UG0639
User Guide
Color Space Conversion

a MICROCHIP company

a Microchip company

Microsemi Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.

50200639. 5.0 2/20

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the current publication.

1.1 Revision 5.0

Updated the RGB data width (Table 4, page 5) and the 4-input LUTs utilization (Table 5, page 6).

1.2 Revision 4.0

The following is a summary of the changes in this revision.

- IOs and equations were updated.
- Figure 1, page 3 and Figure 2, page 4 were updated.
- Input and Output Port were updated. See, Table 2, page 4 and Table 3, page 4.
- Added figure to show Testbench results. See, Figure 5, page 6 and Figure 6, page 6.
- Resource Utilization values were updated. See, Resource Utilizations, page 6.

1.3 Revision 3.0

Updated the Resource Utilization section and the Resource Utilization Report. See Resource Utilizations, page 6.

1.4 Revision 2.0

The following is a summary of the changes in this revision.

- The Testbench section was added to the document as per SAR 76100. For more information, see Testbench, page 6.
- Resource Utilization values were updated as per SAR 76100. For more information, see Resource Utilizations, page 6.

1.5 Revision 1.0

The first publication of this document.

2 Introduction

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

- RGB - Used in computer graphics
- YIQ, YUV, and YCbCr - Used in video compression

The red, green, and blue (RGB) color space is widely used in computer graphics. These are three primary additive colors and represented by a three-dimensional, Cartesian coordinate system. These three colors are used to create any desired color. Therefore, the choice of the RGB color space simplifies the architecture and design of the system. Also, the system that is designed using the RGB color space takes advantage of the large number of existing software routines.

However, RGB is not very efficient in terms of bandwidth as all the three components have to be present in equal bandwidth to produce any color. So an RGB based frame buffer must have same pixel depth and display resolution for each RGB component. Processing an image in RGB color space is usually not the most efficient method. For example: to modify the intensity or color of a given pixel, the three RGB values must be read from the frame buffer, the intensity or color calculated, desired modifications performed, new RGB values calculated and written back into the frame buffer.

The same can be achieved, if the image color properties are stored directly in intensity and color format.

Due to this reason, many video standards use luma and two color difference signals. One of the common color spaces in this format is the YCbCr color space format.

The YCbCr color space was developed as part of ITU-R VT.601 during development of a worldwide digital component video standard. The luma component Y is defined to have a nominal 8-bit range of 16-235 range of values. The color information is represented as Cb and Cr with nominal 8-bit range of 16-240 range of values. There are several YCbCr sampling formats such as 4:4:4, 4:2:2, 4:1:1, and 4:2:0.

Table 1 • **YCbCr Sample Formats**

YCbCr Sample Format	Description
4:4:4	Each sample has a Y, a Cb, and a Cr value represented typically using 8-bits or 10-bits per component. Therefore, each sample in 4:4:4 sampling format requires either 24-bits or 30 bits.
4:2:2	In 4:2:2 sampling format, for every two horizontal Y samples, there is one Cb and Cr value. Each component sample is typically represented as 8-bits or 10-bits. Therefore, each sample in 4:2:2 sampling format requires either 16-bits or 20-bits.
4:1:1	In 4:1:1 sampling format, for every four horizontal Y samples, there is one Cb and Cr value. Each component sample is typically represented as 8-bits. Therefore, each sample in 4:1:1 sampling format requires 12-bits.
4:2:0	In 4:2:0 sampling format, the 2:1 reduction is done on both horizontal and vertical values. It is commonly used in video compression.

The advantages and disadvantages of various color space formats lead to requirements for the color space conversions. The objective is to convert the video inputs into the desired color space before performing any video processing on it. The RGB to YCbCr and vice-versa conversion is one such example.

3 Hardware Implementation

This section describes the implementation of the Color Space Conversion block.

3.1 Design Description

The Color space conversion IP block contains two modules — RGB to YCbCr and YCbCr to RGB. The RGB to YCbCr Color Space Converter IP module implements the equations to convert 24-bit input RGB color samples to 24-bit YCbCr output samples. The YCbCr to RGB Color Space Converter IP module converts vice-versa of the RGB to YCbCr. Both the converters use 4:4:4 sampling format.

