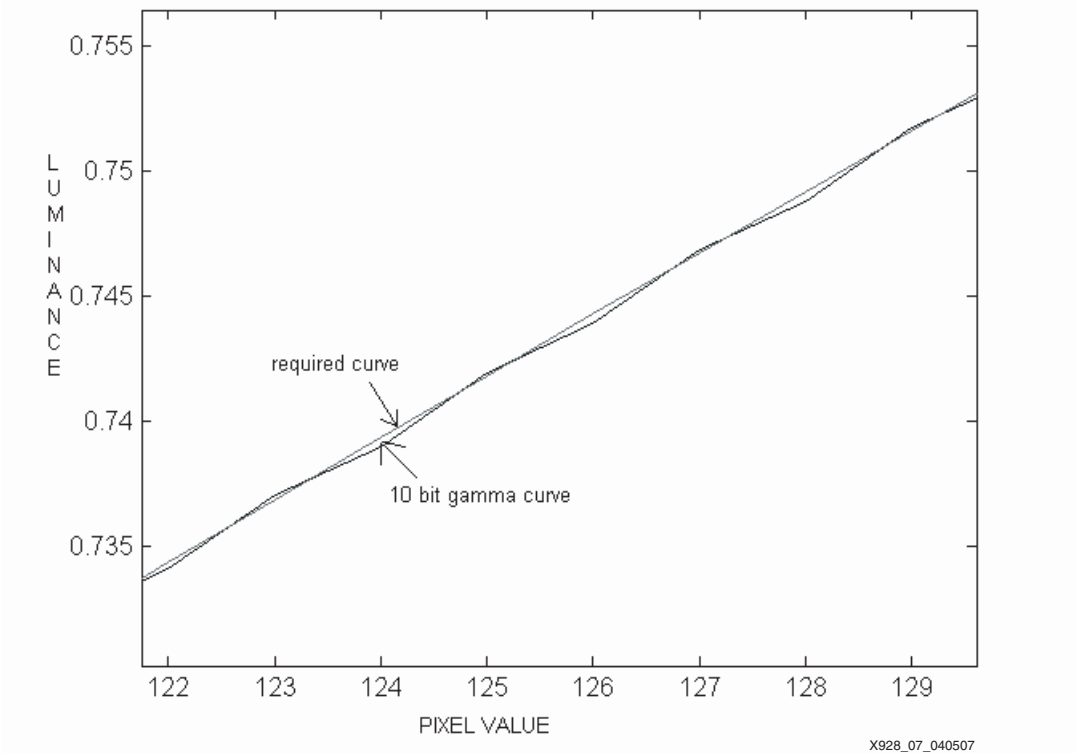

Figure 7: **10-bit Gamma Curve versus a Desired Linear Response**

The figures clearly show that the 8-bit output produces a step-like function and the 10-bit output produces the desired smoothness and fit to the output.

Equation 2 is used for 10-bit gamma correction.

$$Y = ROUND\left[1023 \times \left(\frac{X}{256}\right) \wedge \left(\frac{1}{Gamma}\right)\right] \qquad \textit{Equation 2}$$

Where:

- X = an RGB input (integer representation an individual 8-bit R, G, or B value)
- Y = the 10-bit gamma corrected R', G', B' output
- Gamma = the gamma factor (programmable)

Round (Y) takes the nearest integer if the value of the decimal value is greater than 0.5; otherwise, it truncates the decimal part. The output is in a 10-bit format.

Detailed Datapath Description

External DIP switches are set to the value of gamma to be applied to the input data. The gamma functions are applied individually to each R, G, B color through a series of look-up tables (LUTs). The output data is in the form of 3x10-bit data for R'G'B'.

When precise gamma correction is complete, data-ready flags are enabled for the next module: the image dithering engine. It is also possible to bypass this block via an external DIP switch.

