Firmware—Building Binning Tables for Color Mixing Using EZ-Color HB-LED Controllers

AN47518

Author: Mukund Krishna

Associated Project: No
Associated Part Family: CY8CLED04, CY8CLED08, CY8CLED16
GET FREE SAMPLES HERE
Software Version: PSoC Designer™ 4.4 or PSoC Express™ 3.0
Associated Application Notes: AN16035, AN33640

Application Note Abstract

Cypress' EZ-Color™ HB-LED controllers enable the use of LEDs from different manufacturing bins for high color accuracy color mixing lighting applications. This application note describes the procedure to build the binning tables in firmware for EZ-Color.

Introduction

Cypress' family of high brightness LED controllers enables intelligent and diverse lighting designs using HB-LEDs. Among the common applications using EZ-Color are light sources that are dynamically able to change color output on demand. It also includes tuning of white light to different color temperatures. This is enabled by complex firmware that is discussed in another Cypress application note AN16035, *Firmware–RGB Color Mixing Firmware for EZ-Color*.

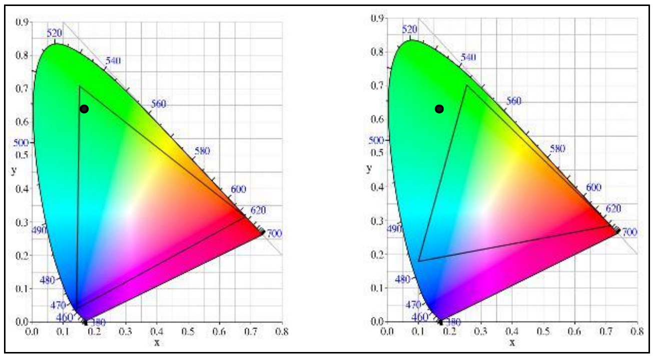
One of the caveats with using HB-LEDs is that they come in different bins. Manufacturers sort LEDs into different bins based upon measured flux (luminosity), color (wavelength), and forward voltage (V_f). This translates to substantial variations in characteristics that must be compensated in the color mixing algorithm. Table 1 shows the effect on light output when changes in the LED characteristics are not compensated for. An advantage of the color mixing firmware developed by Cypress is that significant care and cost need not be borne to procure LEDs of the same bin. This is because the algorithm is designed in a manner that drives the LEDs on the basis of their rated characteristics. The use of this firmware helps to maintain the characteristics of light output, irrespective of change in certain LED characteristics such as rated flux (luminosity) and dominant wavelength (color coordinates).

Table 1. Effect of Changes in LED Characteristics on Light Output

Sl.No.	Parameter	Effect on Light Output
1	Luminosity (flux)	Intensity, Color, Color Rendering Index (CRI), Correlated Color Temperature (CCT)
2	Color (dominant wavelength)	Color, CRI, CCT
3	Forward Voltage	Efficiency of driver

This is true only for colors that are within the gamut formed by the individual LEDs. It is impossible for any algorithm to produce a color outside that gamut. If the color required is in the border of the gamut, there is a good chance of placing that color point completely out of that gamut using LEDs of different bins. This is illustrated in Figure 1 by the black dot representing the color point. During the development stage, ensure that the required colors are produced with all possible bin combinations of the different LEDs used.

Figure 1. Color Point on the Border of Gamut Falls Out of Gamut if LED Bins are Changed



The color mixing firmware assumes that knowledge of the LED characteristics such as color coordinates and rated luminosity are present in firmware and uses that information to determine actual dimming values. This application note describes the method to build the binning tables (for flux and color bins) in firmware in a form that is usable by the color mixing algorithm.

For more detail on color accuracy, refer to the Cypress application note AN33640, *Color Mixing Accuracy with EZ-Color High-Brightness LED Controllers*.

Accompanying Hardware and Software

The firmware described in this application note is included with the CY3261A-RGB demonstration kit available at the Cypress online store. Using this application note, the firmware available with this kit can be modified quickly to accommodate different LED manufacturers. The text file associated with this application note is a header file for use with PSoC Designer™ 4.4.

The implementations that follow require the PSoC Designer software tool. The latest versions of both this software tool and the latest service pack are available for free download at www.cypress.com.