Both the modules take data enable, as inputs and pipeline them accordingly to match the conversion video data outputs.

To convert the floating point constants into integer multiplication, the floating point constants are scaled by multiplying these constants with 2^8 = 256. Then after the computation of the above equations, the output is divided by the scaling factor 2^{16} = 65536.

After scaling, the RGB to YCbCr equations are:

Y= 16 + 65.738*R/256 + 129.057*G/256 + 25.064*B/256

Cb = 128-37.945*R/256 - 74.494*G/256 + 112.439*B/256

Cr = 128+112.439*R - 94.154*G/256 - 18.285*B/256

After scaling, the YCbCr to RGB equations are:

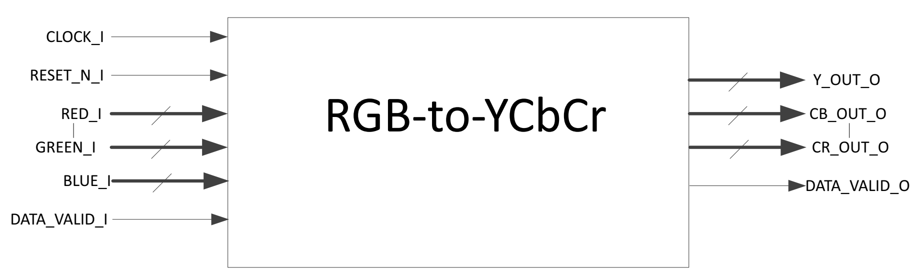
R = 298.082*Y/256 + 408.583 * Cr/256 - 222.921

G = 298.082*Y/256 -100.291*Cb/256 - 208.120*Cr/256 + 135.576

B = 298.082*Y/256 + 516.412*Cb/256 - 276.836

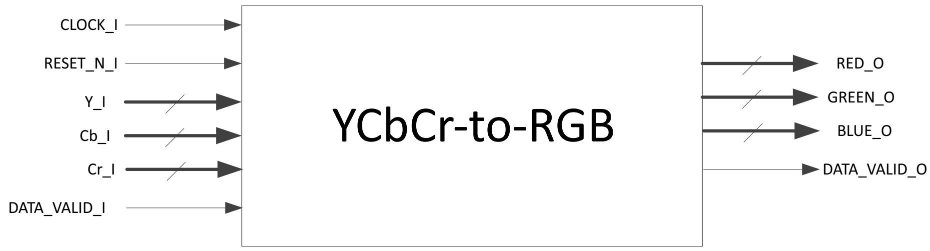
The following figures show the system-level block diagrams of the RGB to YCbCr block.

Figure 1 • **System-Level Block Diagram of RGB to YCbCr**



The following figures show the system-level block diagrams of the YCbCr to RGB blocks.

Figure 2 • **System-Level Block Diagram of YCbCr to RGB**



3.2 Inputs and Outputs

The following tables shows the input and output ports of the RGB to YCbCr.

Table 2 • **Input and Output Ports of the RGB to YCbCr Block**

Signal Name	Direction	Width	Description
RESET_N_I	Input	-	Active low asynchronous reset signal to design
CLOCK_I	Input	-	System clock
RED_I	Input	[(G_RGB_DATA_BIT_WIDTH-1):0]	Red pixel data input
GREEN_I	Input	[(G_RGB_DATA_BIT_WIDTH-1):0]	Green pixel data input
BLUE_I	Input	[(G_RGB_DATA_BIT_WIDTH-1):0]	Blue pixel data input
DATA_VALID_I	Input	-	Input data valid signal
Y_OUT_O	Output	[(G_YCbCr_DATA_BIT_WIDTH-1):0]	Y pixel data output
Cb_OUT_O	Output	[(G_YCbCr_DATA_BIT_WIDTH-1):0]	Cb pixel data output
Cr_OUT_O	Output	[(G_YCbCr_DATA_BIT_WIDTH-1):0]	Cr pixel data output
DATA_VALID_O	Output	-	Output data valid signal

The following tables shows the input and output ports of the YCbCr to RGB blocks.