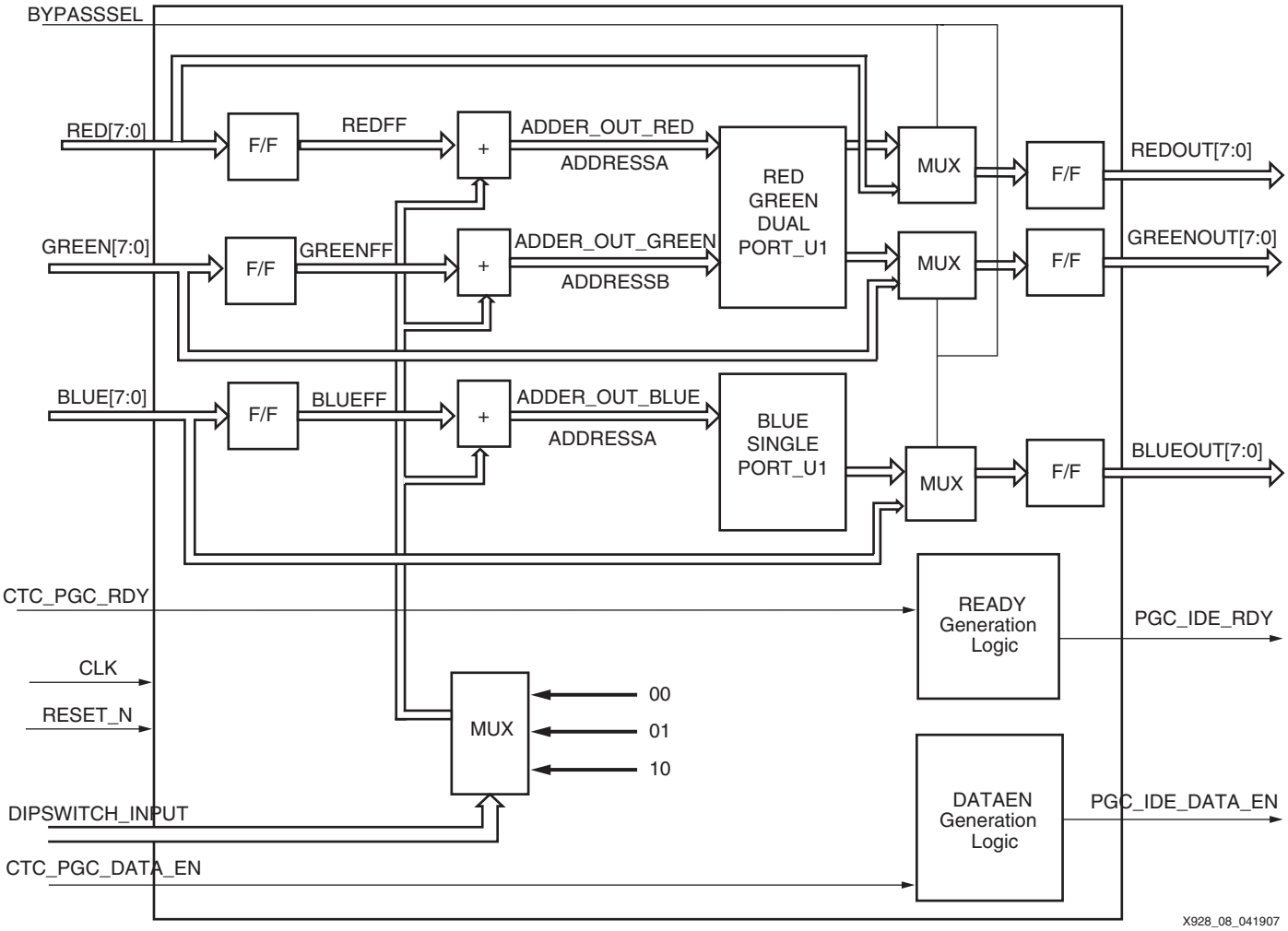

Figure 8: **Block Diagram of Precise Gamma Correction**

Table 7 defines the signals in the PGC module.

Table 7: **Signal Descriptions of the PGC Module**

Signals	I/O	Detailed Description
RESET_N	I	Active Low global reset. This signal is asynchronous.
CLK	I	This signal is the global clock.
CTC_PGC_RDY	I	The CTC module asserts this signal to inform the PGC module that the pixel data is valid.
GREEN[7:0]	I	This eight-bit bus contains the Green input.
BLUE[7:0]	I	This eight-bit bus contains the Blue input.
RED[7:0]	I	This eight-bit bus contains the Red input.
CTC_PGC_DATA_EN	I	The CTC module asserts this signal High for the complete data valid line of the frame.
BYPASSSEL	I	This signal enables data to bypass the gamma correction block.
DIPSWITCH_INPUT[1:0]	I	The user sets the desired gamma value on these signals.
REDOUT[9:0]	O	This 10-bit bus contains the gamma-corrected output from the PGC module.

