

Hierarchical EMFI Analysis on a RISC-V SoC

Dillibabu Shanmugam[✉]
Worcester Polytechnic Institute
Worcester, MA, USA
dshanmugam@wpi.edu

Zhenyuan Liu[✉]
Worcester Polytechnic Institute
Worcester, MA, USA
zliu12@wpi.edu

Patrick Schaumont[✉]
Worcester Polytechnic Institute
Worcester, MA, USA
pschaumont@wpi.edu

Abstract—Electromagnetic fault injection (EMFI) can induce transient faults in SoCs. Such faults are often hard to understand and analyze due to limited hardware observability. We present a scan-chain-assisted EMFI analysis on a custom RISC-V system-on-chip (SoC) called CAPRI1. The scan chain offers full visibility of the impact of EMFI on the SoC state. We then perform hierarchical fault analysis and follow the fault propagation from initial bit-flip through the SoC micro-architecture into the high-level software behavior. We evaluate the method with EMFI experiments on two PIN-verification firmware workloads. From scan captures, we classify faults by source component and by system-level outcome. We also relate the initial bit-flip (x,y) coordinates to layout-level features. This framework improves understanding of design behavior under EMFI fault scenarios and offers early vulnerability assessment.

Index Terms—Scan chain, Power distribution network, Layout-aware analysis, Electromagnetic fault injection

I. Introduction

Fault injection attacks exploit the fact that integrated circuits are not perfectly robust under all operating conditions. By perturbing supply voltage, clock, temperature, or the electromagnetic environment, an adversary can force a secure embedded system into erroneous states that violate control-flow or data-flow integrity. These errors are then analysed to extract secrets or bypass security checks.

Electromagnetic fault injection (EMFI) is particularly attractive to attackers because it can be performed contactlessly, often without invasive modification of the target device. A short, high-intensity current pulse through a small coil placed close to the die induces eddy currents and transient voltages in the chip, which can lead to timing violations and bit upsets. Prior work [1], [2] has shown that carefully timed EMFI can cause instruction skips or arithmetic faults in microcontrollers.

Most published EMFI studies work in a black-box setting. The internal state of the device is not directly observable; only external behaviour such as program output, error flags, or crashes is recorded. To explain the observations, the analyst postulates a fault model—for instance, a single-bit flip in a register, or the replacement of one instruction by another—and tries to match it to the data. In reality, the effect of a physical disturbance may be more complex: multiple bits may flip simultaneously, or the device may visit states that cannot be reached in normal operation. Without visibility into the internal state, these “weird” behaviours [3] remain opaque, complicating validation of countermeasures.

This work addresses that visibility gap using a system-on-chip derived from CROC [4] called CAPRI1. CAPRI1 integrates a microcontroller core, memories, peripherals, and, crucially, a full scan chain that connects all 12756 flip-flops.

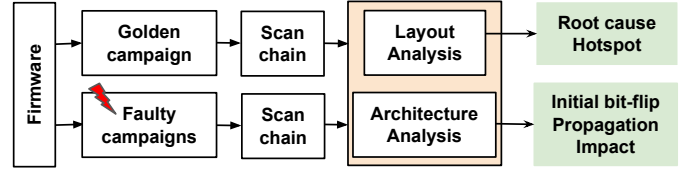


Fig. 1. Scan-assisted EMFI analysis flow on CAPRI1. The golden campaign and faulty campaigns each produce scan-chain dumps, which are processed by two analysis layers: an architecture-level layer (fault propagation) and a layout-level layer (initial bit flip and physical hotspot localisation).

The scan infrastructure allows us to halt the device after a fault injection and dump the full internal state. By comparing these scan dumps against a fault-free reference as depicted in Fig. 1, we can pinpoint the first bit-flip in hardware, track its propagation over subsequent cycles, and relate it to the eventual software-level outcome. First, we study how EMFI affects CAPRI1 as a piece of hardware: magnitudes and locations of bit flips, common patterns, and correlations with physical structures. Second, we analyse how the same hardware faults interact with two firmware workloads, denoted `verifyPIN0` (VP0) and `verifyPIN7` (VP7) firmware [5], leading to either exploitable behaviour or benign(no-effect).

The main contributions of this paper are:

- A scan-assisted EMFI framework on CAPRI1 that provides cycle-resolved, bit-level visibility of the internal state across all 12 756 flip-flops after injection.
- A hierarchical fault analysis that relates physical fault manifestations to architectural effects through a study of the VP0 and VP7 firmware workloads under EMFI.
- We demonstrate that a long-loop PDN coupling model, derived pre-silicon, is consistent with post-silicon scan-observed register bit flips under EMFI.

