

Introduction

The Xilinx LogiCORE™ IP RGB to YCrCb 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 RGB to YCrCb or RGB to YUV 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

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-3	144	199	4	0	176
Spartan-6	152	186	4	0	180
Virtex-5	141	199	4	0	318
Virtex-6	168	186	4	0	369
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™, Platform Studio (XPS)				
Simulation	ModelSim v6.5c, Xilinx ISIM 12.2				
Synthesis Tools	XST 12.2				
Support					
Provided by Xilinx, Inc.					

- For a complete listing of supported devices, see the [release notes](#) for this core.
- Resources listed here are for 8-bit input, 8-bit output width configurations. For more complete performance data, see [Core Resource Utilization and Performance](#), page 16.

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.

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.

The 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 of a limited number of bits is made 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.

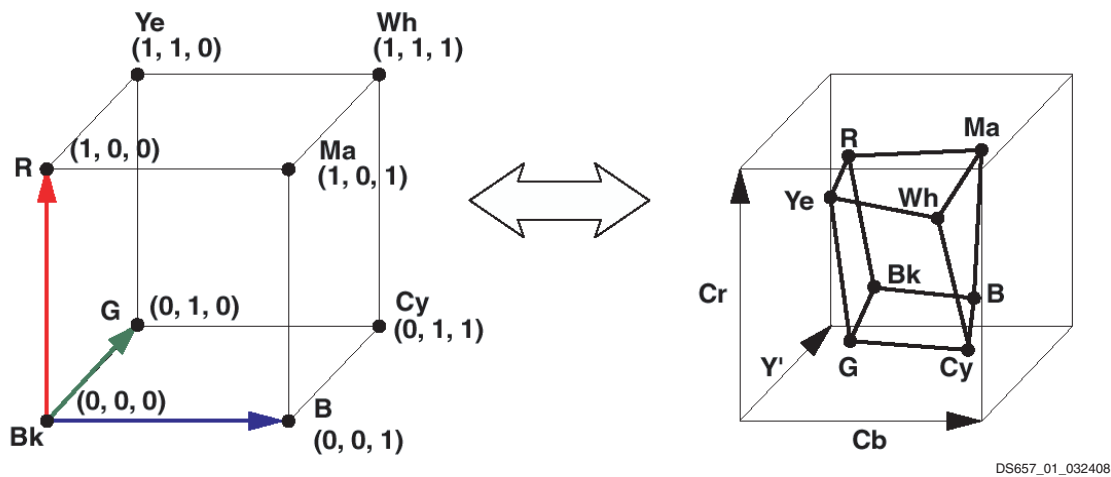


Figure 1: RGB and YCrCb Color Representations

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:

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

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 Equation 2.

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

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

The minimum and maximum values of R-Y are:

$$\begin{aligned} \min_{R-Y} &= RGB_{min} - (CA * RGB_{min} + (1 - CA - CB) * RGB_{max} + CB * RGB_{max}) = (CA - 1) * (RGB_{max} - RGB_{min}) \\ \max_{R-Y} &= RGB_{max} - (CA * RGB_{max} + (1 - CA - CB) * RGB_{min} + CB * RGB_{min}) = (1 - CA) * (RGB_{max} - RGB_{min}) \end{aligned}$$

Thus, the range of R-Y is:

$$2(CA - 1)(RGB_{max} - RGB_{min}) \tag{Equation 3}$$

Similarly, the minimum and maximum values of B-Y are:

$$\begin{aligned} \min_{B-Y} &= RGB_{min} - (CA * RGB_{max} + (1 - CA - CB) * RGB_{max} + CB * RGB_{min}) = (CB - 1)(RGB_{max} - RGB_{min}) \\ \max_{B-Y} &= RGB_{max} - (CA * RGB_{min} + (1 - CA - CB) * RGB_{min} + CB * RGB_{max}) = (1 - CB)(RGB_{max} - RGB_{min}) \end{aligned}$$

Thus, the range of B-Y is:

$$2(CB - 1)(RGB_{max} - RGB_{min}) \tag{Equation 4}$$

In most practical implementations, the range of the luminance and chrominance components should be equal. There are two ways to accomplish this: 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.

