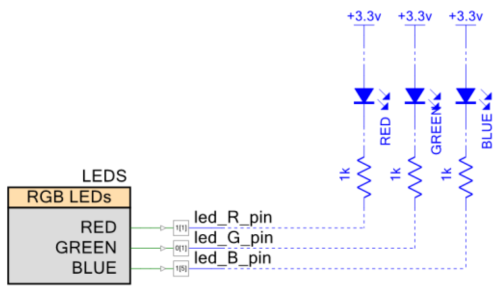
RGB LEDS

MICA

Controlling Leds

Features



- Control RGB LEDs through pin-by-pin basis
- Use software API for controlling colors

General Description

Users should use the RGB LEDs component when there is a Red, Green, and Blue LED connected to the PSoC. This API does not support hardware PWM blocks for driving the LEDs. There are two versions of the component: active high and active low. Active high should be used when the pin is connected to the anode of the LEDs, and active low should be used when the pin is connected to the cathode of the LEDs. Mixed high/low support is currently not available.

Input/Output Connections

This section describes all of the virtual connections for the RGB LEDs component.

RED – Output

This should be connected to the virtual pin that represents the red LED. For active high configurations the pin should be strong drive, for active low configurations, the pin should be open drain, drives low.

GREEN– Output

This should be connected to the virtual pin that represents the green LED. For active high configurations the pin should be strong drive, for active low configurations, the pin should be open drain, drives low.

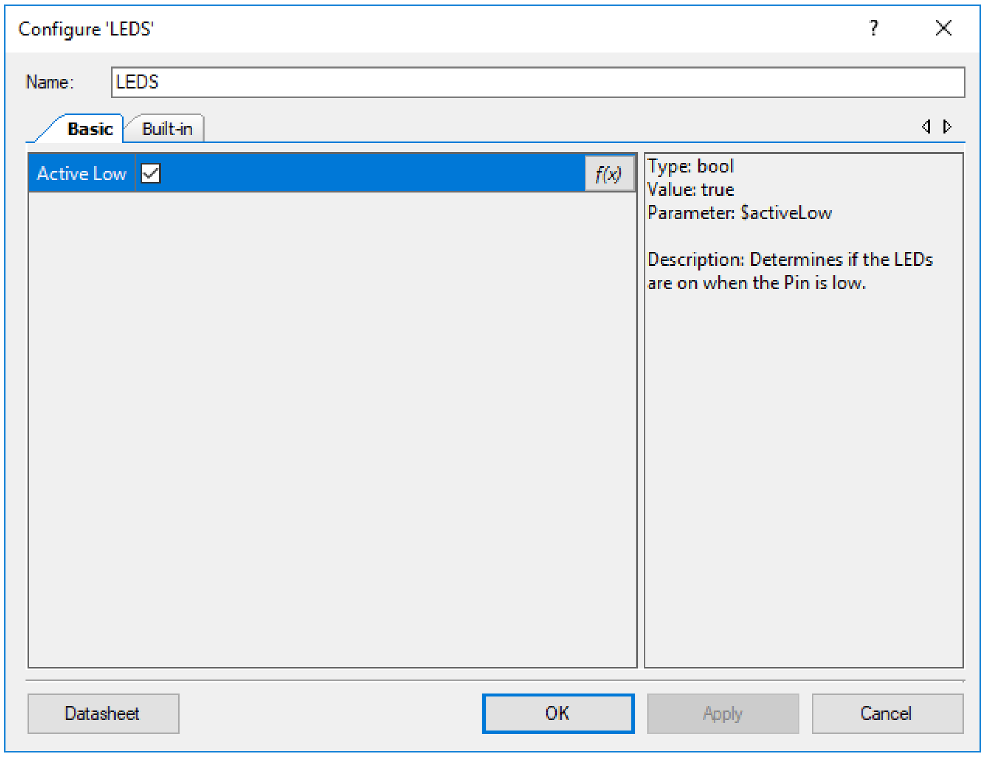
BLUE – Output

This should be connected to the virtual pin that represents the blue LED. For active high configurations the pin should be strong drive, for active low configurations, the pin should be open drain, drives low.