The remainder of this paper is organized as follows. Section II describes the CAPRI1 SoC, the EMFI experimental setup and the fault injection methodology. Section III develops a hierarchical fault analysis that relates physical fault manifestations to architectural effects through an application study of the VP0 and VP7 firmware workloads under EMFI. Section IV links pre-silicon and post-silicon findings by showing that long PDN loops in the top metal layers couple strongly to the EMFI probe, inducing transient loop currents that create a local supply voltage drop. This voltage droop reduces noise and timing margins, explaining the first bit flips observed in the scan-chain data. Section V concludes the paper.

II. CAPRI1 SoC and EMFI Experimental Setup

CAPRI1 is a 32-bit RISC-V microcontroller SoC based on the Ibex (CV32E20) core, developed as a fault injection test

TABLE I
Flip-flop distribution in CAPRI1.

Category	FFs
Peripherals	2722
GPR	992
μ arch (F:283, D:14, CSR:420, LSU:67)	784
Memory (IMEM:4129, DMEM:4129)	8258

platform. The chip includes two on-chip SRAM blocks using DFF(128 words each, for instruction and data memory) and standard peripherals (timer, UART, GPIO, and control/status registers), all accessed via a memory-mapped on-chip bus. The design was fabricated in a 0.18 μ m CMOS process, with a core area of about 2 mm \times 2 mm and a 1.8 V supply voltage. A JTAG-based debug unit provides program loading, break-points, and execution control. CAPRI1 also incorporates full design-for-test (DFT) features (notably a comprehensive scan chain and on-chip debug interfaces), enabling precise control of execution and complete observation of the entire internal state. In the following, we describe the SoC’s architecture and scan chain, highlight key physical layout aspects, and detail the EMFI experimental setup and triggering methodology.

A. Architecture Hierarchy and Scan Chain

CAPRI1’s processor employs a two-stage pipeline (instruction-fetch and decode/execute) implementing the RV32IMC RISC-V instruction set. The architecturally visible state includes the 32 general-purpose registers (GPRs), the program counter (PC), the control and status registers (CSRs), and memory-mapped I/O registers. In contrast, micro-architectural state (e.g., prefetch buffers, pipeline registers, internal control signals) is not visible to software. All sequential elements of the SoC (flip-flops in the CPU core, SRAMs, and peripheral interfaces) are connected into a single scan chain. This chain comprises 12756 flip-flops in total (Fig. 2 and Table I). Each scan cell is associated with a register instance name and physical (x, y) coordinate in the chip layout. The scan chain supports two modes:

- *Scan-load*: an arbitrary state can be shifted into the chain (or a previously captured state can be looped back) to initialize all flip-flops before executing a test program.
- *Scan-capture*: In this mode, after halting the system clock, the content of all flip-flops is shifted out for analysis.

This ability to initialize the device to a known state and later capture its entire post-execution state is central to our fault injection methodology. Mapping each observed bit from a scan dump to its known die coordinates allows us to correlate fault-induced state changes with specific regions of the chip. This capability greatly aids in diagnosing the effects of EMFI.

B. Physical Design and Layout Components

From a layout perspective (Fig. 3), CAPRI1 includes several features that influence its susceptibility to EMFI.

- **Power Distribution Network (PDN):** Top metal layers (M5/M6) form a grid-style VDD/VSS power network covering the entire core. The PDN uses core rings on M3/M4 and M5/M6 (4.75 μ m width, 3 μ m spacing, 8 μ m

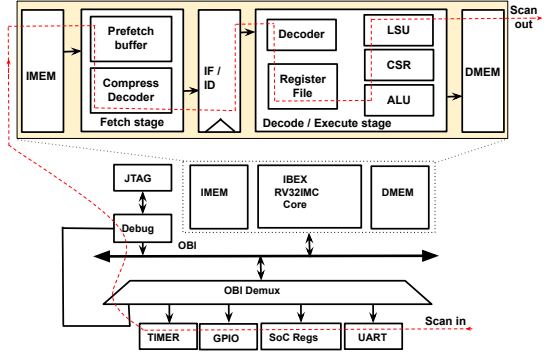


Fig. 2. Block diagram of the CAPRI1 SoC. The Ibex core connects through an OBI demultiplexer to instruction and data SRAMs and to MMIO peripherals. A single scan chain spans all sequential elements, enabling full-state initialisation and readout under EMFI. This architectural organisation is the basis for the cycle-resolved, bit-level fault analysis in later sections.

