Color Space Conversion from RGB to YCbCr based on FPGA

Pei YUAN¹, Zhong-hua HU¹, Gan-xin OUYANG¹ and Xue-jun ZHANG¹,²,*

¹Guangxi University, Nanning 530004
²Key Laboratory of Multimedia Communication and Information Processing, Guangxi University, Nanning 530004, Guangxi

*Corresponding author

Keywords: RGB, YCbCr, ITU-T T.871, FPGA, Verilog, Multiplier.

Abstract. In image processing, color space conversion of RGB-YCbCr is often used. Based on ITU-T T.871 design standard and FPGA operation characteristics, this paper realizes the color space conversion from RGB to YCbCr using Verilog language. In the FPGA chip, the multiplier resources are very scarce. The optimization of multiplier resources and processing speed is often the bottleneck of image processing. In traditional methods, 5-9 multipliers are usually used. By optimizing the formula, compared with the results of other documents, this module only uses 4 multipliers, so the resource consumption is less. The pipeline design is used, the operation speed is fast, and the real-time performance is good. It is suitable for the requirement of JPEG compression and is easy to be realized in the actual project.

Introduction

With the continuous development of digital multimedia technology, digital image processing technology is widely used in various military, civilian, commercial and industrial production areas. The real-time performance of digital image processing has become the focus of attention. In recent years, the development of microelectronic technology and super large scale integrated circuit manufacturing technology, especially the development of field programmable gate array (FPGA), has provided new ideas and methods for improving the performance of image processing system [1-4]. Because the data preprocessing of the bottom layer image is very large, requiring fast processing speed, but the operation result is relatively simple, the image processing system with FPGA as the main processing chip is very good at image preprocessing. Aiming at the demand of real-time image processing, this paper implements a new algorithm from RGB to YCbCr color space on FPGA, and uses FPGA chip of xilinx to complete the design. JinXiaodong [1] and Feng Ansong [2] and others all refer to ITU-R BT.601 standard [5, 6]. The transformation from RGB to YCbCr is achieved by 9 and 5 multipliers, respectively. The range of Y is 16-235, and C's range is 16-240.

In this paper, we refer to ITU-T T.871 standard and use the DSP48 kernel in FPGA to realize multiplications and additions, only using four multipliers to realize color space conversion. The range of values for both y and c is 0-255. It not only saves resources, but also improves the speed of the algorithm, which is suitable for the requirement of JPEG compression.

Color Space

RGB color mode is a color standard [7-9] in industry, which obtains all kinds of colors through the change of the three colors channels of red, green, blue and the superposition on each other. RGB represents the three colors channels of red, green and blue, which is a standard including almost all colors the human eyes can perceive, and it is one of the most widely used color system. The range of RGB is 0~255, and Table 1 lists corresponding R, G, and B values [10] of several colors.

RGB color generation is easy to realize and is widely used in the display system of computer and color TV. However, in scientific research, RGB color space is not generally used, because its hue, brightness, saturation are shown together, which cannot be easily separated, leading the details difficult to digitally adjust.
YCbCr is a kind of color space, which is widely used. JPEG, MPEG, DVD, camera, digital TV and so on all adopt this format. In JPEG compression algorithm, we need to transform patterns (usually RGB models) into YCbCr models, where Y represents luminance, and Cb and Cr represent green and red chromatic aberration respectively. The importance of YCbCr is that its luminance signal (Y) and color difference signal (CbCr) are independent of each other. YCbCr also has another advantage that the characteristics of human eyes can be used to reduce the storage capacity required for digital color images. There are many sampling formats for YCbCr, such as 4: 4: 4, 4: 2: 2, 4: 1: 1 and 4: 2: 0.