Table 7: **Signal Descriptions of the PGC Module *(Continued)***

Signals	I/O	Detailed Description
GREENOUT[9:0]	O	This 10-bit bus contains the gamma-corrected output from the precise gamma correction block.
BLUEOUT[9:0]	O	This 10-bit bus contains the gamma-corrected output from the PGC module.
PGC_IDE_RDY	O	The PGC module asserts this signal to inform the IDE module that the pixel data is valid.
PGC_IDE_DATA_EN	O	The PGC module asserts this signal High to indicate the complete data valid line of the frame to the IDE module.

The reset state of all flip-flops, boundary signals, and registers is logic “0” for all bits.

DIP Switch Settings

Table 8 shows the DIP switch settings for precise gamma correction.

Table 8: **DIP Switch Settings for the PGC Module**

DIP Switch	Port Name	Configuration
DIP_SWITCH_TYPE0 DIP_SWITCH_TYPE1 User Switches SW3, SW2	DIPS_PGC_SEL0_IP DIPS_PGC_SEL1_IP	PGC_TYPE1 & PGC_TYPE0 [Switch SW3, Switch SW2] • 00: Gamma set to 2.2 • 01: Gamma set to 2.4 • 10: Gamma set to 2.6 • 11: Reserved

Device Utilization for the Precise Gamma Correction Module

Table 9 summarizes the resources used in the PGC module.

Table 9: **Resources for the PGC Module**

	Used	Available	Utilization
Logic Utilization			
Number of slice flip-flops	50	29,504	1%
Number of 4-input LUTs	31	29,504	1%
Total Number of 4-Input LUTs	**31**	**29,504**	**1%**
Number of bonded IOBs	63	376	17%
Number of Block RAMs	2	36	6%
Number of GCLKs	1	24	4%
Logic Distribution			
Number of occupied slices	42	14,752	1%
Number of slices containing only related logic	42	42	100%
Number of slices containing unrelated logic	0	42	0%

Image Dithering Engine (IDE)

Introduction to Dithering

Dithering is a technique used to create the illusion of color depth in displays with limited color depth. In a dithered image, colors that are not available are approximated by a mix of colored pixels from within the colors that are available. The human eye perceives the mixture as a different color.

For example, a display with only black or white colors can be used to create an image with gray colors by use of dithering (see Figure 9). The interlaced black and white pixels create the illusion of gray.

Figure 9: **Creating New Colors by Use of Dithering**

Some display devices have color depth less than the color depth of the input data (for example, eight-bit input data and six-bit display color depth). The input data is either truncated or rounded, but this approach usually produces both a loss of detail and may produce large, banded areas of a single color that differs significantly from the original image. Dithering is used to enhance these images.

Image Dithering Algorithm Description

The IDE module receives 30-bit (10 bits of data x3 for R'G'B') pixel streams from the PGC module.

The image dithering engine operates only on the active pixels. It uses a spatial dithering technique with a 2x2 dithering matrix for 10- to 8-bit dithering. When a 3x10 bit data stream comes in, for example, the two least significant bits (LSBs) of the three R,G,B colors contain the most "fine" color information and are selected away from the eight remaining MSB bits.

The two LSB truncated bits effectively are an "error." These two bits can have any value from `00` to `11` (binary) providing four finer color levels (0%, 25%, 50%, and 75%) to the remaining eight bits. This error is spread over adjacent pixels.

The following example shows the weights in the 2x2 matrix:

```
0   1
2   3
```

The following example shows how the above weight table is spread across the display space.

```
0   1   0   1   .....
2   3   2   3   .....
0   1   0   1   .....
2   3   2   3   .....
.   .   .   .   .....
```

Each weight is actually a threshold value. If the last two-bit value of the input pixel is greater than the entry in the table for that position, the energy level represented by those two LSBs is added to the remaining bits, while making sure that the resultant eight-bit number does not overflow.

