LogiCORE IP Gamma Correction v3.0

DS719 September 21, 2010 **Product Specification**

Introduction

The Xilinx Gamma Correction LogiCORE™ provides customers with an optimized hardware block for manipulating image data to match the response of display devices. This core is implemented using a look-up table structure that is programmed to implement a gamma correction curve transform on the input image data. Programmable number of Gamma tables enable having separate gamma tables for all color channels, separate tables for luminance and chrominance channels, or one gamma table to be shared by all color channels.

Features

- Programmable gamma tables
- Single or three color channel independent or shared look-up table structure
- Optional interpolated output values
- Selectable processor interface
  - EDK pCore
  - General Purpose Processor
  - Constant Interface
- Optional double-buffered interface to prevent image tearing
- 8-, 10-, or, 12-bit input and output precision
- Delay match support for up to three sync signals
- For use with Xilinx CORE Generator™ software v12.3 or higher

Applications

- Pre-processing block for image sensors
- Post-processing block for image data adjustment
- Intensity correction
- Video surveillance
- Consumer displays
- Video conferencing
- Machine vision

LogiCORE IP Facts Table				
Core Specifics				
Supported Device Family(1)	Spartan-3A DSP, Spartan-6, Virtex-5, Virtex-6			
Supported User Interfaces	General Processor Interface, EDK PLB 4.6, Constant Interface			
Supported Operating Systems	Windows XP Professional 32-Bit, Windows Vista Business 32-Bit, Red Hat Enterprise Linux WS v4.0 32-bit/64-bit, Red Hat Enterprise Desktop v5.0 32-bit/64-bit (with Workstation Option), SUSE Linux Enterprise (SLE) desktop and server v10.1 32-bit/64-bit			
	Resources(2)(3)			**Frequency**
Configuration	**LUT6-FF Pairs**	**FFs**	**Block RAMs**	**Max. Freq.(3)**
Data Width=8	55	60	0(36k)+ 3(18k)	400
Data Width=10	62	72	0(36k)+ 3(18k)	400
Data Width=12	79	84	3(36k)+ 3(18k)	400
Provided with Core				
Documentation	Product Specification			
Design Files	Netlists, EDK pCore files, C drivers			
Example Design	Not Provided			
Test Bench	Not Provided			
Constraints File	Not Provided			
Simulation Models	VHDL or Verilog Structural, C, and MATLAB™			
Tested Design Tools				
Design Entry Tools	CORE Generator tool, Platform Studio (XPS)			
Simulation	ModelSim v6.5c, Xilinx® ISim 12.2			
Synthesis Tools	XST 12.2			
Support				
Provided by Xilinx, Inc.				

1. For a complete listing of supported devices, see the release notes for this core.
2. Interface with input data width=output data width. For more complete device performance numbers, see Table 7, Table 8, Table 9, and Table 10.
3. Performance numbers listed are for Virtex-6 FPGAs. For more complete performance data, see "Core Resource Utilization and Performance," page 16.

Overview

Gamma correction, also known as gamma compression or encoding, is used to encode linear luminance or RGB values to match the non-linear characteristics of display devices. Gamma correction helps to map data into a more perceptually uniform domain, so as to optimize perceptual performance of a limited signal range, such as a limited number of bits in each RGB component.

Gamma correction is, in the simplest cases, defined by

$$V_{out} = V_{in}^{\gamma}$$

where the input and output values are between 0 and 1 (Figure 1). The case γ <1 is often called gamma compression and γ >1 is called gamma expansion.

When used in conjunction with an embedded or external processor, the Gamma Correction core supports frame-by-frame dynamic reprogramming of the gamma tables. The gamma tables can be reprogrammed with arbitrary functions, supporting a wide range of applications, such as intensity correction, feature enhancement, lin-log, log-lin conversion and thresholding.

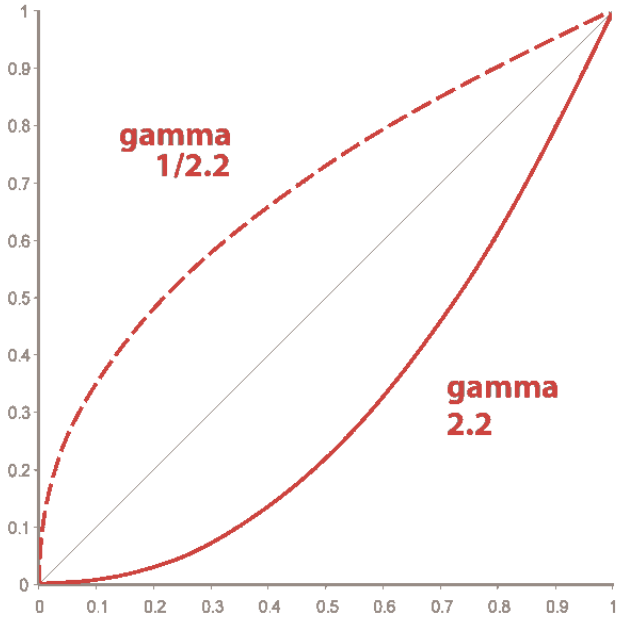

Figure 1: **Gamma Correction**

The Gamma Correction core also offers various configuration options for a designer to optimize the block RAM footprint required by the core.

Optimization Options

Various configuration settings determine how many Block RAM primitives are used. For example, sharing correction curves between two or three color channels, using 8- or 10-bit input, and single-buffering requires only a single Block RAM primitive. Table 1 and Table 2 show Block RAM usage for a number of typical configurations.

