

## Objective

This example shows how to use the PSoC Creator™ Timer Counter Pulse Width Modulator (TCPWM) Component configured as a Timer/Counter in a PSoC® 4 device.

## Overview

This example contains three projects that use the Timer Counter Component. The Counter\_Count project demonstrates the Timer to keep track of the number of button presses with LED to show the count changing. The Counter\_Frequency\_DutyCycle project measures the frequency and duty cycle of an input waveform using counter mode and prints the results over UART. The Counter\_Periodic\_Interrupt uses the timer mode to create periodic interrupts that blink an LED.

## Requirements

**Tool:** PSoC Creator 4.2

**Programming Language:** C (Arm® GCC 5.4.1)

**Associated Parts:** PSoC 4 family

**Related Hardware:** CY8CKIT-042 PSoC 4 Pioneer Kit

## Hardware Setup

This code example is set up for CY8CKIT-042. If you are using a different kit, see .

[Reusing this Example.](#)

For CY8CKIT-042, the USB-UART bridge in KitProg2 module is used:

1. Connect the \UART:rx\ pin P0[4] to P12[7] on header J8.
2. Connect the \UART:tx\ pin P0[5] to P12[6] on header J8.

Other kits use different pins for the UART. Make sure that you select the pins that are right for your kit.

## Software Setup

This design requires a terminal emulator such as PuTTY or Tera Term running on your computer.

## Operation

1. Connect the USB cable between the PC and the PSoC 4 Pioneer Kit.
2. Build the project and program it into the PSoC 4 device. Choose **Debug > Program**. For more information on device programming, see PSoC Creator Help.
3. Open a terminal emulator on your computer and configure the program to the appropriate COM port. Configure the baud rate to 115200, data bits to 8, no parity bits, stop bit as 1, and no control flow.
4. For Counter\_Count project: Press the kit button SW2 and confirm that the count displayed on the terminal increments accordingly.
5. For Counter\_Frequency\_DutyCycle project: Change the period and compare register settings of the PWM to create different periods and duty cycles. Confirm that the frequency and duty cycle are correctly displayed on the terminal.
  - a. An alternative to using the PWM is to use a digital input pin and feed in your own input waveform. Note that this requires an inverter so that Timer\_2 can use the inverse of the input waveform.
6. For Counter\_Interrupt project: confirm that the red LED turns on and off every two seconds. No terminal emulator is needed for this project



## Counter\_Periodic\_Interrupt

In the Counter\_Periodic\_Interrupt example, the following functions are performed:

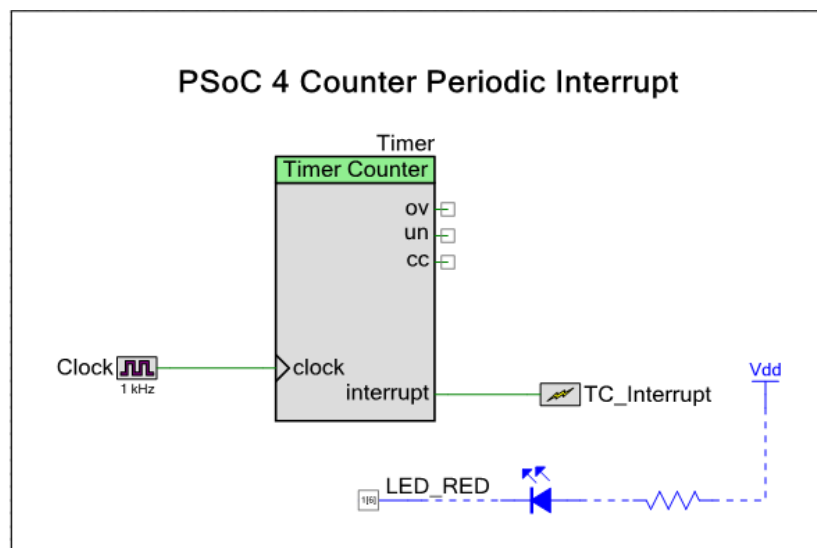
1. The Timer Counter Component is started
2. The timer counter interrupt handler function TC\_InterruptHandler is configured.
3. An interrupt occurs when the timer's count reaches the terminal count, which is determined by the period set for timer.

