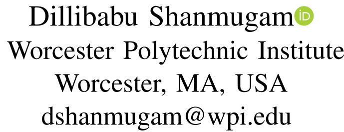
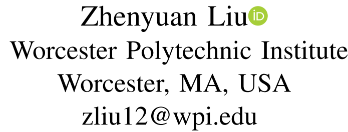
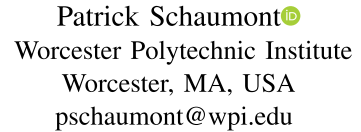
CAPRI6: A Solution for Fault Root Cause Detection


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*—Fault injection is a well-known threat to hardware security. However, the behavior of a digital system under fault injection is poorly understood. It is hard and often impossible to precisely observe the onset of the fault, and the outcome of a fault injection is only observed many clock cycles after fault injection. In the case of a microcontroller core, this leads to poor understanding of fault effects and, consequently, ineffective fault countermeasures. To better understand the fault onset in a microcontroller core, we present `CAPRI6`, an SoC ASIC with six openMSP430 microcontroller cores coupled in lockstep execution. `CAPRI6` supports full-state observation of all cores using a scan chain. We show how the combination of redundancy and scan chain leads to a precise understanding and characterization of fault injection effects. We present practical results for clock-glitching and electromagnetic fault injection (EMFI) on the `CAPRI6` chip, and we validate these measurements against pre-silicon design data (timing and place-and-route results). We propose the use of a scan chain as a practical tool in the analysis of fault injection effects in complex digital systems.**

***Index Terms*—Fault Injection Attacks, Redundant Core Architecture, Scan chain analysis, Sane vs Weird Machine**

I. Introduction

Fault injection is a well-known threat to hardware security, and numerous recent attacks have shown that even commercially deployed systems remain highly susceptible [1]. The asymmetry between exploiting faults and defending against them stems from the fact that the adversary does not need a detailed understanding of the fault onset: only the fault outcome matters. However, from the viewpoint of the designer, effective handling of faults is crucial and faults have to be detected and squashed as early as possible. This is very challenging. Fault injection techniques such as clock-glitching, voltage-glitching and EMFI introduce bit-flips at the lowest hardware level, at a location that depends on the fault injection parameters as well as on low-level design properties including timing, cell placement, and interconnect layout. Furthermore, fault propagation is poorly understood, especially in a multi-layered design that includes software and hardware. The software community has used the term *weird machine* (as opposed to sane machine) to describe the behavior of a software program subject to hardware faults [2]. Indeed, since hardware faults can affect a processor's micro-architecture, even the behavior of individual instructions becomes unpredictable.

In this contribution, we tackle one of the key questions that prevent a deeper analysis of the problem of fault attacks: How can we measure fault injection? Our proposal is to use a scan chain as illustrated in Figure 1. We have implemented a

979-8-3315-3689-3/25/$31.00 ©2025 IEEE

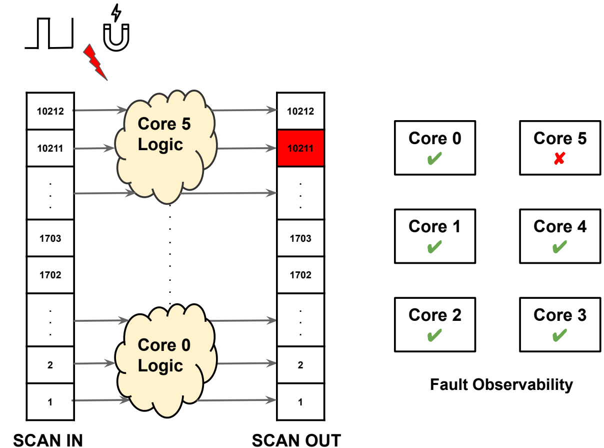


Fig. 1. `CAPRI6`: A six-core openMSP430 microcontroller featuring scan logic for fault observability

six-fold redundant microcontroller design (openMSP430 [3]) with scan-logic. The six-fold redundant implementation is used to easily parallelize fault injection experiments, as each fault injection can affect six cores simultaneously. Each core is identical, except for variations in the layout (place-and-route), and due to process manufacturing. To study fault effects, first an initial state is scanned into each of the six cores. Next, a fault is injected when the cores execute a single clock cycle. Finally, the scan chain is extracted and analyzed. Since all cores perform the same operation, faults can be easily detected by comparing the scan chain contents. Furthermore, since the cores run for exactly one clock cycle, there is no ambiguity on the root cause of any faults identified. The main objective of this paper is to demonstrate that this technique is generic and can characterize faults regardless of the injection source. We demonstrate fault analysis for clock glitching as well as for EMFI. Furthermore, we also show that our empirical data of observed fault response matches the chip design data. Since scan chains are a well understood and widely used test structure, the proposed technique can be applied to a wide variety of designs. Because of the experimental nature of this work, a detailed analysis of the fault's impact on software execution is out of scope of this paper. We refer to other recent work that investigates this question [4].

