Timer

The third use of the TCPWM component is as a timer. In this example we will time how long the switch is held down and translate that into the brightness of the LED by reusing the PWM that we set up in the last lesson.

Copy the PWM project and add a TCPWM Timer component. Rename it and enable four inputs; start, stop, capture and reload. These inputs appear as extra terminals on the left-hand side of the symbol. In a moment you will connect the input pin from the switch to all four of these terminals. But before that modify the modes so that when you press the switch the Timer starts and when you release it the timer stops. The pin is active low and so choose falling-edge for the start and reload inputs and select rising edge for the capture and stop, also enable interrupt on capture event. This configuration will cause the Timer to reset its count register to zero and begin running when the switch is pressed. It will stop when the switch is released and simultaneously issue a capture event, resulting in an interrupt.

Back in the schematic, wire the switch to all four new inputs. You still need a clock for the Timer. To make the math easy, drop in a new clock component and change the frequency to 1kHz. Lastly, add an ISR component to the Interrupt terminal of the TCPWM.

There's a little bit of work to be done in the firmware. In main start the PWM and the Timer. Also you should register the interrupt handler using StartEx and make sure the global interrupts are enabled. Above main, create the interrupt handler. Write the code to read the value of the Timer. This is the number of milliseconds that the switch was pressed. Write that value into the PWM compare register and it will vary the brightness of the LED. If you are detail-obsessed, like me, you'll worry that the Timer may return a value greater than ten thousand if the switch is pressed for over 10 seconds. Since the PWM has a maximum period of ten thousand that would mean the compare is greater than the period, which would probably just leave the LED fully lit but is not a good coding practice. Add a check of the Timer value and make sure it cannot exceed ten thousand. Lastly clear the capture interrupt with the ClearInterrupt function and the argument for a compare event. This is the Timer component name plus 'I, 'N', 'T', 'R' underscore MASK underscore 'CC' underscore MATCH.

Program this into your board and play with it to verify that it works. If you press the switch for a long time the LED is very dim. If you then press it quickly it goes bright. As an extra test, modify the design so that instead of reading pulse times on the input, read the time between button presses.

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