Pulse-Width Modulator (PWM)

My name is Alan Hawse and this is PSoC 101. In this lesson I am going to show you the pulse width modulator, or PWM functionality of the TCPWM. This component generates repeating pulses on its output and lets you control both the period and the duty cycle. The duty cycle is the percentage of time that the output is high. A large duty cycle is a mostly-high signal and a small duty-cycle is mostly low signal. The PWM is really a special case of a Counter. It detects rising edges on the clock input and increments a counter. You can set count values for the period and the compare. The period defines when the counter rolls back to zero and, as a result, sets the output frequency of the PWM. The compare value sets the spot where the output switches from low to high which is how you control the duty-cycle.

In this project we are going to use a PWM to control the intensity of the red LED. You can do this by blinking the LED with different duty cycles. When you blink very fast your eye does not see the flashing but can detect the reduced amount of light being emitted. Varying the PWM duty-cycle makes the LED appear brighter or dimmer.

Make a copy of the Counter project and delete the Counter. Leave the input pin in the design because we'll be using it again in the next lesson. Find, and drop in the PWM, rename it and set the period to 10,000. Set the compare value to any number below that so you do not get an error. The actual value does not matter because you are going to set it in the firmware. Generate the application so that the editor knows about the API functions.

The C code is pretty simple. Just start the PWM and then change the compare value every second. Alternate the value between a very large and very small number so you can really see the difference in brightness when you program the board.

Go ahead and build this project for yourself. To test that you understand it fully, use the PWM to drive two LED pins. The line output has a complement, line underscore n, which is the inverse. If you connect that up to the green LED then it will be dim when the red one is bright, and vice versa. Then use firmware to gradually change the compare value throughout the whole period of the PWM. Use the CyDelay function with a value of 50 milliseconds and change the compare value by 1% each time so that it takes about 5 seconds to change from fully red to fully green.

As always you are welcome to email me at alan_hawse@cypress.com.