Gohan: A Golden-Copy-Aided Platform Enabling Online Hybrid-Interactive Reliability Analysis

Quan Cheng[1,2,†], Haoyuan Li[3,1,†], Wang Liao[4], Feng Liang[3], Longyang Lin[2], Masanori Hashimoto[1,*]

[1]Department of Communications and Computer Engineering, Kyoto University, Kyoto, Japan
[2]School of Microelectronics, Southern University of Science and Technology, Shenzhen, China
[3]School of Microelectronics, Xi'an Jiaotong University, Xi'an, China
[4]School of Systems Engineering, Kochi University of Technology, Kochi, Japan

***Abstract*—Ensuring reliable operation of modern silicon systems in safety-critical domains requires fault injection (FI) platforms that simultaneously achieve accuracy, observability, and efficiency. Traditional simulation-based FI provides full observability but is prohibitively slow, while hardware-based FI improves speed but struggles to provide cycle-level precision, cross-domain support, and comprehensive monitoring. To address this, this work presents Gohan, a golden-copy-aided platform that enables online, hybrid-interactive reliability analysis across multi-clock-domain systems. To preserve cycle-accurate state transitions, it introduces a per-domain golden copy that is generated independently for each domain through simulation. In addition, an FPGA-based host–DUT co-execution loop is used, incorporating clock domain-crossing (CDC)-aware pause-resume mechanisms and scan-chain-based FI. Experimental results on both lightweight RISC-V cores and complex AI processor demonstrate that Gohan achieves 100% consistency with simulation models under repeated pause–resume operations and fault campaigns, while providing 3 orders-of-magnitude speedup over pure simulation. By bridging simulation accuracy and hardware realism, Gohan offers a scalable, low-cost, and high-fidelity solution for reliability evaluation at pre-silicon stage.**

***Index Terms*—Fault Injection, Golden Copy, Multi-Clock Domain, FPGA Emulation, Reliability Analysis**

I. Introduction

Ensuring system reliability under harsh environmental conditions (e.g., radiation-prone environments or safety-critical automotive scenarios) has become a critical issue in modern chip and system design [1], [2]. Soft errors caused by single event upsets (SEUs), or single event transients (SETs), can lead to system failures or even disasters, requiring a thorough reliability evaluation prior to deployment [3]–[5]. In particular, as system complexity increases and hardware moves closer to the edge, the ability to accurately analyze and respond to fault behavior in real time is more important than ever [6].

Traditional software-based reliability evaluation, typically performed through RTL or gate-level simulation, provides full observability and flexibility but suffers from prohibitively low throughput [7]–[9]. Large-scale fault injection (FI) campaigns often require days or even weeks to complete due to the heavy computational overhead of cycle-accurate simulation. In addition, software-based simulations struggle to capture the true timing behavior and interactions of real hardware, particularly in systems with multiple clock domains, under fault conditions [10]. In contrast, hardware-based platforms such as FPGA prototypes offer significantly higher execution speed and the ability to evaluate realistic workloads [11], [12]. However, they introduce a new set of limitations. FI in hardware is often imprecise of controlling FI location and timing at the cycle level. Specifically, FI mechanisms at specific flip-flops (FFs) or memory elements are either unavailable or coarse-grained. Moreover, they do not inherently address challenges in clock domain crossing (CDC), where injected faults may propagate unpredictably across asynchronous domains. Runtime observation in hardware also tends to be limited by "visibility" constraints, typically relying on sparse I/O or logic analyzers, which makes internal state monitoring shallow and unreliable.

To address these challenges, we propose Gohan: A **G**olden-Copy-Aided Platform Enabling **O**nline **H**ybrid-Interactive Reliability **An**alysis. It bridges the accuracy and observability of software simulation with the speed and realism of hardware execution, forming a hybrid environment for precise and efficient fault analysis. Our key contributions are as follows:

- **Golden-copy-aided precise execution framework**: A high-fidelity golden copy is generated from the simulation environment with full observability. The golden copy records fault-free states in a clock-domain-aware manner, preserving the temporal evolution of each domain separately. During hardware evaluation, the host leverages the golden copy to guide DUT execution on FPGA, enabling precise clock scheduling and full access to internal state nodes, ensuring correctness and reproducibility of results.
- **Online hybrid-interactive host-FPGA control loop**: Gohan establishes an online co-execution model between the host and FPGA-deployed DUT. The host manages fine-grained clock control, triggers FIs at precise cycles, and monitors runtime behavior through a lightweight interface. To maintain consistency with the golden copy, Gohan employs a CDC-aware pause-resume mechanism and compensates for clock skews, ensuring multi-domain execution remains aligned and fault effects are faithfully captured. This unified control loop combines the observability of simulation with the realism of hardware.

