LogiCORE IP YCrCb to RGB Color-Space Converter v3.0

DS659 July 23, 2010 **Product Specification**

Introduction

The YCrCb to RGB Color-Space Converter is a simplified 3x3 matrix multiplier converting three input color samples to three output samples in a single clock cycle. The optimized structure uses only four XtremeDSP™ slices by taking advantage of the dependencies between coefficients in the conversion matrix of most YCrCb or YUV to RGB standards.

Features

- Built in support for:
  - SD (*ITU 601*)
  - HD (*ITU 709*) PAL
  - HD (*ITU 709*) NTSC
  - YUV
- Support for user-defined conversion matrices
- Efficient use of DSP blocks
- 8-, 10-, and 12-bit input and output precision
- Delay match support for up to three sync signals
- For use with Xilinx CORE Generator™ software 12.2 or later

Applications

- Pre-processing block for image sensors
- Image compression
- Video surveillance
- Pre-processing block for video analytics

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/64-bit, Windows Vista Business 32-Bit/64-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)				**Frequency**
Configuration	**LUTs**	**FFs**	**DSP Slices**	**Block RAMs**	**Max. Freq.**
Spartan-3A DSP	107	68	4	0	157
Spartan-6	95	108	4	0	196
Virtex-5	98	111	4	0	312
Virtex-6	97	108	4	0	397
Provided with Core					
Documentation	Product Specification				
Design Files	Netlists				
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 and 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. Resources listed here are for 8-bit input, 8-bit output width configurations. For more complete performance data, see "Core Resource Utilization and Performance".

Overview

A color space is a mathematical representation of a set of colors. The three most popular color models are:

- RGB or R'G'B', gamma corrected RGB, used in computer graphics
- YIQ, YUV and YCrCb used in video systems
- CMYK used in color printing

These color spaces are directly related to the intuitive notions of hue, saturation and brightness.

All color spaces can be derived from the RGB information supplied by devices such as cameras and scanners. Different color spaces have historically evolved for different applications. In each case, a color space was chosen for application-specific reasons.

A particular color space choice may be preferred because it requires less storage, bandwidth or computation in analog or digital domains.

The convergence of computers, the Internet, and a wide variety of video devices, all using different color representations, is forcing the digital designer today to convert between them. The objective is to have all inputs converted to a common color space before algorithms and processes are executed. Converters are useful for a number of markets, including image and video processing.

RGB Color Space

The red, green and blue (RGB) color space is widely used throughout computer graphics. Red, green and blue are three primary additive colors: individual components are added together to form a desired color, and are represented by a three dimensional, Cartesian coordinate system, as shown in Figure 1.

Table 1 presents the RGB values for 100% saturated color bars, a common video test signal.

Table 1: **100% RGB Color Bars**

	Normal Range	White	Yellow	Cyan	Green	Magenta	Red	Blue	Black
R	0 to 255	255	255	0	0	255	255	0	0
G	0 to 255	255	255	255	255	0	0	0	0
B	0 to 255	255	0	255	0	255	0	255	0

The RGB color space is the most prevalent choice for computer graphics because color displays use red, green and blue to create the desired color. Also, a system that is designed using the RGB color space can take advantage of a large number of existing software algorithms.

However, RGB is not very efficient when dealing with real-world images. All three components need equal bandwidth to generate arbitrary colors within the RGB color cube. Also, processing an image in the RGB color space is usually not the most efficient method. For example, to modify the intensity or color of a given pixel, all three RGB values must be read, modified and written back to the frame buffer. If the system had access to the image stored in the intensity and color format, the process would be faster.

R'G'B' Color Space

While the RGB color space is ideal to represent computer graphics, 8-bit linear-light coding performs poorly for images to be viewed [Ref 2]. It is necessary to have 12 or 14 bits per component to achieve excellent quality. The best perceptual use is made of a limited number of bits by using nonlinear coding that mimics the nonlinear response of human vision. In video, JPEG, MPEG, computing, digital photography, and many other domains, a nonlinear transfer function is applied to the RGB signals to give nonlinearly coded gamma-corrected components, denoted with symbols R'G'B'. Excellent image quality can be obtained with 10-bit nonlinear coding with a transfer function similar to that of *Rec. 709* [Ref 4] or RGB.

YUV Color Space

The YUV color space is used by the analog PAL, NTSC and SECAM color video/TV standards. In the past, black and white systems used only the luminance (Y) information. Chrominance information (U and V) was added in such a way that a black and white receiver can still display a normal black and white picture.

YCrCb (or YCbCr) Color Space

The YCrCb or YCbCr color space was developed as part of the *ITU-R BT.601* [Ref 3] during the development of a world-wide digital component video standard. YCbCr is a scaled, offset version of the YUV color space. Y has a nominal range of 16-235; Cb and Cr have a nominal range of 16-240. There are several YCbCr sampling formats, such as 4:4:4, 4:2:2 and 4:2:0.

