

Trip points for IC-timing analysis

GET THE MOST OUT OF YOUR TIMING METHODOLOGY.

If you are a budding timing-analysis engineer or even a veteran, understanding trip points, which all major timing-analysis tools incorporate, is essential. Engineers use trip points in timing-analysis tools to calculate delay and transition values on the various nodes of a design. Timing-analysis engineers must become familiar with the proper use of trip points, as there are many nuances to using them. If engineers overlook them, these nuances can cause problems late in the SOC (system-on-chip)-design cycle, when they need to address timing. A quick tutorial and a bit of timing tool-script work can make their job easier.

DEFINING AND DETERMINING TRIP POINTS

A trip point is the percentage of logic-high levels engineers use as a reference to measure slew and delay values. **Figure 1a** illustrates the slew trip points, and **Figure 1b** illustrates delay trip points.

Engineers use trip points to characterize the delay and transition values of the pins of a standard-cell or hard-block-IP (intellectual property), when checking the timing of an SOC. Typically, engineers can find trip-point values in the timing models, most commonly in Liberty (.lib format). Timing-anal-

ysis tools in turn use these values to calculate delays and slews.

Engineers set trip points for a particular technology node while characterizing the libraries for standard cells or hard-IP blocks. The aim of setting a trip point is to ensure that the measured delays and slews are closer to actual Spice waveforms. As **Figure 2** shows, slew values are more accurate when a trip point lies in the linear region of 20 to 80% of the switching waveform than when it lies in the nonlinear region of 10 to 90%. Typically, cell delays that timing tools calculate are closer to Spice results when transition trip points are in the linear region.

Also, the threshold-voltage characteristic for transistors plays an important role in determining a trip point, because the output waveform linearizes after the input voltage crosses the threshold-voltage value for the transistor (**Reference 1**).

Delay thresholds are fixed in the linear region of the input and output waveforms. It does not matter whether the delay trip points are in the 20 to 80% region or at 50%, as long as they lie in the linear part of the waveform. **Listing 1**, which is available at www.edn.com/ms4272, is a snapshot of a typical timing model (in .lib) to indicate the trip points under use.

TIMING TOOLS CALCULATE DELAYS

Timing tools calculate delays using trip points in many ways. When both the driver and the load have the same delay

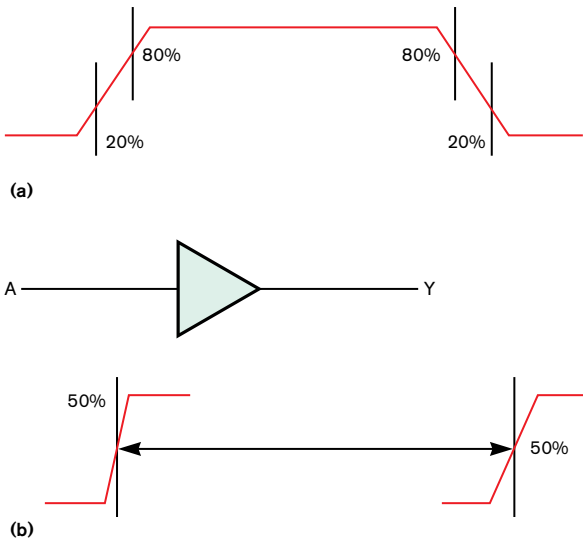


Figure 1 A trip point is the percentage of logic-high levels engineers use as a reference to measure slew (a) and delay (b) values.

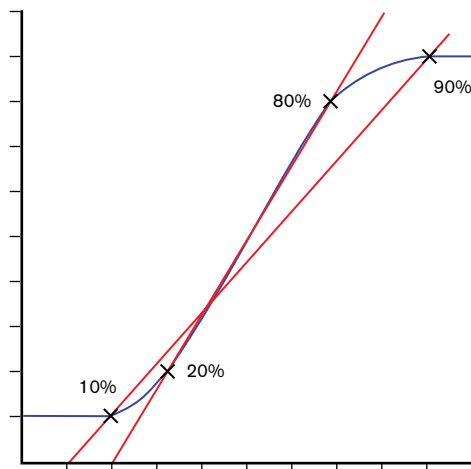


Figure 2 Slew values are more accurate when a trip point lies in the linear region of 20 to 80% of the switching waveform than when it lies in the nonlinear region of 10 to 90%.

thresholds, timing tools calculate the time difference between the driver reaching 50% of its logic-high value and the load reaching 50% of its logic-high value after taking into account slew degradation, which the net causes (Figure 3). Similar explanations hold true for falling signals and delays from the input to the output of a cell. Timing tools then calculate slew values from a number of variables outlined in .lib files. When an interface has different trip points, however, things get a bit more interesting.