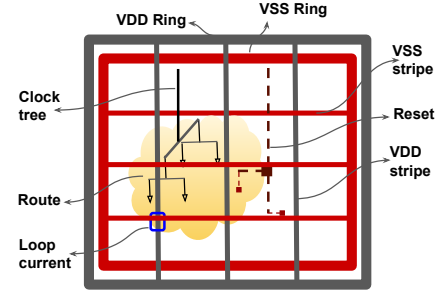


Fig. 3. Key CAPRI1 layout structures—power rings, stripes, and clock/reset routing—form conductive loops that determine the chip’s susceptibility.

offset). Dense M5 horizontal and M6 vertical straps (70 μ m pitch) provide global distribution. M2–M4 carry fewer supply lines. Table II lists total wire lengths for supply (VDD, VSS), PDN, reset, and clock networks. VDD and VSS combined exceed 4.5 m of wiring, mainly on M4–M6, whereas M1 contributes minimally. Under EMFI, disturbances in the top metal cause localized supply fluctuations that may upset nearby flip-flops.

- **Clock Distribution:** The clock tree uses buffered trunk and leaf branches. Trunk segments have slews near 0.18 ns and leaves around 0.23 ns. Clock nets [6] couple strongly to EMFI.
- **Reset Network:** A global active-low reset line spans mid- and low-metal layers and is synchronized to release with low skew relative to the clock; each flip-flop uses an asynchronous reset pin. Under EMFI, a transient at this pin can reset a subset of flip-flops, disrupting state even when the synchronized reset logic remains deasserted.
- **Routing and Cell Placement:** Signals route mainly on Metals 1–4; Metal 3 contributes roughly 50–70% of total length and Metal 4 carries longer runs. High impedance limits EMFI effects on signal nets.

C. EMFI Bench Setup and Triggering Methodology

To assess CAPRI1’s vulnerability to electromagnetic fault injection (EMFI), we performed experiments on a test bench that allowed precise control of the injection parameters. A small coil probe, Fig. 4 was placed just above the chip’s

TABLE II
Global interconnect lengths for supply, reset, clock, and PDN, with PDN length by metal layer (rounded unit).

Network	Total length	PDN layer	Length
VDD (all layers)	2,315,495	METAL6	934,752
VSS (all layers)	2,192,268	METAL5	1,104,051
RST	18,748	METAL4	982,077
CLK (clk_i)	7,858	METAL3	749,330
PDN*	4,507,763	METAL2	688,456
		METAL1	49,096

*Total PDN length equals VDD+VSS.

die using a computer-controlled micro-positioning stage. We used a high-speed pulse generator to deliver current pulses (amplitude up to ± 10 V, width 4–10 ns) through this coil. The coil’s pulses generated a brief magnetic field that inductively coupled into on-chip structures (such as the power distribution network mesh and the clock/reset network), creating transient disturbances in the chip’s voltages and currents.

A Raspberry Pi-based controller orchestrated each EMFI campaign (see Fig. 4). This controller provided the 1 MHz system clock, managed the reset line and JTAG interface, and coordinated the scan operations. Meanwhile, the CAPRI1 firmware toggled a GPIO pin as a trigger once per clock cycle, immediately before executing the instruction under test. In our experiments, the test program ran for N clock cycles. We therefore defined N separate fault-injection campaigns, each targeting a specific clock cycle of the program’s execution. For example, campaign 10 targeted the 10th cycle. By aligning the EM pulse immediately after the clock’s rising edge (Fig. 5), we ensured that the injection occurred outside the flip-flops’ setup/hold window. This timing allowed the EM pulse to directly disturb the flip-flops’ stored output value Q .

Once a fault is injected, the controller immediately switches to scan mode. In scan mode, it froze all flip-flops and shifted out their contents, thereby dumping the internal state. It then returned the chip to functional mode to execute the next clock cycle before switching back to scan mode. This sequence of one clock cycle followed by a scan dump continued until the program completed. We later compared the sequence of scan dumps from each injection campaign to reference traces from a fault-free execution of the program. By correlating the locations of flipped bits with the specific cycle targeted by the EM pulse, we identified which regions or circuit blocks of CAPRI1 were most susceptible to EMFI during that cycle. Thus, our systematic campaign-based analysis enabled us to gauge the SoC’s fault tolerance and pinpoint specific blocks or interconnects that were particularly vulnerable to EM pulses.

III. Hierarchical Fault Analysis

The CAPRI test chip integrates an Ibex RISC-V core with full-scan access, which exposes the internal flip-flop state at cycle granularity. This feature enables precise identification of the first corrupted element after a physical disturbance and allows tracking of its propagation through the microarchitecture. Rather than viewing each experiment only through the final program outcome, we reinterpret the EMFI campaigns using a hierarchical, component-centric perspective.