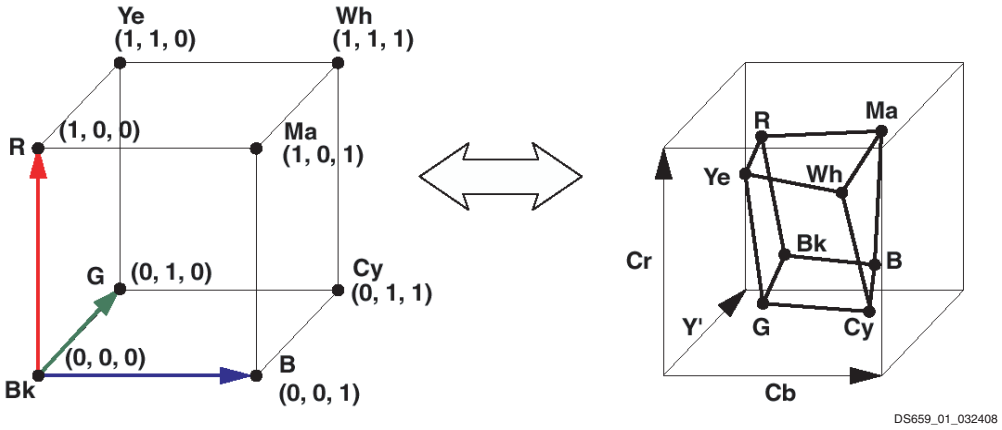

Figure 1: **RGB and YCrCb Color Representations**

Conversion Equations

Derivation of Conversion Equations

To generate the luminance (Y, or gray value) component, biometric experiments were employed to measure how the human eye perceives the intensities of the red, green and blue colors. Based on these experiments, optimal values for coefficients CA and CB were determined, such that:

$$Y = CA*R + (1 - CA - CB)*G + CB*B \qquad \text{Equation 1}$$

Actual values for CA and CB differ slightly in different standards.

Conversion from the RGB color space to luminance and chrominance (differential color components) could be described with the following equation:

$$\begin{bmatrix} Y \\ R-Y \\ B-Y \end{bmatrix} = \begin{bmatrix} CA & 1-CA-CB & CB \\ 1-CA & CA+CB-1 & -CB \\ -CA & CA+CB-1 & 1-CB \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad \text{Equation 2}$$

Coefficients CA and CB are chosen between 0 and 1, which guarantees that the range of Y is constrained between the maximum and the minimum RGB values permitted, RGB_{max} and RGB_{min} respectively.

In most practical implementations, the range of the luminance and chrominance components should be equal. There are two ways to accomplish this: the chrominance components (B-Y and R-Y) can be normalized (compressed and offset compensated), or values above and below the luminance range can be clipped/clamped.

Both clipping and dynamic range compression results in loss of information; however, the introduced artifacts are different. To leverage differences in the input (RGB) range, different standards choose different tradeoffs between clipping and normalization.

The YCrCb to RGB Color-Space Converter core supports only the conversions that fit the following general form:

$$\begin{bmatrix} Y \\ C_R \\ C_B \end{bmatrix} = \begin{bmatrix} CA & 1-CA-CB & CB \\ CC(1-CA) & CC(CA+CB-1) & CC(-CB) \\ CD(-CA) & CD(CA+CB-1) & CD(1-CB) \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} O_Y \\ O_C \\ O_C \end{bmatrix} \qquad \text{Equation 3}$$

CC and CD allow dynamic range compression for B-Y and R-Y, and constants O_Y and O_C facilitate offset compensation for the resulting CB and CR. To avoid arithmetic under- and overflows while converting from the RGB to the YCrCb domain, with RGB values in the [0.1] range, a choice for CC and CD is:

$$CC = \frac{1}{2(1-CA)} \qquad CD = \frac{1}{2(1-CB)} \qquad \text{Equation 4}$$

The YCrCb to RGB core facilitates both range de-compression and optional clipping and clamping. Range, offset, clipping and clamping levels are parameterizable.

By inverting the transformation matrix in Equation 3, the transformation from the YCrCb color space to the RGB color space can be defined as:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 1/CC & 0 \\ 1 & \dfrac{-CA}{CC(1-CA-CB)} & \dfrac{-CB}{CD(1-CA-CB)} \\ 1 & 0 & 1/CD \end{bmatrix} \begin{bmatrix} Y-O_Y \\ C_R-O_C \\ C_B-O_C \end{bmatrix} \qquad \text{Equation 5}$$

Hardware Implementation

The YCrCb to RGB color-space transformation (Equation 5) can be expressed as:

$$R = Y - O_Y + ACOEF(C_R - O_C) \quad \text{Equation 6}$$

$$G = Y - O_Y + BCOEF(C_R - O_C) + CCOEF(C_B - O_C) \quad \text{Equation 7}$$

$$B = Y - O_Y + DCOEF(C_B - O_C) \quad \text{Equation 8}$$

This cannot efficiently utilize the MADD capabilities of XtremeDSP slices. As offsets and coefficients are constants, the preceding equations can be rewritten as:

$$R = ACOEF \bullet C_R + ROFFSET + Y \quad \text{Equation 9}$$

$$G = BCOEF \bullet C_R + CCOEF \bullet C_B + GOFFSET + Y \quad \text{Equation 10}$$

$$B = DCOEF \bullet C_B + BOFFSET + Y \quad \text{Equation 11}$$

This can be directly mapped to the architecture shown in Figure 2. The blue and gray boxes represent logic blocks, which are always implemented using XtremeDSP slices.

