techXclusives

The SRL16E: Advanced Level

How using this exciting mode can lead to "cost saving of an order of magnitude."

(Part 3 of a 3-part series)

By Ken Chapman
Staff Engineer, Core Applications - Xilinx UK

ADVANCED LEVEL:

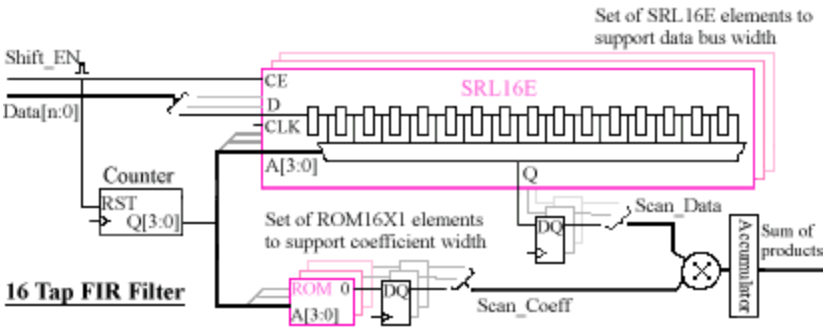
At this advanced level, we will move away from the concept that the SRL16E is used to implement a fixed delay. This means that the select lines to the multiplexer are now dynamically addressed inputs.

Separate Shift Register and Multiplexer

The most important fact to realise about this mode of operation is that the shift register and the multiplexer are totally independent. The shift register takes data from the 'D' input under the control of the clock (CLK) and clock enable (CE) signals. The output at 'Q' depends only on the A[3:0] inputs.

Although it is tempting to believe that an pin called 'Q' is associated with a synchronous clocked output, it must be understood that the multiplexer operation is completely combinatorial and has nothing to do with the clock. The use of the label 'Q' is only an indication of the fact that the multiplexer is selecting one of the 16 shift register flip-flop outputs.

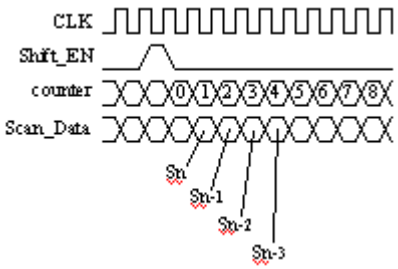
For a synchronous output, the associated flip-flop within the "slice" can be used. Such a flip-flop must use the same clock as the SRL16E, but this is of no issue in a good synchronous design utilising a single clock.



In this application the SRL16E elements form the taps of the FIR filter. As expected a new sample causes all previous samples to move along the shift register and for the oldest sample to be discarded. To calculate the filter output a single multiply and accumulate unit is used over 16 clock cycles.

With the data held static in the shift register (CE is low), a counter addresses the multiplexer within the SRL16E to select each tap in turn and

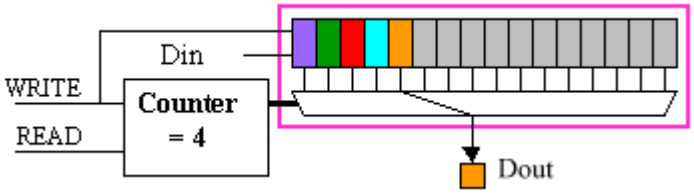
apply the contents to the multiplier. The same address is used to select the appropriate coefficient from a ROM. In the complete implementation the accumulator would be reset at the beginning of each calculation.



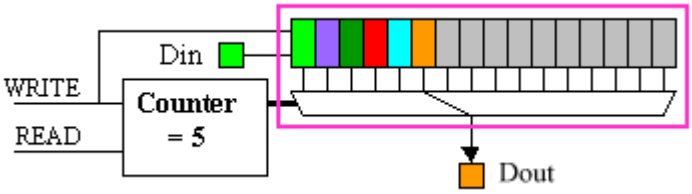
FIFO

A FIFO is normally based on a dual port memory (for simultaneous write and read operations), a pair of address counters (for write and read pointers) and control logic to detect full/empty conditions. The SRL16E can also be used to form a FIFO as illustrated below with the advantage of greatly simplified counters and control logic and twice the density over dual port RAM.