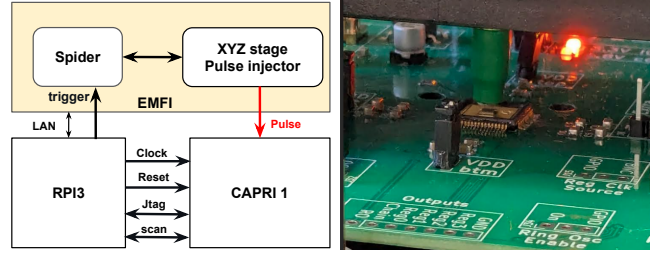


Fig. 4. Experimental EMFI bench. A Raspberry Pi 3 controls the CAPRI1 board, the pulse generator, and a motorised XYZ stage that positions the EM probe above the die. The controller provides clock, reset, JTAG, and trigger signals and collects scan-chain dumps after each injection. This setup enables automated spatial and campaign-based, cycle-resolved EMFI experiments.

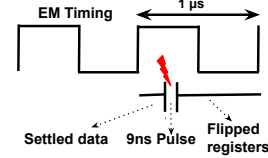


Fig. 5. Alignment of the system clock and EM pulse, illustrating the sampling window and the mechanism of an EM-induced bit flip.

Hierarchical fault analysis studies fault propagation bottom-up. A physical disturbance first perturbs one or more microarchitectural bits (e.g., within the register file or pipeline latches). The resulting upset then travels across hardware components, eventually modifies architectural state, and becomes observable through software interaction with the ISA. This multi-layer view links a concrete physical effect to an architectural manifestation and, finally, to an application-level failure. To capture this chain, we adopt a *component-centric* fault classification instead of a purely output-based taxonomy. Traditional output-based classes (e.g., crash, reset, success, fail) characterise only the externally visible result and thus conflate many distinct root causes. In contrast, a component-centric view attaches each observed outcome to specific hardware blocks, which exposes which structures enable or block the success of a given fault.

A. Component-Centric Fault Classification in CAPRI

Our component-centric hierarchical analysis steps:

- 1) **Initial component upset:** identify the microarchitectural or architectural component that first deviates from the golden execution (e.g., a flip in `ibex_register_file`, a corrupted instruction word in `sram_bank0`, or a mis-aligned PC in the fetch stage).
- 2) **Propagation path across Ibex blocks:** reconstruct how this initial upset propagates through Ibex blocks and pipeline registers, including control-flow divergence and secondary data corruptions.
- 3) **Resulting system-level behavior:** characterise the resulting program behavior, such as abnormal termination, preserved control flow with wrong data, or complete bypass of a security check. At a high level the system behavior is identified as fault patterns [7] such as Detour, Tunnel, Spinner, Wormhole.