[†]Equally Credited Authors (ECAs); [*]Corresponding Author

978-3-9826741-1-7/DATE26/© 2026 EDAA

By combining simulation-generated behavioral baselines with real-time hardware-in-the-loop execution, Gohan enables comprehensive fault analysis with cycle granularity precision and efficiency across clock domains. It forms a critical foundation for the design and validation of reliable systems.

II. Related Work

Recent years have seen the rise of various FI methodologies aiming to evaluate system reliability under soft errors, especially in AI System-on-Chips (SoCs) deployed in safety-critical scenarios. HachiFI [10] proposed a lightweight, architecture-independent FI framework capable of full fault coverage through memory and scan-chain (SC) access. It features a low area overhead ($<2\%$), supports multiple fault patterns. HachiFI is validated on a 22nm Edge-AI processor with a high correlation ($R^2 = 0.984$) between FI and irradiation experiments. However, its applicability to multi-clock-domain systems is limited, as it lacks precise FI control and synchronization across different clock domains, which in turn leads to limited observability.

To improve estimation accuracy under limited observability, Cheng et al. [7] developed a pure software-based FI and reliability analysis platform. By adjusting the time scale, this framework can achieve fine-grained and accurate FI. However, such simulation-based platforms are inherently inefficient, as cycle-accurate simulations require prohibitively long runtimes. Furthermore, in multi-clock-domain systems, achieving high-accuracy FI demands an even finer time scale to capture cross-domain interactions, which further exacerbates the inefficiency.

While both works emphasize accuracy and coverage, ACE-Pro [13] takes a different approach by targeting simulation efficiency. It introduces the ACE propagation graph to prune the FI candidate set in RTL simulation, identifying masked faults. ACE-Pro reduces fault candidates by over 99% without sacrificing fidelity. Nevertheless, it relies heavily on RTL visibility and simulation environments, making it less applicable to gate-level testing scenarios under hardware acceleration. Moreover, RTL abstractions do not always reflect the final gate-level behavior of the design, especially regarding timing closure, synthesis optimizations, and clock-domain interactions, further limiting its applicability for accurate reliability analysis.

In contrast, our proposed platform, Gohan, unifies the strengths of the above works. It inherits the full-system coverage and SC-based FI support like HachiFI [10]. More importantly, Gohan uniquely supports FI across multiple clock domains, enabling comprehensive FI and reliability analysis for arbitrary designs with explicit adaptation to multi-clock-domain systems. With broad architectural compatibility, real-time control, cycle-granularity observability, and full-state accessibility validated against radiation testing, Gohan provides a flexible, low-cost, and high-fidelity solution for SEU resilience evaluation across all development stages.

III. Gohan Platform

Fig. 1 illustrates the architecture of the Gohan platform, which enables comprehensive FI, monitoring, and analysis in ASIC design flows, software-based environments, and FPGA-based execution. The process begins with DFT-ready HDL designs that undergo synthesis and place-and-route to produce a gate-level netlist with SCs. This netlist is simulated using a Verilog/SV testbench. During simulation, the Gohan tracer, a hybrid C/C++/SV co-simulation tool based on DPI-C interface, extracts runtime data and generates a Gohan-defined golden copy as the reference for fault analysis. The host side contains FI controller, logger, clock management, comparator, visualization interface, and peripheral controls. These modules enable real-time FI, behavior monitoring, and result comparison against the golden copy under multi-clock domain. For hardware acceleration or hardware-in-the-loop validation, the same DUT netlist is deployed on an FPGA. A dedicated control interface bridges the host and the FPGA via Ethernet, supporting multiple clock scheduling and FIs. This platform streamlines both pre-silicon simulation-based analysis and emulation-based fault evaluation.

A. Gohan in AISC Desgin flow

Algorithm 1 Gohan Tracer: Per-Domain Runtime Signal Extraction for Golden Copy.

```
Initialize DPI-C interface between SV and C++
Load ff_list, memory_list, clock_domains
Filter lists → selected_ff_list, selected_memory_list
for all domain d ∈ clock_domains do
    Initialize cycle_counter[d] ← 0
    Initialize Golden_Copy[d] ← ∅
end for
while simulation is running do
    for all domain d ∈ clock_domains do
        if triggered by configured clock edge of d then
            Increment cycle_counter[d]
            Build reg_frame[d] from FFs in domain d
            Build mem_frame[d] from MEMs in domain d
            Insert cycle_counter[d] into both frames
            Package into format_data[d]
            send_data(format_data[d])
            Append format_data[d] to Golden_Copy[d]
        end if
    end for
end while
for all domain d do
    Store Golden_Copy[d] file
end for
```

In ASIC design flow, the Gohan platform begins by processing HDL designs integrated with DFT-ready logic. Using commercial DFT tools, SCs are inserted into the synthesized design. To ensure both accessibility to global components and compatibility with diverse design modules, we define two operating modes for the DFT logic. The first mode allows FF-level FI and monitoring via SCs, and the second mode facilitates memory-level FI and monitoring through SCs.