Source of YCbCr

The interpretation of Y, Cb and Cr is derived from the definition of the $E'_Y$, $E'_c_b$ and $E'_c_r$ signal in the 625-line specification of Rec. ITU-R BT.601. These signals are standardized to allow for the use of 8-bit binary encoding of a full range of Y components at a 256 level. Rather, they are specified by the following relationships:

\[
Y = \text{Min}(\text{Max}(0, \text{Round}(255 \times E'_Y)), 255)
\]

\[
C_b = \text{Min}(\text{Max}(0, \text{Round}(255 \times E'_c_b + 128)), 255)
\]

\[
C_r = \text{Min}(\text{Max}(0, \text{Round}(255 \times E'_c_r + 128)), 255)
\]

using the following mathematical definitions:

\[
\text{Round}(x) = [x + 0.5]
\]

\[
\text{Min}(x, y) = \begin{cases} x & x \leq y \\ y & x > y \end{cases}
\]

\[
\text{Max}(x, y) = \begin{cases} x & x \geq y \\ y & x < y \end{cases}
\]

$E'_Y$, $E'_c_b$ and $E'_c_r$ signals are defined in the ITU-R BT.60. The nominal range of $E'_Y$ is from 0 to 1.0 and the nominal range of $E'_c_b$ and $E'_c_r$ in normal colorimetric method are from -0.5 to +0.5. A lot of articles use formula (1):

\[
Y = 16 + (0.257 \times R + 0.504 \times G + 0.114 \times B)
\]

\[
C_b = 128 + (-0.148 \times R - 0.291 \times G + 0.439 \times B)
\]

\[
C_r = 128 + (0.439 \times R - 0.368 \times G - 0.071 \times B)
\]

When using the above formula (1) to transform, the range of RGB, calculated Y luminance signal, and Cr and Cb respectively is 0~255, 16~235 and 16~240. However, it does not apply to JPEG compression requirements, and the values of Y, Cb and Cr must be shown in a range from 0 to 255 in accordance with ITU-T T.871. YCbCr color (each component has 256 levels) can be directly calculated from the RGB color of the full scale 8-bit color. Black is represented by (0,0,0) and white is represented by (255,255,255). The following formula (2) is used.

\[
Y = \text{Min}(\text{Max}(0, \text{Round}(0.299 \times R + 0.587 \times G + 0.114 \times B)), 255)
\]

\[
C_b = \text{Min}(\text{Max}(0, \text{Round}(-0.1687 \times R - 0.3313 \times G + 0.5 \times B + 128)), 255)
\]

\[
C_r = \text{Min}(\text{Max}(0, \text{Round}(0.5 \times R - 0.4187 \times G - 0.0813 \times B + 128)), 255)
\]

<table>
<thead>
<tr>
<th></th>
<th>Normal range</th>
<th>White</th>
<th>Yellow</th>
<th>Cyan</th>
<th>Reddish purple</th>
<th>Red</th>
<th>Green</th>
<th>Blue</th>
<th>Black</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>0~255</td>
<td>255</td>
<td>255</td>
<td>0</td>
<td>255</td>
<td>255</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>G</td>
<td>0~255</td>
<td>255</td>
<td>255</td>
<td>0</td>
<td>0</td>
<td>255</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0~255</td>
<td>255</td>
<td>0</td>
<td>255</td>
<td>0</td>
<td>255</td>
<td>0</td>
<td>0</td>
<td>255</td>
</tr>
</tbody>
</table>

Table 1. Corresponding Values of R,G,B of Several Common Colors.
\[ Y = \text{Min}(\text{Max}(0, \text{Round}(0.299 \times R + 0.587 \times G + 0.114 \times B)), 255) \]

\[ C_b = \text{Min}(\text{Max}(0, \text{Round}(-0.1687 \times R - 0.3313 \times G + 0.5 \times B + 128)), 255) \]

\[ C_r = \text{Min}(\text{Max}(0, \text{Round}(0.5 \times R - 0.4187 \times G - 0.0813 \times B + 128)), 255) \] (2)

**Color Space Conversion Based on FPGA**

In order to improve the image quality, each component of FPGA is generally set to 12-bit precision, then the converted YCbCr is 0~4095. Considering the requirements of saving resources and improving the rate, the formula (2) is first optimized to formula (3).

\[ Y = \text{CA} \times (R - G) + G + \text{CB} \times (B - G) + \text{YOFFSET} \]

\[ C_b = \text{CC} \times (B - Y) + \text{COFFSET} \]

\[ C_r = \text{CD} \times (R - Y) + \text{COFFSET} \]

\[ \text{CA} = 0.299 \]

\[ \text{CB} = 0.114 \]

\[ \text{CD} = 0.7133 \]

\[ \text{CC} = 0.5643 \]

\[ \text{YOFFSET} = 0 \]

\[ \text{COFFSET} = 2048 \] (3)