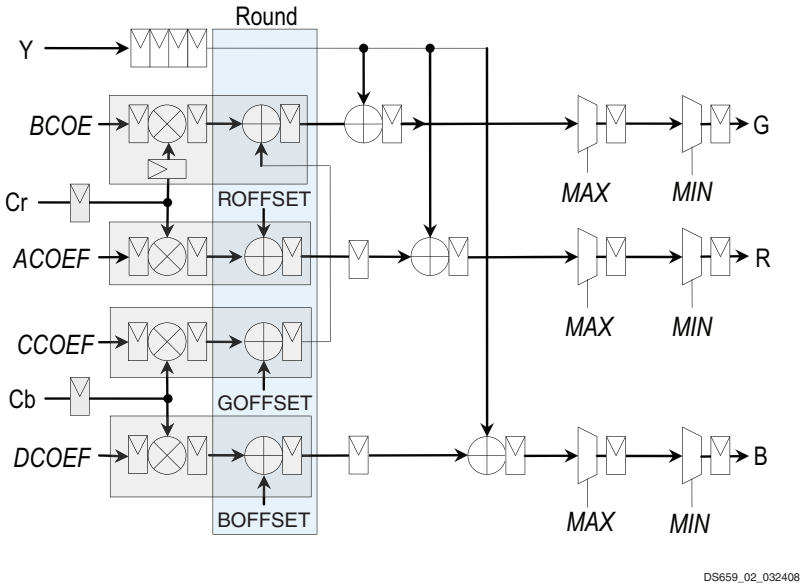

Figure 2: **YcrCb to RGB Schematic**

Assigning Values to Design Parameters

The following section specifies parameter values for some widely used standards. Most parameter values, except for COEF and OFFSET parameters, can be assigned from Table 2, Table 3 and Table 4 directly. These parameters have to be calculated, scaled and rounded before assigning integer values to corresponding VHDL parameters, using the following equations:

$$ACOEF = \frac{1}{CC} \quad \text{Equation 12}$$

$$BCOEF = \frac{-CA}{CC(1 - CA - CB)} \quad \text{Equation 13}$$

$$CCOEF = \frac{-CB}{CD(1 - CA - CB)} \quad \text{Equation 14}$$

$$DCOEF = \frac{1}{CD} \quad \text{Equation 15}$$

Coefficients are passed to the core in CWIDTH bits wide two's complement format. Y, Cr and Cb are passed as IWIDTH bits wide unsigned integers. After multiplication, in devices without DSP blocks, results are truncated to MWIDTH bits. For these families, MWIDTH is user definable, and for families with DSP blocks, MWIDTH is preset to IWIDTH + CWIDTH.

$$\text{ROUNDING\_CONST} = 2^{MWIDTH - OWDITH - 2} \quad \text{Equation 16}$$

$$ROFFSET = \text{ROUNDING\_CONST} - (ACOEF \bullet COFFSET + YOFSET) \bullet \text{SCALE\_M} \quad \text{Equation 17}$$

$$GOFFSET = \text{ROUNDING\_CONST} - ((BOEF + CCOEF) \bullet COFFSET + YOFFSET) \bullet \text{SCALE\_M} \quad \text{Equation 18}$$

$$BOFFSET = \text{ROUNDING\_CONST} - (DCOEF \bullet COFFSET + YOFFSET) \bullet \text{SCALE\_M} \quad \text{Equation 19}$$

$$\text{SCALE\_M} = 2^{MWDITH - IWIDTH - CWIDTH} \quad \text{Equation 20}$$

ITU 601 (SD) and 709 - 1125/60 (NTSC) Standard Conversion Coefficients

Table 2: **Parameterization Values for the SD (ITU 601) and NTSC HD (ITU 709) Standards**

Coefficient/ Parameter	Range		
	16-240	16-235	0-255
CA	0.299		0.2568
CB	0.114		0.0979
CC	0.713	0.7295	0.5910
CD	0.564	0.5772	
YOFFSET	$2^{OWIDTH-4}$		
COFFSET	$2^{OWIDTH-1}$		
YMAX	$240*2^{OWIDTH-8}$		$235*2^{OWIDTH-8}$
CMAX	$240*2^{OWIDTH-8}$		$235*2^{OWIDTH-8}$
YMIN	$16*2^{OWIDTH-8}$	0	$2^{OWIDTH}-1$
CMIN	$16*2^{OWIDTH-8}$	0	$2^{OWIDTH}-1$

Standard ITU 709 (HD) 1250/50 (PAL)

