Question: "What is a best way to measure pulse width of DisplayPort transmitter with PS3600?"
Alexander Roskin

Question:

I need to measure a pulse width of a pattern generated by my DisplayPort transmitter as part of transmitter test. What is the best way to perform such a measurement on PS3600?

Answer:

The overall approach depends on required precision/resolution of the test. If the resolution should be higher than ~1/20 of the UI width, then following test methodology can be suggested:

1. Perform first search for a rising edge using linear-binary (linbin) timing search with high resolution and search range set to 2*UI
2. Perform search for a falling edge using binary timing search with high resolution and modified boundary conditions taking into consideration the results of the first search

Although this methodology would provide best resolution, it shows some drawbacks in terms of throughput. Assuming measurement is being performed on multiple channels and/or multisite set-up, linbin timing search can not be performed in parallel due to nature of binary search, so that the measurements are done in sequential way. Apart from that, recalculation of values for the search and download of the values into the hardware for every step is time consuming process.
Conclusion: use this approach in DV, when precision matters more than throughput.

If, however, the resolution of 1/20 of the UI or less is sufficient (which is usually the case for HVM), a more efficient way to perform the pulse width search is to use pre-defined timing spec sets:

1. Create pre-defined spec sets with different offsets according to required resolution. E.g. first compare strobe at 0ps offset, second one at 50ps offset, third at 100ps offset and so on (resolution 50ps)
2. Create either a
   - single pattern consisting of a burst with a number of sub patterns equal to amount of spec sets and with a CTIM command at the beginning of every sub pattern, or
   - just a normal pattern, which loops inside a Test Method with change timing executed before every single iteration
3. Execute the test, which will be linear and fully parallel on all channels / all sites.
4. Count the number of passes and fails. Based on known resolution and known UI width, calculate resulting pulse width based on the amount of passes.

Depending on the required resolution, this test can be faster by a factor of 5-10 than a binary timing search test.
Conclusion: using this methodology provides the most throughput-efficient way to perform pulse width search for HVM