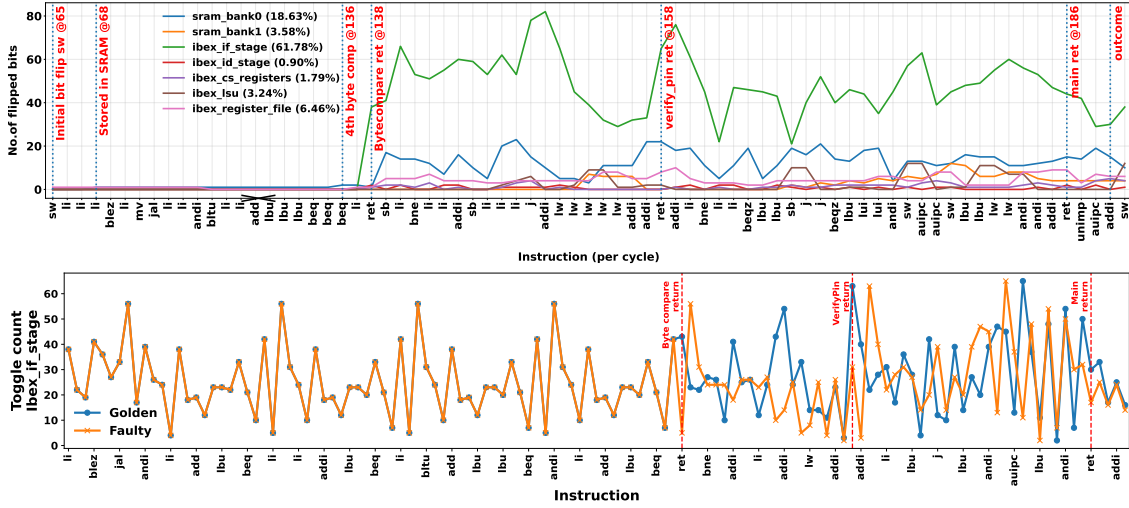


Fig. 6. Timeline of fault propagation for the register-file upset case. Top: number of flipped scan bits per instruction for major Ibex components. After the corrupted instruction is stored in `sram_bank0` and later refetched, the error concentrates in the fetch path: `ibex_if_stage` and `sram_bank0` dominate the flip count, while other blocks remain close to their golden behaviour, illustrating a narrow but effective propagation channel from the initial register-file upset to the final outcome. Bottom: toggle count per instruction for the golden and faulty executions of the `Ibex_if_stage` component; both traces overlap until the corrupted instruction is fetched, after which they diverge around the branch event and the subsequent returns.

- 4) **Component-centric outcome label:** assign a label that captures both the observed effect and the originating component, “register-file–originated data corruption”.

We apply this hierarchical framework to two PIN-verification kernels executed on CAPRI: VP0(unprotected), a naïve implementation that runs for 190 cycles, and VP7(protected), a hardened implementation that completes within 140 cycles. Two case studies are summarized below.

B. Register File Upset:

In the 65th EMFI campaign of VP0, we observe a register-file upset. We compare the golden and faulty executions and summarize our observations below.

Initial component: At clock cycle 65, a single-bit upset flips one bit in register `x15` of the `ibex_register_file`. The corresponding scan-cell coordinates (302, 731) identify the physical location of the disturbed flip-flop in the layout and thus the origin of this initial flip.

Propagation: The corrupted register value remains benign until it is used as an operand of a store instruction, which inadvertently overwrites an instruction word in the firmware’s instruction memory. Later, when that memory address is fetched (e.g., during a comparison loop), the altered instruction executes and writes a constant value (0x04) into a register involved in the PIN verification, making its content equal the expected reference. This satisfies the branch-if-equal condition at the final PIN check. However, a misaligned program counter (caused by the fault-induced prefetch anomaly) introduces a one-instruction delay that prevents the branch from actually being taken, shifting the fault’s effect from the data path to the control-flow path. This control-flow divergence then propagates through the pipeline, ultimately corrupting additional registers and even a word in data memory. As shown in Fig. 6, over 60% of the flipped bits in this faulty execution occur in the fetch stage, around 19% in the instruction memory,

and only 6% in the register file; other components are only minimally affected. This distribution highlights the fetch stage as the dominant hotspot of error propagation.

System-level behaviour: The core neither traps nor resets. It completes execution and returns a status value, but the result is incorrect and the PIN check is bypassed. As shown by the `ibex_if_stage` toggle trace in Fig. 6 (bottom), once the fault reaches the fetch pipeline the execution enters a distinct high-activity yet bounded mode, indicating stable but incorrect control flow that still satisfies the program’s termination conditions.

Component-centric label: This case is a register-file–originated silent data corruption with fault-induced *success*. A single register-file upset propagates through instruction memory and the fetch subsystem, alters the program return value (over 60 bits), and yields apparent success for an invalid PIN without raising an error.

C. Fault outcome and origin analysis

Table III summarizes VP0 and VP7 over 331 injections. Most campaigns show no observable effect (209/331); the dominant observable outcomes are resets (54) and crashes (63). VP7 exhibits no fault-induced successes, whereas VP0 shows two.

Among crash/success campaigns with an identified origin, the first deviation most often occurs in the instruction-fetch (IF) unit (29/65), followed by the load–store unit (LSU, 10) and decode/execute (DE, 9). The register file is rarely the earliest deviating block (2). This distribution is expected: IF logic is active every cycle and immediately controls the next instruction, so small upsets quickly amplify into control-flow divergence, whereas register upsets are often overwritten or masked before affecting a security-relevant comparison. Source code: <https://github.com/Secure-Embedded-Systems/capri1>.

TABLE III
Fault injection summary for VP0 and VP7: (a) campaign outcomes and (b) identified fault origins.

