# The Last Byte

# Big Data, Big Faults

**Scott Davidson**

■ **One of the keynote** addresses at the just concluded International Test Conference noted that even very high fault coverage (over 99%) in billion transistor circuits will still leave hundreds of thousands of untested faults. The speaker was concerned about the quality issues due to this, and also security issues. If all these untested faults represent a block of untested logic, how can we be sure that the untested logic wasn't inserted by some third party?

This problem resonated with me since I've had the opportunity to look at this kind of fault list. My untested fault list did not have any security problems that I could see. There were some chunks of untested logic between scan wrappers isolating memory from the rest of the logic, but these were unlikely to cause issues. Most of the rest of the untested faults list consisted of faults that would get tested in real life, but which were marked as untested by the test generation software for a variety of good reasons. Some faults in multiple-use logic blocks showed up many times with slightly different names. Deal with one and you deal with many of these.

None of this proves that there aren't insidious untestable faults hidden in the design, faults that could cause failures at board, system, and field levels. How do we find the few bad faults in the sea of untested faults that do get tested in normal test programs?

Having just traveled by air, I saw that this problem like that of detecting suspicious passengers. We can use data analysis methods to screen out faults/passengers unlikely to pose problems. For instance, faults associated with scan enable lines or clock lines would likely be detected during a test but will show up as untestable in a fault report. Similarly, a person with a history of travel between two airports is probably not worthy of further scrutiny.

Frequent travelers willing to pay can precheck themselves and skip most of the screening. Perhaps we should let netlists precheck some of their faults to eliminate them from undetected fault lists. This would be easier if we use data analysis methods to define similar faults so that marking one fault in a class as not a problem will mark them all that way.

Some of us are more likely to be faulty, or have ailments, than others. Genetic testing can find this. Several companies today offer ways of analyzing cell libraries to find realistic defects in them, some of which might not be covered by stuck-at or delay fault testing. However, not all these faults are equally probable, depending on the layout of cells in a library. If we can state that a person has a probability of getting sick depending on genetics, maybe we can assign probabilities to the occurrence of these faults. Test all of them if possible, but when analyzing untested fault lists unlikely faults might go to the bottom.

Big data has helped semiconductor manufacturing to control processes that produce millions of components, but since our parts now have millions of internal components, we should be able to leverage methods used for big data in the large for big data in the nanometer small. ■

*Digital Object Identifier 10.1109/MDAT.2019.2957362*
*Date of current version: 6 February 2020.*

■ Direct questions and comments about this department to Scott Davidson; davidson.scott687@gmail.com.