High Level Overview

In the color mixing firmware project, the binning information is stored in a header file called *bin_tables.h* (see Figure 2). Of the three types of bins, information stored in this file pertains to color and rated flux bins.

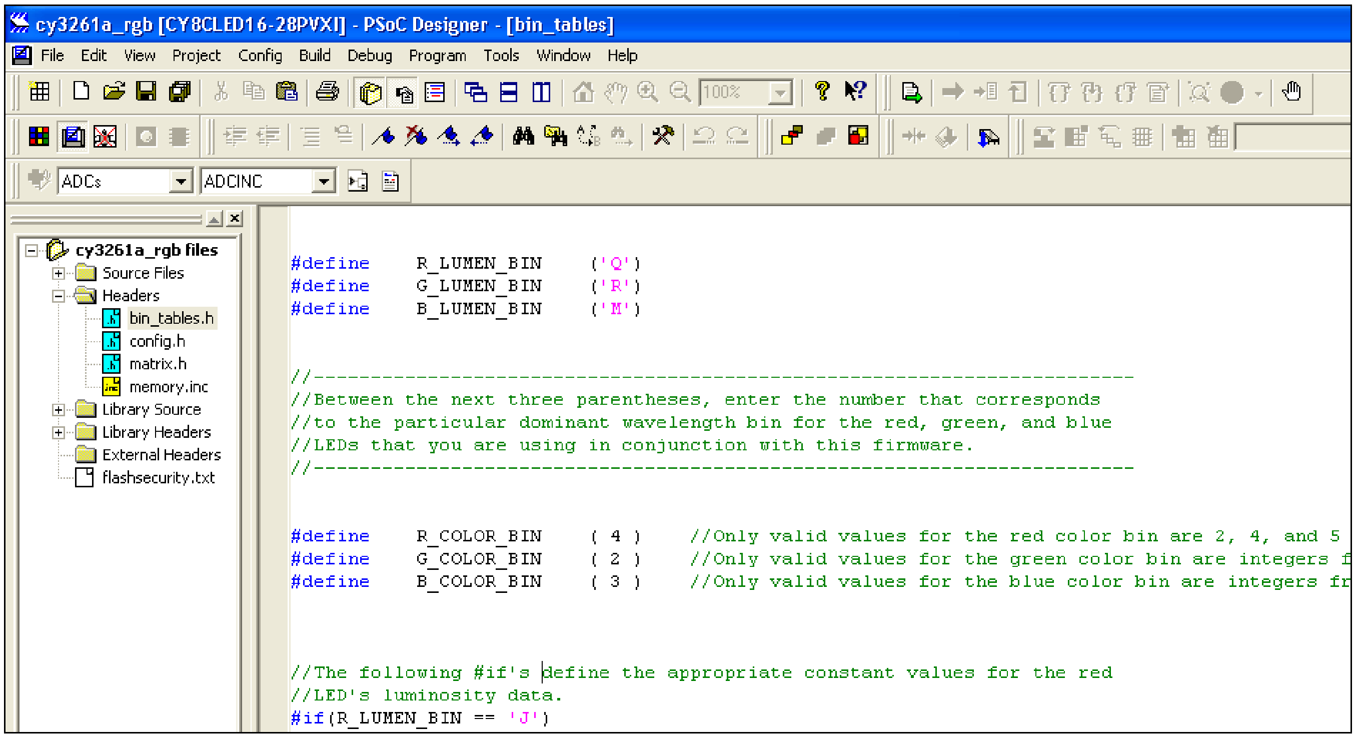
Pre-processor statements in C language are useful to implement structures such as binning tables. Using the #DEFINE and #IF statements, a one-time definition of all the bins are performed. A simple statement at the beginning informs the algorithm of the actual bin being used in that instance. Therefore, the entire binning table can be designed and written into firmware during the development stage and the same firmware can be used throughout production, with just a simple change in the letter code of the actual bin in use. The LED bins used during development stage and production need not even be the same.

It must be noted that the accuracy and consistency of light output depends on the accuracy of the information stored in the binning tables.

Steps to Build Binning Table in Firmware

Figure 2 shows the location of the header file in the firmware's file structure for the CY3261A-RGB demonstration board.