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

The RGB to YCrCb color space conversion core facilitates both range compression and optional clipping and clamping. Range, offset, clipping and clamping levels are parameterizable. The core supports conversions that fit the following general form:

$$\begin{bmatrix} Y \\ C_R \\ C_B \end{bmatrix} = \begin{bmatrix} CA & 1 - CA - CB & CB \\ CC(1 - CA) & CC(CA + CB - 1) & CC(-CB) \\ CD(-CA) & CD(CA + CB - 1) & CD(1 - CB) \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} O_Y \\ O_C \\ O_C \end{bmatrix} \tag{Equation 5}$$

CC and CD allow dynamic range compression for R-Y and B-Y, and constants OY and OC facilitate offset compensation for the resulting Y, C_B and C_R components.

Based on Equation 3 and Equation 4, to constrain the resulting chrominance components (C_B and C_R) into the [0,1] range, the chrominance offset (OC) and the chrominance range compression constants (CC, CD) should be selected as follows (OC=0.5):

$$CC = \frac{1}{2(1 - CA)(RGB_{max} - RGB_{min})} \tag{Equation 6}$$

$$CD = \frac{1}{2(1 - CB)(RGB_{max} - RGB_{min})} \tag{Equation 7}$$

When RGB values are also in the [0,1] range, using the following equations avoids arithmetic under- and overflows (OC=0.5).

$$CC = \frac{1}{2(1 - CA)} \quad CD = \frac{1}{2(1 - CB)} \tag{Equation 8}$$

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	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

Hardware Implementation

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

$$Y = CA*(R - G) + G + CB*(B - G) + YOFFSET \tag{Equation 9}$$

$$Cr = CC*(R - Y) + COFFSET \tag{Equation 10}$$

$$Cb = CD*(B - Y) + COFFSET \tag{Equation 11}$$

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

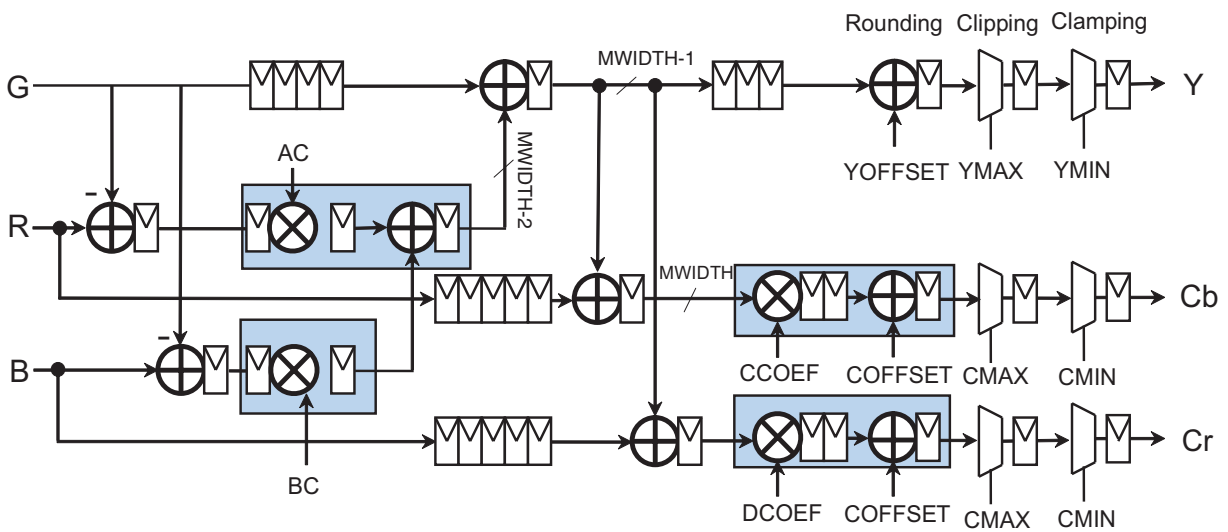


Figure 2: Application Schematic

DS657_02_032408

CORE Generator – Graphical User Interface

The main screen of the Graphical User Interface (GUI) of CORE Generator allows quick implementation of standard RGB to YCrCb or RGB to YUV converters without having to manually enter values from Tables 2, 3 and 4. The Color-Space Converter core also supports proprietary (non-standard) converter implementations. This is done 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 5.