The ISR\_Timer function does the following:

1. Clears the interrupt for terminal count.
2. Toggles the LED ON/OFF state.

The top-level schematic of the PSoC Creator project is shown in [Figure 3](#).

Figure 3. Schematic, Counter\_Periodic\_Interrupt



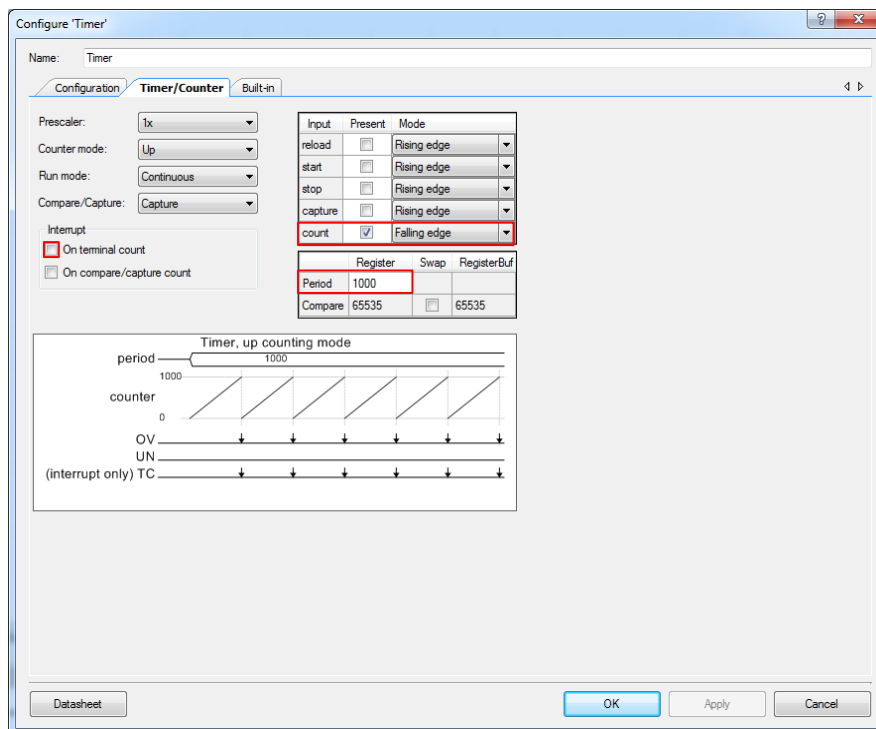
## Components and Settings

[Table 1](#) lists the PSoC Creator Components used in this example, how they are used in the design, and the non-default settings required so they function as intended.

Table 1. PSoC Creator Components

Component	Instance Name	Purpose	Non-default Settings
Timer Counter (TCPWM mode) [v2.10]	Timer	Handle the Timer/Counter operation	See <a href="#">Figure 4</a>
Timer Counter (TCPWM mode) [v2.10]	Timer_1 & _2	Handle the Timer/Counter operation	See <a href="#">Figure 5</a>
Timer Counter (TCPWM mode) [v2.10]	Timer (for periodic interrupt project)	Handle the Timer/Counter operation	Set period register to 2000.
PWM (TCPWM mode) [v2.10]	PWM	Handle the PWM operation	Set period register to 4999. Set compare register to 2499.
UART (SCB mode) [v4.0]	UART	Handle UART communication	None
Digital Input Pin	Pin_Switch	Handle the SW2 connection on device	See <a href="#">Figure 6</a>