Figure 2. Location of Header File with Binning Tables in CY3261A-RGB Firmware



```
#define     R_LUMEN_BIN     ('Q')
#define     G_LUMEN_BIN     ('R')
#define     B_LUMEN_BIN     ('M')


//------------------------------------------------------------------------
//Between the next three parentheses, enter the number that corresponds
//to the particular dominant wavelength bin for the red, green, and blue
//LEDs that you are using in conjunction with this firmware.
//------------------------------------------------------------------------


#define     R_COLOR_BIN     ( 4 )     //Only valid values for the red color bin are 2, 4, and 5
#define     G_COLOR_BIN     ( 2 )     //Only valid values for the green color bin are integers f
#define     B_COLOR_BIN     ( 3 )     //Only valid values for the blue color bin are integers fr


//The following #if's define the appropriate constant values for the red
//LED's luminosity data.
#if(R_LUMEN_BIN == 'J')
```

This section details the steps to build or modify the binning tables for flux and color bins in firmware given the LED characteristics. This includes 'C' language syntax, methods to calculate the numbers used, and the format to store them.

The exact LED part number must be known and the relevant data sheet and binning document must be available. The following data is required from the data sheet or binning document:

- Luminous flux bin codes and their minimum and maximum flux values.
- Color bin (wavelength) codes and their maximum and minimum wavelength values.
- Color coordinates of each color bin. **Note** Sometimes this information is not present in the data sheet or binning document and must be obtained from the manufacturer, either in the form of test data or color purity value (from which color coordinate is calculated).

Part A: Flux Bins

1. Using the bin codes for the flux bins, a number of C pre-processor code lines are written (or modified) for each color LED that is used.

   Note Some manufacturers have tighter luminous flux bins (such as CREE 7090XR) that are sub groups of main bins. For example, K2 and K3 are tighter sub bins of the bin K. In the firmware, the pre-processor statements do not accept multi-byte characters. So the bins are labeled as numbers and K2 and K3 are 2 and 3 respectively. Bins M2 and M3 are 4 and 5 respectively.

   ```
   Example:
   #if(R_LUMEN_BIN == 'F')
   #define    R_MAX_LUMENS   ( 3149 )
   //12.3 * 2^8
   #endif
   ```

   This section of pre-processor code checks if the constant R_LUMEN_BIN (refers to the Red LED's bin) is 'F' and accordingly sets the constant R_MAX_LUMENS (the rated flux of the bin). From the CREE Xlamp-7090XR device binning and labeling document (as an example), it is seen that for bin 'F',

   a. The minimum flux at 350 mA is 10.7
   b. The maximum flux at 350 mA is 13.9
   c. The average flux of this bin is (10.7+13.9)/2 = 12.3
   d. This number is scaled by 2^8 so it is not a floating point number. Therefore, 12.3 x 256 = 3148.8 ~= 3149.

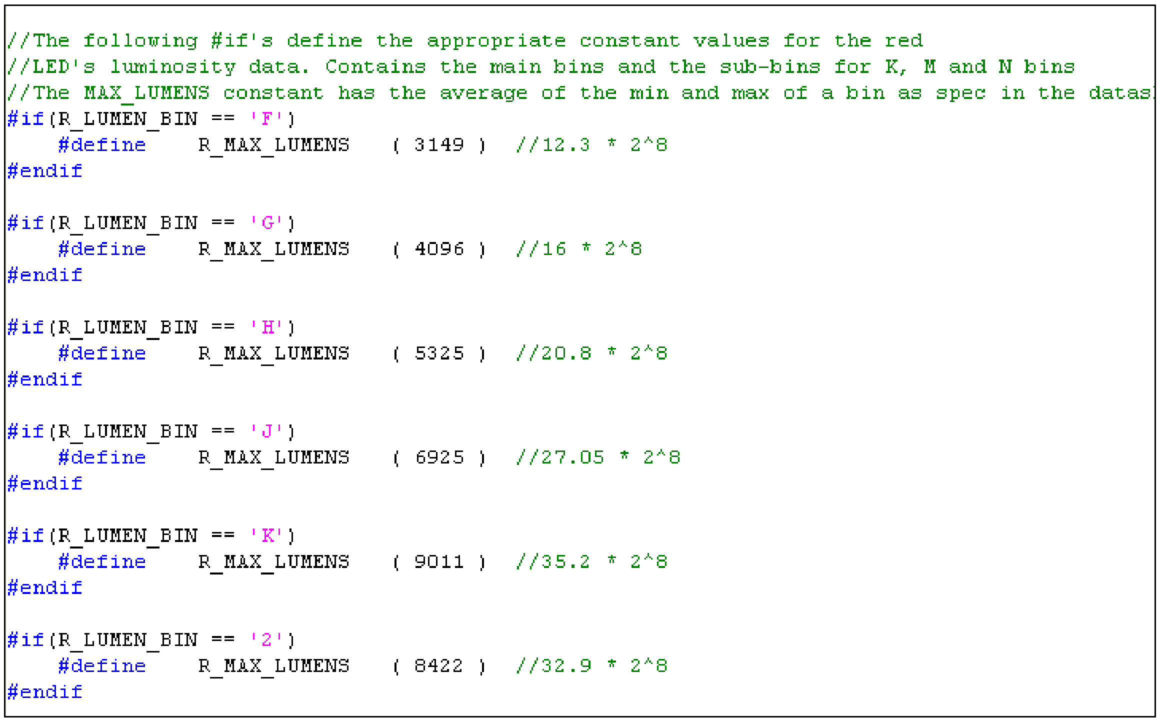
2. The steps outlined in step 1 (a) to (d) are repeated for every unique flux bin that is present in the LED binning and labeling document. Figure 3 illustrates what the statements look like for the Red LED.

