

# Implementing an LED (RGB) Controller on an IGLOO<sup>®</sup> FPGA

## 1.0 Introduction

The main objective of this design is to mix colors using the pulse width modulation (PWM) method with red-green-blue (RGB) LEDs. The color mixing control is available through hardware as well as software using the serial interface.

## 2.0 General Implementation Overview

Three 8-bit PWM blocks are used to control the intensity of individual colors (red, green, or blue of an RGB LED).

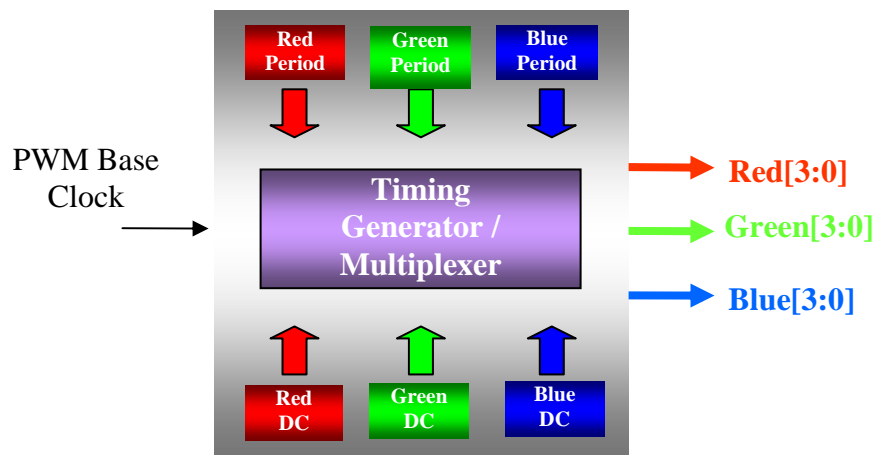


Figure 1. RGB Color Mixing IP

## 3.0 Reference Design Implementation

Since each of these 8-bit PWMs can be controlled individually, a range of colors can be generated by a combination of variations in the period and duty cycle of these PWMs. Using 3 PWMs for the red, green, and blue LEDs, different color combinations can be generated (0-255 red, 0-255 green, and 0-255 blue).

The individual controls for each LED are provided as hardware control signals (PLUS and MINUS). A PLUS control signal increases the duty cycle while the MINUS control signal decreases the duty cycle of the corresponding LED. The overall brightness of the combination of LEDs can also be controlled using the BRT and DIM control signals, as shown in Figure 2. A serial interface is also provided to directly write the desired duty cycle value for each of the RGB LEDs, as shown in Figure 3.