Figure 4. Timer Parameter Settings



The 'Configure Timer' dialog box is shown with the 'Timer/Counter' tab selected. The settings are as follows:

- Name: Timer
- Prescaler: 1x
- Counter mode: Up
- Run mode: Continuous
- Compare/Capture: Capture
- Interrupt:
  - ☒ On terminal count
  - ☐ On compare/capture count

Input	Present	Mode
reload	<input type="checkbox"/>	Rising edge
start	<input type="checkbox"/>	Rising edge
stop	<input type="checkbox"/>	Rising edge
capture	<input type="checkbox"/>	Rising edge
count	<input checked="" type="checkbox"/>	Falling edge

	Register	Swap	RegisterBuf
Period	1000	<input type="checkbox"/>	
Compare	65535	<input type="checkbox"/>	65535

Timer, up counting mode

period: 1000

counter: 0

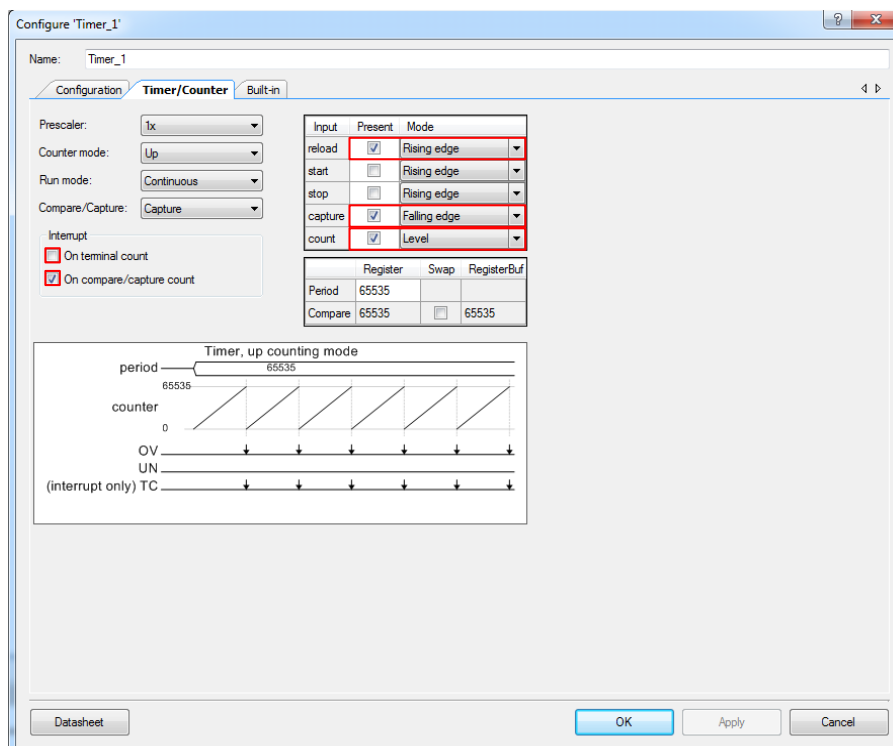
OV

UN

(interrupt only) TC

Buttons: Datasheet, OK, Apply, Cancel

Figure 5. Tiimer\_1 (2) Parameter Settings



The 'Configure Timer\_1' dialog box is shown with the 'Timer/Counter' tab selected. The settings are as follows:

- Name: Timer\_1
- Prescaler: 1x
- Counter mode: Up
- Run mode: Continuous
- Compare/Capture: Capture
- Interrupt:
  - ☐ On terminal count
  - ☒ On compare/capture count

Input	Present	Mode
reload	<input checked="" type="checkbox"/>	Rising edge
start	<input type="checkbox"/>	Rising edge
stop	<input type="checkbox"/>	Rising edge
capture	<input checked="" type="checkbox"/>	Falling edge
count	<input checked="" type="checkbox"/>	Level

	Register	Swap	RegisterBuf
Period	65535	<input type="checkbox"/>	
Compare	65535	<input type="checkbox"/>	65535

