Status Register

My name is Alan Hawse and this is PSoC 101. Now I have a UART to print out information I want to show you a couple of extra components that are really useful for digital design. These are the control and status registers. These registers are the interface between the schematic and your firmware.

I want to reuse the toggle flip-flop project we did a while back and, instead of driving LEDs, connect the flip-flops to a status register. In the C code I'll read the register and print the result to the UART.

Start by making a copy of the flip-flop project. Add a status register to the right of the flip-flops. Give it a sensible name and reduce it to just a 2-bit register. Note that the bits are in transparent mode, which means that the register constantly updates with the value on the input terminals, as opposed being sticky, which means they hold their value until read from firmware. In sticky mode the register must be clocked from the schematic. However, if all the bits are transparent this is not necessary and so I will just place a logic high on the clock terminal for completeness.

Now, add the UART, give it a short name, and use the DWR file to pick the pins that connect to the kitprog. Generate the application to build the API functions.

The C code is really easy. Start the UART first. In the main loop, read the register and, if its value has changed, write the value to the UART. If you write a carriage return character, but no line feed, then the UART will continually write the value of the status register to the same position on screen. When you press the switch it increments the value in the status register.

For your challenge, just as you did with the toggle flip-flops before, extend this design to be a 3-bit counter instead of just 2.

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