

# TAPS All Over My Chips - position statement

Teresa L. McLaurin

ARM Inc  
1250 S. Capital of TX Hwy, Bldg 3, Ste 560  
Austin, TX 78746

Long ago and far away (at least in semiconductor technology time) there came a standard called 1149.1 that was created to help solve board interconnect testing issues. People saw it and used it and said, "This is good". As time progressed, people said, "I can use this 1149.1 for other things that need standardization". 1149.1 was upgraded and utilized for things such as debug and memory BIST. People saw this and said, "This is kind of good". As time went on, more functionality was put onto a chip. Former chips became modules (or cores) within the chips. Many of these former chips contained 1149.1 TAP controllers. As the number of cores grew inside the chip, there came many TAP controllers on one chip. People said, "This is not so good" There was much confusion about how to control these multiple TAPs through the top level JTAG pins. If that was not enough to contend with, those chips became cores and now there were multiple TAPs in a core. Not only were there multiple TAPs, they were hierarchically embedded. Of course there were solutions to come out. Some of these solutions were not 1149.1 compliant, but some were. All were invariably patented so that no one else could use them without paying a price.

Many core providers have no boundary scan on their cores, though there may be legacy TAP controllers embedded for debug or other such features. Whoever uses those features must contend with the legacy TAP controllers and come up with extra logic to control them while in the mode that utilizes the TAP controller. In this position statement, I will

discuss the testing required for production, such as ATPG, MBIST or LBIST.

Standard ATPG tools do not understand TAP controllers. In fact, there is no reason to add this type of logic onto a core for scan test. It adds extra logic to the core, which affects the test coverage, makes it more difficult to create the test patterns and makes it more difficult for a core user to integrate the core into a system-on-chip (SoC).

TAP controllers are also used to direct memory BIST or logic BIST controllers. This is fine if the BIST is for a chip. However, if the BIST is for a core, this creates an extra level of complexity. It does not seem clever to add an extra level of complexity to a core when it does not have to be there. Instead, put "hooks" in the core to allow a TAP controller to be added external to the core, if desired. This gives the core user the choice to use a TAP or not to use a TAP. Choice is good.

The best practice for a core provider is to deliver a core that is easily integrated and testable in the many environments in which it will eventually reside. This is best done by allowing the core user as many options as possible. Adding a TAP controller to a core is taking one option away from the core user. There is no practical reason to add a TAP controller to test a core and many practical reasons to leave it off.