After the design is finalized, it is crucial to extract runtime internal signals for reliability analysis. To this end, we leverage the DPI-C interface of Synopsys VCS and construct a hybrid C/C++ and SV co-simulation environment, referred to as

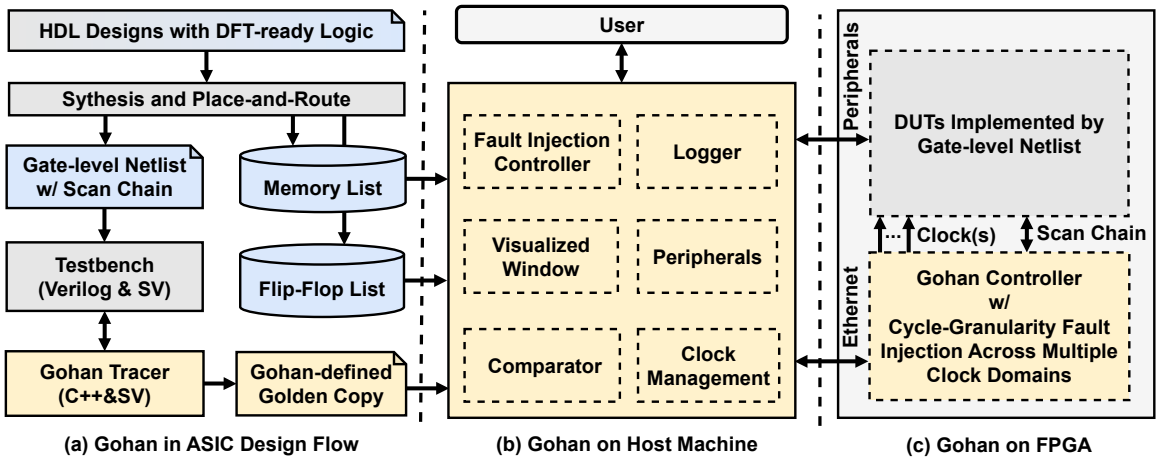

Fig. 1. Overview of Gohan platform. (a) ASIC design flow with scan chain integration and simulation tracing. (b) Host-side FI and monitoring environment. (c) FPGA-based DUT execution with runtime control via the Gohan control interface.

the Gohan tracer shown in Algorithm 1. It captures internal state data during simulation and records it as a golden copy across different clock domains, which serves as an accurate behavioral reference for subsequent FI and analysis in DUTs. The extraction and reuse of the golden copy can reduce the simulation overhead during FI campaigns. As simulation results are fully observable and functionally correct, we only need to extract data once. The golden copy can be reused during FPGA-based FI, eliminating the need to rerun simulation for each case and greatly improving overall efficiency.

Since modern designs often contain multiple clock domains, the Gohan tracer explicitly manages domain-level extraction. For each clock domain, a dedicated cycle counter is maintained, and FF/memory states are recorded with respect to the counter. Crucially, these counters are also used to derive the relative timing offsets across domains. This is important because inter-domain communication may require one or more cycles of latency, and such delays are not fixed but subject to physical propagation and false-path effects. By recording and preserving these timing differences, Gohan ensures that the golden copy reflects both intra-domain execution and inter-domain synchronization behavior. During FI, the DUT clocks are then controlled and aligned to the reference golden counters, so that even across multiple domains, timing relationships are faithfully reproduced. In Algorithm 1, the recording resolution is finer than the fastest clock, guaranteeing sufficient accuracy for preserving inter-domain alignment. Additionally, we extract two lists from the post-layout design (i.e., memory and FF lists). These lists are transferred to the host and serve as the basis for FI operations during runtime evaluation.

B. Gohan on Host Machine

On the host side, Gohan leverages the golden copy generated during simulation to enforce precise clock management (cycle-level). Two challenges arise when synchronizing hardware execution with the golden copy. First, frequent pauses can desynchronize DUT clocks from the golden copy. Second, physical deployment introduces subtle timing skew between clock domains. To address the first issue, Gohan employs