Timer, up counting mode

period: 65535

counter: 0

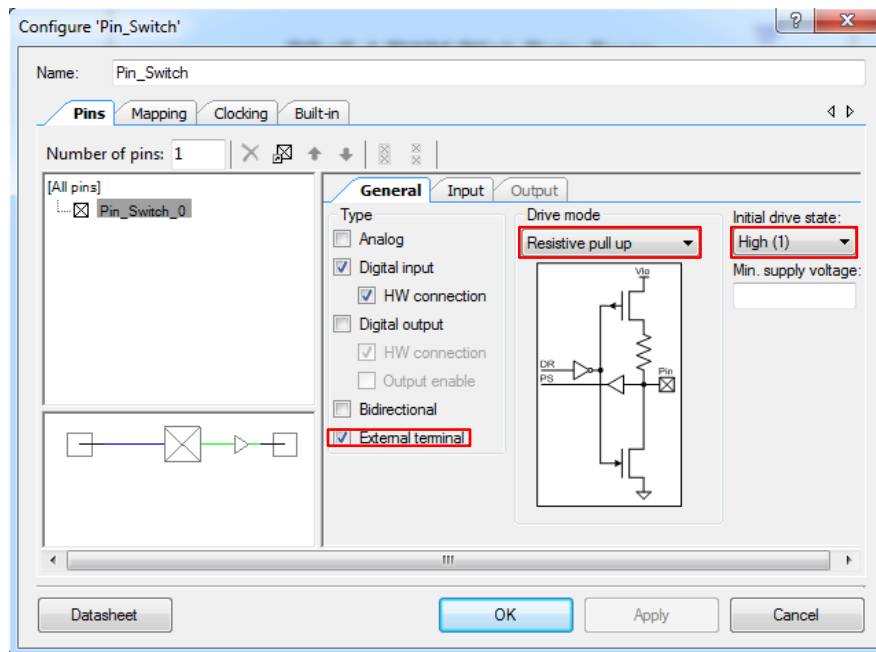
OV

UN

(interrupt only) TC

Buttons: Datasheet, OK, Apply, Cancel

Figure 6. Pin\_Switch Parameter Settings



For information on the hardware resources used by a Component, see the [Component datasheet](#).

## Reusing this Example

This example is designed for the CY8CKIT-042 pioneer kit. To port this design to a different PSoC 4 device, kit, or, both, do the following:

1. In PSoC Creator, select **Project > Device Selector** to change the target device. Select your device as listed in [Table 2](#).
2. Make sure that the **SysClk Desired frequency** is set to 24 MHz after the device is changed.
3. In the PSoC Creator Workspace Explorer, select the **Clocks** interface listed under **Design Wide Resources**.
4. Set the **SysClk Desired Frequency** to 24 MHz, if it is not already.
5. Route \UART:tx\ and \UART:rx\ to the pins listed in [Table 3](#). For the CY8CKIT-048, install jumper wires for \UART:tx\ and \UART:rx\ to P12[6] and P12[7] on header J16, respectively.

Table 2. Development Kits and Associated Devices

Development Kit	Device
CY8CKIT-041	CY8C4146AZI-S433
CY8CKIT-042	CY8C4245AXI-483
CY8CKIT-042-BLE	CY8C4247LQI-BL483
CY8CKIT-044	CY8C4247AZI-M485
CY8CKIT-046	CY8C4248BZI-L489

Table 3. Pin Assignments for Different Kits

Pin Name	Development Kit					
	CY8CKIT-041	CY8CKIT-042	CY8CKIT-042-BLE	CY8CKIT-044	CY8CKIT-046	CY8CKIT-048
\UART:rx\	P0[4]	P0[4]	P1[4]	P7[0]	P3[0]	P0[4]
\UART:tx\	P0[5]	P0[5]	P1[5]	P7[1]	P3[1]	P0[5]

In some cases, a resource used by a code example (for example, an IP block) is not supported on another device. In that case the example will not work. If you build the code targeted at such a device, you will get errors. See the device datasheet for information on what a device supports.