The main screen is shown in Figure 3. Descriptions of the options provided in the GUI screens are included in this section.

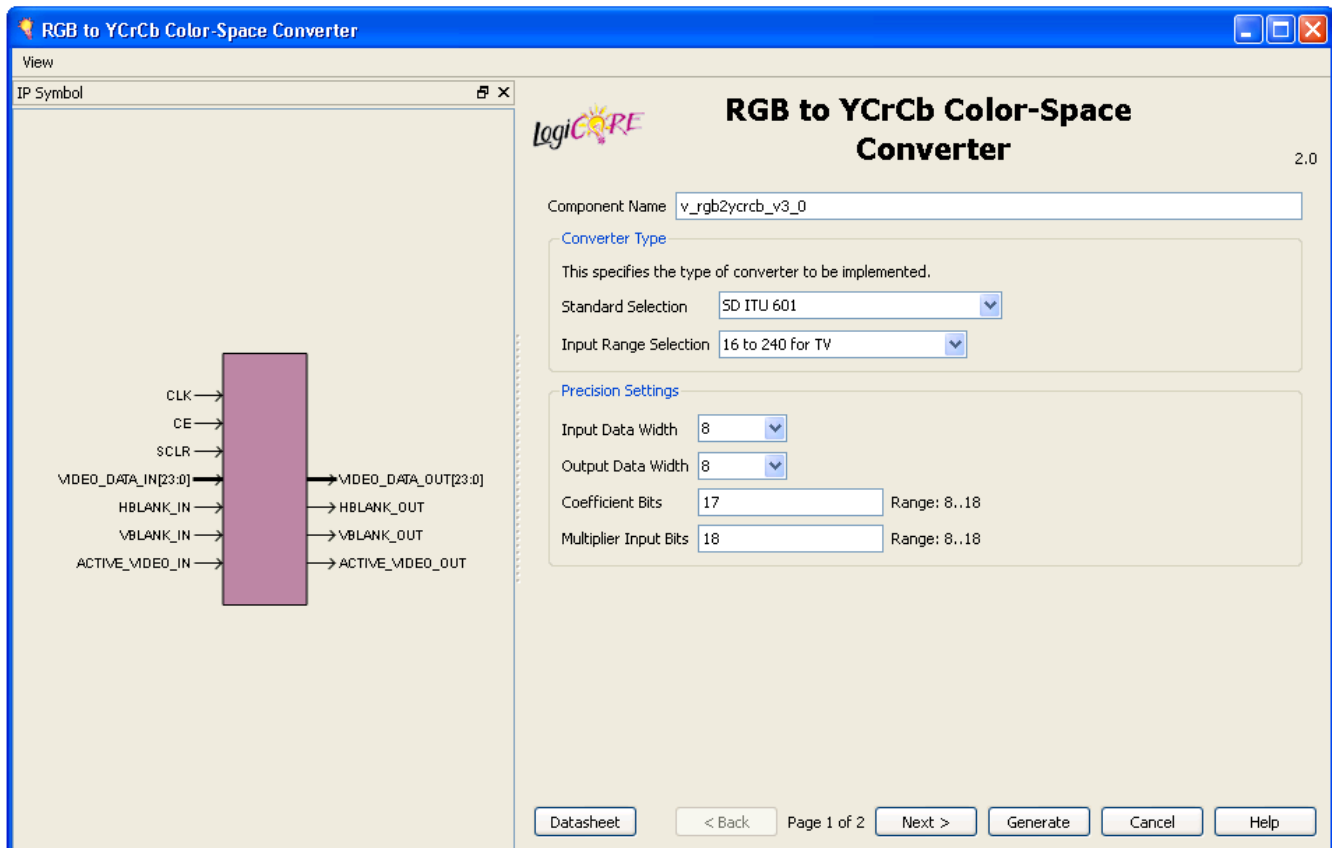


Figure 3: Color-Space Converter Main Screen

The first page of the GUI displays the following options:

- **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 be used to implement the selected standard.