Table 3 • **Input and Output Ports of the YCbCr-to-RGB Block**

Signal Name	Direction	Width	Description
RESET_N_I	Input	-	Active low asynchronous reset signal to design
CLOCK_I	Input	-	System clock
Y_I	Input	[(G_YCbCr_DATA_BIT_WIDTH-1):0]	Y pixel data input
Cb_I	Input	[(G_YCbCr_DATA_BIT_WIDTH-1):0]	Cb pixel data input
Cr_I	Input	[(G_YCbCr_DATA_BIT_WIDTH-1):0]	Cr pixel data input
DATA_VALID_I	Input	-	Input data valid signal
RED_O	Output	[(G_RGB_DATA_BIT_WIDTH-1):0]	Red pixel data output
GREEN_O	Output	[(G_RGB_DATA_BIT_WIDTH-1):0]	Green pixel data output
BLUE_O	Output	[(G_RGB_DATA_BIT_WIDTH-1):0]	Blue pixel data output
DATA_VALID_O	Output	-	Output data valid signal

3.3 Configuration Parameters

The following table shows the configuration parameters used in the hardware implementation of RGB to YCbCr and YCbCr to RGB blocks. These are generic parameters and can be varied as per the application requirements.

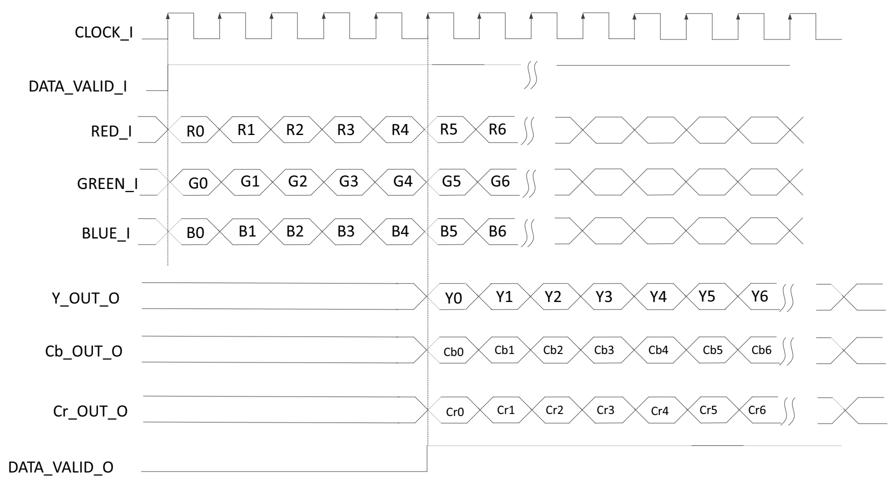
Table 4 • **Configuration Parameters**

Name	Description
G_RGB_DATA_BIT_WIDTH	RGB data bit width (supports 8,10, and 12)
G_YCbCr_DATA_BIT_WIDTH	YCbCr data bit width (supports 8,10, and 12)

3.4 Timing Diagrams

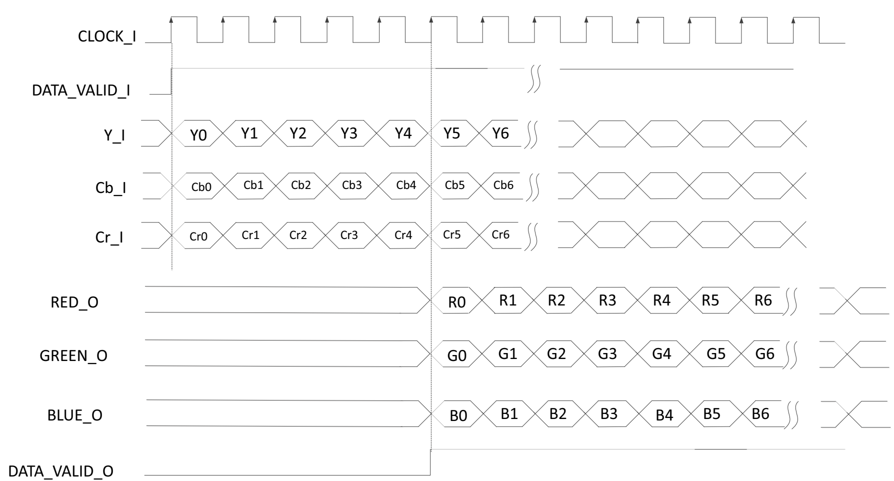
The following figures show the timing diagrams of RGB to YCbCr.