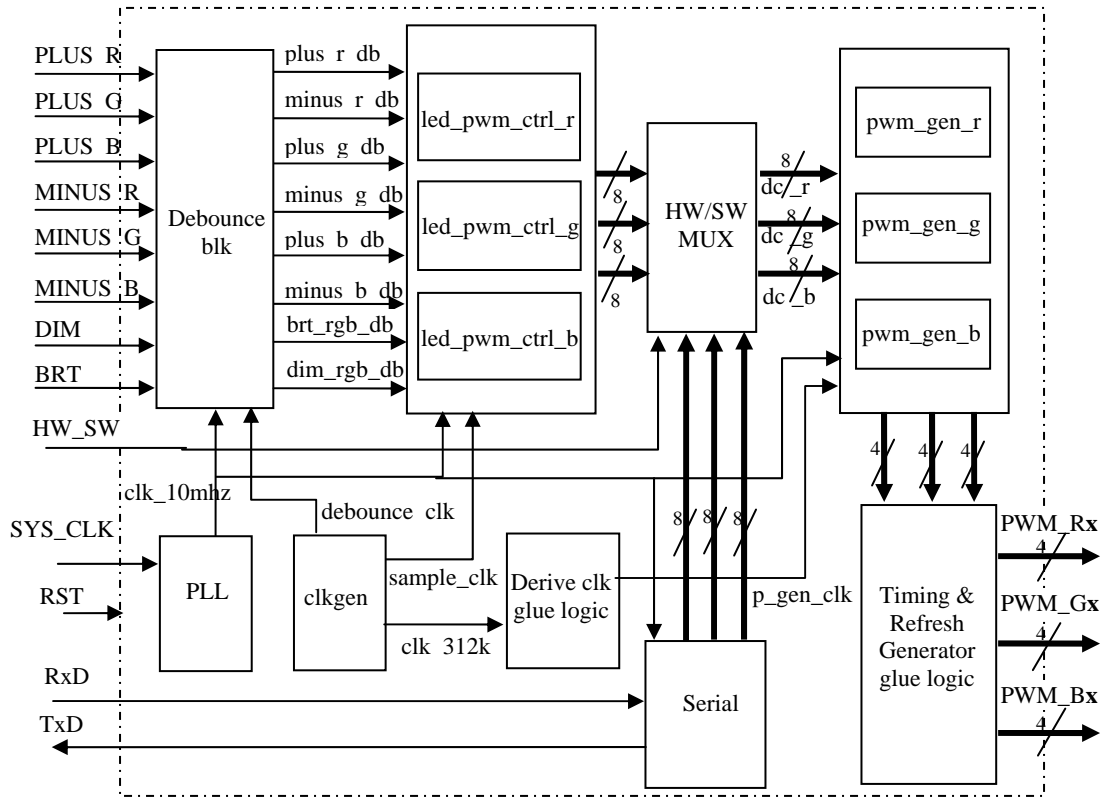


Figure 2. RGB – Color Mixing Logic Block Diagram

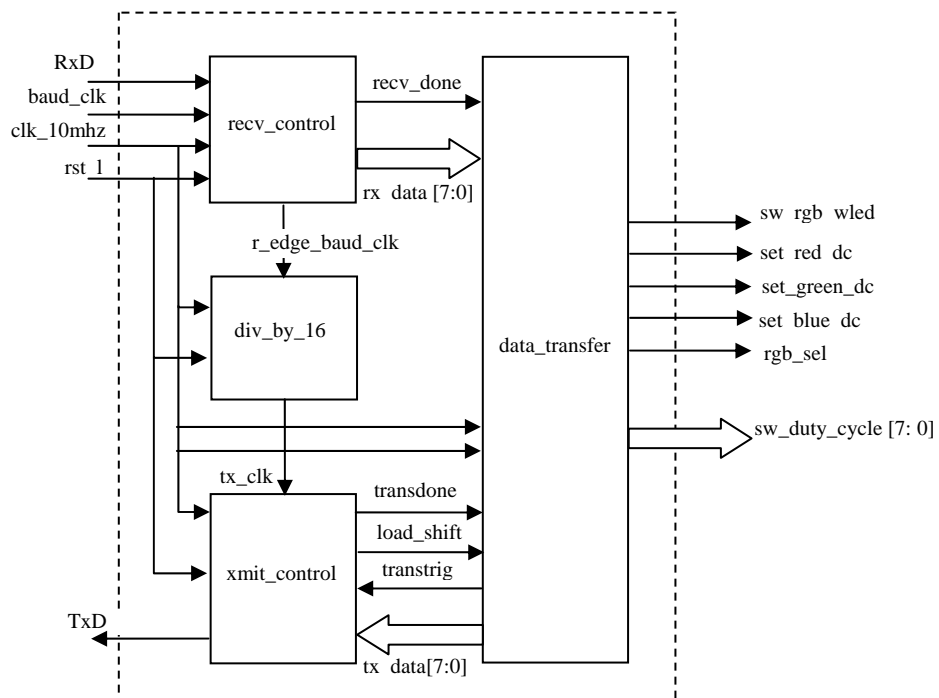


Figure 3. Serial Interface (RGB)

## 4.0 I/Os

The following table describes the main I/O's in the design:

Signal	Input/Output	Description
PWM_R[3:0]	Output	Output for RED LED
PWM_G[3:0]	Output	Output for GREEN LED
PWM_B[3:0]	Output	Output for BLUE LED
TXD	Output	Transmit – Serial Output
PLUS_R	Input	Input to increase the brightness of RED
PLUS_G	Input	Input to increase the brightness of GREEN
PLUS_B	Input	Input to increase the brightness of BLUE
MINUS_R	Input	Input to decrease the brightness of RED
MINUS_G	Input	Input to decrease the brightness of GREEN
MINUS_B	Input	Input to decrease the brightness of BLUE
DIM	Input	Input to decrease the brightness of RGB
BRT	Input	Input to increase the brightness of RGB
HW_SW	Input	Hardware / Software selection 1 = Hardware 0 = Software
RXD	Input	Receive - Serial Input
SYS_CLK	Input	System Clock – 20Mhz
SYS_RESET_N	Input	Master Reset – Active Low