- **Output Range Selection:** This selection governs the range of outputs Y, Cr and Cb by affecting the conversion coefficients as well as the clipping and clamping values. The core supports the following 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

Output clipping and clamping values are the same for luminance and chrominance channels. To set an asymmetric value, such as 16 to 235 for Cr and Cb and 16 to 240 for Y, select “custom” for the standard, then manually modify the clipping and clamping values on page 3.

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):** Specifies the width of inputs R, G and B.
 - **Output Width (OWIDTH):** Specifies the width of outputs Y, Cr and Cb.
 - **Coefficient Bits:** Sets the number of bits used to represent CA, CB, CC and CD. As displayed in [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.
 - **Multiplier Input Bits:** Allows the user to control to the width of operands (MWIDTH) for the CC and CD multipliers ([Figure 2](#)). Similar to the coefficient width setting, this advanced control allows trading off precision and logic resource counts.

The Conversion Matrix, Offset Compensation, Clipping and Clamping screen (Figure 4) displays and enables editing of conversion coefficients, similar to Equation 9, Equation 10 and Equation 11. Contents are editable only when “custom” is selected as the standard on page 1.

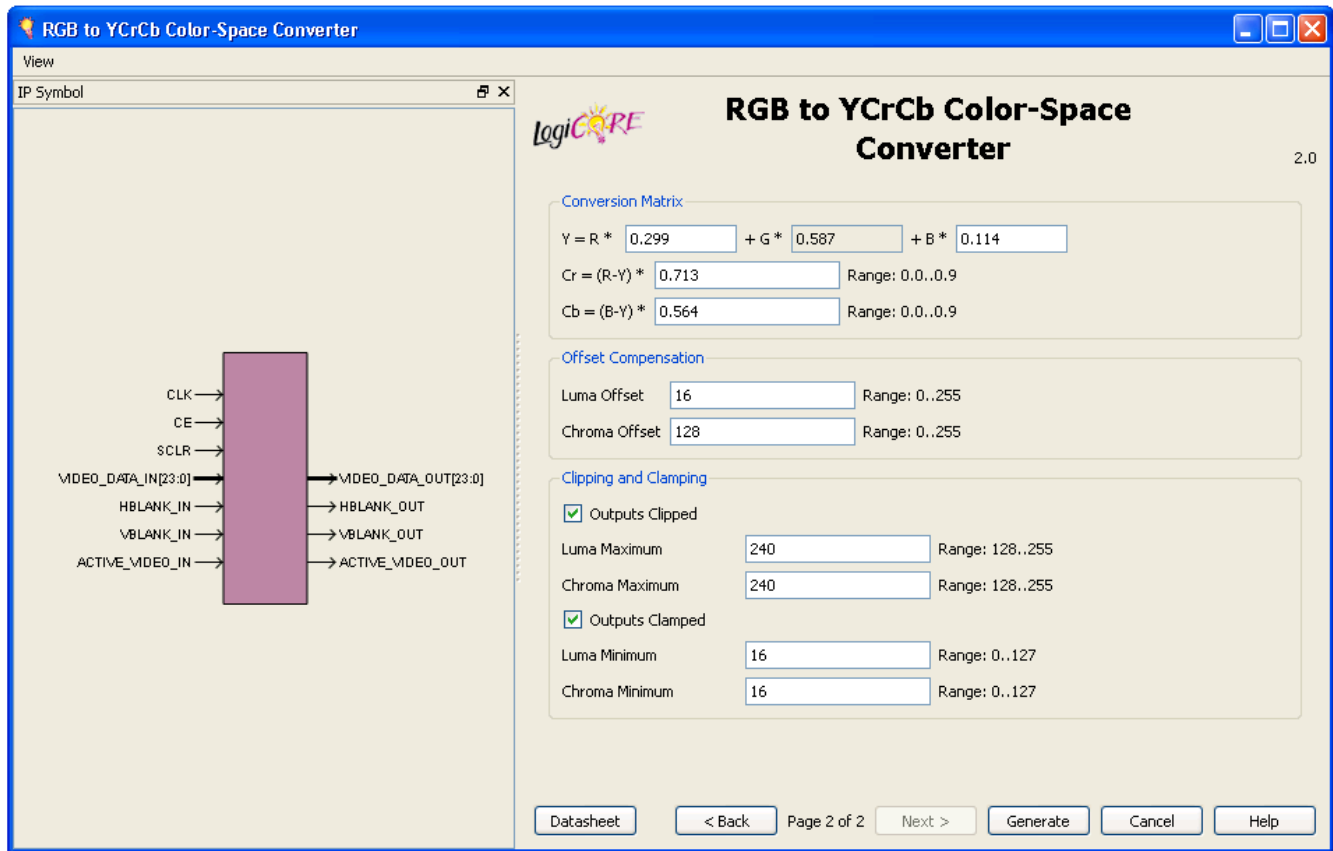


Figure 4: Conversion Matrix, Offset Compensation, Clipping and Clamping Screen

- **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 (YOFFSET and COFFSET in Equation 9). 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 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 RGB to YCrCb 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 5](#) 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 RGB to YCrCb 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 RGB to YCrCb 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_CSC_IN, BUS_STD = XSVI, BUS_TYPE = TARGET
PORT active_video_in  = active_video, DIR = IN,  BUS = XSVI_CSC_IN
PORT hblank_in       = hblank,          DIR = IN,  BUS = XSVI_CSC_IN
PORT vblank_in       = vblank,          DIR = IN,  BUS = XSVI_CSC_IN
PORT video_data_in   = video_data,     VEC = [0:((IWIDTH*3)-1)], DIR = IN, BUS = XSVI_CSC_IN
```

Output Side:

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

For more information on the MPD format, see [UG642: Platform Specification Format Reference Manual](#)

The RGB to YCrCb 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.

