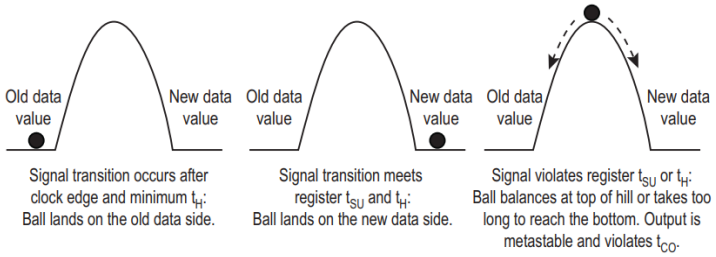
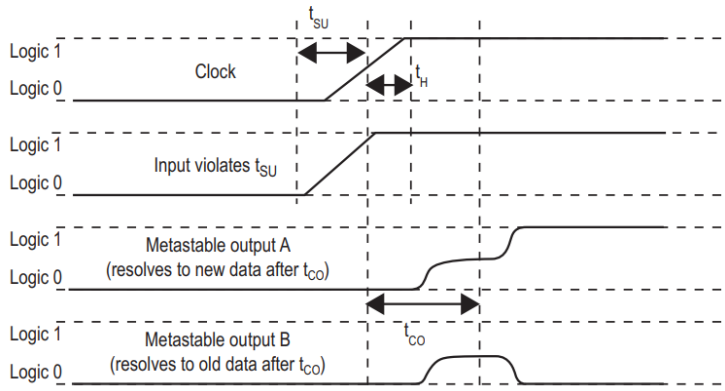
Understanding Metastability in FPGAs

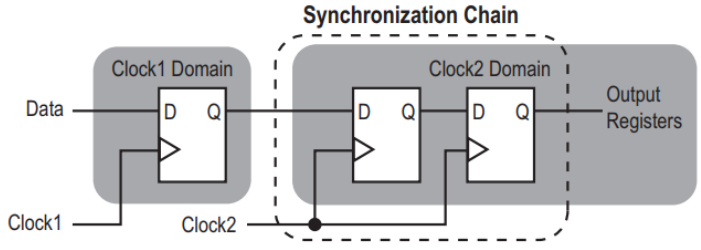
In digital devices like FPGAs, timing is an important subject. Data that arrives to FF's input must be stable some time before (setup) the capture edge and must be stable some time after (hold) the capture edge. If this does not happen than the FF can go into metastable situation that is not either valid logic one or logic zero for a time, a not valid intermediate value. That's a not good thing. This could happen if a design is not meet the timing constraint or data transfer between two different clock domains or even with no clock domain to clock domain (asynchronous). In asynchronous system, there is no way to know or guarantee that the signal goes to FF is meet the timing (setup/hold). Metastability can be illustrated as a ball dropped on a hill, see the below figure.



Example of metastable output signals are shown below figure.



If the FF's output does not resolve to a valid state in the defined clock period (before the next FF captures the data) than system can be failed, and the metastable situation can propagate throughout the design. To avoid that, synchronization registers must be used for one bit data transfer between different clock domains or asynchronous domains to clock domain. The synchronization registers must be placed in the used clock domain. It is rare for the last FF in the chain to be metastable because the registers give additional time resolve to a valid state before it is sampled in the design. An example schematic with 2 synchronization register is shown below.



To transfer more bits in different clock domains, the above example is not enough. So, the dual FIFO (First In First Out) or handshaking logic must be used.

MTBF (Mean Time Between Failures) is an estimation of average time between failures, and it can also be used for metastable errors. For the critical devices like avionics, medical etc. MTBF should be higher than normal/uncritical devices like user entertainments like video gaming etc. MTBBF formula for the synchronizers is like below;

$$MTBF = \frac{e^{t_{MET}/C_2}}{C_1 \cdot f_{CLK} \cdot f_{DATA}}$$

C_1 and C_2 constants that come from device process and operating conditions. f_{clk} and f_{data} are define the frequency of clock and data respectively. If the destination clock frequency or input data change rate is faster, than the MTBF will be worse. t_{met} is the metastability resolving time (available timing slack beyond the FF's t_{co}). For the synchronization registers, t_{met} is the sum of the resolving time for each register. See the below for failure rate of design;

$$failure\_rate_{design} = \frac{1}{MTBF_{design}} = \sum_{i=1}^{number\ of\ chains} \frac{1}{MTBF_i}$$

When the process or transistor technology improves like moving from 180 nm to 90 nm in FPGA, the MTBF is getting better, but this is not true for voltage. When the supply voltage is decreased, the metastable FF takes longer time to recover from this state. That's why it is a big consideration for FPGA vendors unless they take action for metastability. Also, the biggest effect in the MTBF formula is exponential part and both designers and FPGA vendors can focus on it.

If a design has high MTBF for all synchronizers except one, this design will have low MTBF because of that one synchronizer. That's why, think the design as a whole. Also adding an extra FF to synchronization chain improves the t_{met} so the MTBF get better but also it adds one clock cycle latency. The designer must evaluate the correct FF number, pros and cons.

To sum up, timing is critical subject in the digital design and asynchronous signal must be evaluated correctly in the design to avoid failures. Also do not remember the MTBF.