(a) Campaign outcomes						
Workload	Campaigns	Reset	Crash	No effect	Success	Unclassified
VP0	191	31	39	118	2	1
VP7	140	23	24	91	0	2
Total	331	54	63	209	2	3

(b) Identified fault origins						
Work load	Instruction Fetch	Decode Execute	Load Store	Control & Status Register	Register ALU	Register file SRAM
VP0	19	5	6	5	4	1
VP7	10	4	4	3	2	1
Total	29	9	10	8	6	2

IV. Direction-Aware PDN Loop Model for EMFI

a) *EMFI loop premise.*: An EMFI pulse induces eddy currents only in closed conductive loops. A dense PDN mesh contains many small cells with opposing circulation, which reduces net pickup. A small set of large, low-impedance boundary loops can enclose larger flux and can dominate pickup [8], [9]. The PDN is therefore reduced to a compact subset of long VDD/VSS loops for coupling analysis, not absolute rail-voltage prediction.

b) *Scan-derived anchor (first flip).*: Each campaign uses a post-injection scan dump and a fault-free reference. The first mismatching scan bit in scan order defines the *first flip*. Scan-to-instance mapping localizes this bit to a placed storage element with die coordinate (x_{ff}, y_{ff}) . In the register-file upset, the first flip maps near $(302, 731) \mu\text{m}$.

c) *PDN graph and loop sampling.*: VDD and VSS routes are extracted from DEF SPECIALNETS. One multilayer graph is built per rail: nodes denote junctions; edges denote wire segments and vias with metal-layer tags and physical lengths. The PDN contains exponentially many simple cycles, so full enumeration is not used. Candidate loops are sampled as fundamental cycles from randomized spanning trees. Repeated sampling yields a loop set $\mathcal{L}_r = \{\ell\}$ for each rail $r \in \{\text{VDD}, \text{VSS}\}$. A small subset (top 5 shown in the table IV) is retained after ranking, since long boundary loops dominate pickup under EMFI [8], as highlighted in Fig. 8.

d) *Direction-aware METAL6 loop metric.*: EM probe induces an electric field with tangential direction around the probe center. Anti-parallel traversal across METAL6 segments yields opposing contributions as shown in Fig 7. This cancellation is captured by a signed METAL6 accumulation.

Let a METAL6 segment e be traversed from endpoint $\mathbf{q}_0 = (x_0, y_0)$ to $\mathbf{q}_1 = (x_1, y_1)$. Define the traversal vector and midpoint:

$$\Delta \ell_e = \mathbf{q}_1 - \mathbf{q}_0, \quad \mathbf{m}_e = \frac{\mathbf{q}_0 + \mathbf{q}_1}{2}. \quad (1)$$

For probe center $\mathbf{p} = (x_p, y_p)$, define $\mathbf{r}_e = \mathbf{m}_e - \mathbf{p}$. A tangential unit vector at \mathbf{m}_e is approximated as the 90° rotation:

$$\hat{\mathbf{t}}_e(\mathbf{p}) = \frac{1}{\|\mathbf{r}_e\|} \begin{bmatrix} -r_{e,y} \\ r_{e,x} \end{bmatrix}. \quad (2)$$

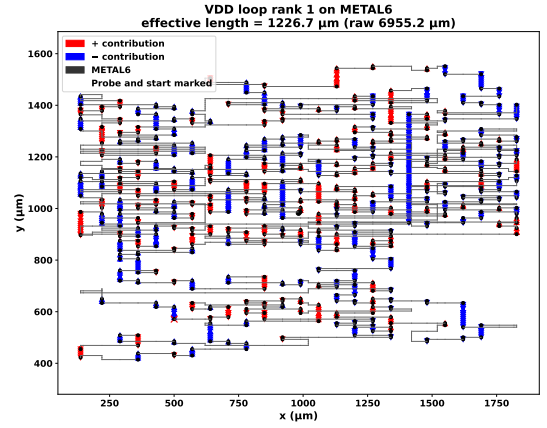


Fig. 7. Example METAL6 loop with sign assignment from (1)–(4). Red/blue segments contribute with opposite signs; the probe is at center(1000,1000). Illustration of loop current direction cancellation in M6 segments.

TABLE IV
Top-5 signed effective loop lengths L_{eff} on METAL6 (columns are ranks).

Net	R1	R2	R3	R4	R5
VDD	1226.7	1223.8	1220.7	1218.6	1218.2
VSS	1455.4	1454.2	1440.9	1439.2	1438.0

The signed length contribution of segment e is

$$a_e(\mathbf{p}) = \hat{\mathbf{t}}_e(\mathbf{p}) \cdot \Delta \ell_e, \quad (3)$$

so $a_e > 0$ adds and $a_e < 0$ subtracts. The direction-aware effective METAL6 length of loop ℓ is then

$$L_{\text{eff}}(\ell; \mathbf{p}) = \left| \sum_{e \in \ell \cap \text{M6}} a_e(\mathbf{p}) \right|. \quad (4)$$