To eliminate a noticeable pattern, the weight table can be rotated every two lines or every frame, which adds a temporal dithering over the spatial dither.

Also, the matrix weight can be randomly rotated to provide randomness over the dithering behavior, which is good for a video screen display. When dithering with the least significant two bits of input data, the IDE uses spatial, tempo-spatial, and random spatial dithering with 2x2 pixel blocks.

When complete, the 24-bit dithered data (eight bits for each R, G, and B) are sent to the LVDS/DVI TX interface along with a dither_data_ready signal.

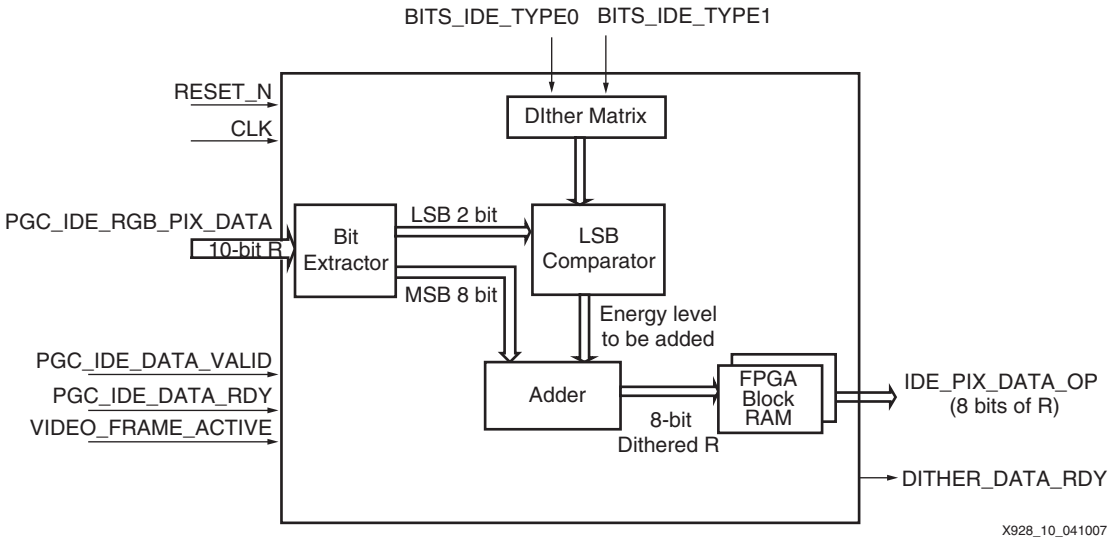

Figure 10: **Functional Description of Image Dithering Engine (one color shown)**

Table 10 defines the signals in the IDE module.

Table 10: **Signal Descriptions of the IDE Module**

Port Name	I/O	Detailed Description
RESET_N	I	This input is asserted Low to reset the IDE module.
CLK	I	The digital clock manager (DCM) generates this system clock.
BITS_IDE_TYPE0	I	Dither type selection. See Table 11 for the encoding.
BITS_IDE_TYPE1	I	Dither type selection. See Table 11 for the encoding.
PGC_IDE_DATA_VALID	I	The PGC module asserts this signal to the IDE module to indicate the input pixel data is valid. This signal is held asserted for the entire line duration.
PGC_IDE_DATA_RDY	I	The PGC module generates a pulse on this signal to the IDE module when gamma corrected data is available.
PGC_IDE_RGB_PIX_DATA[29:0]	I	The PGC module places 10-bit R, G, and B gamma corrected pixel data on this bus for the IDE module.
VIDEO_FRAME_ACTIVE	I	This signal is asserted High to indicate a valid frame. It goes Low when the frame ends.
IDE_PIX_DATA _OP[23:0]	O	This output bus provides the dithered 8-bit RGB values.
DITHER_DATA_RDY	O	This signal is asserted to indicate dithered data is available.

DIP Switch Settings

The IDE module is modified by using the user push button settings (see Table 11).

