Position Statement: Tool Interoperability Issues

Teresa L. McLaurin
Motorola
7700 West Parmer Lane
MD: TX32/PL35
Austin, TX 78729-8084

The steps of our basic flow for going from an idea of functionality to silicon tape-out are pretty well established. The design is written in behavioral code or modeled. The model is synthesized, verified, timed, placed and routed, timed again and taped out. This is highly simplified, but the major steps are noted. Somewhere in that flow, the design may be scan inserted or some other type of DFT installed and the test logic verified.

Each EDA vendor tool or in-house tool works on a specific section of the process and may be blind to the needs of other sections of the process, especially DFT - the newest comer to the scene. This is the area on which this statement will focus.

If each section of a process is evaluated by itself, it usually works very well. When the full process is put together, there are several issues, that if unknown, can be very painful in time and effort to go back and fix. If they are all known about ahead of time, the pain lessens, but, there are always some surprises. The next few paragraphs describe just a few of the “gotchas” when trying to make all of the tools work together within one process.

One such example occurs when timing analysis is run on the scan inserted gate level model. The fastest path in this gate level model happens to be the Q output of one Mux-DFF to the SDI input of the next Mux-DFF. The timing analysis tool adds buffers or inverters to slow the path down...we wouldn’t want hold time violations. At some point, this netlist is given to place and route. The tool will disconnect the wires to the SDI inputs, place the gates and rewire the scan chains to their nearest neighbor. But wait! There are buffers in the scan chain and the tool thinks this is a separate scan chain. This causes problems. So, the buffers are removed...somehow. The scan optimization is rerun. The scan flip-flops are connected to their nearest neighbor. This probably causes hold violations, because the fastest path on the design was just made shorter. More buffers are added.

Since we are in the era of SOCs, let’s consider wrappers. A wrapper is a boundary around a core that either shares functional flip-flops of registered inputs and outputs or has a dedicated test cell at the input or output of a core for control and observe purposes. The wrapper may be on the same clock as the core and many tools perform scan optimization utilizing the clock as the criteria of how the scan chains can be mixed. These tools connect the scan cells to their nearest neighbor. The wrapper chain cannot be mixed with any other scan chain. So, the wrapper chain is left connected during placement and ignored during scan optimization. This may disturb the timing of the functional placement if the wrapper cells are shared. Timing issues of the wrapper scan chain itself may occur.

Lock-up latches may be a necessity when there are several clock domains that only contain a few flip-flops each. The lock-up latches are carefully put between flip-flops of different clock domains while concatenating them into one scan chain. It’s time for scan optimization during place and route. The place and route tool does not understand the purpose of lock-up latches and will mix all of the scan cells on that particular scan chain in a manner that is undesirable.

What if scan observe registers are needed? The output of the observe register is connected only to a wire until scan insertion occurs. There may be many of these. During synthesis these cells are usually optimized out of the synthesized netlist as they appear to serve no purpose until they are scan inserted. Precautions must be taken to prevent this from occurring. This may cause a hiccup in the flow that mandates synthesis be redone.

Memory BIST and other test related HDL generators (e.g. JTAG) generate HDL, yet the HDL generated does not always pass scan test rules.

There are many tool inter-operability issues that must be addressed, these are just a few. Each tool vendor must have some understanding of the entire process in order to build a process capable tool.