Table 1: **Block-RAM Footprint When Interpolation is Not Used**

IWIDTH	OWIDTH	DBLBUF	Look-up* Tables	Spartan-6	Virtex-6	
				RAMB16	RAMB18	RAMB36
8	8	0	1,2	2	2	0
8	8	0	3	3	3	0
8	8	1	3	3	3	0
8	10	0	1,2	2	2	0
8	10	0	3	3	3	0
8	10	1	3	3	3	0
8	12	0	1,2	2	2	0
8	12	0	3	3	3	0
8	12	1	3	3	3	0
10	8	0	1,2	2	2	0
10	8	0	3	3	3	0
10	8	1	3	3	3	0
10	10	0	1,2	2	2	0
10	10	0	3	3	3	0
10	10	1	3	6	0	3
10	12	0	1,2	2	2	0
10	12	0	3	3	3	0
10	12	1	3	6	0	3
12	8	0	1,2	4	2	0
12	8	0	3	6	0	3
12	8	1	3	12	0	6
12	10	0	1,2	6	2	2
12	10	0	3	9	0	3
12	10	1	3	15	0	6
12	12	0	1,2	6	2	2
12	12	0	3	9	0	3
12	12	1	3	18	0	9

* The number of lookup tables used is as follows:
3: when Independent look-up tables for each Color Channel is selected
2: when Identical look-up tables for Chrominance Channels Only is selected
1: when Identical look-up tables for all Color Channels is selected

Table 2: **Block-RAM Footprint When Interpolation is Used**

IWIDTH	OWIDTH	DBLBUF	Look-up* Tables	Spartan-6	Virtex-6	
				RAMB16	RAMB18	RAMB36
12	8	0	3	3	3	0
12	8	1	3	6	6	0
12	10	0	3	3	3	0
12	10	1	3	6	6	0
12	12	0	3	3	3	0
12	12	1	3	6	6	0

* The number of lookup tables used is as follows:
3: when Independent look-up tables for each Color Channel is selected
2: when Identical look-up tables for Chrominance Channels Only is selected
1: when Identical look-up tables for all Color Channels is selected

Shared Look-Up Tables

When multiple channels require the same correction curve, a single look-up table may be shared between two or more channels. Sharing look-up tables between multiple channels reduces the number of write operations required to update the look-up tables for EDK pCore and General Purpose Processor interfaces. It also can reduce the number of Block RAM resources used (see Table 1).

Interpolation of Look-up Table Contents

When the gamma function is configured for 12-bit input data, an optional look-up table interpolation is provided to reduce the size of look-up tables and thereby the number of block RAMs. This interpolation stores every 4th sample in the look-up table (Figure 2), which can reduce the number of block RAMs used by 75%.

The Gamma Correction core supports linear interpolation, which trades off block RAM(s) for adders to implement the 1-to-4 interpolation. When used to interpolate sufficiently smooth functions, such as the power functions used for gamma correction, the interpolation error is orders of magnitude smaller than the output quantization error.

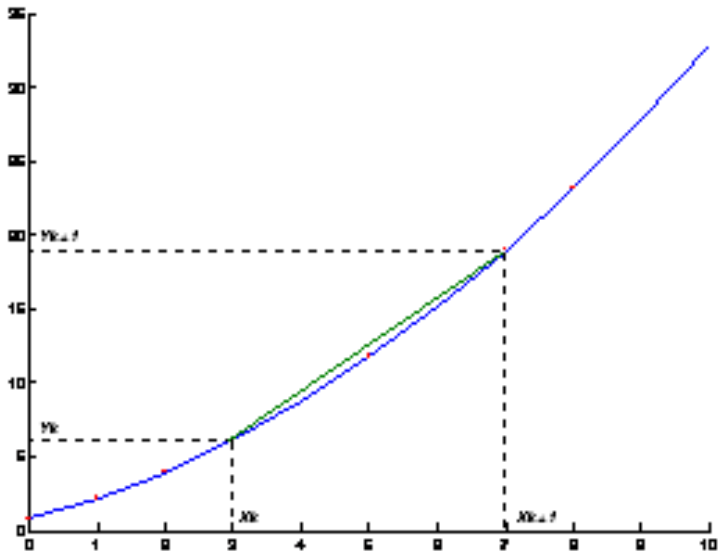

Figure 2: **Interpolation of Look-up Table Contents**

Processor Interfaces

The Gamma Correction core supports the following four processor interface options:

- EDK pCore Interface
- General Purpose Processor Interface
- Constant Interface

The Constant Interface results in a static gamma converter, where the designer has to specify the gamma coefficient (γ) or initialization function at generation time for the table initialization. The EDK pCore and General Purpose Processor interfaces use the gamma coefficient at generation time as a default for gamma table initialization and also allow run-time, dynamic updates to the gamma tables.

EDK pCore Interface

Many imaging applications have an embedded processor that can dynamically control the parameters within the core. The EDK pCore Interface creates a pCore that can be added to an EDK project as a hardware peripheral. This pCore provides a memory-mapped interface for the programmable registers within the core, which are described in Table 3.

Table 3: **EDK pCore Interface Register Descriptions**

Address Offset (hex)	Register Name	Access Type	Default Value	Description
BASEADDR + 0x00	gamma_reg0_control	R/W	0x00000001	Bit 0: Software Enable • 0 – Not enabled • 1 – Enabled Bit 1: Table Update When using Double Buffering, bit 1 is a control bit for triggering the update of the active look-up table on the next VBlank rising-edge. • 0 – Normal Operating Mode, continue to operate with current look-up table. • 1 – look-up table update. The inactive look-up table with newly written data will become the active look-up table on the next VBlank rising edge.
BASEADDR + 0x04	gamma_reg1_reset	R/W	0x00000000	Bit 0: Software Manual Reset • 0 – Not Reset • 1 – Force immediate reset, hold until bit is cleared Bit 1: Software Auto-synchronized Reset • 0 – Not Reset • 1 – Reset will occur automatically on the next rising-edge of vblank_in
BASEADDR + 0x08	gamma_reg2_status	R	0x02030000	Bits [15-0]: Not Used Bits [31-16]: DRIVER_PARAMS & DRIVER_VERSION
BASEADDR + 0x0C	gamma_reg3_addr_data	R/W	0x00000000	Bits [15-0]: Value to write to gamma look-up table Bits [31-16]: Target address in look-up table

All of the registers are readable, enabling you to verify writes or read back current values.

The Gamma Correction core has a feature that allows it to be enabled or disabled. This halts the operation of the core by blocking the propagation of all video signals. This function is controlled by setting the Software Enable, bit 0 of `gamma_reg0_control` register, to 0; the default value of Software Enable is 1 (enabled).

