Late Contribution: VeriSide: A Modified Verilator for Leakage Assessment at the RTL Level

Late Contribution: VeriSide: A Modified Verilator for Leakage Assessment at the RTL Level

Behnam Farnaghinejad, Antonio Porsia, Annachiara Ruospo, Alessandro Savino, Stefano Di Carlo and Ernesto Sanchez
Department of Control and Computer Engineering, Politecnico di Torino, Turin, Italy

***Abstract*—Leakage assessment at the Register Transfer Level (RTL) is essential for identifying vulnerabilities in various designs, including cryptographic systems, AI models, and other applications handling sensitive data during the design phase. This paper introduces VeriSide, an innovative framework built as a modified version of Verilator to generate compact format files that directly capture `side-oriented` information, such as Hamming Distance (HD) or Hamming Weight (HW) of the signals. VeriSide streamlines the power side-channel (PSC) analysis process by providing efficient and scalable solutions for large-scale designs.**

Traditional methods relying on verbose Value Change Dump (VCD) or Switching Activity Interchange Format (SAIF) files face significant scalability and resource challenges, especially for complex systems-on-chip (SoCs). These methods incur substantial storage and processing overheads. VeriSide overcomes these limitations by drastically reducing file size and eliminating post-simulation memory usage, while maintaining analysis accuracy.

***Index Terms*—Power Side-Channel (PSC), Register-Transfer Level (RTL), Early-Stage Vulnerability Assessment**

I. Introduction

Power side-channel (PSC) attacks exploit the relationship between a system's power consumption and processed data to extract sensitive information. For instance, power consumption during Sbox operations in Advanced Encryption Standard (AES) rounds can exhibit data-dependent variations. An attacker can correlate power traces with distinguishers, which are statistical metrics used to identify key-dependent patterns.

Pre-silicon assessment at the RTL stage has emerged as a proactive strategy to address PSC attacks, offering flexibility for design modifications but lower accuracy than gate-level or post-silicon analysis. Frameworks like RTL-PSC [2] and RTL-PAT [1] have enhanced PSC analysis at this level but face scalability and resource constraints. RTL-PSC captures switching activity using SAIF (Switching Activity Interchange Format) files, while RTL-PAT improves efficiency by using compressed VCD (Value Change Dump) files to track signal changes over time. Despite this, RTL-PAT still struggles with storage and analysis overhead in large SoC designs. This paper proposes VeriSide, a modified version of Verilator to extract switching activity (SA) traces for side-channel attack analysis. Compared to existing solutions, it provides a more scalable and efficient approach for security assessment at the RTL level.

This work was funded by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU.

II. Background

Verilator, a widely used open-source SystemVerilog simulator, is the foundation for VeriSide. Verilator translates SystemVerilog code into C++/SystemC code, enabling high-speed simulation [3]. However, Verilator workflows involve generating and processing large VCD files, particularly for SoCs and large designs, and consequently, the PSC analysis of these VCD files becomes resource-intensive and time-consuming. The new proposal, called VeriSide, addresses this limitation by generating custom `side` format files containing Hamming Distance (HD) or Hamming Weight (HW) data grouped into encryption time windows. This eliminates the need for post-simulation power trace extraction, significantly reducing resource requirements while maintaining analysis accuracy.

III. Proposed Approach

In order to better highlight the improvements obtained by VeriSide, two strategies similar to the ones proposed by the state-of-the-art are used here as references for comparison: (i) a **standard VCD Parser** that processes targeted signals from the VCD file over a given time interval, computing Hamming Distance (HD) or Hamming Weight (HW) in post-processing, with the downside of having to store intermediate data; (ii) an **optimized VCD Parser** that directly computes and stores HD or HW values while reading the VCD file, filtering only required signals and time intervals, reducing memory usage.

To evaluate the performance obtained by VeriSide, a case study was developed involving a cryptographic accelerator connected to the CVA6 core [4]. In addition to this, two configurations of the 64-bit CVA6 core were implemented and analyzed to observe the impact of design modifications, signal counts, and VCD file size. In particular, one configuration includes an FPU, while the other one does not.

A. Framework Description

Figure 1 illustrates the block diagram for profiling power traces using the *Standard VCD Parser* with Verilator. This process extracts the signal's switching activity, including the timing and electrical values. After simulation, a large VCD file is generated and parsed to extract SA traces. The *Optimized VCD Parser* follows a similar structure but omits the "Extract

HD or HW" block. Instead, it directly computes and stores the SA traces during the VCD parsing process, eliminating the need to save the time and value of every signal. This optimization significantly reduces memory usage. Section III provides a detailed comparison of these approaches.