Figure 3 • **Timing Diagram of RGB to YCbCr**



The following figures show the timing diagrams of YCbCr to RGB blocks.

Figure 4 • **Timing Diagram of YCbCr to RGB**

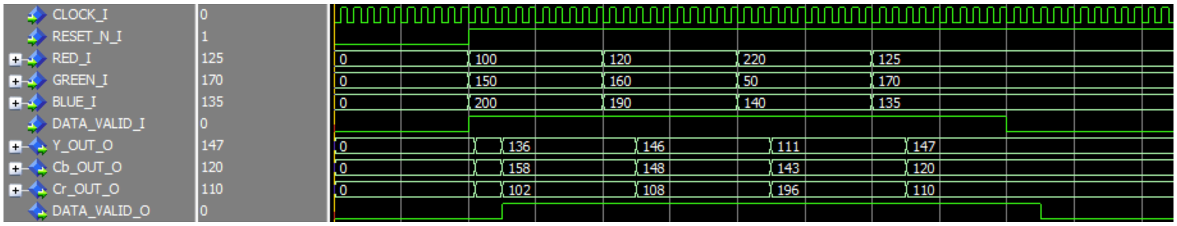


Note: All other input signals get delayed by the clock cycles as shown in Figure 4, page 5.

3.5 Testbench

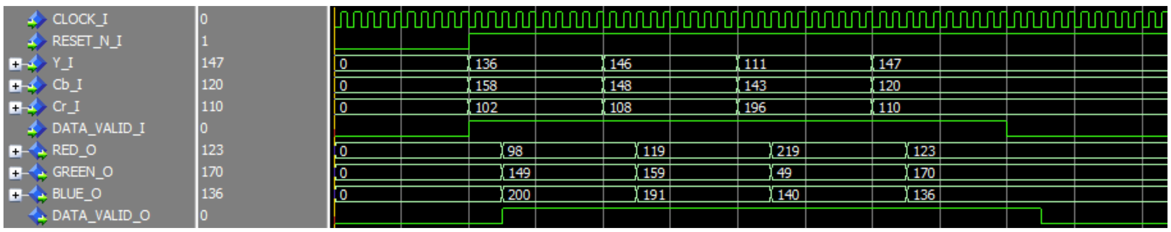
A testbench is provided to check the functionality of Color Space Converter core. The following figure shows testbench result for RGB to YCbCr color space conversion.

Figure 5 • RGB to YCbCr Testbench Results



The following figure shows testbench result for YCbCr to RGB color space conversion.

Figure 6 • YCbCr to RGB Testbench Results



3.6 Resource Utilizations

The color space conversion block is implemented on an M2S150T SmartFusion®2 System-on-Chip (SoC) FPGA in the FC1152 package) and PolarFire FPGA (MPF300TS_ES - 1FCG1152E package).The following table shows the Resource Utilization of RGB to YCbCr.

Table 5 • Resource Utilization of RGB to YCbCr

Resource	Usage
DFFs	51
4-input LUTs	85
MACC	9
RAM1kx18	0
RAM64x18	0

The following table shows the Resource Utilization of YCbCr to RGB.

Table 6 • Resource Utilization of YCbCr to RGB

Resource	Usage
DFFs	85
4-input LUTs	134
MACC 5	5
RAM1kx18	0
RAM64x18	0