Table 11: **DIP Switch Settings for the IDE Module**

DIP Switch	Port Name	Configuration
DIP_SWITCH_TYPE0 DIP_SWITCH_TYPE1 [User Push Button PSW0, PSW1]	DIPS_IDE_TYPE0_IP DIPS_IDE_TYPE1_IP	IDE_TYPE1 & IDE_TYPE0 [PSW1, PSW0] • 00: IDE disabled • 01: Spatial Dithering • 10: Tempo-Spatial Dithering • 11: Random Dithering

Device Utilization for the IDE Module

Table 12 summarizes the resources used in the IDE module.

Table 12: **Resources For the IDE Module (XC3S1600EFG484)**

	Used	Available	Utilization
Logic Utilization			
Number of slice flip-flops	98	29,504	1%
Number of 4-input LUTs	170	29,504	1%
Logic Distribution			
Number of occupied slices	119	14,752	1%
Number of slices containing only related logic	119	119	100%
Number of slices containing unrelated logic	0	119	0%
Total Number of 4-input LUTs	**188**	**29,504**	**1%**
Number used as logic	170		
Number used as a route-thru	18		
Number of bonded IOBs	62	376	16%
Number of GCLKs	1	24	4%

LVDS TX/DVI Interface

The LVDS transmitter converts 28 bits of LVCMOS/LVTTL data into four LVDS data streams. A phase-locked transmit clock is transmitted in parallel with the data streams over a fifth LVDS link. Every cycle of the transmit clock's 28 bits of input data are sampled and transmitted. Both the LVDS and DVI interfaces are provided simultaneously on the Spartan-3E Display Development Board.

The RGB Output interface consists of 24-bit RGB data and four bits of timing and control signals (Clock, HSYNC, VSYNC, Data Enable).

LVDS Transmitter Functional Block Diagram

The LVDS TX module (see Figure 11) consists of a DCM and a serdes_4b_7to1 module. The DCM generates the clkx3p5 and clkx3p5not clocks, which are each 3.5 multiples of the input clock. These two clocks are used inside the serdes_4b_7to1 module for 7-to-1 serialization of the 28-bit data in four lines.

This module does the serialization of the 28 data lines into four channels by multiplexing, using the clkx3p5 and clkx3p5not clocks.

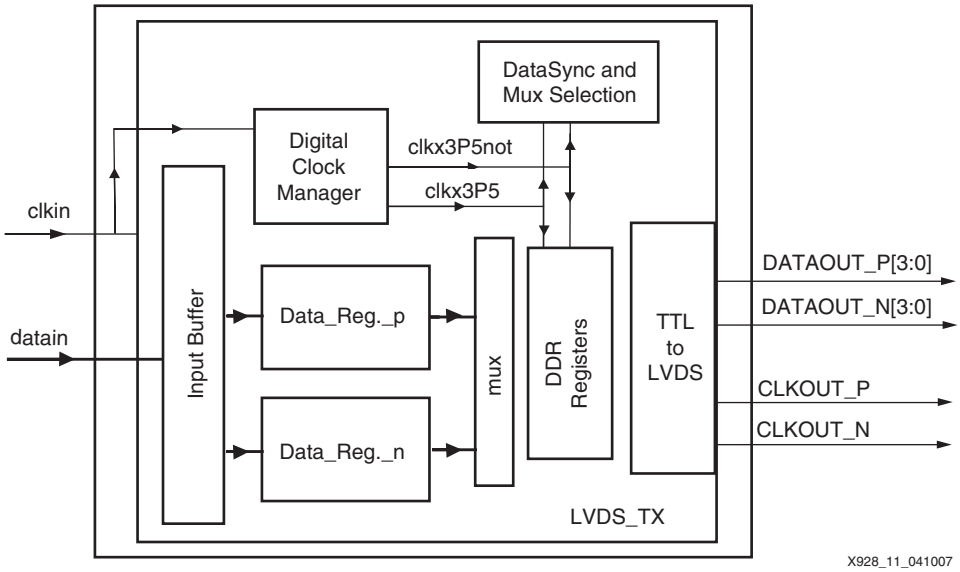

Figure 11: **Block Diagram of LVDS Transmitter**

Table 13 shows the data bit encoding for the LVDS channels (the data is transmitted LSBs first).