For fixed (\mathbf{p}, r) , the dominant loop is the maximizer of L_{eff} over \mathcal{L}_r .

e) *Loop excitation and delivery to a victim tap.*: Each retained loop ℓ is modeled as a series R – L loop driven by an induced emf in the frequency domain:

$$I_\ell(\omega; \mathbf{p}, D) \approx \frac{\kappa_\ell(\mathbf{p}, D) V_0(\omega)}{Z_\ell(\omega)}, \quad Z_\ell(\omega) = R_\ell + j\omega L_\ell. \quad (5)$$

$\kappa_\ell(\mathbf{p}, D)$ captures coupling efficiency. A geometry term derived from $L_{\text{eff}}(\ell; \mathbf{p})$ is used to reflect direction-dependent cancellation on METAL6. The dominant spectral component [10] can be tied to pulse rise time t_r via $\omega \approx 2\pi \cdot 0.35/t_r$.

Voltage disturbance at a victim tap t is modeled on each rail $r \in \{\text{VDD}, \text{VSS}\}$ by linear superposition over the retained loops:

$$\Delta V_{t,r}(\omega) \approx \sum_{\ell \in \mathcal{L}_r} I_{\ell,r}(\omega) Z_{\ell \rightarrow t,r}(\omega), \quad (6)$$

where $Z_{\ell \rightarrow t,r}(\omega)$ denotes an effective delivery-path impedance from loop ℓ to tap t on rail r , extracted from the PDN graph.

f) *Worst-case differential swing bound.*: The register supply is $S(t) = V_{\text{DD}}(t) - V_{\text{SS}}(t)$. A phase-agnostic bound uses magnitude addition:

$$|\Delta S|_{\text{worst}} = |\Delta V_{t,\text{VDD}}| + |\Delta V_{t,\text{VSS}}|. \quad (7)$$

This bound supports ranking of probe locations and PDN loops by tap vulnerability without phase assumptions.

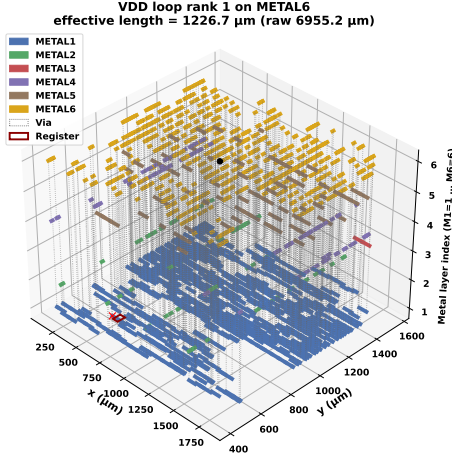


Fig. 8. VDD loop rank 1 across metal layers (M1–M6) with vias. The square marker indicates the victim neighborhood near (302, 731) μm .

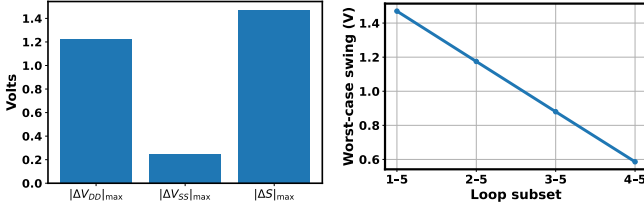


Fig. 9. Modeled rail disturbance at the victim tap. The bars correspond to $|\Delta V_{DD}|_{\max}$, $|\Delta V_{SS}|_{\max}$ and the combined worst-case differential swing $|\Delta S|_{\max}$. (Right) Worst-case differential swing for loop subsets (1–5, 2–5, 3–5 and 4–5), showing the swing decreases as larger PDN loops are removed.

As shown in Fig. 10(left), a pre-silicon model evaluates the worst rail disturbance at a register tap by resolving contributions from the VDD and VSS rails and computing the combined differential swing. With a 1 mm probe height, 10 V injection and 4 ns pulse (10% rise time) the model predicts $|\Delta V_{DD}|_{\max} \approx 1.23 \text{ V}$ and $|\Delta V_{SS}|_{\max} \approx 0.24 \text{ V}$, giving $|\Delta S|_{\max} \approx 1.47 \text{ V}$ —a level sufficient to disrupt the noise margin and potentially cause a bit-clear fault in the register [8]. A pre-silicon analysis that omits package parasitics, process variability and high-frequency resonances may deviate from empirical observations. A more detailed study of model parameters is left for future work.

