Two Simple Solutions for *Tricky Problems*

Here are two solutions that can help you create trouble-free designs.

by Peter Alfke, Xilinx Applications Engineering, Xilinx, Peter@xilinx.com

How to Eliminate False Triggering

As FPGA flip-flops become faster, they can respond to very short clock pulses and undesired glitches from clock reflections and ground bounce. Here's a simple solution.

Fast clock transitions often lead to reflections on long printed circuit clock lines, and such reflections can result in overshoot, undershoot, and ringing. This can lead to multiple crossings of the flip-flop input thresholds and thus false triggering. There are well-known methods to keep reflections under control, but they require an understanding of high-frequency, transmission-line effects; they may also be difficult and expensive to implement, and impossible to retrofit.

Systems with multiple un-correlated clocks are especially vulnerable, even when the clock edges are noise free. Ground bounce caused by one clock can occur during the transition of another clock, especially when this transition is slow, and can cause the apparently monotonic transition to be interpreted as multiple clock transitions inside the chip. In the past, FPGAs were slow enough to be forgiving, so that many high-frequency problems could be ignored. Now, double pulses separated by one or a few nanoseconds, as shown in Figure 1, can lead to double triggering and result in system failure.

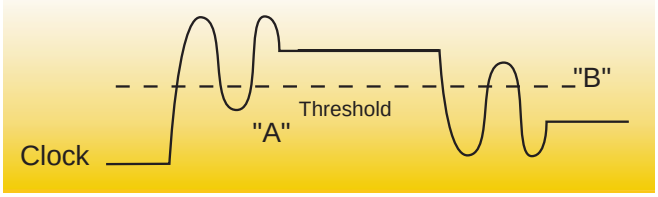

Figure 1- Reflections Causing Clock Noise

Here are two effective remedies against such problems. The description assumes rising-edge triggering, but can easily be modified for falling-edge triggering.

(A) Double clocking on the rising edge – this means that a reflection caused the internal clock signal to go Low-High-Low-High (L-H-L-H) instead of a simple Low-High (L-H) transition. The second L-H transition might clock the flip-flops again, if they are fast enough to respond to two clock edges that are only a few nanoseconds apart.

The simple solution is to give all affected flip-flops sufficient delay in front of the D-inputs, so they cannot change in the short time between the two rising clock edges, as shown in Figure 2. Use redundant LUTs or additional routing to slow the few flip-flops that are sensitive; typically they are the least-significant bits of a counter.

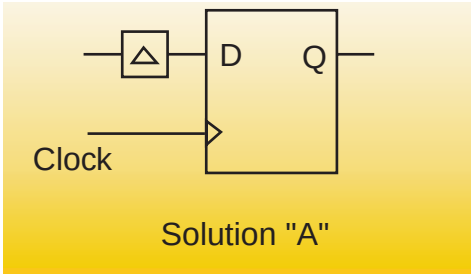

Figure 2 - Delaying the D-input

(B) Clocking on the wrong (falling) edge – this is always caused by a H-L-H-L sequence of clock transitions within a few nanoseconds. (No flip-flop can possibly be affected by the wrong edge. On the other hand, no flip-flop will ever ignore the edge it is supposed to trigger on.)

This false triggering which seems to occur on the wrong edge in the middle of the clock period cannot be cured by slowing down the flip-flop inputs. Instead, a slightly delayed and inverted version of the incoming clock signal must be used as Clock Enable to the flip-flops, as shown in Figure 3. CE will be active High before and during the legitimate rising clock edge, but will be inactive (Low) before and during the unintended clock glitch, caused by a reflection of the falling clock edge.

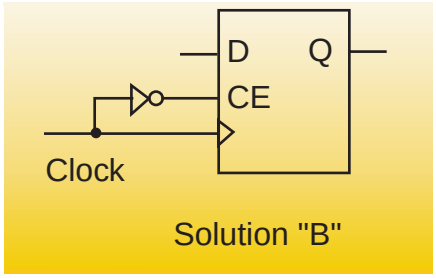

Figure 3 - Inverting the Clock

These two problems (on the rising and the falling edge), although caused by the same phenomenon, require two very different solutions. Luckily, both solutions are simple, and can be used together.

Clock reflections are best avoided by proper printed circuit board design, but accidents do happen, and it is nice to have a simple solution, especially one that is so easily applied in an FPGA without any harmful side-effects. These techniques can be used to retrofit older designs that fail when faster devices (which are more susceptible to noise) are used.

Proper Use of Mode-Pin Pull-Up Resistors

XC4000-series mode pins have an internal pull-up resistor that guarantees a logic High level on an unconnected mode pin during power-up. After configuration, the default bitstream turns these resistors off. Subsequent re-configurations can, therefore, fail.

For all modes except Master Serial (where all three mode pins are being pulled Low), we recommend either external pull-up resistors to guarantee a High level upon reconfiguration, or an explicit change of the configuration bitstream.

Here's how to change the configuration bitstream:

- From command-line: use the following options upon invoking bitgen:
  bitgen -g M0Pin:Pullup -g M1Pin:Pullup -g M2Pin:Pullup <design_name>.ncd
- From the Design Manager GUI: under the Configuration tab of the Edit Configuration Options Window, select the PullUp radio buttons for the M0, M1, and M2 Configuration pins.