When using a single-buffered interface, write operations to the `gamma_reg3_data` port take effect immediately on the active look-up tables in the Gamma Correction core, and may cause image tearing if updated when the `active_video signal` = '1'.

When using a double-buffered interface, write operations to the `gamma_reg3_data` port take effect in a secondary inactive look-up table, and can occur at any time safely. When write operations are complete, assert `gamma_reg0_control` bit 1 to automatically swap the active look-up table with the inactive look-up table at the next rising edge of VBlank, which will cause the following frame to be processed using the newly updated gamma function.

The Software Reset register `gamma_reg1_reset` can be used to reset the state of the Gamma Correction core. Setting bit 0 of `gamma_reg1_reset` to 1 resets the core's state immediately, while setting bit 1 of `gamma_reg1_reset` to 1 causes a reset to occur automatically on the next rising edge of VBlank. In both cases, reset does not alter the contents of the gamma look-up tables, which can only be updated by explicitly writing new contents to the core.

Descriptions of `gamma_reg2_status` and `gamma_reg3_addr_data` are detailed in "Updating the Gamma Tables Using the EDK pCore Interfaces."

Double-Buffering

When using a Double-Buffered interface, all but the control and software reset registers are double-buffered in hardware to ensure no image tearing when gamma look-up table values are written by the processor during the active area of a frame. One buffer, the active buffer, is used actively by the core, while the inactive buffer can be updated by the processor. Double-buffering provides a more flexible and easier-to-use core because it decouples updating of the gamma tables from the video signal blanking period. This allows software to update the gamma tables at any time within the video frame, without degrading or corrupting the image quality. Using double-buffering may effectively double the number of block RAM primitives allocated by the core depending on the configuration of the core.

Note: When updating look-up tables using a double-buffered interface, it is highly recommended that the entire look-up table for all color channels be updated before the table is committed by asserting the Table Update bit (Bit 1 of gamma_reg0_control).

Due to the swapping of inactive and active look-up tables, performing only partial table updates may produce unexpected results.

Single-Buffering

The option for single-buffered registers can cause image tearing or corruption if the tables are not fully updated in the blanking period. If the end-product can tolerate partially updated gamma tables, or the processor can ensure complete updates for the entire gamma table(s) during the vertical blanking period, a single-buffered interface can be used. This single-buffer configuration offers a savings in the number of block RAMs but trades off a tighter period that registers can be updated without detrimental video effects.

Updating the Gamma Tables Using the EDK pCore Interfaces

The double- and single-buffered interfaces require that each write operation contain a valid address and data. Bits [31-16] of register `gamma_reg3_addr_data` are designated as the look-up table address, while bits [15-0] represent the number of words to be written into the gamma look-up table(s). The valid address range for the data depends on the input width, number of shared look-up tables, and whether interpolation is used, as shown in Table 4.

Table 4: **Valid Address Ranges for Gamma Correction Look-up Table Contents**

Input Width	Look-up Tables*	Interpolation	Red/U/Cr Baseaddr, Range	Blue/V/Cb Baseaddr, Range	Green/Y Baseaddr, Range
8	3	0	0x0000, 0x00FF	0x0100, 0x01FF	0x0200, 0x02FF
8	2	0	0x0000, 0x00FF	N/A	0x0200, 0x02FF
8	1	0	0x0000, 0x00FF	N/A	N/A
10	3	0	0x0000, 0x03FF	0x0400, 0x07FF	0x0800, 0x0BFF
10	2	0	0x0000, 0x03FF	N/A	0x0800, 0x0BFF
10	1	0	0x0000, 0x03FF	N/A	N/A
12	3	0	0x0000, 0x0FFF	0x1000, 0x1FFF	0x2000, 0x2FFF
12	2	0	0x0000, 0x0FFF	N/A	0x2000, 0x2FFF
12	1	0	0x0000, 0x0FFF	N/A	N/A
12	3	1	0x0000, 0x03FF	0x0400, 0x07FF	0x0800, 0x0BFF
12	2	1	0x0000, 0x03FF	N/A	0x0800, 0x0BFF
12	1	1	0x0000, 0x03FF	N/A	N/A

* The number of lookup tables used is as follows:
3: when Independent look-up tables for each Color Channel is selected
2: when Identical look-up tables for Chrominance Channels Only is selected
1: when Identical look-up tables for all Color Channels is selected

EDK pCore API Functions

This section describes the EDK pCore API functions.

GAMMA_Enable(uint32 BaseAddress);

- This macro enables a Gamma instance.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h).

GAMMA_Disable(uint32 BaseAddress);

- This macro disables a Gamma instance.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h).

GAMMA_RegUpdateEnable(uint32 BaseAddress);

- When using a double-buffered interface, enabling the RegUpdateEnable causes the gamma correction look-up table to update on the next rising edge of `VBlank_in`. This action causes the new values written to the inactive look-up table to become the active look-up table when `VBlank_in` rising edge occurs. The user must manually disable the register update after a sufficient amount of time to prevent continuous updates.
- This function only works when the Gamma core is enabled.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h)

GAMMA_RegUpdateDisable(uint32 BaseAddress);

- When using a double-buffered interface, disabling the Register Update prevents the gamma correction look-up table from updating. It is recommended that the Register Update be disabled while writing to the inactive look-up table in the gamma correction core, until the write operation is complete. While disabled, writes to the inactive look up table are stored, but do not effect the core's behavior.
- This function only works when the Gamma core is enabled.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h)

GAMMA_Reset(uint32 BaseAddress);

- This macro resets a Gamma instance. This reset effects the core immediately, and may cause image tearing. This reset resets the Gamma's configuration registers, and holds the core's outputs in their reset state until GAMMA_ClearReset() is called.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h)

GAMMA_ClearReset(uint32 BaseAddress);

- This macro clears the Gamma's reset flag (which is set using GAMMA_Reset(), and returns it to normal operation. This ClearReset effects the core immediately, and may cause image tearing.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h).