Table 3: **Parameterization Values for the PAL HD (ITU 709) Standard**

Coefficient/ Parameter	Input range		
	16-240	**16-235**	**0-255**
CA	0.2126		0.1819
CB	0.0722		0.0618
CC	0.6350	0.6495	0.6495
CD	0.5389	0.5512	
YOFFSET	$2^{OWIDTH-4}$		
COFFSET	$2^{OWIDTH-1}$		
YMAX	$240*2^{OWIDTH-8}$	$235*2^{OWIDTH-8}$	$2^{OWIDTH}-1$
CMAX	$240*2^{OWIDTH-8}$	$235*2^{OWIDTH-8}$	$2^{OWIDTH}-1$
YMIN	$16*2^{OWIDTH-8}$		0
CMIN	$16*2^{OWIDTH-8}$		0

YUV Standard

Table 4: **Parameterization Values for the YUV Standard**

Coefficient/ Parameter	Value		
	16-240	**16-235**	**0-255**
CA	0.299		
CB	0.114		
CC	0.877283		
CD	0.492111		
YOFFSET	$2^{OWIDTH-4}$		
COFFSET	$2^{OWIDTH-1}$		
YMAX	$240*2^{OWIDTH-8}$	$235*2^{OWIDTH-8}$	$2^{OWIDTH}-1$
CMAX	$240*2^{OWIDTH-8}$	$235*2^{OWIDTH-8}$	$2^{OWIDTH}-1$
YMIN	$16*2^{OWIDTH-8}$		0
CMIN	$16*2^{OWIDTH-8}$		0

Clipping and Clamping

Output Quantization Noise

Coefficients CC and CD in Equation 3 allow standard designers to trade off output quantization and clipping noise. Actual noise inserted depends on the probability statistics of the Cb and Cr variables, but in general if CC and CD are larger than the maximum values calculated in Equation 4, output values may clip, introducing clipping noise. However, the lower CC and CD values are chosen, the worse Cb and Cr values will use the available dynamic range, thus introducing more quantization noise. Therefore, the designer's task is to equalize output quantization and clipping noise insertion by carefully choosing CC and CD values based on knowing the statistics of Cb and Cr values. For instance, when probabilities of extreme chrominance values are small, it is beneficial to increase CC and CD values, as the extra noise inserted by occasional clipping is less than the gain in average signal power (and thus SQNR).

Output Clipping Noise

If coefficients CC and CD in Equation 3 are larger than the maximum values calculated in Equation 4, Cr and Cb output values may get larger (overflow) than the maximum or smaller (underflow) than minimum value the output representation can carry. If overflow occurs and the design does not have clipping logic, binary values wrap around and insert substantial noise to the output. If clamping/clipping logic is used, output values saturate and less noise is introduced, as shown in Figure 3.

Use of clipping and clamping increases slice count of the design by approximately 6*OWIDTH slices.

If a targeted standard limits output of values to a predefined range other than those of binary representation, such as *ITU-R BT.601-5* [Ref 3], use of clipping and clamping logic facilitates constraining output values to the predefined range by setting RGB_{max} and RGB_{min} values according to the standard specifications.

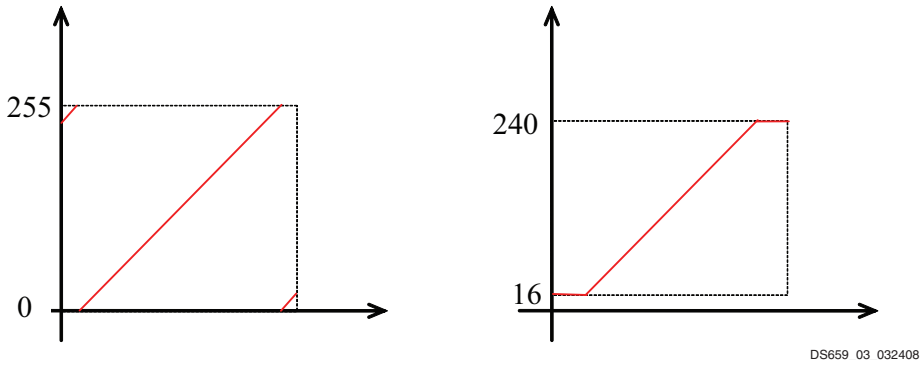

Figure 3: **Wrap-Around and Saturation**

CORE Generator – Graphical User Interface

The main screen of the Graphical User Interface (GUI) of the CORE Generator, shown in Figure 4, allows quick implementation of standard YCrCb to RGB or YUV to RGB converters without having to manually enter values from Table 2, Table 3 and Table 4. The Color-Space Converter core also supports proprietary (non-standard) converter implementations, by selecting "custom" from the Standard Selection drop-down menu, as long as the custom conversion matrix can be transformed to the form of Equation 3. The front pages of the RGB to YCrCb and the YCrCb to RGB GUIs are very similar, allowing easy generation of complementing converter pairs.