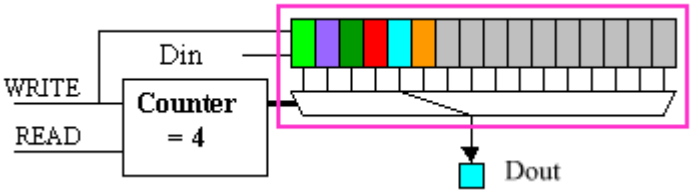
1) At the point at which this example begins, 5 data words have already been written to the FIFO. The counter is at value 4 to select the 5th tapping point and hence the oldest data is available at the output.



2) Data WRITE. The writing of new data has caused all data in the shift register to advance by one position. However, the counter has been incremented and hence the multiplexer is still selecting the oldest data to be presented to the output.

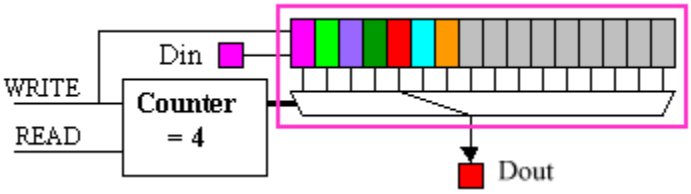


3) Data READ. The reading of data does not have any effect on the contents of the shift register. However, the counter has been decremented such that the next oldest data has been selected by the multiplexer to be presented at the output.



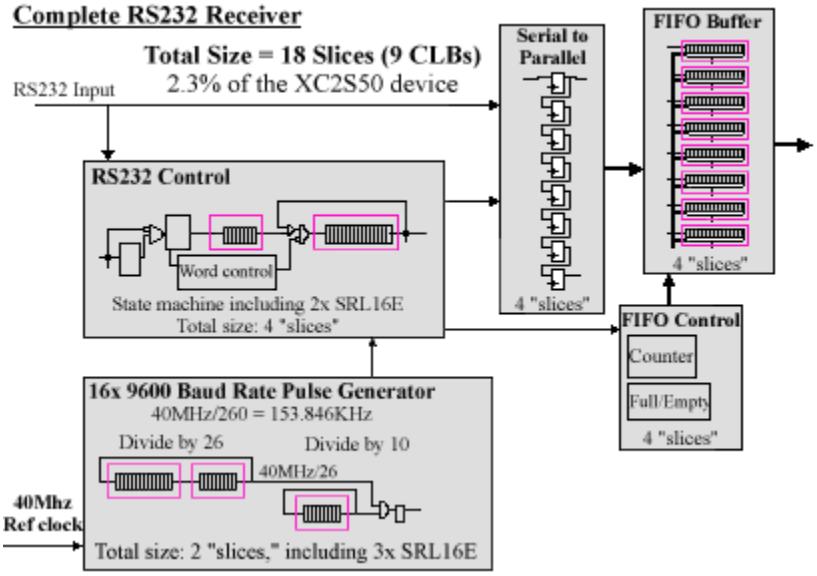
4) Simultaneous data WRITE and data READ. In this case the shift register has advanced by one position but the counter has remained the same

value such that the next oldest data has moved into the selected position. The previously read data, and now unwanted data, is still contained in the shift register but will eventually be lost completely.



COMBINED APPLICATION EXAMPLE:

When combining several of the previous examples, it is possible to realise the potential of the SRL16E in designs and the variety of modes in which this primitive can be utilised. This combined example illustrates a complete RS232 receiver with baud rate generator and FIFO data buffer.



It is interesting to note that these 18 "slices" are used to provide 207 flip-flops which alone yield a density of 138 gates/CLB (based on just 6 gates per flip-flop). 13 function generators are used in SRL16E mode, and 21 are used in standard logic mode.

Share your comments, questions and ideas with Ken Chapman and other interested designers at the "SRL16E" FORUM.