Using GreenPAK's I2C Port to Control an RGB LED Breathing Pattern

Authors: Chuck Husted and Luke Thomas

Date: March 01, 2016

Introduction

This app note explains how to reconfigure the SLG46531's registers via I2C. Specifically, it shows how to change the registers associated with counters that set the PWM ramp for driving the R, G, and B cathodes of an LED. This creates a breathing pattern of light.

It is important to note that any reconfigurations via I2C are volatile and will revert to the programmed code after the POR inside the GreenPAK resets.

GreenPAK Benefits

GreenPAK is a very versatile, low current consumption IC. It can offload functions from other microcontrollers and larger SOC's. For example, a microcontroller can be active, drawing several mA of current, and can write via I2C to the SLG46531V to set the RGB "breathe time" (or PWM ramp pattern). The GreenPAK would manage the RGB light function while the microcontroller enters a deep sleep mode to save system current. When desired, the GreenPAK can wake up the microcontroller using another I/O pin as an interrupt.

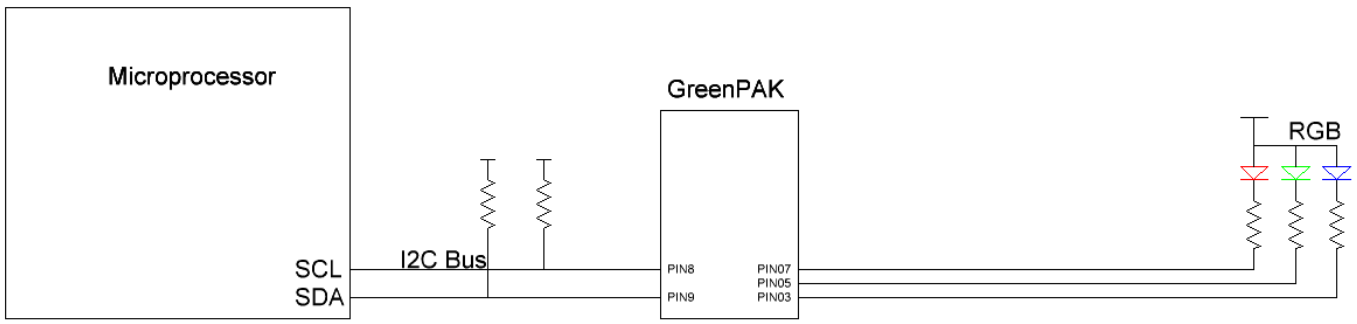

Figure 1. GreenPAK Used as an RGB Driver in a Larger System

GreenPAK Configuration

The GreenPAK design shown in Figure 2(a) implements a simple RGB LED driver. On the GreenPAK Universal Dev Board, connect Pin 3 to the cathode of the blue LED, Pin 5 to the cathode of the green LED and Pin 7 to the cathode of the red LED. The common anode should be connected to the VDD pin of the GreenPAK. Remember to use current limiting resistors in series with the LEDs as needed.

The breathing pattern is generated through a constant change between two counters. Each counter will output a high pulse for one clock cycle of their programmed period. By programming CNT3 to be one count higher than CNT4, we have generated an offset of one clock width between the output pulses of the counters. These two signals are then XOR'd into the clock input of a DFF configured to toggle on each rising edge of its clock input.

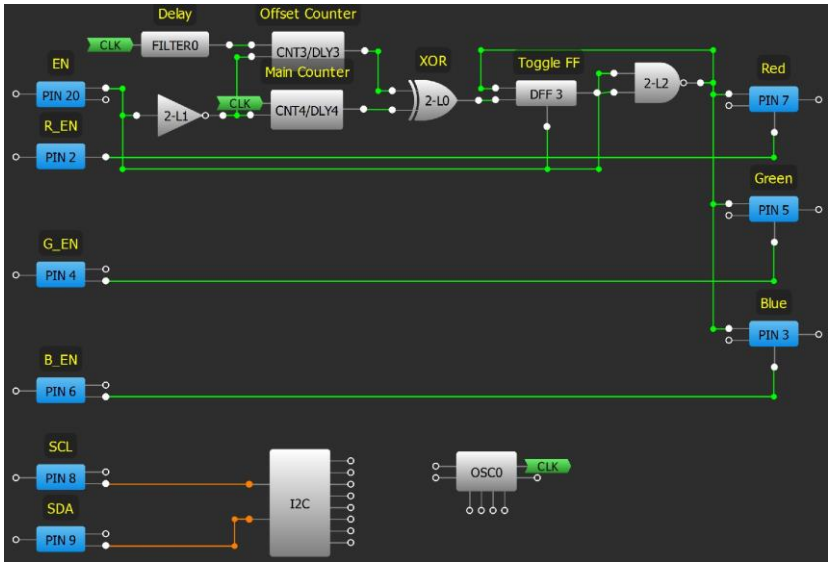

Figure 2(a). GreenPAK block diagram with counters for setting the PWM of each color