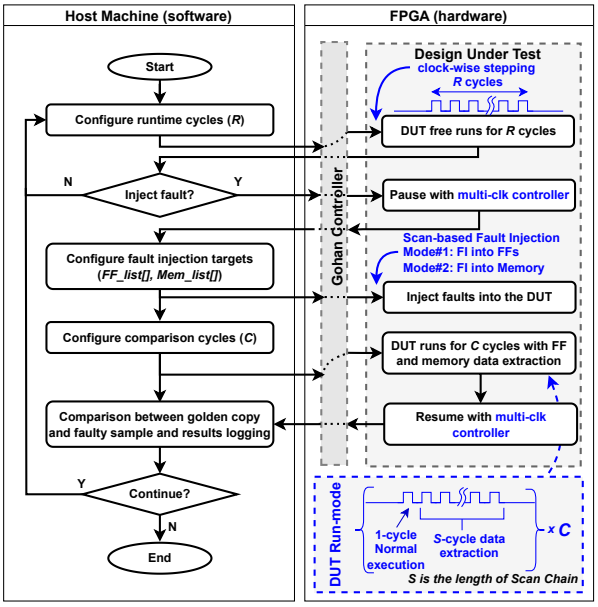

Fig. 2. Online hybrid-interactive flow in Gohan.

least-common-multiple (LCM)-based clock management in Section III-C, aligning all DUT domains to a LCM period so that independent clocks remain synchronized without drift. To address the second, Gohan further introduces an event-based clock control scheme in Section III-D, which dynamically gates and schedules clocks such that inter-domain offsets observed in the golden copy are faithfully reproduced on the DUT. This mechanism guarantees that even under multi-clock-domain conditions, FI can be specified and executed at precise cycles consistent with the golden copy.

With this foundation, FI is performed by mapping FF and memory operations to dedicated SC operations under two DFT modes (Section III-A), ensuring cycle-accurate FI across multiple clock domains. During DUT execution, runtime information on host is logged through a dedicated monitor, and a comparator evaluates system state against the golden copy, enabling accurate detection of divergence and fault effects under radiation-equivalent fault models.

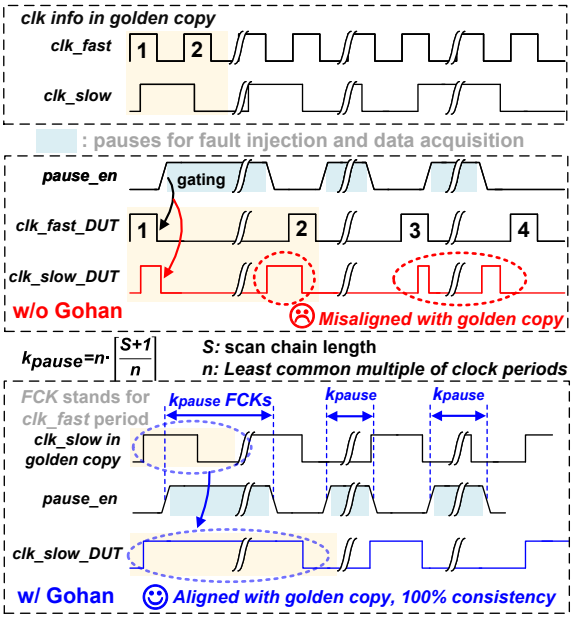

Fig. 3. LCM-based clock management for Gohan.

C. Gohan on FPGA

For real-time, online reliability evaluation, the DUT with integrated SCs is deployed directly on the FPGA. To support dynamic monitoring and FI, a lightweight interface is implemented on the FPGA and connected to the host machine via an Ethernet port. This interface enables specific clock control, scan-based FI, and data extraction in real time, forming a complete closed-loop system for efficient reliability analysis.

In Fig. 2, the host manages the evaluation campaign by first configuring the runtime cycles and FI targets, and then issuing corresponding commands to FPGA via Gohan controller. On the FPGA, the DUT executes normal cycles under clock-wise stepping until a pause request is received. The clock controller then halts all domains synchronously, ensuring consistent FI timing. The host triggers FI via SC(s), targeting either FFs or memory elements. While FF FI follows the standard scan-shift procedure, Gohan extends the scan interface with custom logic that enables direct overwriting of memory cells, thus unifying FF and memory FI. After FI, the DUT advances for C comparison rounds (CRs). Each CR consists of one functional cycle followed by S scan-extraction cycles, where S equals the SC length. During extraction, the DUT's functional clocks are paused, and the FF/memory contents are shifted out through the SC(s) to the host. The host then compares this faulty trace with the golden copy and determines whether to proceed to the next iteration. Thus, all domains can be checked. Through this interactive cycle of configure–inject–extract–compare, the FPGA and host collaboratively provide a scalable and efficient platform to evaluate reliability under various fault scenarios.

The proposed LCM-based clock management is shown in Fig. 3. Gohan framework requires repeated pauses with cycles of single-step execution, SC extraction, and re-execution, where accumulated disturbances from frequent gating can quickly desynchronize the DUT clock from the golden copy.