GAMMA_AutoSyncReset(uint32 BaseAddress);

- This macro resets a Gamma instance, but differs from GAMMA_Reset() in that it automatically synchronizes to the `VBlank_in` input of the core to prevent tearing. On the next rising-edge of `VBlank_in` following a call to GAMMA_AutoSyncReset(), all of the core's configuration registers and outputs are reset, then the reset flag is immediately released, allowing the core to immediately resume default operation.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h)

GAMMA_ReadReg(uint32 BaseAddress, uint32 RegOffset)

- Read the given register.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h).
- RegOffset is the register offset of the register (defined in Table 5).
- This macro returns the 32-bit unsigned integer value of the register.

GAMMA_WriteReg(uint32 BaseAddress, uint32 RegOffset, uint32 Data)

- Write the given register.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h).
- RegOffset is the register offset of the register (defined in Table 5).
- Data is the 32-bit value to write to the register.

GAMMA_GetIWIDTH(uint32 BaseAddress)

- Read the Gamma's instantiation parameter IWIDTH from the Status register.
- Return the corresponding integer value.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h)

GAMMA_GetOWidth(uint32 BaseAddress)

- Read the Gamma's instantiation parameter OWIDTH from the Status register.
- Return the corresponding integer value.
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h)

GAMMA_GetINTPOL(uint32 BaseAddress)

- Read the Gamma's instantiation parameter INTPOL from the Status register.
- Return the corresponding integer value (1 if interpolation is used, 0 otherwise).
- BaseAddress is the Xilinx EDK base address of the Gamma core (from xparameters.h)

General Purpose Processor Overview

The General Purpose Processor interface exposes the control, status, address and data registers as ports. This option is very useful for users designing a system with a user-defined bus interface (decoding logic and register banks) to an arbitrary processor. The ports for the General Purpose Processor Interface are described in detail in "Core Symbol and Port Descriptions."

Constant Interface Overview

The Constant Interface assumes the gamma look-up table values are constants, which eliminates the processor interface. The core is not programmable. It can be reset and enabled/disabled using the SCLR and CE ports. The ports for the Constant Interface are described in detail in "Core Symbol and Port Descriptions."

CORE Generator – Graphical User Interface

The Gamma Correction core is easily configured to meet the user's specific needs through the CORE Generator GUI. This section provides a quick reference to parameters that can be configured at generation time. Figure 3 shows the main Gamma Correction screen.

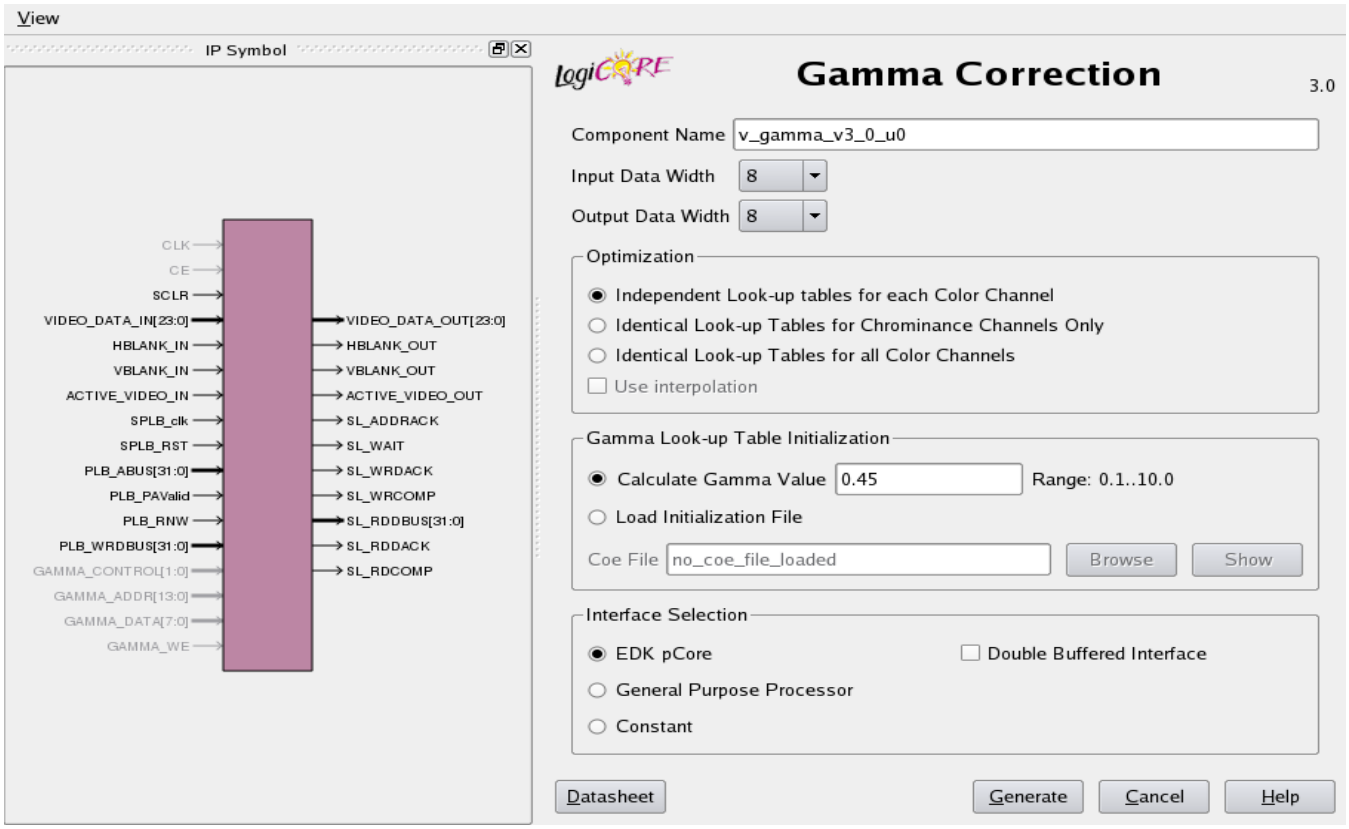

Figure 3: **Gamma Correction Main Screen**

The main screen displays a representation of the IP symbol on the left side, and the parameter assignments on the right side, which are described as follows:

- **Component Name:** The component name is used as the base name of output files generated for the module. Names must begin with a letter and must be composed from characters: a to z, 0 to 9 and "_". **The name v_gamma_v3_0 is not allowed.**
- **Input Data Width (IWIDTH)**: Specifies the bit width of the input color channel for each component. Permitted values are 8, 10 and 12 bits.
- **Output Data Width (OWIDTH)**: Specifies the bit width of the output color channel for each component. Permitted values are 8, 10 and 12 bits.
- **Optimization**: Specifies options to reduce memory usage.
  - **Independent Look-up Tables for each Color Channel:** When selected, each color channel uses a separate look-up table, permitting each channel to implement a distinct function. This option requires the most Block RAM resources (see "Optimization Options").
  - **Identical Look-up Tables for Chrominance Channels Only:** When selected, the chrominance channels (Cr, Cb) will share the same look-up table contents. (This also applies to the U and V channels for YUV or the Red and Blue channels of RGB). When two channels can use the same look-up table, the required number of write operations to modify the function stored in the look-up tables is reduced, and in some cases the number of Block RAM resources required is reduced (see "Optimization Options").
  - **Identical Look-up Tables for all Color Channels:** When selected, the red, green, and blue (or luminance and chrominance channels) all share the same look-up table contents. When all channels can use the same look-up tables, the required number of write operations to modify the function stored in the look-up tables is reduced, and in some cases the number of Block RAM resources required is reduced (see "Optimization Options").
  - **Use Interpolation:** Interpolation is used to reduce block RAM counts when using 12-bit input from 4k entries per look-up table to only 1k per look-up table (see "Optimization Options").
- **Gamma Look-Up Table Initialization**
  - **Calculate Gamma Value:** When selected, specifies the gamma value for initializing the look-up tables. Permitted values are floating-point values from 0.1 to 10.
  - **Load Initialization File:** When selected, the Load Initialization File feature allows a custom COE file to be loaded which specifies the contents of the gamma look-up tables. This permits the Gamma Correction core to be used to implement any function for a variety of tasks.
- **Interface Selection**: As described previously in this document, this option allows for the configuration of four different interfaces for the core.
  - **Double-Buffered Interface:** Double-buffering is used to eliminate tearing of the output images by writing to an inactive look-up table, then providing the ability to swap inactive and active look-up tables. This feature is only available for the EDK pCore and General Purpose Processor Interfaces. However, using this feature may increase the memory footprint of the core. See "Double-Buffering" in the "EDK pCore Interface" section of "Processor Interfaces", and "Optimization Options".
  - **EDK pCore**: CORE Generator generates a pCore that can be easily inserted into an EDK project as a hardware peripheral. Gamma table values are reprogrammable via the PLB bus interface. Note that the Gamma Correction pCore cannot be modified from within the EDK, and must be regenerated from the CORE Generator to modify the core's configuration.
  - **General Purpose Processor Interface:** CORE Generator software will generate a set of ports to be used to program the core. See the"General Purpose Processor Interface".
  - **Constant Interface:** The gamma tables are constant, and therefore no programming is necessary.

Core Symbol and Port Descriptions

As discussed previously, the Gamma Correction core can be configured with three different interface options, each resulting in a slightly different set of ports. The Gamma Correction IP core uses a set of signals that is common to all of the Xilinx Video IP cores called the Xilinx Streaming Video Interface (XSVI). The XSVI signals are common to all interface options and are shown in Figure 4 and described in Table 5.

Xilinx Streaming Video Interface

The Xilinx Streaming Video Interface (XSVI) is a set of signals common to all of the Xilinx video cores used to stream video data between IP cores. XSVI is also defined as an Embedded Development Kit (EDK) bus type so that the tool can automatically create input and output connections to the core. This definition is embedded in the pCORE interface provided with the IP, and it allows an easy way to cascade connections of Xilinx Video Cores. The Gamma Correction IP core uses the following subset of the XSVI signals:

- video_data
- vblank
- hblank
- active_video

Other XSVI signals on the XSVI input bus, such as `video_clk, vsync, hsync, field_id,` and `active_chr` do not affect the function of this core.

Note: These signals are neither propagated, nor driven on the XSVI output of this core.

The following is an example EDK Microprocessor Peripheral Definition (.MPD) file definition. IWIDTH and OWIDTH are the values you selected when you generated the IP in CORE Generator (i.e., 8, 10, or 12).

Input Side:

```
BUS_INTERFACE BUS = XSVI_GAMMA_IN, BUS_STD = XSVI, BUS_TYPE = TARGET

PORT active_video_in  = active_video, BUS = XSVI_GAMMA_IN, DIR = IN,
PORT hblank_in        = hblank,       BUS = XSVI_GAMMA_IN, DIR = IN
PORT vblank_in        = vblank,       BUS = XSVI_GAMMA_IN, DIR = IN
PORT video_data_in    = video_data,   VEC = [0:((IWIDTH*3)-1)], BUS=XSVI_GAMMA_IN, DIR = IN
```

Output Side:

```
BUS_INTERFACE BUS = XSVI_GAMMA_OUT, BUS_STD = XSVI, BUS_TYPE = INITIATOR,

PORT active_video_out = active_video, BUS = XSVI_GAMMA_OUT,  DIR =OUT,
PORT hblank_out       = hblank,      BUS = XSVI_GAMMA_OUT  DIR = OUT
PORT vblank_out       = vblank,       BUS = XSVI_GAMMA_OUT, DIR = OUT,
PORT video_data_out   = video_data, VEC =[0:((OWIDTH*3)-1)],BUS = XSVI_GAMMA_OUT, DIR=OUT,
```

The Gamma Correction IP core is fully synchronous to the core clock, clk. Consequently, the input XSVI bus is expected to be synchronous to the input clock, `clk`. Similarly, to avoid clock resampling issues, the output XSVI bus for this IP is synchronous to the core clock, `clk`. The `video_clk` signals of the input and output XSVI buses are not used.

Constant Interface

As this interface does not provide additional programmability, the Constant Interface has no ports other than the Xilinx Streaming Video Interface, clk, ce, and sclr signals. The Constant Interface Core Symbol is shown in Figure 4.