Component Parameters

Double click on the RGB LEDs component to open the Configure dialog.

Basic Tab



Configure 'LEDS'

Name: LEDS

Active Low ✓

Type: bool
Value: true
Parameter: $activeLow

Description: Determines if the LEDs are on when the Pin is low.

The RGB LED Component has the following parameters.

Active Low

When selected, this indicates that the LEDs connected to the component will be on when a logical low is present on the hardware pin. It is suggested (but not required) that the virtual pin be designated Open Drain, Drives Low if this is selected.

Application Programming Interface (API)

API routines allows users to control the LEDs from software. The default name for the component is “LEDS”, which can be changed in the configure dialogue. All functions and constants are generated based on this name.

Functions

Function	Description
LEDS_Write()	Sets the state of the LED control register, setting the color of the LEDs on the board.
LEDS_Read()	Returns the current value of the LED control register
LEDS_Sleep()	Prepares the component for sleep. Sleeps the control Register.
LEDS_Wakeup()	Wakes up the component from sleep. Wakes the control Register.
LEDS_R_Write()	Sets the state of the Red LED.
LEDS_G_Write()	Sets the state of the Green LED.
LEDS_B_Write()	Sets the state of the Blue LED.
LEDS_Test()	Runs through the testing routine for the LEDs Expected Outcome: 0. All LEDs off 1. Red LED on 2. Green LED on 3. Blue LED on 4. Yellow (RB) 5. Cyan (BG) 6. Magenta (RG) 7. White (RGB) LEDs on

uint8 LEDS_Write(uint8 state)

Description:	Sets the state of the LED control register, setting the color of the LEDs on the board.
Parameters:	uint8: The new state mask to write to the control register. It is recommended that users only pass the color macros defined by the component.
Return Value:	uint8: Value of the control register after it was written.

uint8 LEDS_Read(void)

Description: Returns the current value of the LED control register

Return Value: uint8: Value held in the LED control register

void LEDS_Sleep(void)

Description: Prepares the component for sleep. Sleeps the LED control Register

Side Effects: Places the LED control registers in sleep, it does not however, change the status of the virtual pins. I.e. an active low virtual pin will remain open drain drives low, and not Hi-Z.

void LEDS_Wakeup(void)

Description: Wakes up the component from sleep. Wakes the control Register.

Side Effects: Sleep does not change the status of the virtual pins. I.e. a pin that has been placed in Hi-Z will not be set back to open drain drives low.

uint8 LEDS_R_Write(bool state)

Description: Sets the state of the Red LED.

Parameters: bool: Desired state of the red pin. Users should utilize the LED_ON and LED_OFF macros defined by the component.

Return Value: uint8: Value of the entire LED control register.

uint8 LEDS_G_Write(bool state)

Description:	Sets the state of the Green LED.
Parameters:	bool: Desired state of the green pin. Users should utilize the LED_ON and LED_OFF macros defined by the component.
Return Value:	uint8: Value of the entire LED control register.

uint8 LEDS_B_Write(bool state)

Description:	Sets the state of the Blue LED.
Parameters:	bool: Desired state of the blue pin. Users should utilize the LED_ON and LED_OFF macros defined by the component.
Return Value:	uint8: Value of the entire LED control register.

void LEDS_Test(uint8 runs)

Description: Runs through the board level testing routine for the LEDs.

Parameters: uint8: The number of times to iterate through the test. 0 means infinite.

Expected Outcome:

0. All LEDs off
1. Red LED on
2. Green LED on
3. Blue LED on
4. Yellow (RB)
5. Cyan (BG)
6. Magenta (RG)
7. White (RGB) LEDs on

Change Log

This sections lists changes to the component from previous versions

Version	Description of Changes	Reason for Changes / Impact
v1.0 r1.0	Update datasheet	Updated MICA component template.
v1.0 r0.0	Initial implementation of the component and datasheet	