The RGB to YCrCb core symbol is shown in [Figure 5](#). Descriptions of each port are shown in [Table 5](#).

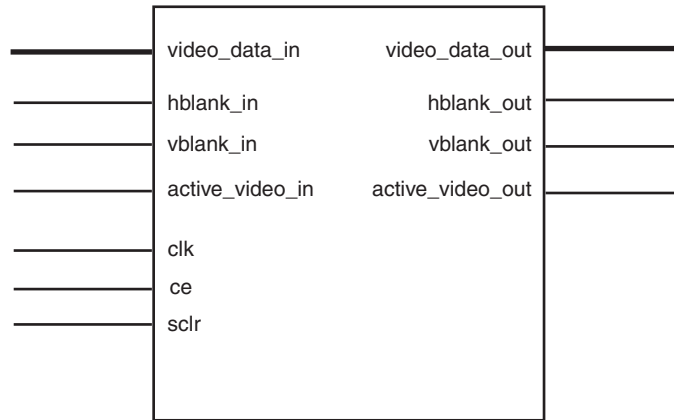


Figure 5: Core Symbol

Table 5: Port Descriptions for the RGB to YCrCb 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 color inputs in the following order. Color values are expected in *IWIDTH* bits wide unsigned integer representation.

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

- **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 converted 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 converted 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/SRL32s* are flushed. Unwanted results can be avoided if *SCLR* is held active until *SRL16/SRL32s* 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 luminance and chrominance outputs in the following order from MSB to LSB [*Cb*: *Cr*: *Y*]. Luminance and Chrominance values are expected in *OWIDTH* bits wide unsigned integer representation.

Bits	3OWIDTH-1:2OWIDTH	2OWIDTH-1:OWIDTH	OWIDTH-1:0
Video Data Signals	Cb	Cr	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.

Error Analysis

The following analysis, based on DSP fundamentals [Ref 5], presents mean-square-error (MSE) calculations for RGB to YCrCb, assuming *IWIDTH* bit RGB input data, *OWIDTH* bit wide YCrCb output data, and *CWIDTH* bits for coefficient precision. [Ref 6] arrives to similar results for fixed coefficient values and input and output representations.

Taking rounding/quantization into account, the structure illustrated on [Figure 2](#) implements the following equations:

$$Y_{RAW} = [ACOE\cdot (R - G) + BCOE\cdot (B - G)]_{MWIDTH-2} + G \quad \text{Equation 12}$$

$$Y = [Y_{RAW}]_{OWIDTH} + YOFF\text{SET} \quad \text{Equation 13}$$

$$Cb = [CCOE\cdot (B - Y_{RAW})]_{OWIDTH} + COFF\text{SET} \quad \text{Equation 14}$$

$$Cr = [DCOE\cdot (R - Y_{RAW})]_{OWIDTH} + COFF\text{SET} \quad \text{Equation 15}$$

where $[]_k$ denotes rounding to k bits. The architecture contains three possible operators that might introduce noise. Quantization noise is inserted when data is rounded.

1. Data is rounded to $MWIDTH-2$ bits after calculating Y_{raw}
2. Data is rounded to $OWIDTH$ bits at the output.
3. If $CCOE$ and $DCOE$ are chosen such that Cb and Cr may over- or underflow, clipping noise gets inserted to the signal flow.

Before analyzing the effects of these noise sources, first look at the input Signal to Quantization Noise Ratio (SQNR). Assuming uniformly distributed quantization error,