3. The whole set of maximum lumen definitions are then repeated for every unique color of LED. For most manufacturers, the LED flux bins are common across all colored LEDs. Hence, the firmware must define the base lumens for each unique color LED. At the end of this step, there should be **'N'** sets of pre-processor definitions with each set having **'M'** individual definitions.

   Here 'N' is the number of unique color LEDs. For a 3-channel color mix system, N = 3. 'M' is the number of unique flux bins present in the data sheet. As an example, for CREE 7090XR series, M = 14.

   In total, there are 14 x 3 = 42 pre-processor definition sets for flux bins.

Figure 3. Luminous Flux Bins for Red CREE 7090XR Series



```
//The following #if's define the appropriate constant values for the red
//LED's luminosity data. Contains the main bins and the sub-bins for K, M and N bins
//The MAX_LUMENS constant has the average of the min and max of a bin as spec in the datas
#if(R_LUMEN_BIN == 'F')
    #define     R_MAX_LUMENS    ( 3149 )   //12.3 * 2^8
#endif

#if(R_LUMEN_BIN == 'G')
    #define     R_MAX_LUMENS    ( 4096 )   //16 * 2^8
#endif

#if(R_LUMEN_BIN == 'H')
    #define     R_MAX_LUMENS    ( 5325 )   //20.8 * 2^8
#endif

#if(R_LUMEN_BIN == 'J')
    #define     R_MAX_LUMENS    ( 6925 )   //27.05 * 2^8
#endif

#if(R_LUMEN_BIN == 'K')
    #define     R_MAX_LUMENS    ( 9011 )   //35.2 * 2^8
#endif

#if(R_LUMEN_BIN == '2')
    #define     R_MAX_LUMENS    ( 8422 )   //32.9 * 2^8
#endif
```

4. To complete the flux binning compensation, the actual bins used in conjunction with this firmware should be stated at the start of the file.

   The lines that perform this function are:

```
#define     R_LUMEN_BIN     ('G')
   //From the CREE binning document
#define     G_LUMEN_BIN     ('H')
#define     B_LUMEN_BIN     ('J')
```

   These pre-processor statements tell the firmware to fix the base lumens values of the individual LEDs according to the bins specified. As an example, since R_LUMEN_BIN is defined as ‘G’, the firmware searches for the appropriate #IF(R_LUMEN_BIN == ‘G’) and fixes the rated lumen accordingly.

Part B: Color Bins

5. The next part of the binning firmware is the color bins. These bins define the dominant wavelength for each colored LED.

   Many LED manufacturers supply only the dominant wavelength values for each bin and omit the actual color coordinate data. If necessary, this data must be procured from the LED manufacturer.