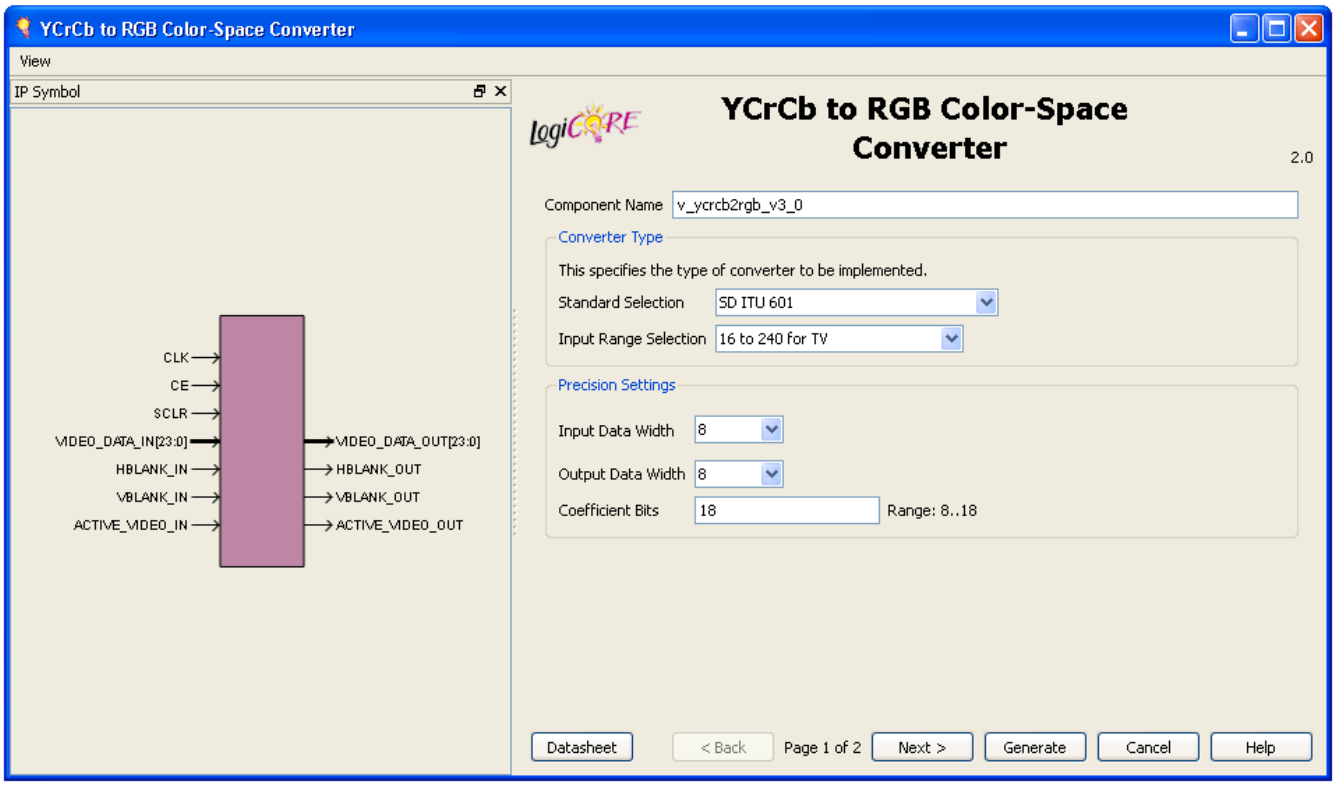

Figure 4: **Color Space Converter Main Screen**

- **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 "_".
- Converter Type
  - **Standard Selection**: Select the standard to be implemented. The offered standards are:
    - YCrCb *ITU 601* (SD)
    - YCrCb *ITU 709* (HD) 1125/60 (PAL)
    - YCrCb *ITU 709* (HD) 1250/50 (NTSC)
    - YUV
    - custom

  Selecting "custom" enables the controls on page 2 of the GUI, so conversion settings can be customized. Otherwise, page 2 only displays the parameters to implement the selected standard.

- **Output Range Selection**: This selection governs the range of outputs R, G and B by affecting the conversion coefficients as well as the clipping and clamping values. The core supports typical output ranges:
  - 16 to 235, typical for studio equipment
  - 16 to 240, typical for broadcast or television
  - 0 to 255, typical for computer graphics

  The previously-mentioned ranges are characteristic for 8-bit outputs. If 10- or 12-bit outputs are used, the ranges are extended proportionally. For example, 16 to 240 mode for 10-bit outputs will result in output values ranging from 64 to 960.

- Precision Settings
  - **Input Width** (IWIDTH): This field specifies the width of inputs Y, Cr and Cb.
  - **Output Width** (OWIDTH): This field specifies the width of outputs R, G and B.
  - **Coefficient Bits**: Sets the number of bits used to represent CA, CB, CC and CD. As displayed on Figure 2, the width of coefficients affects the width of multiplier results, which may affect the size of fabric-based adders further down the processing pipe. Reducing the coefficient size may save some slices by trading off precision with logic resources.

The Conversion Matrix, Offset Compensation, Clipping and Clamping screen (Figure 5), displays and enables editing of conversion coefficients, similar to Equation 3. Contents are editable only when "custom" is selected as the standard on page 1 (Figure 4).