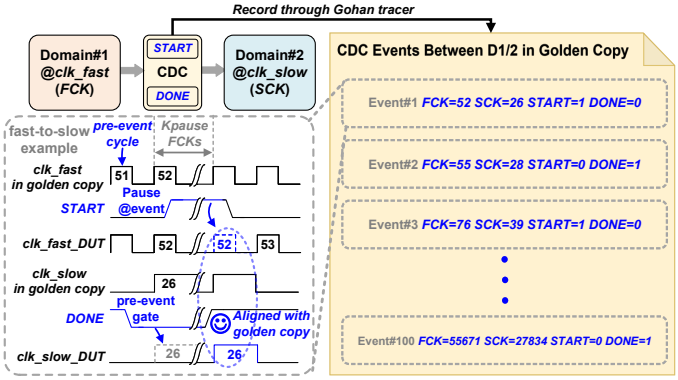

Fig. 4. Event-based clock controller for Gohan.

As cycle-accurate comparison is essential, maintaining clock alignment is critical. When pause cycles are not properly managed, the slow clock delivered to the DUT (clk_slow_DUT w/o Gohan) diverges from the golden copy clock (clk_slow) after a few pauses, leading to unreliable comparisons. To prevent this, a multi-clock controller is introduced. During golden copy simulation, the LCM of fast and slow clock periods is recorded. The pause length of the fast clock is then enforced as the smallest multiple of LCM exceeding the SC length, while the slow clock (clk_slow_DUT w/ Gohan) is frozen by holding its last state. A glitch-less gating module further ensures that the fast clock halts only after completing a full cycle. Notably, we assume that all clock domains are generated from a single source, which is the common practice in modern SoCs. Thus, the relative phase relation across domains is fixed, and the clock controller strictly preserves it during repeated pause–resume operations. With this strategy, the DUT clocks remain fully consistent with the golden copy, eliminating pause-induced mismatches in reliability evaluation.

D. Event-Based Clock Gating for Multi-Clock Domains

In practical hardware deployment, slight clock skew between domains can cause communication mismatches. For example, a receiving domain may observe incoming data one cycle later than expected. Such discrepancies would normally break cycle-accurate comparison against the golden copy, since even minor misalignments could cause divergence and invalidate subsequent fault analysis. To avoid this, Gohan does not let simulation and hardware executions drift independently. Instead, it enforces the golden copy's execution schedule onto FPGA-deployed DUTs through event-based clock gating mechanism. Namely, every cycle in hardware is explicitly aligned with the golden reference, ensuring that cross-domain timing relations remain consistent. This guarantees that simulation and hardware traces stay synchronized, allowing accurate and reproducible fault analysis.

As illustrated in Fig. 4, the Gohan tracer records a complete event sequence during simulation, explicitly mapping fast-clock (FCK) and slow-clock (SCK) cycles. During FPGA execution, these events are replayed to orchestrate clock control in an event-driven manner. In the case of fast-to-slow data transfer, the SCK domain is gated (`DONE=0`) one cycle

TABLE I
DUT EXPERIMENTAL SETUP

	Hummingbird E203	Tiny RISC-V+ NVDLA-based Accel.
Execution Freq.	HCLK@16MHz LCLK@32kHz	Accel.@150MHz RISC-V@25MHz
Memory size	128kB	608kB
Scan FF number	11825	66813
Application	Dhrystone (10 runs)	LeNet5 (MNIST) (1 run)
Execution Cycle (FCK)	19,408	87,750

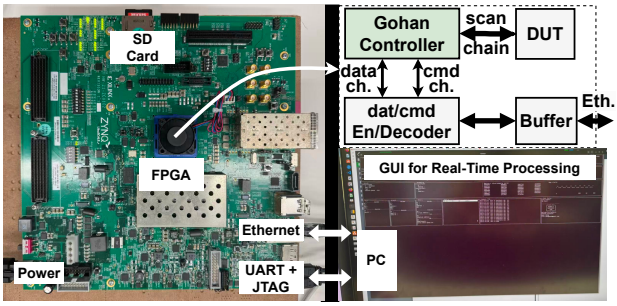

Fig. 5. Experimental setup for Gohan evaluation.

before the scheduled event to freeze its state, the FCK then performs the designated event cycle, and both domains remain paused for a LCM number of FCK cycles to guarantee state consistency. The subsequent release of the SCK (`DONE=1`) is itself logged as an event, with the FCK `START` signal issued one cycle later since the FCK-side operation has already been executed. This protocol ensures deterministic alignment of multi-clock-domain behavior between FPGA execution and the golden copy. The same event-driven protocol also applies to slow-to-fast transfers, where the FCK domain is pre-event gated and both domains remain paused for an LCM number of FCK cycles, ensuring consistent alignment.