   Sometimes, LED manufacturers supply this information as sample test data with four corner points marking the boundaries of the region where the color coordinates of an LED from that particular bin could fall into. These points are usually close to each other on the CIE chart and the average of the x and y coordinates is calculated and used as the color coordinate point of the LED of that bin.

Figure 4. Color Coordinates of Color Bin R2

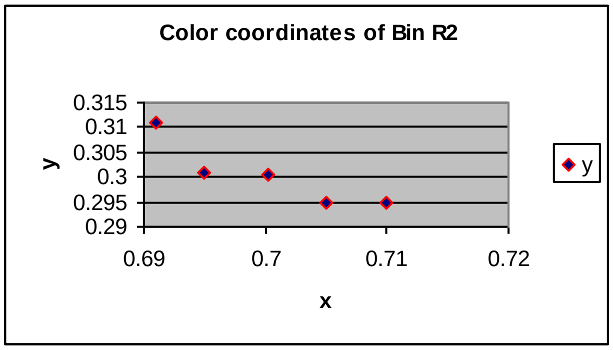


Figure 4 shows what the distribution of color coordinates of a sample of Red LEDs of the color bin R2 could look like. LED manufacturers often do not publish this data and it must be procured from them. The point in the center denotes the point formed by averaging the x and y coordinates of the sample LEDs. These coordinates are used in the firmware corresponding to the bin code R2 as explained in step 6.

Note that although the graph in Figure 4 seems to imply that the points are far apart, the entire space represented by the graph forms a very small part of the actual x-y color space, with a difference of only 0.02 between the x and y extremities.

[+] Feedback

6. For each color bin, the average dominant wavelength number is also calculated from the minimum and maximum values provided.

   The following pre-processor code defines the values for each color bin:

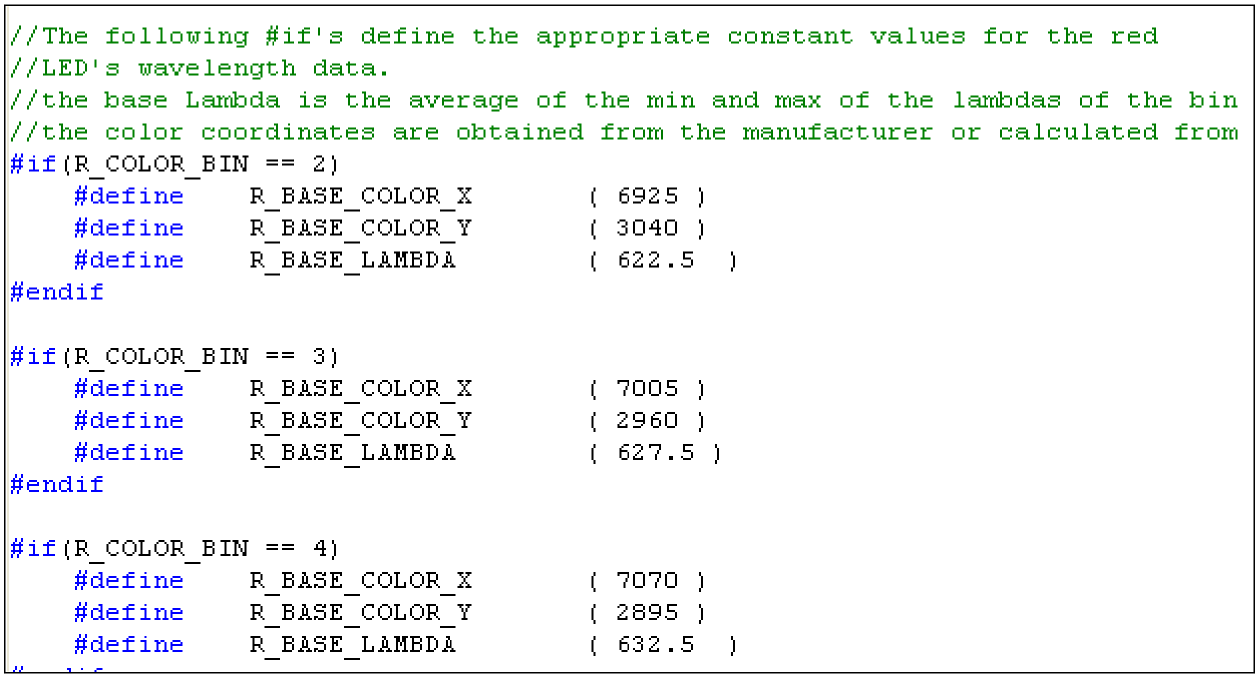
```
#if(R_COLOR_BIN == 2)
    #define    R_BASE_COLOR_X       (
7003 )
    #define    R_BASE_COLOR_Y       (
3005 )
    #define    R_BASE_LAMBDA        (
622.5  )
#endif
```