Table 13: **Data Bit Encoding on the LVDS Channels**

LVDS Channel	Position of datain Bits
Channel 0	0, 4, 8, 12, 16, 20, 24
Channel 1	1, 5, 9, 13, 17, 21, 25
Channel 2	2, 6, 10, 14, 18, 22, 26
Channel 3	3, 7, 11, 15, 19, 23, 27

Top-Level Design Hierarchy

The design files are used in the following hierarchy, including testbenches. The design files are located on the Xilinx website at http://www.xilinx.com/bvdocs/appnotes/xapp928.zip.

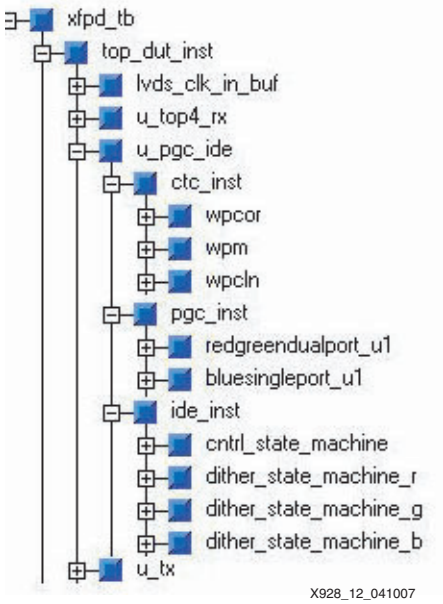

Figure 12: **Module Hierarchy**

Device Utilization for Top-Level Entity

Table 14 provides the synthesis results for the entire system.

Table 14: **Complete System Synthesis Results**

	Used	Available	Utilization
Logic Utilization			
Total Number of Slice Registers	**5,748**	**29,504**	**19%**
Number used as flip-flops	5,743		
Number used as latches	5		
Number of 4-input LUTs	3,878	29,504	13%
Number used as logic	3,878		
Number used as a route-thru	33		
Number used as shift registers	88		
Total Number of 4-Input LUTs	**3,999**	**29,504**	**14%**
Number of bonded IOBs	93	376	25%
Number of Block RAMs	6	36	17%
Number of MULT18X18s	4	36	11%
Number of GCLKs	7	24	24%
Number of DCMs	3	8	38%
Number of RPM macros	2		
Logic Distribution			
Number of occupied slices	3,709	14,752	25%
Number of slices containing only related logic	3,709	3,709	100%
Number of slices containing unrelated logic	0	3,821	0%

System-Level I/O Signals

Figure 13 shows an I/O diagram of the display panel solution in a Spartan-3E FPGA.

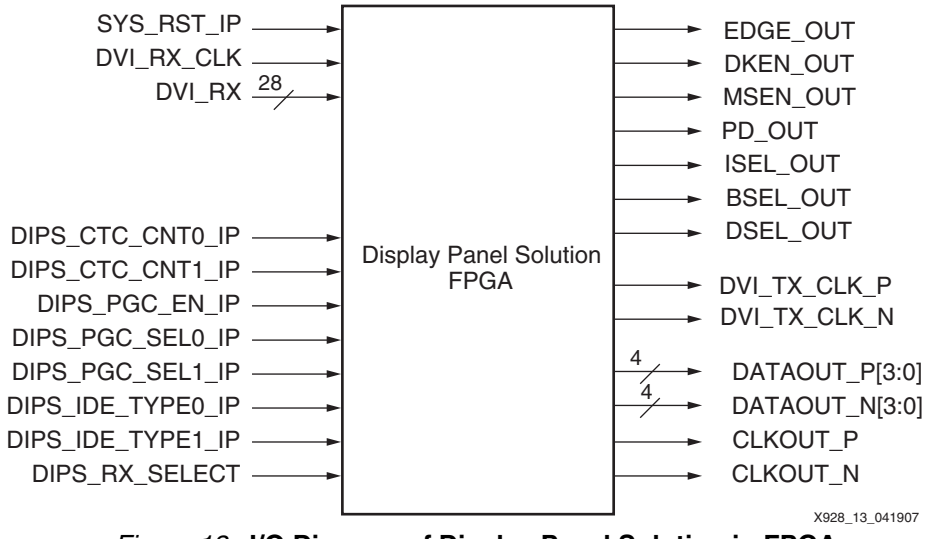

Figure 13: **I/O Diagram of Display Panel Solution in FPGA**

The following signal types are taken from the CORE specifications:

- I: Input is a standard input-only signal
- O: Output is standard output

The following signal naming conventions apply:

- All active-Low signals have a suffix of "_N"
- The suffix '+' denotes the positive line of a differential signal
- The suffix '-' denotes the negative line of a differential signal

Table 15 summarizes the signals for the Spartan-3E Display Development Board.

