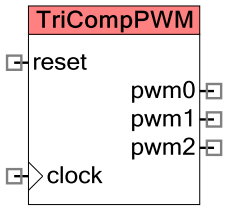
Single Counter PWM with Three Compare Outputs

TriCompPWM v1.00

Features



- Three compare outputs
- Intended for RGB LED Control
- Efficient resource usage

General Description

The Component implements two independent 16-bit PWMs within 2 UDBs. Each PWM has its own period, compare and counter registers. See the Functional Description section for a Timing Diagram.

Input/Output Connections

Pin	Type	Function
resetHW	Input	reset
clock	Input	twice the operating frequency of the PWM
pwm0	Output	pwm corresponding to 1st compare value
pwm1	Output	pwm corresponding to 2nd compare value
pwm2	Output	pwm corresponding to 3rd compare value

Component Parameters

Parameter Name	Default Value	Description
Compare0_init	0	8-bit initial compare value for PWM0
Compare1_init	0	8-bit initial compare value for PWM1
Debug	False	Enables the state machine pins to assist in debugging
EnableReset	False	Enables an optional reset terminal

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. The following table lists and describes the interface to each function. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name “PWM_CMP3_1” to the first instance of this component in a given design. You can the rename the instance to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol. For readability, the instance name used in the following table is “PWM_CMP3.”

Function	Description
PWM_CMP3_Start()	Enable Component and set up initial configuration
PWM_CMP3_ReadPeriod()	Read period value for PWM1
PWM_CMP3_WritePeriod()	Write period value for PWM1
PWM_CMP3_ReadComparex()	Read compare value for PWMx (x = 0, 1, 2)
PWM_CMP3_WriteComparex()	Write compare value for PWMx (x = 0, 1, 2)
PWM_CMP3_WriteIntensityx()	Write intensity value for PWMx (x = 0, 1, 2)
PWM_CMP3_ReadCounter0()	Read current counter value
PWM_CMP3_WriteCounter0()	Write current counter value

void PWM_CMP3_Start(void)

Description: Configures/Initializes the PWM, sets up its initial period

Parameters: None

Return Value: None

Side Effects: None

void PWM_CMP3_WriteIntensityx(uint8 intensity)

Description: Meant for ease of use with LEDs – larger value implies more brightness (x = 0,1, or 2)

Parameters: uint8 intensity value

Return Value: None

Side Effects: None

All the rest of the ‘functions’ are implemented as macros.

uint16 PWM_CMP3_ReadPeriod (void)

Description: Reads the period value of counter

Parameters: None

Return Value: uint16 period value

Side Effects: None

void PWM_CMP3_WritePeriod (uint16 period)

Description: Writes the period value of counter

Parameters: uint16 period value

Return Value: None

Side Effects: None

uint16 PWM_CMP3_ReadComparex(void)

Description: Reads the compare value of pwm x (x = 0, 1, or 2)

Parameters: None

Return Value: uint16 compare value

Side Effects: None

void PWM_CMP3_WriteComparex(uint16 compare)

Description: Writes the compare value of pwm x (x = 0, 1, or 2)

Parameters: uint16 compare value to be written

Return Value: None

Side Effects: None

uint16 PWM_CMP3_ReadCounter0(void)

Description: Reads the counter value

Parameters: None

Return Value: uint16 counter value

Side Effects: None

void PWM_CMP3_WriteCounter0(uint16 value)

Description: Writes the counter value

Parameters: uint16 counter value to be written

Return Value: None

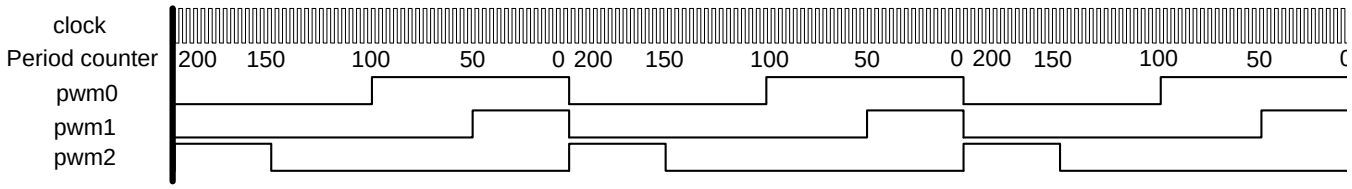
Side Effects: None

Functional Description

The PWM_CMP3 implements a single period-counter, triple compare 8-bit PWM in a single UDB.

Figure 1 shows the behavior of the PWM_CMP3 Component with a period of 300 and compare0, compare 1, compare2 as 150, 75 and 75, respectively.

Figure 1. Custom PWM_CMP3 Component Behavior



Notable Facts

- The PWM_CMP3 is implemented as a down-counter with a less-than compare function (i.e. only when the period counter is less than the compare value is the PWM output asserted)
- The PWM_CMP3 period counter updates at half the frequency of the ‘clock’ input (due to the implementation)

Resources

Resource Type				
Datapath Cells	**Macrocells**	**P Terms**	**Status Cells**	**Control Cells**
1	3	7	0	0

API Memory Usage

Minimal. TBD

Support

PSoC Creator Community Components are developed and supported by the Cypress Developer Community. Go to www.cypress.com/CommunityComponents to discuss this and other Community Components.

© Cypress Semiconductor Corporation, 2013. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control, or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

This work is licensed under a Creative Commons Attribution 3.0 Unported License.

http://creativecommons.org/licenses/by/3.0/deed.en_US

You are free to:

- Share — to copy, distribute and transmit the work
- Remix — to adapt the work
- Make commercial use of the work