The average color coordinates calculated in step 5 are scaled by a factor of 10000 to eliminate decimal points. Therefore, the x-coordinate is represented as 7003 (0.70025 x 10000) and the y-coordinate is represented by 3005 (0.3005 x 10000). The wavelength number, although present, is not used by the firmware at this point.

For every color bin, a set of pre-processor statements is written similar to the one shown earlier. So, if three LEDs are used and there are three color bins per LED, the total number of such statements is 3 x 3 = 9.

Figure 5 shows the color bin definitions in the *bin_tables.h* file of the color mixing firmware.

Figure 5. Definition of Color Bins for Red LED



```
//The following #if's define the appropriate constant values for the red
//LED's wavelength data.
//the base Lambda is the average of the min and max of the lambdas of the bin
//the color coordinates are obtained from the manufacturer or calculated from
#if(R_COLOR_BIN == 2)
    #define    R_BASE_COLOR_X       ( 6925 )
    #define    R_BASE_COLOR_Y       ( 3040 )
    #define    R_BASE_LAMBDA        ( 622.5  )
#endif

#if(R_COLOR_BIN == 3)
    #define    R_BASE_COLOR_X       ( 7005 )
    #define    R_BASE_COLOR_Y       ( 2960 )
    #define    R_BASE_LAMBDA        ( 627.5 )
#endif

#if(R_COLOR_BIN == 4)
    #define    R_BASE_COLOR_X       ( 7070 )
    #define    R_BASE_COLOR_Y       ( 2895 )
    #define    R_BASE_LAMBDA        ( 632.5  )
```

7. The final step to completing the color bin firmware is to define which color bin LEDs are actually used in the current project. Similar to the pre-processor statements described in step 4, the color bin is defined at the start of the file before all other pre-processor statements:

```
#define    R_COLOR_BIN     ( 2 )
#define    G_COLOR_BIN     ( 4 )
#define    B_COLOR_BIN     ( 3 )
```

   This code is also seen in Figure 2. The #define statements declare that the color bins used for R, G, and B LEDs are 2, 4, and 3 respectively. As before, the color bins may be called R2, G4, and B3. Here, 2 refers to R2 when used in reference with the constant R_COLOR_BIN and G2 when used in reference with the constant G_COLOR _BIN and so on.

Part C: Forward Voltage Bins

The change in forward voltage with different bins is a detail that affects the regulator or driver circuit responsible for driving current through the LEDs. Therefore, the tolerance for different forward voltage bins must be accounted for in the regulator and driver circuit design.

The header file containing the binning information as explained in this application note is attached as a text file for reference. It gives an idea as to how the complete file looks.

Summary

This application note describes the steps to create or modify LED binning tables in firmware for use with Cypress's HB-LED controllers. This process is performed during development phase and allows the evaluation of a multitude of LED flux and color bins. It also enables the use of LEDs from multiple bins in a production design.

Note that color accuracy directly depends on the LED characteristics programmed into firmware. Data from LED data sheets and binning documents may not be sufficient for extremely high color accuracy applications. For more detail on color accuracy, refer to the Cypress Application Note AN33640, *Color Mixing Accuracy with EZ-Color High-Brightness LED Controllers*.

About the Author

Name:	Mukund Krishna
Title:	Applications Engineer
Background:	Mukund has a graduate degree in Electrical Engineering from the University of Southern California. He currently works in the EZ-Color lighting solutions group, where he is responsible for aiding customer designs, designing collateral, and product definitions.
Contact:	Email: Mukund.Krishna@cypress.com Ph: (408)-432-7058

Document History

Document Title: Firmware—Building Binning Tables for Color Mixing Using EZ-Color HB-LED Controllers

Document Number: 001-47518

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	2542293	UKK	07/23/08	New Application Note

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com/