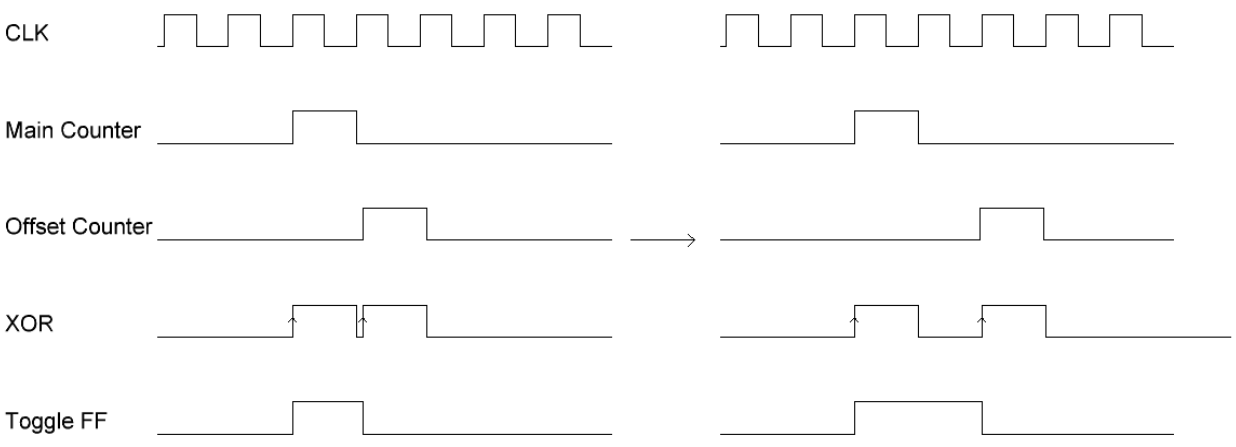

Figure 2(b). Counters and logic timing

Figure 2(b) shows the first two periods of the counters. You can see the shift of the Offset counter relative to the main counter by one clock cycle each period of the Main counter. The time difference between counter outputs determines the pulse width of the output LED signal. The relative difference between the outputs of the Main and Offset counters will grow until the output signal is at full PWM. It will then flip around and begin to decrease the PWM until reaching the minimum duty cycle which will start the whole process over again.

To set the RGB color, this design was configured to accept three separate external PWM signals on pins 2, 4 and 6. Pin 20 is used as an enable for the three LED outputs.

Creating an I2C Command to Write Register Bits

This application note will not cover the basic GPAK I2C command format. Instead, we will emphasize the specific commands to implement the RGB LED. In the example GreenPAK design file, we have used a default chip address of 0x00. Note '[' represents an I2C start bit and ']' represents a stop bit. Since the design uses two counters we need to know the address of each counter's register.

Counter	Byte Address (Hex)
CNT3	0xC1
CNT4	0xC2

Table 1. Counter register address

Table 1 provides the register address for each of the counters used in this design. Changing these registers will impact the PWM ramp time of the LED.

By putting together all of this information, we can create a command to turn on and off the three LED's in a pulsing pattern by writing the following to the GreenPAK:

To write to CNT3 we use this command which will write 21 (decimal) to the CNT3 register:

[0x00 0xC1 0x15]

Similarly, for CNT4 we use this command to write 20 (decimal) to the CNT4 register:

[0x00 0xC2 0x14]

Using the GreenPAK Development Kit Emulator

We can use the GreenPAK Development kits emulator to create the signals that we need to drive the LEDs properly. First we must connect the development kit to a computer or laptop's USB port. Second, place a SLG46531V chip into the socket. Third, open the design file inside the GreenPAK Designer software and click the "Emulator" button in the upper toolbar. This should appear as in Figure 3(a). Next we must enable the I2C tools within the software. This is the button to the right side of the screen.

After enabling the I2C tools, we must select the I2C "Virtual Inputs" button. This tool allows us to write directly to the counter registers. Please see Figure 3(b).

