Accelerated Analysis of Simulation Dumps through Parallelization on Multicore Architectures

P. Bernardi, A. Calabrese, S. Littardi, S.Quer
Dipartimento di Automatica e Informatica, Politecnico di Torino, Torino, Italy

With the explosion of the size of the chips in terms of the number of gates and the advent of novel technology related to failure modes, a major concern for testing and reliability is constituted by simulation and fault simulation times. Unfortunately, while new fault models and stress metrics are objects of investigation, computations based on ATPG engines can be insufficient to reach the desired result accuracy. In this framework, many strategies often adopt a post-analysis of simulation dumps [1][2].

In principle, when ATPG cannot help too much, it is often useful to record a simulation trace and save into a file all values and timings for the set of selected signals (i.e., possibly all signals of the circuit). A commonly used file format is the Value Change Dump (VCD) [3]. This is composed of two sections as shown in figure 1. The first part locates all signals selected for the dump, during the simulation, into a hierarchical description. The second section reports all transitions affecting such signals.

In recent experiments, related to the effectiveness of burn-in techniques, we evaluated the coverage of stress metrics including the "straightforward" toggle activity plus some more complex "topology-related" metrics [4]. Our case study is an automotive device, including around 30 million gates; whose dump trace is generated by a typical simulation and it occupies up to hundreds of GBs. In this context, we developed a method able to mitigate the evaluation time of a dump by exploiting parallelization and some other optimizations, related to advanced features of modern programming languages and operating systems.

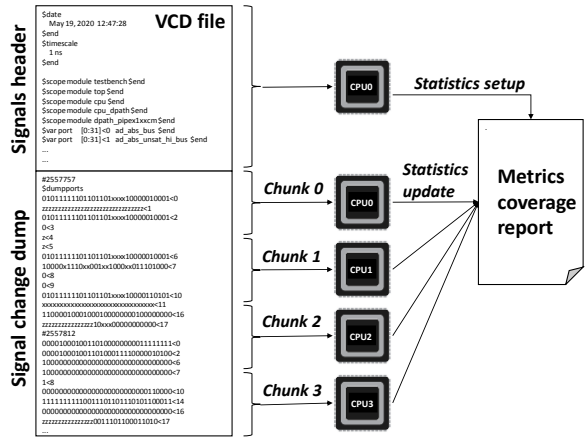

Fig. 1: simulation dump analysis

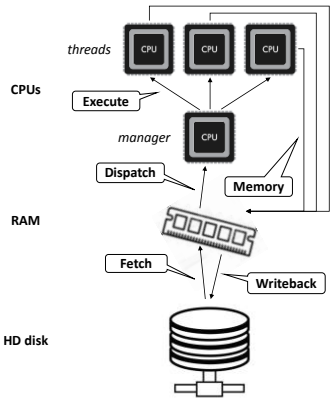

Fig 2: working principle

Given a VCD file, which initially resides in the mass storage of the system where it was stored by the simulator, we propose to divide it into several slices to be assigned to multiple computational cores. Figure 1 illustrates the principle of dividing the VCD signal change dump into many chunks. More in detail figure 2 shows the hardware resources involved in the VCD analysis flow. This flow includes 5 stages: We first transfer the file content from mass to central memory (stage 1); then a manager thread dispatches it to many working threads (stage 2); then, each working thread analyzes its chunk (stage 3); working threads progressively store their result in an efficient data structure (stage 4); threads finally write back their results to the mass memory (stage 5).

Since the size of the VCD file may be extremely large, not all chunks can be read and manipulated at the same time. Therefore, to be as effective as possible, the system has to be organized in a pipeline fashion where all stages ideally occupy the same quantum of time. Figure 3 visually describes the pipeline evolution of our analysis flow. Using more cores allows us to mitigate the time requirements of the execution stage; avoiding stalls is crucial to reach an accurate balance of the timing between stages.

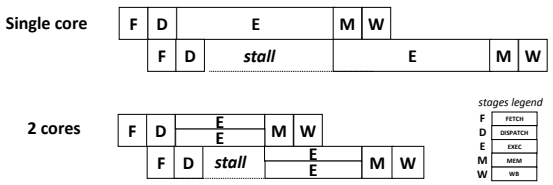

Fig. 3: Assessment of the parallelization effectiveness

Table I: Experimental results about full

Size	SINGLE TREAD	4 TREADS		8 TREADS		32 TREADS	
6 GB	3m28s	1m28s	2.36X	1m11s	2.93X	1m2s	3.35X
60 GB	46m44s	32m48s	1.65X	21m39s	2.16X	18m12s	2.57X
60 GB*	9m12s	5m48s	1.59X	4m32s	2.03X	2m4s	4.45X

**producing statistics only, instead of all gates results report*

We have evaluated the proposed solution on VCD files generated by the simulation of large industrial circuits. On these circuits, we analyzed the toggle activity created by scan patterns during burn-in. Table I shows how much time it is required to analyze two VCD files with different hardware and software configurations. The speed-up looks consistent and it increases with larger files. We may forecast a time gain almost linear with the number of cores adopted when all stalls will be properly eliminated.

REFERENCES

[1] K. Tsai, R. Guo and W. Cheng, "A Robust Automated Scan Pattern Mismatch Debugger," 2008 17th Asian Test Symposium, Sapporo, 2008, pp. 309-314, doi: 10.1109/ATS.2008.45.

[2] K. Marcinek and W. A. Pleskacz, "AGATE - towards designing a low-power chip multithreading processor for mobile software defined radio systems," 2012 IEEE 15th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS), Tallinn, 2012, pp. 26-29, doi: 10.1109/DDECS.2012.6219018.

[3] IEEE Standard Verilog Hardware Description Language," in IEEE Std 1364-2001 , vol., no., pp.1-792, 28 Sept. 2001, doi: 10.1109/IEEESTD.2001.93352.

[4] D. Appello et al., "A comprehensive methodology for stress procedures evaluation and comparison for Burn-In of automotive SoC," Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017, Lausanne, 2017, pp. 646-649, doi: 10.23919/DATE.2017.7927068.