$$SQNR_{RGB} = 10\log \frac{P_x}{P_N} = 10\log \frac{\int_{RGBMIN}^{RGBMAX} x^2 dx}{\frac{1}{\Delta} \int_{-\Delta/2}^{\Delta/2} e^2 dx} \quad \text{Equation 16}$$

Substituting $LSB = 2^{-INBITS}$, where $INBITS$ is the input (RGB) precision, $SQNR_{RGB}$ becomes a function of the input dynamic range. In the next three calculations, when calculating $SQNR_{RGB}$ for the typical dynamic ranges, $INBITS = 8$ for all three cases.

When RGB values are in the (0, 255) range:

$$SQNR_{RGB} = 10\log \frac{\frac{1}{255} \int_0^{255} x^2 dx}{\int_{-1/2}^{1/2} x^2 dx} = 10\log \frac{\frac{1}{3 \cdot 255} [255^3]}{\frac{1}{12}} = 54.15\text{dB} \quad \text{Equation 17}$$

when RGB values are in the (16, 240) range:

$$SQNR_{RGB} = 10\log \frac{\frac{1}{224} \int_{16}^{240} x^2 dx}{\int_{-1/2}^{1/2} x^2 dx} = 53.92\text{dB} \quad \text{Equation 18}$$

and when RGB values are in the (16, 235) range:

$$SQNR_{RGB} = 10\log \frac{\frac{1}{219} \int_{16}^{235} x^2 dx}{\int_{(-1)/2}^{1/2} x^2 dx} = 53.74\text{dB} \quad \text{Equation 19}$$

The first rounding noise source can be practically eliminated by the careful choice of $MWIDTH$. Approximating SQNR by $6.02 \cdot MWIDTH$ [dB], intuitively the rounding noise can be reduced by increasing $MWIDTH$. However, $MWIDTH$ affects the resource usage and carry chain length in the design (thereby affecting maximum speed). Choosing $MWIDTH > 18$ would significantly increase the dedicated multiplier count of the design.

Therefore, optimal *MWIDTH* values, in the *IWIDTH*+4 to 18 range, do not significantly increase resource counts but assure that quantization noise inserted is negligible (at least 20 dB less than the input noise).

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 and Equation 5, 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 knowing the statistics of Cb and Cr values. For instance, when probabilities of extreme chrominance values are very small, it can be 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).

Though a quantitative noise analysis of the signal flow graph based on Figure 2 is possible by replacing quantizers with appropriate AWGN sources, the complexity of the derivation of a final noise formula which addresses clipping noise as well is beyond the scope of this document. Instead, Table 6 illustrates noise figures for some typical (see Table 2 on page 5) parameter combinations.

Table 6: Input and Output SNR Measurement Results [dB] for ITU-REC 601 (SD)

SNR	<i>IWIDTH</i> = <i>OWIDTH</i> = 8 Bits	<i>IWIDTH</i> = <i>OWIDTH</i> = 10 Bits	Input Range
SNR _{RGB} (input)	54.1	66.2	[0..255] (8bit)