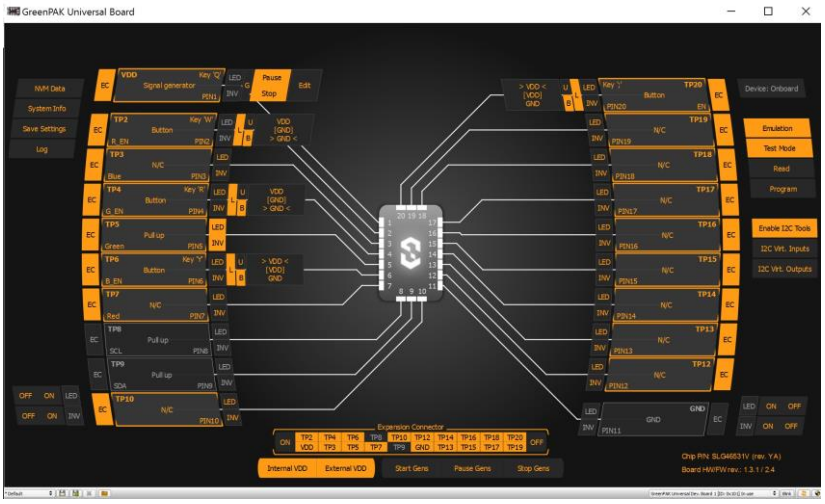

Figure 3(a). GreenPAK emulator with I2C Virtual Inputs

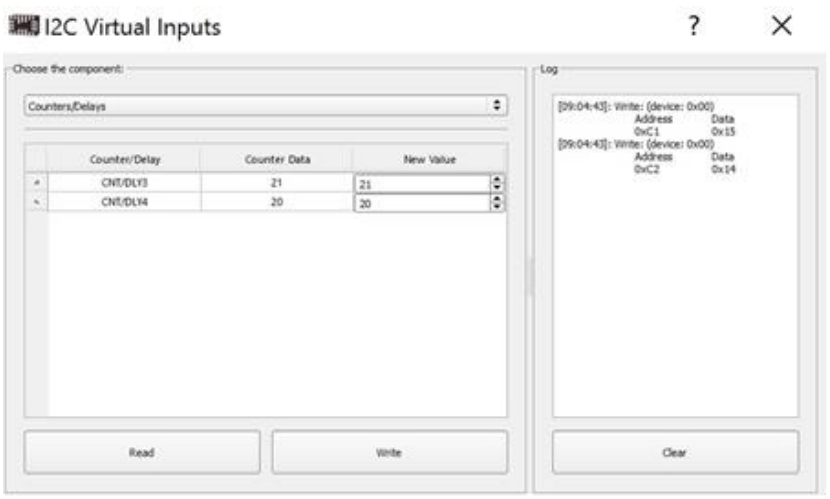

Figure 3(b). GreenPAK emulator I2C Virtual Inputs window

Enter the value of 21 (decimal) into the "New Value" box for counter 3 (CNT3). This sets the offset counter. Next enter 20 into the "New Value" box for Counter 4 (CNT4) and press the "Write" button. This will enable the RGB LED breathing pattern and continue as long as the OE pin (Pin 20) is set to a high value and the individual PWM inputs are high.

Please see Figure 3(c) for what the breathing PWM pattern looks like.

Likewise, as long as a value within the CNT3 and CNT4's register range is entered, and CNT3's register is one value higher than CNT4, any length of time for the breathing pattern can be created.

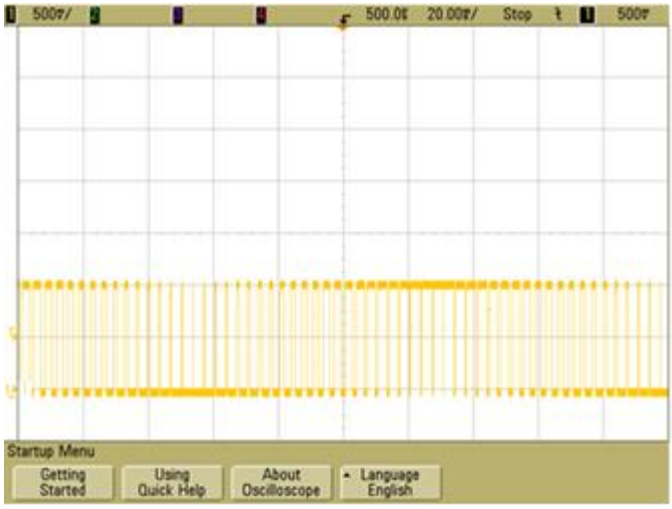

Figure 3(c). Oscilloscope capture showing PWM ramp up and down for the LEDs

Conclusion

We can use a GreenPAK SLG46531V to implement RGB LED driver functions. By implementing a system architecture this way, a microcontroller or other system SOC can be put in sleep mode to save overall system power consumption which is desirable for battery based portable and wearable systems.

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers skilled in the art designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only for development of an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising out of your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Rev.1.0 Mar 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.