## 5.0 Waveforms

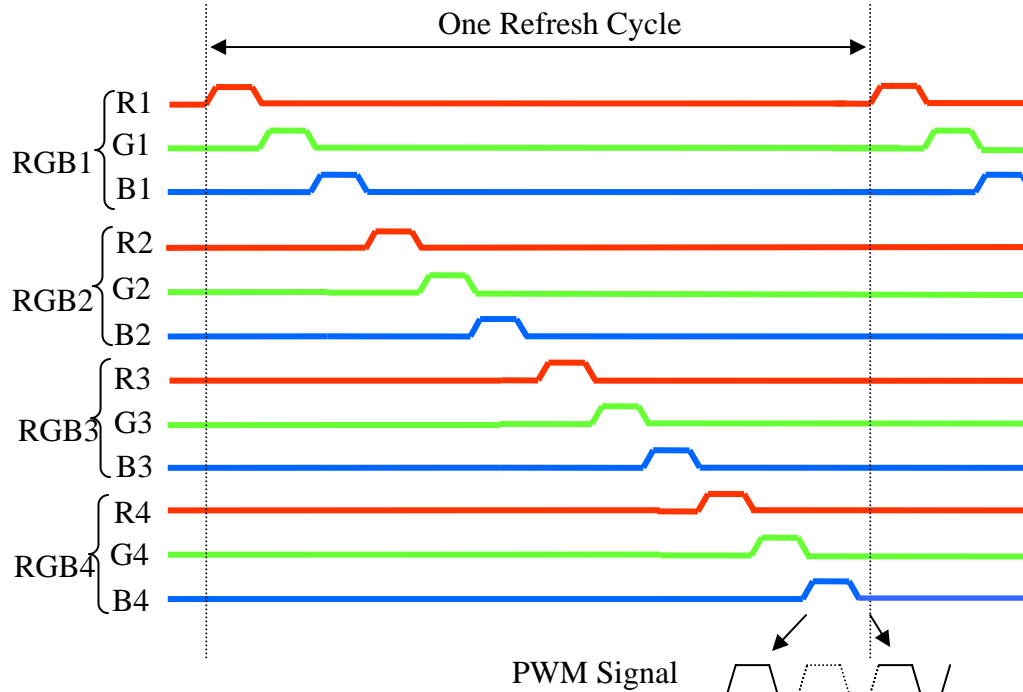


Figure 4. Time-Multiplexed Refresh Cycle (RGB)

The PWM outputs vary according to the value of the duty cycle programmed. If the user selects BRT as the input control, the intensity of all four RGB LEDs increases at the same time. If the user selects DIM as the input control, the intensity of all four RGB LEDs decreases at the same time. For individual control of the red, green, or blue colored LEDs, the intensity of that particular color increases (PLUS\_R, PLUS\_G, PLUS\_B) or decreases (MINUS\_R, MINUS\_G, MINUS\_B) in all four RGB LEDs at the same time. The time multiplexing minimizes the power consumption as all these LEDs are not lit at the same time. The process is repeated for every refresh cycle, as shown in Figure 4 above.

## **6.0 Conclusion**

This scheme can be used to generate keypad backlight or LCD backlight of any color or illuminate a particular area with required color using the RGB LEDs. Pulse width modulation allows for control of brightness through power management for enhanced battery life. As each RGB LED is lit in a time-multiplexed fashion, this method is beneficial in saving on total power consumption (as shown in Figure 4).

## **Appendix A – RGB LED Controller Design Example**

### **Design Files Summary**

<b>Files</b>	<b>Functionality</b>
clk_by_2.v	Divides input clock by 2 -- Toggle F/F
clk_gen.v	Clock Generator
data_transfer.v	Transfers the data according to commands given by PC
debounce.v	Debounce Logic Block
debounce_blk.v	Interconnects all debounce blocks
divide_by_16.v	Divide by 16 block for serial communication – baud clock
divide_by_5.v	Derived clock for internal use
led_pwm_ctrl.v	Dutycycle for PWM Unit
pwm_gen.v	PWM Generator
recv_control.v	This block receives data serially on RxD
xmit_control.v	This block transmits data serially on TxD.
serial.v	This block connects xmit_control, recv_control, data_transfer
rgb_ip.v	This block interconnects rgb_pwm blocks, serial etc
top_rgb_ip.v	This block interconnects rgb_ip and PLL
top_tb.v	Test_bench for rgb_ip

## **About Ishnatek**

Ishnatek offers FPGA design and hardware prototyping services. Ishnatek offers Design and Verification Services for embedded solutions. We have IPs such as 8031, RTC, Timers, Enhanced PWM, UART/SIO/IrDA, I2C, LED Driver and Key scan, Parallel Port, ECP/EPP, etc., which can be building blocks for your embedded controller solutions.