This event-based strategy eliminates behavioral discrepancies caused by clock skew between physical execution and the golden copy. Although it incurs some performance overhead due to additional pauses, the intrinsic speed of FPGA execution ensures that the loss in efficiency is negligible compared to the gain in accuracy and consistency.

IV. EXPERIMENT AND EVALUATION

A. Experimental Flow and Platform Setup.

To evaluate Gohan, we conduct experiments on two DUTs: a Hummingbird E203 RISC-V core [14], an AI processor integrating a lightweight Tiny RISC-V core [15] and a downsized NVDLA-based AI accelerator [16] featuring a 16×16 INT8 processing element array. These two DUTs differ significantly in scale and application scenarios, ranging from lightweight processor workloads to AI inference tasks, thus providing a broad and representative evaluation for Gohan. Moreover, since both DUTs naturally incorporate multiple clock domains, they serve as ideal platforms to validate Gohan's capability of performing cycle-granularity FI across multiple clock domains, ensuring realistic assessment of cross-domain synchronization and system-level reliability analysis.

TABLE II
CONSISTENCY EXPERIMENT RESULTS.

		Hummingbird E203	Tiny-RISC-V+ NVDLA-Based Accel.
FAR @P=500	**Baseline-1**	73.6%	49.4%
	Gohan	**100%**	**100%**
PFAR	**Baseline-2**	84.60%@P=1k 14.23%@P=10k	76.34%@P=1k 9.39%@P=10k
	Gohan	**99.98%@P=1k** **98.67%@P=10k**	**99.97%@P=1k** **98.58%@P=10k**
FI Consistency		**100%**	**100%**

P: number of pause; PFAR: event-based clock control for fair comparison.

The experimental setup (Fig. 5) is implemented on a Xilinx ZCU102 FPGA board, integrating multiple components to support FI and analysis. The system adopts a ZYNQ SoC architecture, where the Processing System (PS) communicates with the host (PC) through Ethernet interfaces. The DUTs are synthesized and routed with DFT logic, where the generated SCs are used for FI and runtime monitoring. On the host side, experiments are managed by a Linux server equipped with an Intel i9-10850K processor and a 10 Gbps Ethernet connection to the FPGA, which operates FI campaigns and collects data for analysis via a custom host software in real time.

Table I summarizes the DUT configurations. The E203 RISC-V core is configured with two clock domains: a high-speed clock (HCLK) at 16 MHz for normal units and a low-speed clock (LCLK) originally specified at 32.768 kHz for always-on units (e.g., watch-dog). For experimental convenience, LCLK is adjusted to 32 kHz so that HCLK and LCLK share an LCM, simplifying synchronized testing. The E203 integrates 128kB memory and contains 11,825 FFs, executing 10 runs of Dhrystone in 19,408 cycles. Another DUT, including Tiny RISC-V and NVDLA-based accelerator, operates with two clock domains (RISC-V at 25 MHz and accelerator at 150 MHz), integrates 608kB memory, and consists of 66,813 FFs. It executes LeNet-5 (MNIST dataset) in 87,750 cycles. For DUTs, the back-end netlists with SC are deployed on FPGA PL side, operating at their respective clock domains. A lightweight Gohan controller, implemented in Verilog, serves as DUT interface between the host and DUTs. It not only orchestrates FI in coordination with the host but also manages precise clock management across multiple clock domains and enables cycle-accurate monitoring of DUT execution. Besides, this controller consumes about 3k LUTs (programmable logic) on ZCU102 with two clock-domain support.

B. Experimental Results

For DUTs, we first execute them on Gohan to obtain and store their complete golden copies. To validate the feasibility of Gohan, we conduct a series of consistency experiments. Here, Baseline-1 refers to execution without event-based clock control, and Baseline-2 refers to execution without LCM-based clock control. In the first experiment, both DUTs are executed on FPGA without any FI to evaluate the influence of clock skews in physical DUT executions. During a complete fault-

free run, we randomly selected 500 frames for comparison. To reach each selected frame and perform scan-out, we applied the LCM-based pause–resume scheme in both configurations. This guarantees fairness by preventing timing hazards that could otherwise accumulate under repeated pauses. At each selected frame, the FF and memory states were extracted via the scan chain and compared against the corresponding golden copy frames. The evaluation metric, Frame Alignment Rate (FAR), is defined as the ratio of matched frames between the hardware execution and the golden copy. Pause points are selected outside the event windows already recorded in the golden copy. As summarized in Table II, the Baseline-1 setup suffers from physical clock skew, causing divergence between hardware execution and the golden copy. This mismatch is particularly evident in the more complex Tiny-RISC-V and NVDLA accelerator, where errors accumulated over long execution sequences. In contrast, Gohan replays the recorded events and applies clock-specific control, thereby eliminating skew accumulation and achieving 100% FAR.

