Breker provides the industry’s first commercially available suite of tools delivering on the promise of portable stimulus, providing automatically generated C/C++ test cases for software-driven verification, transactions for UVM testbenches, or a combination of the two. These test cases are portable from IP to full-chip level, and from simulation to silicon. Verification engineers, embedded programmers, and bring-up teams can easily generate, reuse, and debug powerful test cases, achieving automatic coverage closure.

Breker’s Portable Stimulus Solution

Breker Products

TrekSoC: Software-Driven Verification

A system-on-chip (SoC) can be verified effectively and efficiently only by fully automated test cases running on its embedded processors. Breker TrekSoC™ automatically generates complex multi-threaded, multi-processor, self-verifying C/C++ test cases for the SoC. The methodology is modular, extensible, and scalable based on a hierarchy of graph-based scenario models capturing design and verification intent. As additional layers of models are added, more complex test cases are generated.

The generated test cases:
- Run efficiently in simulation or acceleration on "bare metal" without the overhead of an operating system
- Include a built-in library of extensible system services such as memory management and interrupt handling
- Minimize embedded memory usage

Many test cases require synchronization between processors and transactions in the testbench. TrekSoC provides this synchronization automatically via the TrekBox™ module, which connects directly to standard UVM verification components on the SoC’s I/O ports. TrekBox also provides a novel runtime display showing how the C test cases execute on multiple threads across multiple processors. Progress through each realistic use case scenario is shown visually. This helps the user understand and document the generated test cases, and debug when test cases uncover design errors in the SoC.
TrekSoC-Si: Hardware and Silicon Support

Using the same scenario models as TrekSoC, TrekSoC-Si automatically generates bare metal test cases optimized to run efficiently on in-circuit emulation (ICE), FPGA-based prototypes, or actual silicon in the bring-up lab. The complex multi-threaded, multi-processor, self-verifying C/C++ test cases provide the same visualization and coverage closure capabilities as the TrekSoC test cases running in simulation. When a test case uncovers a design error in the SoC, TrekBox visually annotates the failing step in the test case map, the failing line in the source code, and all relevant error messages. Silicon validation is often a manual task; TrekSoC-Si automates the process with its ease of use and rich debugging capabilities.

Cache Coherency TrekApp: Pushbutton Verification

There is a clear trend in the semiconductor industry for the most complex designs to move to SoC architectures with multiple embedded processors, multi-level caches, and cache-coherent interconnects. Many of these chips, especially in server and networking applications, contain dozens or even hundreds of processors linked by a common cache-coherent multi-level on-chip bus, chip fabric, or a network-on-chip (NoC) interconnect. This type of design introduces new verification challenges. The Breker Cache Coherency TrekApp™ was developed specifically to address these challenges.

The Cache Coherency TrekApp automatically generates sophisticated test cases that stress every aspect of the SoC’s processors, caches, memories, and interconnects.

The generated test cases:
- Are highly complex, exercising single-processor and multi-processor cache state transitions
- Include many different varieties of memory reads and writes as well as cache snoops across the multi-level cache architecture
- Can reflect different page tables with various security and coherency rules
- Can exercise processor instructions for different sizes of data and bursts
- Have a high degree of interleave and synchronization among the threads on the multiple processors

The test cases can run on any platform, from simulation and acceleration to ICE, FPGA prototypes, and silicon. The result is far higher cache coherency coverage than could ever be achieved by hand-written tests or simple generators. This is truly a pushbutton solution; the user provides only basic information on the configuration of processors, memories, caches, and cache line widths. No knowledge of graphs or scenario models is required.