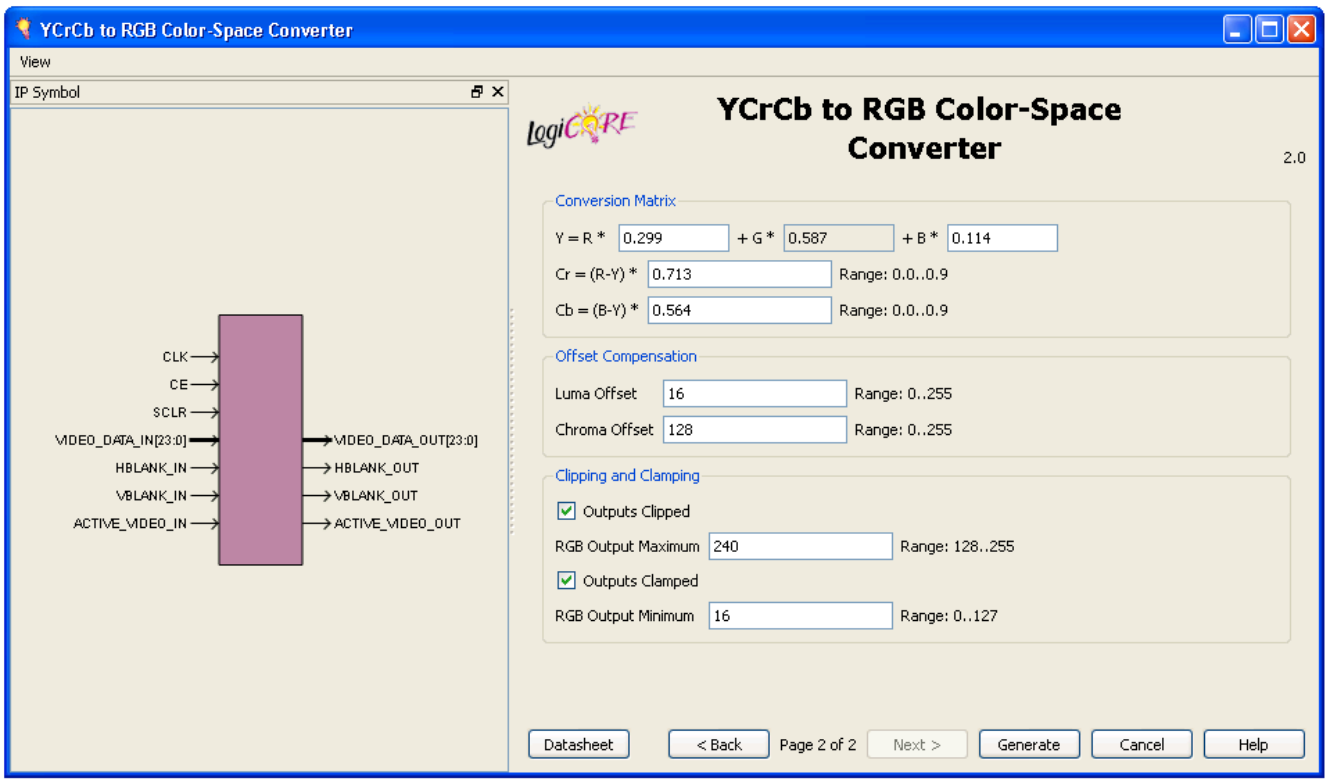

Figure 5: **Conversion Matrix, Offset Compensation, Clipping and Clamping Screen**

- **Conversion Matrix**: Enter floating-point conversion constants, ranging from 0 to 1, into the four fields representing CA, CB, CC and CD.
- **Offset Compensation**: Enter the offset compensation constants (OY and OC in Equation 17, Equation 18 and Equation 19). These constants are scaled to the output representation. If OY and OC are in the 0.0 - 1.0 range, and the output is represented as 10-bit unsigned integers, then luminance and chrominance offsets should be entered as integers in the 0 - 1023 range.
- **Outputs Clipped/Outputs Clamped**: These check boxes control whether clipping/clamping logic will be instantiated in the generated netlist. The clipping/clamping logic ensures no arithmetic wrap-arounds happen at overflows, at the expense of extra slice-based logic resources.
- **Minimum and Maximum Values**: Similar to offset values, the edit boxes take unsigned integer values in the range permitted by the current output representation.

Core Symbol and Port Descriptions

The YCrCb to RGB core uses a set of signals that is common to all of the Xilinx Video cores called the Xilinx Streaming Video Interface (XSVI). This core has no ports other than the Xilinx Streaming Video Interface, clk, ce, and sclr signals. The core symbol with the clk, ce, sclr, and XSVI signals is shown in Figure 6 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. For a complete description of this interface, see the UG762: Xilinx Streaming Video Interface User Guide. XSVI can also be defined as an Embedded Development Kit (EDK) bus type. This allows the EDK tool to automatically create input and output connections to EDK pCores that include this interface definition, and provide an easy way to cascade connections of Xilinx Video IP cores.

Note: The YCrCb to RGB core is not currently available with a pCore interface. Consequently, the core cannot be directly added to an EDK project and the tool cannot directly recognize the XSVI bus type. To use this core in an EDK project, you must import the core (see "Importing Color Space Conversion Cores into EDK as pCore with XSVI Bus") and define the signals as an XSVI bus type. The tool allows easy connection of the signals to other video IP cores with XSVI bus type.