Then, we investigate the effect of repeated pause-resume operations, which are required after FI to assess error propagation. Directly pausing and resuming the DUT can introduce inconsistencies across clock domains. To evaluate robustness, we insert P (1k and 10k) random pauses during execution. At each pause, we perform a single SC readout to capture one frame and compare it with the golden copy; PFAR is the fraction of these P frames that match. In Table II, Baseline-2 exhibits a rapidly decreasing PFAR as P grows, reflecting accumulated timing errors and rendering it unsuitable for FI. With Gohan's clock controller, however, such clock misalignment is almost entirely eliminated, maintaining a near-100% PFAR even under intensive pause-resume operations.

Finally, the FI correctness is verified. Identical faults are injected in both VCS simulation and the hardware DUT, repeated for 1000 trials. After each FI, the DUT executes 8 CRs and the results are compared against the simulation. In all cases, Gohan achieves 100% alignment with the software model, confirming that the proposed framework provides reliable and reproducible FI across hardware and simulation domains. Regarding throughput, the FI rate is fundamentally determined by the SC length, as each FI requires a complete scan shift for state overwrite and extraction. With the DUTs studied here, the hardware-accelerated flow on ZCU102 achieves more than 1500$\times$ (3 orders) higher simulation efficiency compared to VCS running on Intel i9-10850K, demonstrating the practicality of Gohan for large-scale FI campaigns.

C. Comparison with Fault Evaluation SOTAs

Compared to prior evaluation approaches as summarized in Table III, Gohan demonstrates clear advantages in cycle-level FI accuracy and multi-clock domain applicability. In [7], FI is performed through pure simulation with full backdoor access, which provides full visibility but suffers from extremely low efficiency and lacks cycle-level precision, making it impractical for large-scale workloads or pre-silicon validation. In [10], a software-hardware co-designed framework significantly improves efficiency through SC-based FI, achieving nearly three orders of magnitude speedup over pure simulation. However, it remains constrained to single-clock-domain observability and does not support cycle-level accurate control, leading to potential underestimation of CDC-related fault effects. In contrast, Gohan attains comparable acceleration (3 orders of magnitude faster than pure simulation) while further combining golden copy–assisted event replay with a CDC-aware SC-based FI mechanism. This capability not only ensures 100% consistency with simulation results but also enables scalable, low-cost fault analysis on complex SoCs, bridging the gap between simulation accuracy and hardware realism.

TABLE III
COMPARISON WITH FAULT EVALUATION SOTAS.

	DAC'24 [7]	**HachiFI [10]**	**This work**
Technique	Simulation (VCS)	Software-Hardware Co-Design	**Golden Copy Assisted; SW/HW Co-Design; Multi-Clock FI**
FI Mechanism	Backdoor	Scan Chain	**Scan Chain**
FI Efficiency	Low	High	**High**
Cycle-Accurate FI Control	No	No	**Yes**
Clock-Aligned Observability	Yes	Partial Single Clock Observability	**Yes Multiple Clock Observability**
FI Analysis Cost (HW)	Low	Middle	**Middle**
Target Designs	Netlist Designs	Netlist Designs w/ SC(s)	**Netlist Designs w/ SC(s)**

V. CONCLUSION

This paper presents Gohan, a golden-copy-aided FI and reliability evaluation platform designed to unify the advantages of software simulation and FPGA-based execution. By generating per-domain golden copies during simulation and replaying them to orchestrate CDC-aware, cycle-accurate clock control on FPGA, Gohan eliminates mismatches caused by physical skew and repeated pause-resume operations. Furthermore, its scan-chain-based FI framework, extended with memory overwriting capability, enables flexible and precise fault targeting at both flip-flop and memory levels. Our evaluation on representative DUTs demonstrates 100% alignment between hardware and simulation traces across consistency tests and FI campaigns. Compared with prior approaches, Gohan not only supports designs with multiple clock domains but also achieves high efficiency without sacrificing accuracy. These results establish Gohan as a practical and scalable solution for SEU resilience evaluation, bridging the gap between pre-silicon analysis and post-silicon deployment.

ACKNOWLEDGMENT

This work was supported in part by the Grant-in-Aid for Scientific Research (S) from Japan Society for the Promotion of Science (JSPS) under Grant 24H00073; the National Natural Science Foundation of China under Grant 62274081; Guangdong Projects under Grants 2023QN10X177 and 2025B0101180002; and the Grant-in-Aid for Early-Career Scientists from JSPS under Grant JP21K17721.

REFERENCES

[1] V. Vargas et al., "Radiation Experiments on a 28 nm Single-Chip Many-Core Processor and SEU Error-Rate Prediction," in IEEE Transactions on Nuclear Science, vol. 64, no. 1, pp. 483-490, Jan. 2017, doi: 10.1109/TNS.2016.2638081.