The loop-subset analysis in Fig. 10(right) shows that the worst-case swing falls monotonically from about 1.47 V with all five loops to roughly 1.17 V, 0.88 V and 0.59 V for the 2–5, 3–5 and 4–5 loop subsets, confirming that eliminating larger PDN loops reduces the induced disturbance. In a METAL6 grid with a 70 μm pitch, adjacent 70 $\mu\text{m} \times 70 \mu\text{m}$ loops largely cancel because they carry equal and opposite induced currents, so the outermost loops dominate the disturbance; increasing mesh density therefore elevates the central path resistance and channels induced currents to the periphery.

g) Bridging pre- and post-silicon results(and limitation).: The top 20% predicted region contains 29 of 41 (70.6%) scan-localized first flips (binomial $p = 9.2 \times 10^{-6}$), showing that first flips cluster in areas the model ranks as most susceptible. *Note:* The susceptibility heatmap reflects only the predicted worst-case supply swing $|\Delta S|_{\text{worst}}$. The measured first-flip density

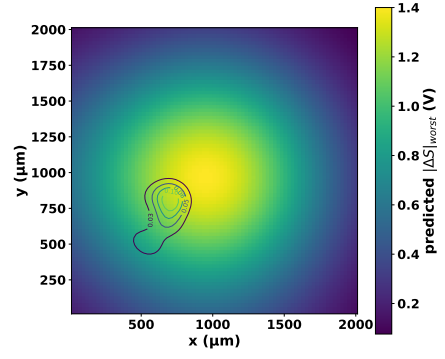


Fig. 10. Predicted susceptibility heatmap ($|\Delta S|_{\text{worst}}$) with scan-derived first-flip density contours.

depends on both the disturbance and the spatial distribution of sequential elements.

V. Conclusion

We presented a scan-chain-assisted EMFI study on the CAPRI1 RISC-V SoC. Full-scan visibility enables cycle-level localization of first-flip events and tracking of fault propagation across Ibex microarchitectural blocks. Across VP0 and VP7, most identified fault origins occur in the instruction-fetch stage, with LSU and decode/execute as secondary sources. To link these architectural effects to physical coupling, we derived a PDN-loop-based susceptibility metric from routed DEF data and used a worst-case supply-swing bound to rank probe locations. The resulting susceptibility map aligns with scan-localized first-flip clustering, providing a practical scan-to-layout workflow to identify EMFI-vulnerable structures and guide early hardening of security-critical logic.

References

- [1] S. Ordas, L. Guillaume-Sage, and P. Maurine, “Electromagnetic fault injection: The curse of flip-flops,” *J. Cryptogr. Eng.*, 2017.
- [2] R. Boulifa, Y. Sun, G. Di Natale, and P. Maistri, “Real-time fault analysis in risc-v processors: A case study using the internal state monitor,” in *2025 IEEE 31st International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2025, pp. 1–7.
- [3] T. Dullien, “Weird machines, exploitability, and provable unexploitability,” *IEEE Transactions on Emerging Topics in Computing*, 2020.
- [4] P. Sauter, T. Benz, P. Scheffler, H. Pochert, L. Wüthrich, M. Povišer, B. Muheim, F. K. Gürkaynak, and L. Benini, “Croc: An end-to-end open-source extensible risc-v mcu platform to democratize silicon,” 2025. <https://arxiv.org/abs/2502.05090>
- [5] L. Dureuil, G. Petiot, M. Potet, T. Le, A. Crohen, and P. de Choudens, “FISSC: A fault injection and simulation secure collection,” in *35th International Conference, SAFECOMP 2016*.
- [6] M. Ghodrati, B. Yuce, S. Gujar, C. Deshpande, L. Nazhandali, and P. Schaumont, “Inducing local timing fault through em injection,” in *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*. <https://doi.org/10.1109/DAC.2018.8465836>
- [7] Z. Liu, D. Shanmugam, and P. Schaumont, “Faultdetective: Explainable to a fault, from the design layout to the software,” *IACR Trans. Cryptographic Hardware and Embedded Systems*, 2024. <https://tches.iacr.org/index.php/TCHES/article/view/11804>
- [8] M. Dumont, M. Lisart, and P. Maurine, “Modeling and simulating electromagnetic fault injection,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2021.
- [9] A. Ghosh, M. Nath, D. Das, S. Ghosh, and S. Sen, “Electromagnetic analysis of integrated on-chip sensing loop for side-channel and fault-injection attack detection,” *IEEE Microwave and Wireless Components Letters*, 2022.
- [10] EM GeoSci. (2018) Faraday’s law in the frequency-domain. https://em.geosci.xyz/content/maxwell1_fundamentals/formative_laws/faraday.html#faraday-s-law-in-the-frequency-domain