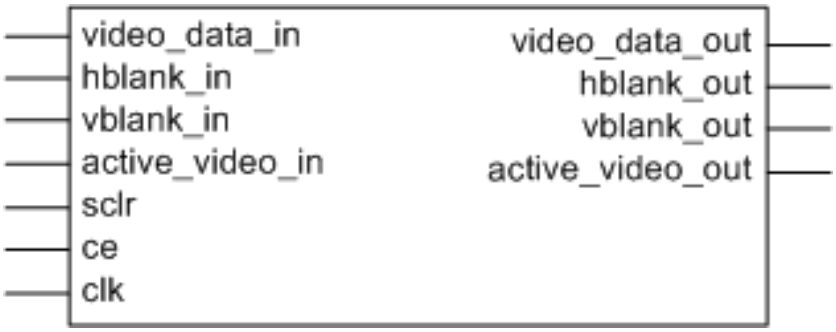

Figure 4: **Core Symbol for the Constant Interface**

Table 5: **Port Descriptions for the Constant Interface**

Port Name	Port Width	Direction	Description
video_data_in	3*IWIDTH	IN	Data input bus
hblank_in	1	IN	Horizontal blanking input
vblank_in	1	IN	Vertical blanking input
active_video_in	1	IN	Active video signal input
video_data_out	3*OWIDTH	OUT	Data output bus
hblank_out	1	OUT	Horizontal blanking output
vblank_out	1	OUT	Vertical blanking output
active_video_out	1	OUT	Active video signal output
clk	1	IN	Rising-edge clock
ce	1	IN	Clock enable (active high)
sclr	1	IN	Synchronous clear – reset (active high)

- **video_data_in**: This bus contains the three individual color inputs in the following order from MSB to LSB [red : blue : green]. Color values are expected in IWIDTH bits wide unsigned integer representation. The Gamma Correction core supports RGB, YUV, and YCrCb input formats.

Bits	3IWIDTH-1:2IWIDTH	2IWIDTH-1:IWIDTH	IWIDTH-1:0
Video Data Signals	Red/U/Cr	Blue/V/Cb	Green/Y

- **hblank_in:** The `hblank_in` signal conveys information about the blank/non-blank regions of video scan lines. This signal is not actively used in the Gamma Correction core, but passed through the core with a delay matching the latency of the corrected data.
- **vblank_in**: The `vblank_in` signal conveys information about the blank/non-blank regions of video frames, and is used by the Gamma Correction core to detect beginning of a frame, when user registers can be copied to active registers to avoid visual tearing of the image. This signal is passed through the core with a delay matching the latency of the corrected data.

- **active_video_in**: The `active_video_in` signal is high when valid data is presented at the input. This signal is not actively used in the Gamma Correction core, but passed through the core with a delay matching the latency of the corrected data.
- **clk - clock**: Master clock in the design, synchronous with, or identical to the video clock.
- **ce - clock enable**: Pulling `CE` low suspends all operations within the core. Outputs are held, and no input signals are sampled, except for reset (`SCLR` takes precedence over `CE`).
- **sclr - synchronous clear**: Pulling `SCLR` high results in resetting all output ports to zero or their default values. Internal registers within the XtremeDSP™ slice and D-flip-flops are cleared. However, the core uses `SRL16/SRL32` based delay lines for `hblank_out`, `vblank_out`, and `active_video_out` generation, which are not cleared by `SCLR`. This may result in non-zero outputs after `SCLR` is deasserted, until the contents of `SRL16/SRL32`s are flushed. Unwanted results can be avoided if `SCLR` is held active for 16/32 clock cycles until `SRL16s/SRL32s` are flushed. Also, SCLR does not reset the contents of the gamma look-up tables within the core.
- **video_data_out:** This bus contains video output in the same order as `video_data_in`. Color values are represented as OWIDTH bits wide unsigned integers. The format of the output (RGB, YUV, or YCrCb) will match the input.

Bits	3*OWIDTH-1:2*OWIDTH	2*OWIDTH-1:OWIDTH	OWIDTH-1:0
Video Data Signals	Red/U/Cr	Blue/V/Cb	Green/Y

- **hblank_out** and **vblank_out**: The corresponding input signals are delayed so blanking outputs are in phase with the video data output, maintaining the integrity of the video stream. The blanking outputs are connected to the corresponding inputs via delay-lines matching the propagation delay of the video processing pipe. Unwanted blanking inputs should be tied high, and corresponding outputs left unconnected, which will result in the trimming of any unused logic within the core.
- **active_video_out**: The `active_video_out` signal is high when valid data is present at the output. The `active_video_out` signal is connected to `active_video_in` via delay-lines matching the propagation delay of the video processing pipe. The `active_video` signal does not affect the processing behavior of the core. Asserting or deasserting it will not stall processing or the video stream, nor will it force video outputs to zero.

Single- and Double-Buffered EDK pCore Interfaces

The Single- and Double-Buffered EDK pCore Interfaces generate additional Processor Local Bus (PLB4.6) interface ports in addition to the `clk`, `ce`, `sclr`, and XSVI signals. The PLB bus signals are automatically connected when the generated pCore is inserted into an EDK project. The Core Symbol for the Single- and Double-Buffered EDK pCore Interfaces is shown in Figure 5. The Xilinx Streaming Video Interface `clk`, `ce`, and `sclr`, signals are described in the previous section. For more information on the PLB bus signals, see Processor Local Bus (PLB) v4.6.

