Creating Efficient Multi-Tap Shift Registers

The Virtex LUTs can be configured as shift registers whose depth is determined by the four inputs to the LUT.

by Paul Gigliotti, Field Applications Engineer, Xilinx, paul.gigliotti@xilinx.com

Using the method described here, creating a 16-bit shift register requires only 0.25 CLBs, while creating a 16-bit shift register using flip-flops would use four CLBs. The depth of the shift register is dynamically configurable by simply changing the inputs to the LUT.

This mechanism can also be used to "snoop" into the shift register. Between shift cycles, the LUT inputs act as addresses into the shift register, allowing various points within the shift register to be examined. This is useful for various DSP filtering applications, as well as pattern/waveform detection.

Virtex LUT-based Shift Registers

A 16-deep by eight-bit wide shift register requires eight LUTs, which is just two Virtext CLBs. The LUT's address lines are tied to "F" (hex), and the control signals are wired in parallel. The shift register's D-Q pair (see **Figure 1**) are tied to a specific bit of the input/output bus. For depths greater than 16, the shift register LUTs can be cascaded, with the address lines for the low order banks tied to F (hex), and the final bank tied to modulo16 (depth).

Virtex LUT as a Shift Register

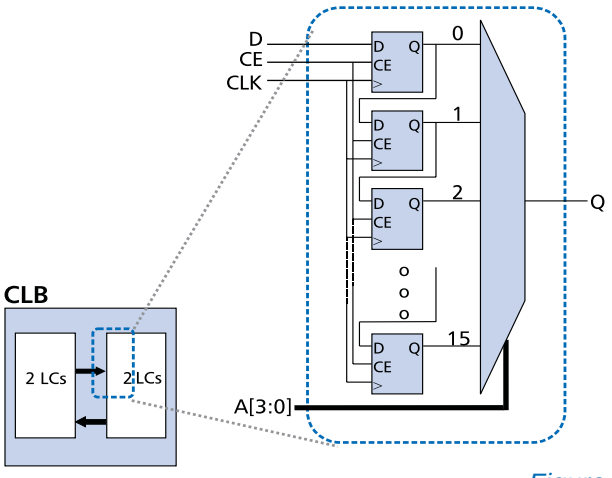

Figure 1

Snooping Into a Shift Register

As can be seen in **Figure 1**, the output of the MUX is an asynchronous read of the LUT. Snooping is accomplished by doing reads of the LUT between shift cycles. The waveform in **Figure 2** assumes that a 16-deep shift register is needed, but the design also requires that the data at Tap N be available. The design is run at twice the frequency, to allow the snooping to occur, as follows:

Snooping

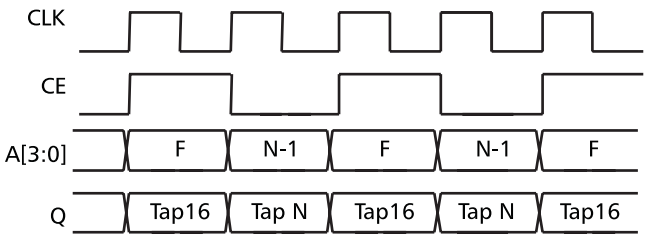

Figure 2

Dynamically Resizable Multi-Tap Shift Register

The design in **Figure 3** can be extended by cascading the multiple shift register banks to add additional depth. When the clock enable is high, a shift cycle is occurring, and the data is shifted from one bank to the next. To extend the design further, more snoop cycles can be added between shift cycles. For example, running the clock at 4X allows for one shift cycle and three snoop cycles. Note, we can only perform three snoops per shift register bank.

Tap locations can be "moved" into other shift register banks, by controlling the depth on each bank. For example, if I need to snoop on taps 3, 7, 9, and 11, the first shift register bank must be 10 deep, at most, so that Tap 11 resides in the

Dynamically Resizable Multi-Tap Shift Register

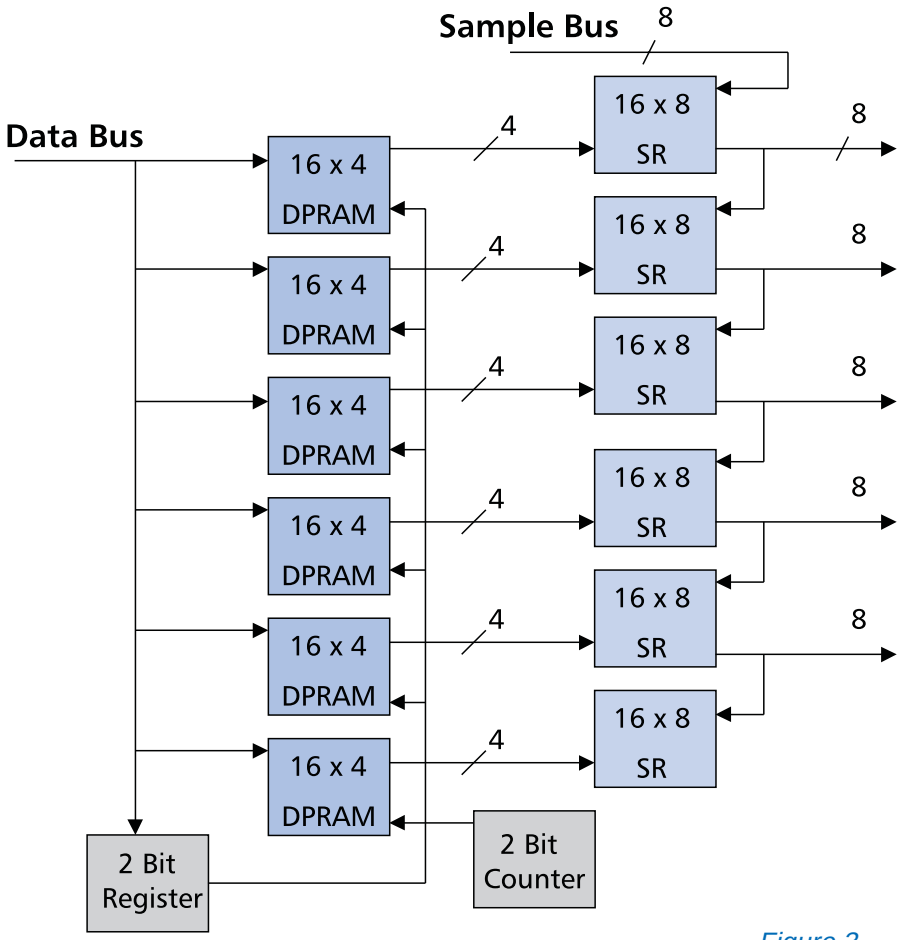

Figure 3

second bank of RAM. Controlling the shift and snoop points is most efficiently accomplished by using ROM blocks to store the look-up values. The block diagram in **Figure 3** has a maximum depth of 96 taps, and a maximum of 18 tap points.

Finally, the whole structure can be made "programmable" by storing the shift/snoop points in distributed dual port RAM, with the read port controlling the shift registers, and the write port mapped into a processor's memory space.

Conclusion

The above design requires 72 LUTs and four flip-flops, for a total of 19 CLBS. A more traditional approach, using flip-flop based shift registers, would require 196 flip-flops or 49 CLBs, just for the shift register portion of the design alone. Also, note that only four locations in the dual port RAMs are required to control the one shift and three snoops. By placing the two MSB address bits under processor control, it is possible to cache addition shift/snoop configurations into the dual port RAMs, and "hot swap" in the new settings.