SNR _Y	51.9	64.0	Or
SNR _{Cr}	47.0	58.9	[0..1023] (10 bit)
SNR _{Cb}	47.0	58.9	
SNR _{RGB} (input)	54.0	65.9	[16..240] (8bit)
SNR _Y	51.8	63.9	Or
SNR _{Cr}	46.9	58.8	[64..960] (10 bit)
SNR _{Cb}	46.9	58.8	
SNR _{RGB} (input)	53.8	65.8	[16..235] (8bit)
SNR _Y	51.5	63.6	Or
SNR _{Cr}	46.9	58.8	[64..920] (10 bit)
SNR _{Cb}	46.9	58.8	

Output Clipping Noise

If coefficients CC and CD in Equation 3 are larger than the maximum values calculated in Equation 4 and Equation 5, 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 (HAS_CLIPPING=0), binary values wrap around and insert substantial noise to the output. If HAS_CLIPPING=1, output values saturate, introducing less noise (Figure 6).

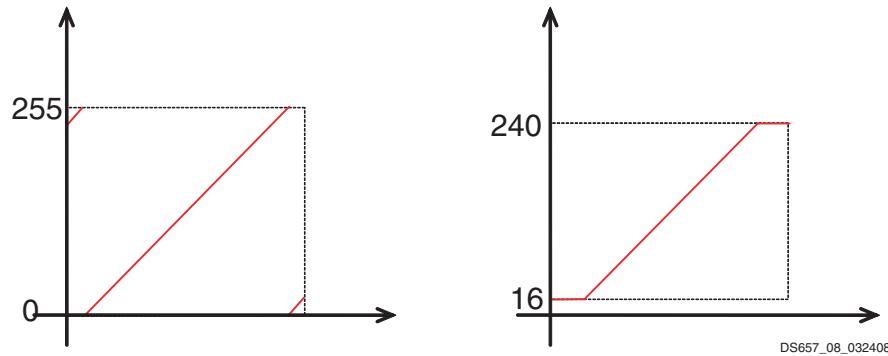


Figure 6: Wrap-Around and Saturation

Similarly, clamping logic is included in the design if HAS_CLAMPING=1. 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. These values are constrained to the predefined range by setting YMAX and YMIN values (constraining luminance), as well as CMAX and CMIN values (constraining chrominance) according to the standard specifications.

Control Signals and Timing

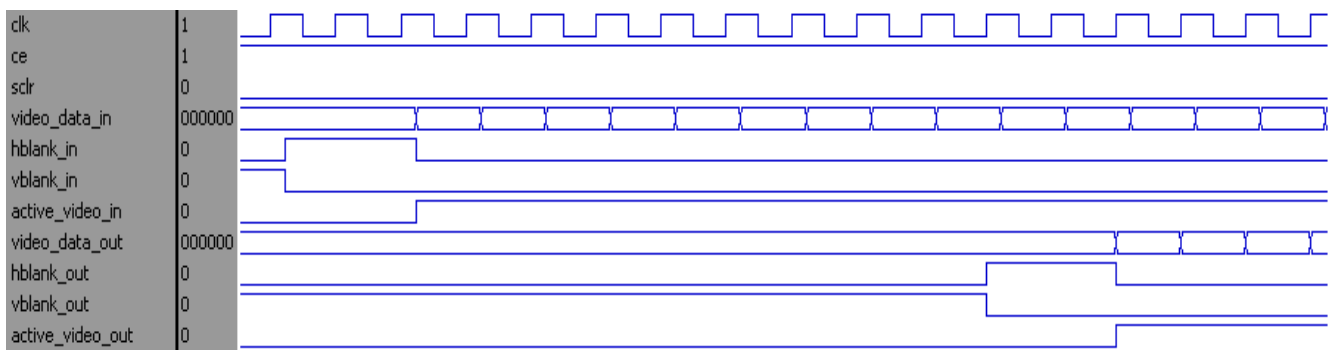


Figure 7: Timing Example

The propagation delay of the RGB to YCrCb 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 following equation:

$$\text{Latency} = 9 + 1(\text{if has clipping}) + 1(\text{if has clamping})$$

This code evaluates to 11 clock cycles for typical cases (unless in “custom” mode the clipping and/or clamping circuits are not used).

See ["Core Symbol and Port Descriptions" on page 10](#) for an explanation about other ports affecting the timing behavior of the core.

Core Resource Utilization and Performance

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

[Table 7](#), [Table 8](#), [Table 9](#), and [Table 10](#) present the resource usage of the RGB to YCrCb core for different families with default parameterization for all permitted input/output width combinations.

Table 7: Spartan-3A DSP - XC3SD3400A, Speed Grade=4 Resource Utilization

Input Width	Output Width	FFs	LUTs	Slices	DSP48s
8	8	199	144	109	4
8	10	215	158	117	4
8	12	262	264	169	4
10	8	209	156	114	4
10	10	225	170	122	4
10	12	272	276	174	4
12	8	236	188	129	4
12	10	252	202	127	4
12	12	299	308	189	4

Table 8: Spartan-6 - XC6SLX25T, Speed Grade=3 Resource Utilization

Input Width	Output Width	FFs	LUTs	DSP48s
8	8	186	152	4
8	10	200	177	4
8	12	214	181	4
10	8	196	166	4
10	10	210	192	4
10	12	224	196	4
12	8	206	177	4
12	10	220	201	4
12	12	234	204	4