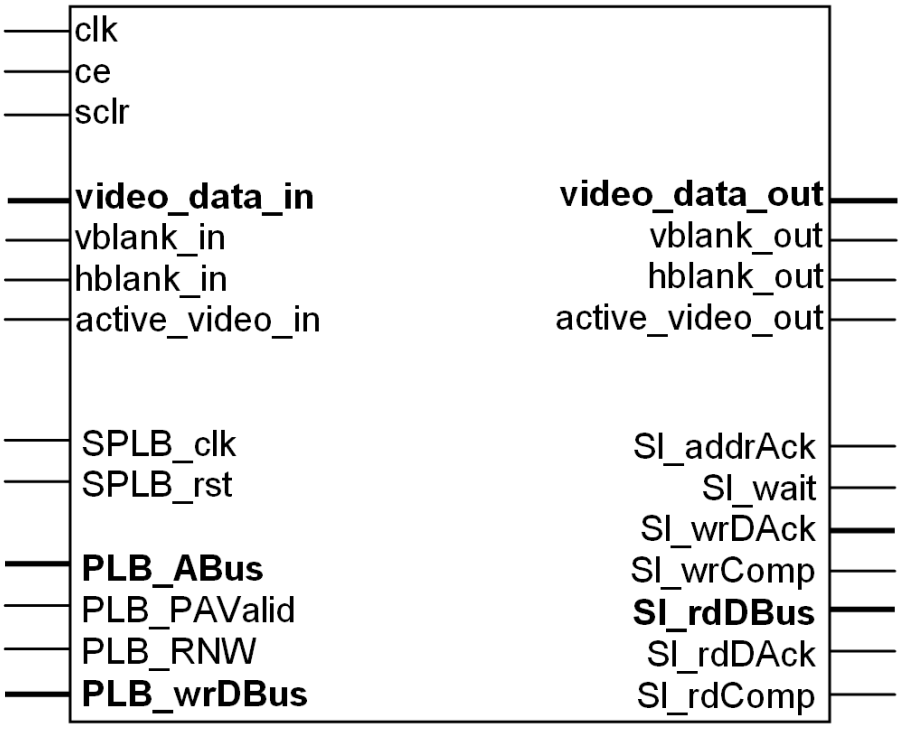

Figure 5: **Core Symbol for the EDK pCore Interface**

General Purpose Processor Interface

The General Purpose Processor Interface exposes the address and data buses, necessary to initialize look-up table contents, as ports. The Core Symbol for the General Purpose Processor Interface is shown in Figure 6. The Xilinx Streaming Video Interface is described in the previous section. The ports are described in Table 6.

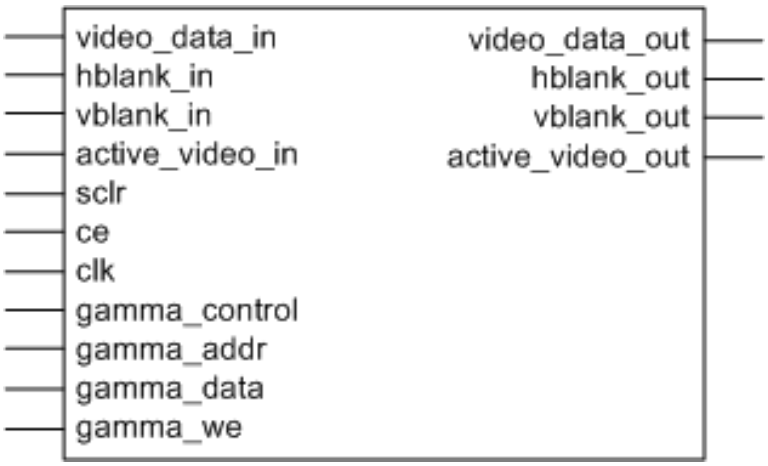

Figure 6: **Core Symbol for the General Purpose Processor Interface**

Table 6: **Optional Ports for the General Purpose Processor Interface**

Port Name	Port Width	Direction	Description
gamma_control	2	IN	Look-up table update, software enable
gamma_addr	14	IN	Look-up table address bus (2 additional MSBs are necessary only when no optimizations are used)
gamma_data	OWIDTH	IN	Look-up table initialization data corresponding to address provided by gamma_addr
gamma_we	1	IN	Look-up table write enable

- **gamma_control:** This port conveys the functionality of the EDK interface register `gamma_reg0_control`, defined in Table 3. The Software Enable bit can enable (1) or disable (0) core functionality. The Look-up Table Update bit, if asserted (1), will trigger the update of the active look-up tables with the contents of the inactive look-up tables at the next rising edge of VBlank.
- **gamma_addr**: Address bus for writing the gamma correction look-up table contents. This bus is always 14 bits wide, but the valid address range for accessing the internal look-up table contents varies depending on configuration. Table 5 shows the valid address ranges for each configuration:
- **gamma_data:** Data bus for writing internal look-up table contents.
- **gamma_we:** Active '1' on `gamma_we` enables writing `gamma_data` into the look-up table address specified by `gamma_addr`.

Control Signals and Timing

The propagation delay of the Gamma Correction core is always four clock cycles. Deasserting `CE` suspends processing, which may be useful for data-throttling, to temporarily cease processing of a video stream to match the delay of other processing components.

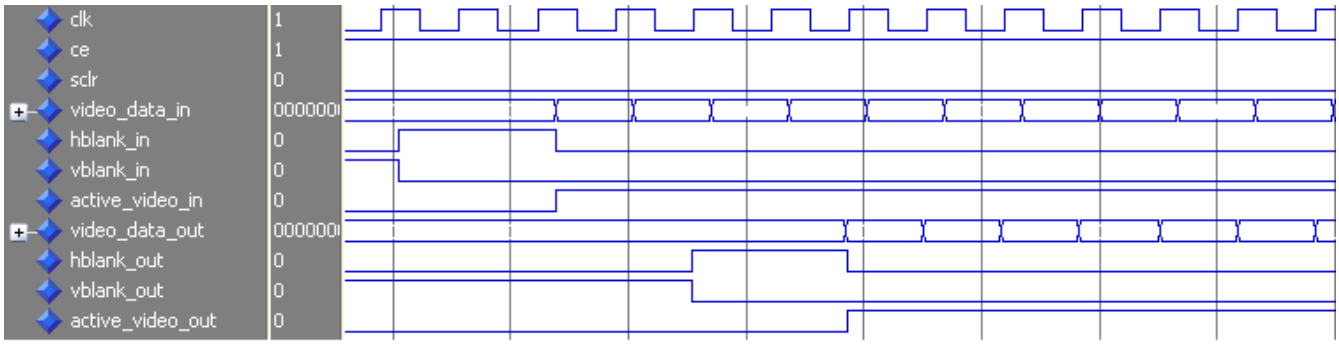

Figure 7: **Timing Example**

Core Resource Utilization and Performance

For an accurate measure of the usage of device resources (for example, block RAMs, flip-flops, and look-up tables) for a particular instance, click **View Resource Utilization** in CORE Generator after generating the core.