After simplification, only 4 multipliers can achieve transformation, which greatly saves the chip resources compared with the original formula. Here Verilog hardware description language is used for text input and description. Firstly, defining input and output variables:

```verilog
input clk;
input wire [11:0] data_R, data_G, data_B;
output wire [11:0] data_Y, data_Cb, data_Cr;
```

Then making a macro-definition of the constant used in the program and quantifying to [0:4095]:

```verilog
parameter signed CA=13'd1225;
parameter signed CB=13'd467;
parameter signed CC=13'd2921;
parameter signed CD=13'd2311;
```

The configuration interface of multiplier “core” is as follows:

![Configuration interface for multiplier “core”](image)

Finally, describing the whole function module. According to the formula (3), the RGB component is input to calculate the Y component first, and instantiate 2 multipliers IP Core to perform addition or subtraction. At the same time, the R and B components are cached in parallel, and then the calculated Y is multiplied and added to obtain Cr and Cb. Using the pipeline design, each input data is processed in parallel and real-time, and the results are output sequentially after a number of clocks. But in the end, the output data should be non-negative and anti-overflow, making each component of the YCbCr is within the range of [0:4095].

The design software ISE13.4 of xilinx is used for compiling and routing. First, create a new
project and add the already written “.v” file. Then create the test file in which write the random number as the signal source and input into the color space conversion module. Calling a third party EDA software Modelsim Se from ISE to perform function simulation. The simulation results are as follows:

Figure 2. Simulation Results of RGB Component to YCbCr Function Module.

From the simulation results, we can see that when the input clock signal is CLK and the input data RGB is (1316,1545,3713), the output signal YCbCr is (1723,3170,1757) after 9 clocks delay, which is consistent with the theoretical calculation results. The xilinx Virtex4 chip is selected for hardware test, and the comprehensive report is obtained after synthesis. In the report, you can see the resource consumption of the module in the chip, and the maximum frequency that can be run. The report reads as follows:

Device utilization summary:
Number of Slices: 167 out of 67584 0%
Number of Slice Flip Flops: 284 out of 135168 0%
Number of 4 input LUTs: 199 out of 135168 0%
Number used as logic: 149
Number used as Shift registers: 50
Number of IOs: 101
Number of bonded IOBs: 101 out of 768 13%
Number of GCLKs: 1 out of 32 3%
Number of DSP48s: 4 out of 96 4%

Timing Summary:
Speed Grade: -12
Minimum period: 2.273ns (Maximum Frequency: 439.937MHz)
Minimum input arrival time before clock: 2.050ns
Maximum output required time after clock: 4.227ns
Maximum combinational path delay: No path found

As you can see from the above report, this module uses very few resources with less than 1% utilization rate of Lut and Reg, and only 4% of the multiplier. The chip has a lot of resources left to do other image processing algorithms. Moreover, the maximum operating frequency of the module can reach 439.937Mhz and it can be used in many ultra-high resolution image processing.

Table 2. The results of this paper are compared with those of references 1 and 2 as follows.

<table>
<thead>
<tr>
<th>Document 1</th>
<th>Multiplier resource</th>
<th>Running speed (Mhz)</th>
<th>Calculation accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>651.976</td>
<td>8-bit</td>
<td></td>
</tr>
<tr>
<td>Document 2</td>
<td>5</td>
<td>189.034</td>
<td>8-bit</td>
</tr>
<tr>
<td>This paper</td>
<td>4</td>
<td>439.937</td>
<td>12-bit</td>
</tr>
</tbody>
</table>

It can be seen from Table 2 that the method used in this paper has the advantages of less use of multiplier, high calculation precision, and fast running frequency of the program, achieving the requirement of being real-time.

Conclusions

YCbCr is a color space which is often used in image processing, which has many forms to reducing the sampling rate of a particular signal. It can save the bandwidth of image transmission and reduce the loss of color components, which can hardly be detected by human eyes. As a universal processing chip, FPGA has flexible design, parallel execution of programs and high bandwidth of
signal processing. In this paper, Verilog hardware language and FPGA rich high-density integrated logic gate resources are used to realize the spatial conversion from RGB to YCbCr, with less resource consumption and high processing rate. It is highly editable, real-time and practical.

References