Table 9: Virtex-5 - XC5VSX50T, Speed Grade = 1 Resource Utilization

Input Width	Output Width	FFs	LUTs	Slices	DSP48s
8	8	199	141	72	4
8	10	215	157	78	4
8	12	262	240	113	4
10	8	209	153	76	4
10	10	225	169	85	4
10	12	272	252	121	4
12	8	236	176	84	4
12	10	252	192	102	4
12	12	299	275	131	4

Table 10: Virtex-6 - XC6VLX75T, Speed Grade = 1 Resource Utilization

Input Width	Output Width	FFs	LUTs	DSP48s
8	8	186	168	4
8	10	200	192	4
8	12	214	199	4
10	8	196	181	4
10	10	210	205	4
10	12	224	213	4
12	8	206	195	4
12	10	220	221	4
12	12	234	228	4

Performance Guidelines

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

Table 11: Performance and Resource Guidelines

Family	Part	Typical Operating Frequency
Spartan-3A DSP	XC3SD1800A-CS484 (-4)	176 MHz
Spartan-6	XC6SL25T-FGG484 (-3)	174 MHz
Virtex-5	XC5VSX35T-FF665 (-1)	318 MHz
Virtex-6	XC6VLX75T-FF784 (-1)	354 MHz

Known Issues

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*.

Licensing Options

The RGB to YCrCb 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

The following table shows the revision history for this document:

Date	Version	Description of Revisions
03/24/08	1.0	Initial Xilinx release.
04/24/09	2.0	Updated supported devices to Spartan-3A DSP and Virtex-5 device families only. Supported tools updated for the ISE 11.1 release. Placed new legal disclaimers.
07/23/10	3.0	Updated for 12.2 release. Updated supported devices to Spartan-6 and Virtex-6. Added supported operating systems. Removed licensing information. Replaced legal disclaimer.

Notice of Disclaimer

Xilinx is providing this product documentation, hereinafter "Information," to you "AS IS" with no warranty of any kind, express or implied. Xilinx makes no representation that the Information, or any particular implementation thereof, is free from any claims of infringement. You are responsible for obtaining any rights you may require for any implementation based on the Information. All specifications are subject to change without notice. XILINX EXPRESSLY DISCLAIMS ANY WARRANTY WHATSOEVER WITH RESPECT TO THE ADEQUACY OF THE INFORMATION OR ANY IMPLEMENTATION BASED THEREON, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OR REPRESENTATIONS THAT THIS IMPLEMENTATION IS FREE FROM CLAIMS OF INFRINGEMENT AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Except as stated herein, none of the Information may be copied, reproduced, distributed, republished, downloaded, displayed, posted, or transmitted in any form or by any means including, but not limited to, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Xilinx.