Figure 4a depicts a situation in which a driver's delay trip point is 20% and the load-cell trip point is 50%. In such a case, a signal could quickly reach the driver at its delay-trip-point value and arrive slowly at the trip point for the load signal. Hence, the net delay for such an interface can be more than in a situation when the driver is also 50%. Timing tools calculate this extra delay using linear or nonlinear scaling (see sidebar "Types of scaling").

Conversely, when a driver's trip point is 50% and its load cell trip point is 20%, the loading signal reaches its delay-trip-point value much sooner than it reaches the trip point for the driver signal (Figure 4b). This situation can result in a "negative delay" due to timing tools' linear and nonlinear scaling features. Although you cannot move backward in a time domain, the timing tool must take this delay into account so that the overall path delay from the signal's starting point—in this case, the input pin of the driver cell—to the endpoint—in this case, the output pin of the load cell—is close to the real-world delay, which you can determine using Spice.

Users must also take into account several miscellaneous is-

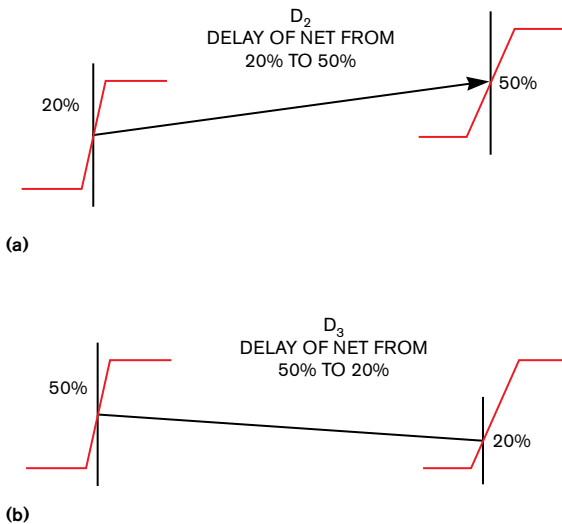


Figure 4 When a driver's delay trip point is 20% and the load-cell trip point is 50%, a signal could quickly reach the driver at its delay-trip-point value and slowly arrive at the trip point for the load signal (a). When a driver's trip point is 50% and its load-cell trip point is 20%, the loading signal more quickly reaches its delay trip-point value than it reaches the trip point for the driver signal (b).

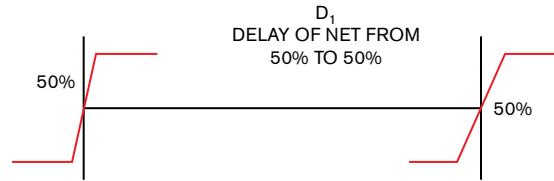


Figure 3 When both the driver and the load have the same delay thresholds and both have equal timing, timing tools calculate the time difference between the driver's reaching 50% of its logic-high value and the load's reaching 50% of its logic-high value.

issues associated with trip points. First, if timing tools encounter negative delays, it is likely that those same negative delays will make their way into the SDF (standard-delay-format) file engineers use for gate-level simulations. Gate-level simulators can't handle negative delays. They either flag an error message or annotate a zero delay for such cases. To work around the problem, designers can write a script to adjust the load- or driver-cell delay in accordance with the calculated negative net delay. Listing 2, which is available at www.edn.com/ms4272, contains this script.