Table 15: **Display Development Board Signal Descriptions**

Signal	I/O	Detailed Description
System Signals		
SYS_RST_IP	I	This system reset signal is asserted to reset all processes in the module.
RGB Signals		
DVI_RX_CLK	I	DVI clock input
DVI_RX[27:0]	I	DVI data inputs (24-bit RGB data, VSYNC, HSYNC, and DE)
LVDS Display Interface Signals		
EDGE_OUT	O	DVI_TX chip-specific signal for the TFP410 (DVI transmitter)
DKEN_OUT	O	DVI_TX chip-specific signal for the TFP410
MSEN_OUT	O	DVI_TX chip-specific signal for the TFP410
PD_OUT	O	DVI_TX chip-specific signal for the TFP410
ISEL_OUT	O	DVI_TX chip-specific signal for the TFP410
BSEL_OUT	O	DVI_TX chip-specific signal for the TFP410
DSEL_OUT	O	DVI_TX chip-specific signal for the TFP410
DVI_TX[27:0]	O	DVI data outputs (24-bit RGB data, VSYNC, HSYNC, and DE)
DVI_TX_CLK	O	DVI output clock
DVI_TX_CLK_N	O	DVI_TX chip-specific signals for the TFP410
LVDS TX Signals		
DATA_OUT_P[3:0] DATA_OUT_N[3:0]	O	LVDS differential output pair
CLKOUT_P CLKOUT_N	O	LVDS clock output pair

Figure 14 shows sync timing relations among VSYNC, HSYNC, and DE signals. (Note: the Sync polarities can change depending on the target panel resolution.)

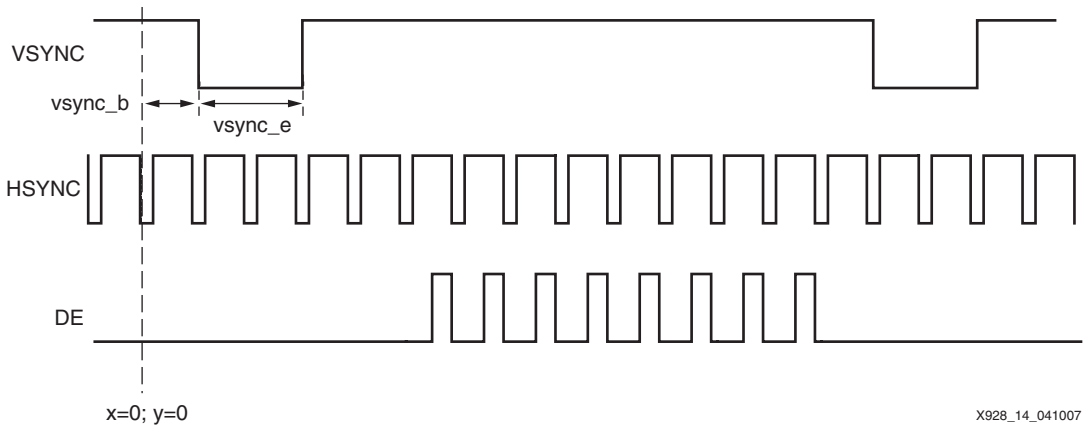

Figure 14: **Signal Relations in Reference Design**

Conclusion

These design files present a series of video processing algorithms on a Xilinx powered development board to quickly start a display development program. Any of the IP blocks can be customized to your needs on any target Xilinx device. In addition, new video processing blocks can be added to this system to quickly validate your display video enhancement algorithms.

Appendix: LVDS Timing

See XAPP486, *7:1 Serialization in Spartan-3E FPGAs at Speeds Up to 666 Mbps*, for more information.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
03/23/07	1.0	Initial Xilinx release.
04/19/07	1.1	Minor text and figure changes.