## Related Documents

Application Notes	
<a href="#">AN79953</a> Getting Started with PSoC® 4	Describes PSoC 4 and shows how to build the attached code example
Code Examples	
<a href="#">CE224593</a> PSoC4 TCPWM PWM	Demonstrates PWM driving an LED with changing blink rates and also three PWM 120 degrees out of phase from each other driving LEDs.
<a href="#">CE224595</a> PSoC 4 TCPWM QuadDec	Demonstrates the use of QuadDec to detect the direction of count and direction status is displayed using a LED. Two PWMs are used to simulate the rotational direction as one leads the other.
PSoC Creator Component Datasheets	
<a href="#">TCPWM</a>	Multifunctional component that can implement the following functionalities: PWM, Timer/Counter, and Quadrature Decoder.
<a href="#">SCB</a>	A multifunction hardware block that implements the following communication Components: I2C, SPI, UART, and EZI2C
<a href="#">General Purpose Input/Output (GPIO)</a>	A multifunctional component that allows hardware resources to connect to a physical port-pin and provides access to external signals through an appropriately configured physical IO pin.
<a href="#">Interrupt</a>	The interrupt component defines hardware triggered interrupts. There are three types of system interrupt waveforms that can be processed by the interrupt controller: Level, Pulse, and Edge.
Device Documentation	
<a href="#">PSoC 4 Datasheets</a>	<a href="#">PSoC 4 Technical Reference Manual</a>
Development Kit (DVK) Documentation	
<a href="#">CY8CKIT-042 PSoC 4 Pioneer Kit</a>	
<a href="#">PSoC 4 Kits</a>	
Tool Documentation	
<a href="#">PSoC Creator</a>	Go to the <b>Downloads</b> tab for Quick Start and User Guides

## Document History

Document Title: CE224594 – PSoC 4 Timer/Counter

Document Number: 002-24594

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6371623	SYAO	12/20/2018	New code example



## Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at [Cypress Locations](#).

### Products

Arm® Cortex® Microcontrollers	<a href="http://cypress.com/arm">cypress.com/arm</a>
Automotive	<a href="http://cypress.com/automotive">cypress.com/automotive</a>
Clocks & Buffers	<a href="http://cypress.com/clocks">cypress.com/clocks</a>
Interface	<a href="http://cypress.com/interface">cypress.com/interface</a>
Internet of Things	<a href="http://cypress.com/iot">cypress.com/iot</a>
Memory	<a href="http://cypress.com/memory">cypress.com/memory</a>
Microcontrollers	<a href="http://cypress.com/mcu">cypress.com/mcu</a>
PSoC	<a href="http://cypress.com/psoc">cypress.com/psoc</a>
Power Management ICs	<a href="http://cypress.com/pmic">cypress.com/pmic</a>
Touch Sensing	<a href="http://cypress.com/touch">cypress.com/touch</a>
USB Controllers	<a href="http://cypress.com/usb">cypress.com/usb</a>
Wireless Connectivity	<a href="http://cypress.com/wireless">cypress.com/wireless</a>

### PSoC® Solutions

[PSoC 1](#) | [PSoC 3](#) | [PSoC 4](#) | [PSoC 5LP](#) | [PSoC 6 MCU](#)

### Cypress Developer Community

[Community Forums](#) | [Projects](#) | [Videos](#) | [Blogs](#) | [Training](#) | [Components](#)

### Technical Support

[cypress.com/support](http://cypress.com/support)

All other trademarks or registered trademarks referenced herein are the property of their respective owners.



Cypress Semiconductor  
198 Champion Court  
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2018. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress does not assume any liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit [cypress.com](http://cypress.com). Other names and brands may be claimed as property of their respective owners.