Another issue involves calculating the delay between the port and the bidirectional pad pin of I/O cell. Because the net running from the port to the pad pin of the I/O cell is typically outside the chip because the net is a bond wire, a timing tool must still take the delay of the net into account. This approach is complicated, because there is no accurate physical

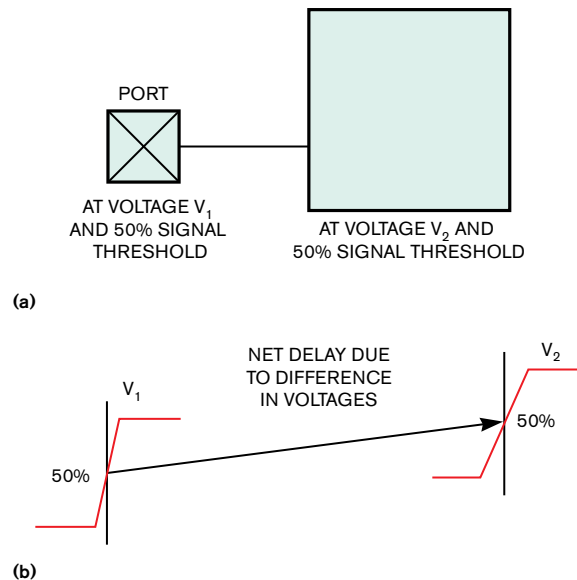


Figure 5 Without a timing model available for the ports, users must manually define a default trip point and voltage level to help the tool calculate the delays.

information for the net. Without a timing model available for the ports, users must manually define a default trip point and voltage level to help the tool calculate the delays (Figure 5). To manually define a default trip point and voltage level, users should first define the operation conditions for ports that are the same as that of I/O cells and then write a script to annotate zero delay for such nets.

It is common to find that libraries are missing trip points. A timing model that does not contain trip-point-threshold or voltage-level values, may have incorrect delays for interfaces to and from them. Because timing tools need at least default values of trip points and voltage levels for analyzing such paths, engineers should consult with their library teams to define the missing trip points.

TIMING-TOOL QUIRKS

A timing tool may fail to perform scaling, resulting in silicon failures because the calculated delays do not match the Spice results. This problem has no easy solution, but it is good practice is to perform Spice analyses for all interfaces that have different trip points. Another method is to use the same trip points for all the modules (hard blocks) in an SOC.

A Spice analysis for multithreshold paths is always a good idea to garner more confidence while cleaning up timing issues. Although thresholds do not exist in the Spice world, the timing models mention them to facilitate the use of timing-analysis tools.

Timing tools typically include a feature that al-

TYPES OF SCALING

Most of the industry-standard timing tools use either linear or nonlinear scaling as their modus operandi. In linear scaling, the tool assumes a linear ramp at both thresholds. This method uses the concept of similar triangles to scale the delay from driver to load cell. In nonlinear scaling, tools use current-source models to define the ramps. Calculating the delays requires complex mathematical equations.

lows users to dump timing paths in the form of a Spice netlist into a Spice tool. To complete this task, users must also provide the Spice tool with a stimulus file containing the input vectors. Spice-simulation tools can read the dumped Spice netlist and the Spice netlists of the standard cells and hard blocks and then source the stimulus file to simulate the critical paths. Engineers need to analyze the generated waveforms to determine whether the path meets timing. One thing to keep in mind while measuring the delay and transition values of such paths in Spice is to use the same trip points that the timing models mention. This approach ensures accuracy.

To properly perform IC-timing analysis, engineers must become familiar with trip points, their nuances, and how timing tools handle them. Failing to properly understand trip points can mean big headaches for a design team and even cause silicon failures. By better studying what they are and how timing tools use them and with a bit of creative script writing, engineers can use trip points to get the most out of their timing methodologies. **EDN**

MORE AT EDN.COM

[Go to www.edn.com/ms4272](http://www.edn.com/ms4272) and click on Feedback Loop to post a comment on this article.

AUTHOR'S BIOGRAPHIES



Sunit Bansal is a design engineer at Freescale Semiconductors focusing on static-timing analysis and timing closure of complete SOCs. He holds a bachelor's degree in electronics and communication from Delhi College of Engineering (Delhi, India).



Ateet Mishra is a design engineer at Freescale Semiconductors focusing on static-timing analysis and timing closure of complete SOCs. He holds a bachelor's degree in electronics and electrical engineering from Punjab Engineering College (Chandigarh, India).



Naveen Sampath Krishna is a design engineer at Freescale Semiconductors, where he is responsible for static-timing analysis and timing closure of SOCs. He holds a bachelor's degree in electrical engineering from Mahad College of Engineering, University of Mysore (Mysore, India).