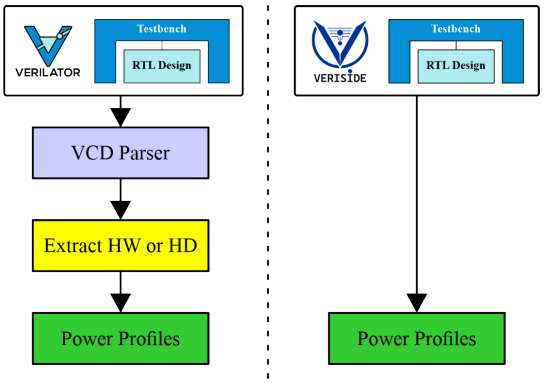

Fig. 1. Block diagram of power trace profiling using Verilator and VeriSide

In VeriSide, specific instances (e.g., the AES core or Sbox) and trigger signals (e.g., a CSR register signal) are defined before simulation to focus on critical design parts. Trigger signals mark when to extract SA traces, such as during AES encryption in our case study. This approach reduces unnecessary data collection, minimizing simulation overhead and storage needs.

The framework collects the Hamming Distance (HD) or Hamming Weight (HW) data from all submodules within the specified instance during the active periods defined by the trigger signal. By targeting only the relevant time intervals and modules, VeriSide streamlines the analysis process, eliminating the overhead associated with parsing large VCD files and reducing resource requirements. At the end of the simulation, VeriSide generates a custom `side` format file, which contains the HD-based or HW-based traces for each encryption time interval defined by the trigger.1.

IV. Performance Evaluation

A. Case Study

A C program was developed and executed in both configurations to perform 1000 AES encryption tasks. The program was run on the cryptographic accelerator connected to the CVA6 core [4]. The objective was to evaluate the AES modules inside the cryptography accelerator.

Simulations were conducted using two configurations, RV64IMAC and RV64IMFDC, both incorporating the Zkne extension. The results show that the RV64IMFDC configuration has a total of 98,523 signals, while the RV64IMAC configuration has 87,828 signals, with both configurations having 368 signals specifically related to the AES module—representing the subset of signals actively involved in its operation during encryption, including inputs, outputs, and internal connections selected to focus the analysis on relevant signals for power assessment. Additionally, the execution of the C program required 283,858 clock cycles in the RV64IMFDC configuration and 280,080 clock cycles in the RV64IMAC configuration.

B. Detailed Results

Tables I and II compare VeriSide's performance with Standard and Optimized VCD analysis for RV64IMAC and RV64IMFDC configurations, highlighting significant improvements in resource efficiency and speed. The output SA traces from all methods are identical, confirming that VeriSide's optimizations do not affect the accuracy of the experiment.

TABLE I
Performance Comparison for RV64IMAC Configuration

Metric	VeriSide	Verilator (Norm.)	Verilator (Opt.)
Simulation			
CPU Time (ms)	86,708	93,057	93,057
Wall Time (ms)	109,487	112,026	112,026
Disk Usage	3.0 MB	5.25 GB	5.25 GB
After Simulation (Parsing VCD file and extract HD/HW)			
Time (s)	—	759.84	611.34
RAM Usage	—	38.5 GB	12.7 GB

TABLE II
Performance Comparison for RV64IMFDC Configuration

Metric	VeriSide	Verilator (Norm.)	Verilator (Opt.)
Simulation			
CPU Time (ms)	109,464	121,395	121,395
Wall Time (ms)	109,487	141,176	141,176
Disk Usage	3.8 MB	5.55 GB	5.55 GB
After Simulation (Parsing VCD file and extract HD/HW)			
Time (s)	—	753.00	588.00
RAM Usage	—	46 GB	14 GB

V. Discussion and Conclusion

VeriSide addresses the limitations of existing VCD-based frameworks for power side-channel analysis by eliminating the need for resource-intensive VCD file generation and post-simulation parsing. Instead, it uses compact `side` format files that provide immediate trace readiness after simulation, significantly reducing disk usage, memory requirements, and processing time. Interestingly, VeriSide reduces disk usage by over 99% compared to both normal and optimized VCD analysis. Moreover, it eliminates the need for power trace extraction after simulation, avoiding post-simulation RAM usage and streamlining the analysis workflow for faster results.

References

[1] N. Pundir et al., "Power Side-Channel Leakage Assessment Framework at Register-Transfer Level," *IEEE Trans. VLSI Syst.*, vol. 30, no. 9, pp. 1207–1218, Sep. 2022, doi: 10.1109/TVLSI.2022.3175067.

[2] M. He et al., "RTL-PSC: Automated Power Side-Channel Leakage Assessment at Register-Transfer Level," *Proc. IEEE VTS*, Apr. 2019, pp. 1–6, doi: 10.1109/VTS.2019.8758600.

[3] W. Snyder et al., *Verilator*, Accessed: Jan. 2025. [Online]. Available: https://github.com/verilator/verilator

[4] F. Zaruba and L. Benini, "The Cost of Application-Class Processing: Energy and Performance Analysis of a Linux-Ready 1.7-GHz 64-Bit RISC-V Core," *IEEE Trans. VLSI Syst.*, Jul. 2019, doi: 10.1109/TVLSI.2019.2926114.