The YCrCb to RGB IP Core uses the following subset of the XSVI signals:

- video_data
- vblank
- hblank
- active_video

Other XSVI signals on the XSVI 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.

Importing Color Space Conversion Cores into EDK as pCore with XSVI Bus

1. Parameterize and generate the core.
2. Create a wrapper file, using the provided instantiation template, either the .veo or .vho file.
3. Open EDK and follow the Create and Import Peripheral Wizard. This tool is documented in the UG111: Embedded System Tools Reference Manual.
4. Modify the .mpd file created by the Create and Import Peripheral Wizard. This file is in the Data directory created by the Create and Import Peripheral Wizard.

   You must define the XSVI bus type and appropriately tag the signals as shown in the following example. 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_YCRCB2RGB_IN, BUS_STD = XSVI, BUS_TYPE = TARGET
PORT active_video_in = active_video, DIR = IN,  BUS = XSVI_YCRCB2RGB_IN
PORT hblank_in       = hblank,       DIR = IN,  BUS = XSVI_YCRCB2RGB_IN
PORT vblank_in       = vblank,       DIR = IN,  BUS = XSVI_YCRCB2RGB_IN
PORT video_data_in   = video_data,   VEC = [0:((IWIDTH*3)-1)], DIR = IN, BUS = XSVI_YCRCB2RGB_IN
```

Output Side:

```
BUS_INTERFACE BUS = XSVI_YCRCB2RGB_OUT, BUS_TYPE = INITIATOR, BUS_STD = XSVI
PORT active_video_out = active_video, DIR = OUT, BUS = XSVI_YCRCB2RGB_OUT
PORT hblank_out       = hblank,       DIR = OUT, BUS = XSVI_YCRCB2RGB_OUT
PORT vblank_out       = vblank,       DIR = OUT, BUS = XSVI_YCRCB2RGB_OUT
PORT video_data_out   = video_data, VEC = [0:((OWIDTH*3)-1)], DIR = OUT, BUS = XSVI_YCRCB2RGB_OUT
```

For more information on the MPD format, see UG642: Platform Specification Format Reference Manual

The YCrCb to RGB 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.

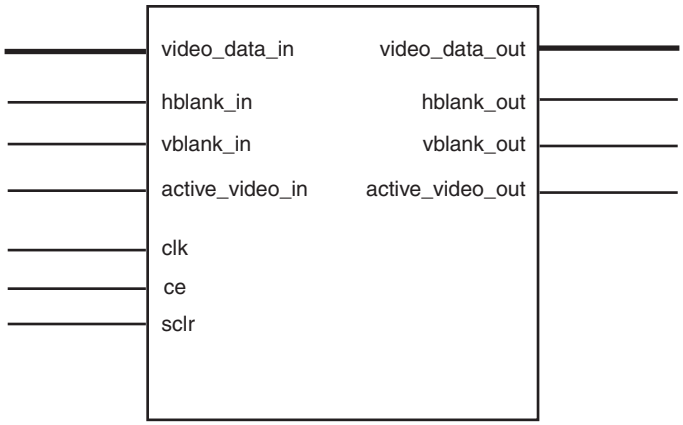

Figure 6: **Core Symbol**

Table 5: **Port Descriptions for the YCrCb to RGB Core**

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 inputs in the following order. Luminance and chrominance values are expected in IWIDTH bits wide unsigned integer representation.

Bits	3IWIDTH-1:2IWIDTH	2IWIDTH-1:IWIDTH	IWIDTH-1:0
Video Data Signals	Cb	Cr	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 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. 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 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, 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. Internal registers within the XtremeDSP slice and D-flip-flops are cleared. However, the core uses `SRL16/SRL32` based delay lines for `hblank`, `vblank` and `active_video` 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 until `SRL16/SRL32`s are flushed. `SCLR` should be held active for the duration of the processing latency of the core. The latency is defined in the "Control Signals and Timing" section.
- **video_data_out**: This bus contains the three individual RGB color outputs in the following order from MSB to LSB [red: blue: green]. Color values are expected in OWIDTH bits wide unsigned integer representation.

Bits	3OWIDTH-1:2OWIDTH	2OWIDTH-1:OWIDTH	OWIDTH-1:0
Video Data Signals	Red	Blue	Green

- **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.

Control Signals and Timing

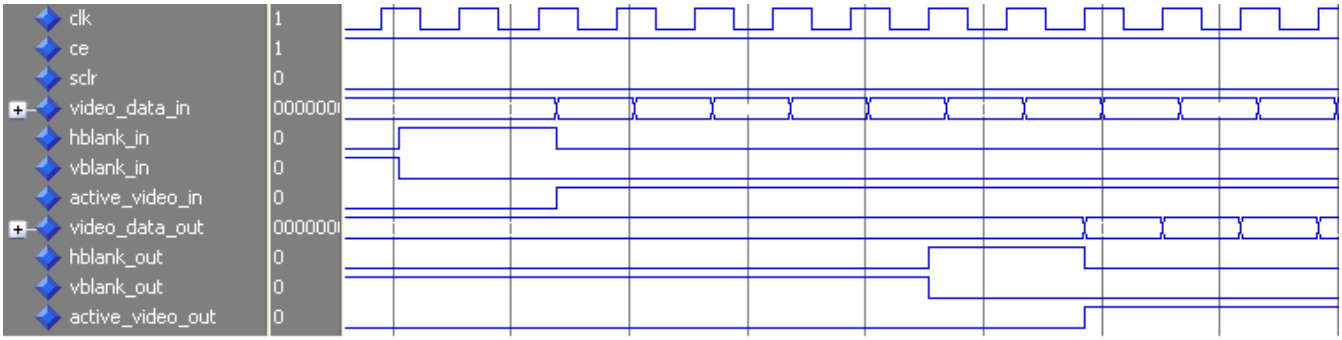

Figure 7: **Timing Example**

The propagation delay of the YcrCb to RGB core is dependent on parameterization but independent of actual signal (`video_data`,`hblank`, `vblank`, `active_video`) values. Deasserting `CE` suspends processing, which may be useful to temporarily cease processing of a video stream in order to match the delay of other processing components.

The processing latency of the core is shown in the follow equation:

```
Latency = 5+ 1(if has clipping) + 1(if has clamping)
```

This example evaluates to seven clock cycles (Figure 7) for typical cases (unless in "custom" mode the clipping and/or clamping circuits are not used).

See "Core Symbol and Port Descriptions," page 11, for an explanation on how other ports may affect the timing behavior of the core.

Core Resource Utilization and Performance

The design was tested using ISE® 12.2 tools with default tool options for characterization data. As resource counts are functions of tool options (such as XST optimizations, map packing factor), the actual resource counts corresponding to the quoted operating frequencies are also listed.

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

The YcrCb to RGB core does not use any block RAMs or dedicated I/O or clock resources.

Table 6, Table 7, Table 8, and Table 9 provide the resource usage of the YCrCb to RGB core for different families with default parameterization for all permitted input/output width combinations

Table 6: **Spartan-3A DSP - XC3SD3400A, Speed Grade=4, Resource Utilization**

Input Width	Output Width	FFs	LUTs	DSP48s
8	8	68	107	4
8	10	80	122	4
8	12	116	230	4
10	8	68	107	4
10	10	82	130	4
10	12	118	238	4
12	8	68	107	4
12	10	82	130	4
12	12	124	262	4

Table 7: **Spartan-6 - XC6SLX25T, Speed Grade=3, Resource Utilization**

Input Width	Output Width	FFs	LUTs	DSP48s
8	8	108	95	4
8	10	126	116	4
8	12	144	123	4
10	8	112	99	4
10	10	132	126	4
10	12	150	132	4
12	8	116	103	4
12	10	136	130	4
12	12	156	141	4

Table 8: **Virtex-5 - XC5VSX50T, Speed Grade=1, Resource Utilization**

Input Width	Output Width	FFs	LUTs	DSP48s
8	8	111	98	4
8	10	129	122	4
8	12	183	197	4
10	8	115	98	4
10	10	135	130	4
10	12	189	205	4
12	8	127	98	4
12	10	147	130	4
12	12	207	229	4

Table 9: **Virtex-6 - XC6VLX75T, Speed Grade=1, Resource Utilization**

Input Width	Output Width	FFs	LUTs	DSP48s
8	8	108	97	4
8	10	126	119	4
8	12	144	125	4
10	8	112	101	4
10	10	132	128	4
10	12	150	135	4
12	8	116	105	4
12	10	136	132	4
12	12	156	144	4

Known Issues

For for the latest Known Issues, see XTP025.

References

1. Jack, Keith. 2004. *Video Demystified*, 4th Edition. Burlington, MA: Newnes: pp 15-19.
2. Poynton, Charles. 2003. *Digital Video and HDTV*. San Francisco: Morgan Kaufmann: pp 302 - 321.
3. *ITU Recommendation BT.601-5*, International Telecommunication Union, 1995.
4. *ITU Recommendation BT.709-5,* International Telecommunication Union, 2002.
5. Proakis, John G., and Dimitris G. Manolakis. *Digital Signal Processing*, 3rd edition. Upper Saddle River, NJ: Prentice Hall: pp 755-756.
6. Sullivan, Gary. 2003. Approximate theoretical analysis of RGB to YCbCr to RGB conversion error. Presented for Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6), July 22-24, in Trondheim, Norway.

Support

Xilinx provides technical support for this Xilinx LogiCORE™ IP 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 YCrCb to RGB core is provided at no cost with the ISE tools. You are not required to license the core before instantiating it in your design.

Revision History

Date	Version	Revision
03/24/08	1.0	Initial Xilinx release.
04/24/09	2.0	Removed support for legacy Virtex and Spartan device families. Updated supported tools to ISE 11.1.
07/23/10	3.0	Updated devices for Spartan-6 and Virtex-6. Updated for 12.2 release. Removed licensing information. Replaced Legal Disclaimer.

Notice of Disclaimer