[2] M. Wirthlin, E. Johnson, N. Rollins, M. Caffrey and P. Graham, "The reliability of FPGA circuit designs in the presence of radiation induced configuration upsets," 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, 2003. FCCM 2003., Napa, CA, USA, 2003, pp. 133-142, doi: 10.1109/FPGA.2003.1227249.

[3] H. . -B. Wang, Y. . -S. Wang, J. . -H. Xiao, S. . -L. Wang and T. . -J. Liang, "Impact of Single-Event Upsets on Convolutional Neural Networks in Xilinx Zynq FPGAs," in IEEE Transactions on Nuclear Science, vol. 68, no. 4, pp. 394-401, April 2021, doi: 10.1109/TNS.2021.3062014.

[4] N. Miskov-Zivanov and D. Marculescu, "Multiple Transient Faults in Combinational and Sequential Circuits: A Systematic Approach," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 29, no. 10, pp. 1614-1627, Oct. 2010, doi: 10.1109/TCAD.2010.2061131.

[5] P. E. Dodd and L. W. Massengill, "Basic mechanisms and modeling of single-event upset in digital microelectronics," in IEEE Transactions on Nuclear Science, vol. 50, no. 3, pp. 583-602, June 2003, doi: 10.1109/TNS.2003.813129.

[6] D. A. Santos et al., "Neutron Irradiation Testing and Analysis of a Fault-Tolerant RISC-V System-on-Chip," 2022 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), Austin, TX, USA, 2022, pp. 1-6, doi: 10.1109/DFT56152.2022.9962335.

[7] Q. Cheng et al., "How accurately can soft error impact be estimated in black-box/white-box cases?–a case study with an edge AI SoC–," in Proc. 61st ACM/IEEE Design Automation Conf. (DAC), 2024.

[8] R. Balasubramanian and K. Sankaralingam, "Understanding the impact of gate-level physical reliability effects on whole program execution," 2014 IEEE 20th International Symposium on High Performance Computer Architecture (HPCA), Orlando, FL, USA, 2014, pp. 60-71, doi: 10.1109/HPCA.2014.6835976.

[9] R. Natella, D. Cotroneo, and H. S. Madeira, "Assessing dependability with software fault injection: A survey," ACM Computing Surveys (CSUR), vol. 48, no. 3, pp. 1–55, 2016.

[10] Q. Cheng, W. Liao, R. Zhang, H. Yu, L. Lin and M. Hashimoto, "HachiFI: A Lightweight SoC Architecture-Independent Fault-Injection Framework for SEU Impact Evaluation," 2025 Design, Automation & Test in Europe Conference (DATE), Lyon, France, 2025, pp. 1-7, doi: 10.23919/DATE64628.2025.10993139.

[11] N. Khoshavi, C. Broyles, Y. Bi and A. Roohi, "Fiji-FIN: A Fault Injection Framework on Quantized Neural Network Inference Accelerator," 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 2020, pp. 1139-1144, doi: 10.1109/ICMLA51294.2020.00183.

[12] I. Tuzov, D. de Andrés, J. -C. Ruiz and C. Hernández, "BAFFI: a bit-accurate fault injector for improved dependability assessment of FPGA prototypes," 2023 Design, Automation & Test in Europe Conference & Exhibition (DATE), Antwerp, Belgium, 2023, pp. 1-6, doi: 10.23919/DATE56975.2023.10137300.

[13] D. -A. Yang, Y. -T. Chang, T. -S. Hsu, J. -J. Liou and H. H. Chent, "ACE-Pro: Reduction of Functional Errors with ACE Propagation Graph," 2021 IEEE International Test Conference (ITC), Anaheim, CA, USA, 2021, pp. 10-19, doi: 10.1109/ITC50571.2021.00008.

[14] Y. Tong and Y. Xia, "Implementation of Hummingbird E203 Based Electric DC Motor Controller," 2023 3rd International Conference on Electronic Information Engineering and Computer Science (EIECS), Changchun, China, 2023, pp. 1220-1223, doi: 10.1109/EIECS59936.2023.10435587.

[15] J. Zhou, G. Qin, L. Li, C. Guo and W. Wang, "ISA Extensions of Shuffling Against Side-Channel Attacks," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 43, no. 3, pp. 761-773, March 2024, doi: 10.1109/TCAD.2023.3323165.

[16] F. Farshchi, Q. Huang and H. Yun, "Integrating NVIDIA Deep Learning Accelerator (NVDLA) with RISC-V SoC on FireSim," 2019 2nd Workshop on Energy Efficient Machine Learning and Cognitive Computing for Embedded Applications (EMC2), Washington, DC, USA, 2019, pp. 21-25, doi: 10.1109/EMC249363.2019.00012.