Information presented in Table 7, Table 8, Table 9, and Table 10 is a guideline to the resource utilization of the Gamma Correction core (with the Constant Interface, Independent Look-up tables for each Color Channel, not using double buffering) for Virtex-5, Spartan-3ADSP, Spartan-6, and Virtex-6 FPGA families. The block RAM count depends on parameterization, and this core does not use any xTremeDSP slices, dedicated I/O, or clock resources. The design was tested using Xilinx ISE® v12.3i tools with default tool options.

Table 7: **Resource Utilization and Target Speed for Spartan-3A DSP - xc3sd3400a,fg676,-5 Speedfile (PRODUCTION 1.33 2010-09-13)**

Input Width	Output Width	Interpolation	FFs	Slices	BRAM18	Clock Frequency (MHz)
8	8	No	81	57	3	324
8	10	No	93	66	3	317
8	12	No	105	75	3	317
10	8	No	87	60	3	324
10	10	No	99	69	3	317
10	12	No	111	78	3	317
12	8	No	93	63	6	330
12	10	No	105	72	9	305
12	12	No	117	81	9	293
12	8	Yes	129	138	3	167
12	10	Yes	150	157	3	161
12	12	Yes	168	173	3	154

Table 8: **Resource Utilization and Target Speed for Spartan-6 - xc6slx150,fgg676,C,-2 Speedfile (PRELIMINARY 1.12 2010-09-13)**

Input Width	Output Width	Interpolation	FFs	LUT6-FF Pairs	BRAM16	BRAM8	Clock Frequency (MHz)
8	8	No	60	44	0	3	243
8	10	No	66	54	0	3	256
8	12	No	72	60	0	3	256
10	8	No	66	57	0	3	256
10	10	No	72	59	3	0	256
10	12	No	78	56	3	0	193
12	8	No	72	66	6	0	200
12	10	no	84	70	6	3	225
12	12	No	72	61	9	0	230
12	8	Yes	81	176	0	3	149
12	10	Yes	87	194	3	0	136
12	12	Yes	93	218	3	0	136

Table 9: **Resource Utilization and Target Speed for Virtex-5 - xc5vlx50t,ff665,-1 Speedfile (PRODUCTION 1.71 2010-09-13, STEPPING level 0)**

Input Width	Output Width	Interpolation	FFs	LUT6-FF Pairs	BRAM36	BRAM18	Clock Frequency (MHz)
8	8	No	57	57	0	3	444
8	10	No	63	63	0	3	444
8	12	No	69	69	0	3	444
10	8	No	63	63	0	3	444
10	10	No	69	69	0	3	444
10	12	No	75	75	0	3	444
12	8	No	69	69	3	0	444
12	10	No	75	75	3	3	444
12	12	No	81	81	3	3	444
12	8	Yes	81	142	0	3	262
12	10	Yes	87	149	0	3	267
12	12	Yes	93	165	0	3	262

Table 10: **Resource Utilization and Target Speed for Virtex-6 - xc6vsx315t,ff1156,C,-1 Speedfile (PRODUCTION 1.10 2010-09-13)**

Input Width	Output Width	Interpolation	FFs	LUT6-FF Pairs	BRAM36	BRAM18	Clock Frequency (MHz)
8	8	No	60	55	0	3	400
8	10	No	66	57	0	3	400
8	12	No	72	55	0	3	400
10	8	No	66	64	0	3	400
10	10	No	72	62	0	3	400
10	12	No	78	69	0	3	400
12	8	No	72	73	3	0	400
12	10	No	78	77	3	3	400
12	12	No	84	79	3	3	400
12	8	Yes	81	184	0	3	274
12	10	Yes	87	213	0	3	274
12	12	Yes	96	226	0	3	250

Known Issues

For for the latest Known Issues see XTP025.

References

1. Processor Local Bus (PLB) v4.6

Support

Xilinx provides technical support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

License Options

The Gamma Correction core provides the following three licensing options:

- Simulation Only
- Full System Hardware Evaluation
- Full

After installing the required Xilinx ISE software and IP Service Packs, choose a license option.

Simulation Only

The Simulation Only Evaluation license key is provided with the Xilinx CORE Generator tool. This key lets you assess core functionality with either the example design provided with the Gamma Correction core, or alongside your own design and demonstrates the various interfaces to the core in simulation. (Functional simulation is supported by a dynamically generated HDL structural model.)

No action is required to obtain the Simulation Only Evaluation license key; it is provided by default with the Xilinx CORE Generator software.

Full System Hardware Evaluation

The Full System Hardware Evaluation license is available at no cost and lets you fully integrate the core into an FPGA design, place-and-route the design, evaluate timing, and perform functional simulation of the Gamma Correction core using the example design and demonstration test bench provided with the core.

In addition, the license key lets you generate a bitstream from the placed and routed design, which can then be downloaded to a supported device and tested in hardware. The core can be tested in the target device for a limited time before timing out (ceasing to function), at which time it can be reactivated by reconfiguring the device.

To obtain a Full System Hardware Evaluation license, do the following:

1. Navigate to the product page for this core.
2. Click Evaluate.
3. Follow the instructions to install the required Xilinx ISE software and IP Service Packs.

Full

The Full license key is available when you purchase the core and provides full access to all core functionality both in simulation and in hardware, including:

- Functional simulation support
- Full implementation support including place and route and bitstream generation
- Full functionality in the programmed device with no time outs

To obtain a Full license key, you must purchase a license for the core. Click on the "Order" link on the Xilinx.com IP core product page for information on purchasing a license for this core. After doing so, click the "How do I generate a license key to activate this core?" link on the Xilinx.com IP core product page for further instructions.

Installing Your License File

The Simulation Only Evaluation license key is provided with the ISE CORE Generator system and does not require installation of an additional license file. For the Full System Hardware Evaluation license and the Full license, an email will be sent to you containing instructions for installing your license file. Additional details about IP license key installation can be found in the ISE Design Suite Installation, Licensing and Release Notes document.

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
04/24/09	1.0	Initial Xilinx release.
07/23/10	2.0	Updated for Core Version 2.0.
09/21/10	3.0	Updated for Core Version 3.0.

Notice of Disclaimer