Recent studies have explored models involving fault adversaries [5]. Techniques for k-fault-resistant partitioning were demonstrated using OpenTitan [6]. Our redundant and scan chain approach to root cause analysis distinguishes itself through its novel methodology for fault root cause analysis in a multicore SoC ASIC environment. The proposed method

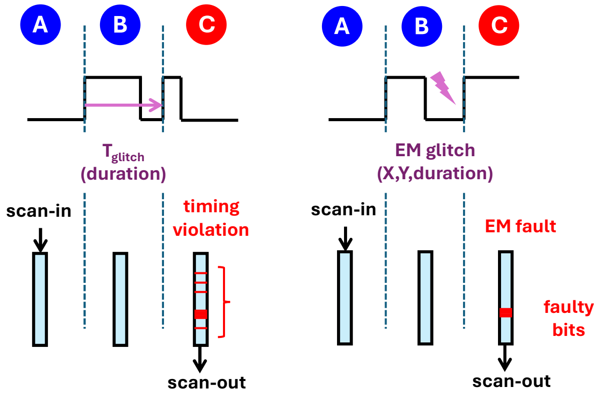


Fig. 2. (Left) Clock glitch fault injection methodology; (Right) EMFI fault injection methodology.

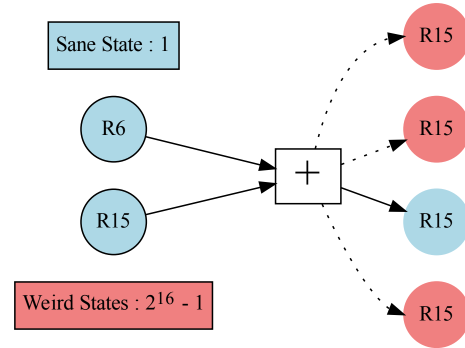


Fig. 3. Illustration of sane and weird machine behavior: When R6 and R15 are added, the expected result is R15 (sane state). Any deviation results in one of the possible (weird) states, demonstrating the impact of fault-induced anomalies on the state transitions.

has potential application in various processor architectures. As a proof of concept, the methodology is explored in the ASIC.

Building upon the utilization of STA and layout analysis [7] for identifying faults, this research demonstrates their implementation in validating the root cause analysis.

The contributions of our work are as follows.

- We present a method to measure the immediate impact of fault injection using a scan chain. We demonstrate this method on a test-chip containing six identical open-MSP430 cores.
- We characterize each core's response on EMFI and clock glitching for a selected number of test instructions (scan configurations), representative of typical software activity.
- We validate the measured fault response against the pre-silicon design data of the same chip and show correlation based on well understood fault response effects. Clock glitches cause timing violations, and EMFI can cause local faults provided that the EM pulse is directed to a specific region of the chip.

The remainder of the paper is structured as follows. Section 2 describes the methodology and validation of root cause, Section 3 elaborates `CAPRI6` architecture and experimental setup. Section 4 and 5 share experimental results of clock glitching and EMFI respectively. Section 6 concludes with key findings and suggestions for enhancing fault resilience.

II. Methodology

The proposed methodology leverages the sane versus weird machine framework to enable precise root cause analysis for fault detection. A sane machine represents the expected system behavior, transitioning only between known states as defined by the design. In contrast, a weird machine arises when faults cause transitions to unknown or unintended states, leading to unpredictable system behavior. This distinction is pivotal for diagnosing fault-induced anomalies and understanding their propagation.

A. Basic principles of scan chain analysis

Figure 2 illustrates two fault injection methodologies: Clock glitch attack and Electromagnetic Fault Injection (EMFI). In a Clock Glitch Attack, the process is initiated with the configuration of registers through the scan-in interface (A). Subsequently, a clock glitch is introduced by varying the glitch duration (T_{glitch}) from 0 to 10 ns (B), inducing a timing violation in the system. Finally, the registers are scanned (C) to observe and analyze the impact of the glitch, identifying any deviations or errors resulting from the timing violation.

Similarly, in an EMFI Attack, the registers are initially configured through the scan-in interface (A). An electromagnetic pulse is then applied to the target circuit by varying parameters, such as the pulse duration and spatial location(XY coordinates) (B). This induces faults in registers and disrupts normal operations. The state of the registers is subsequently extracted through the scan-out interface (C) to detect and analyze any faulty bits caused by the EM pulse. Both methodologies emphasize the identification of system vulnerabilities through the systematic injection of faults and observation of their effects on registers.

To enhance the observability and controllability of state transitions, this methodology employs a single clock cycle scan chain technique. This technique captures the state of the system after each clock cycle and provides detailed insights into state transitions. If a transition occurs from a known state to an expected state, the system remains in the sane machine. However, transitions to unknown states indicate weird machine behavior and potential faults. An example is illustrated in the figure 3. By identifying the specific clock cycle and state responsible for the deviation, this technique enables precise root cause identification.

The methodology is particularly effective under stressful conditions, such as clock glitches [8] or thermal-induced faults, where the system behavior can deviate unpredictably. Real-time monitoring of the scan chain allows the system to detect faults early and to analyze their progression. By addressing faults at their origin, the methodology enhances the robustness of the system, reduces fault recurrence, and improves the overall reliability and security.

B. Validation of Root Cause Analysis

Validation of the root cause analysis integrates static timing analysis (STA) and electromagnetic fault injection (EMFI) layout correlation to ensure comprehensive accuracy. In STA, validation focuses on the signal slack time at flip-flops relative to the clock edge during state transitions. Clock glitches, which

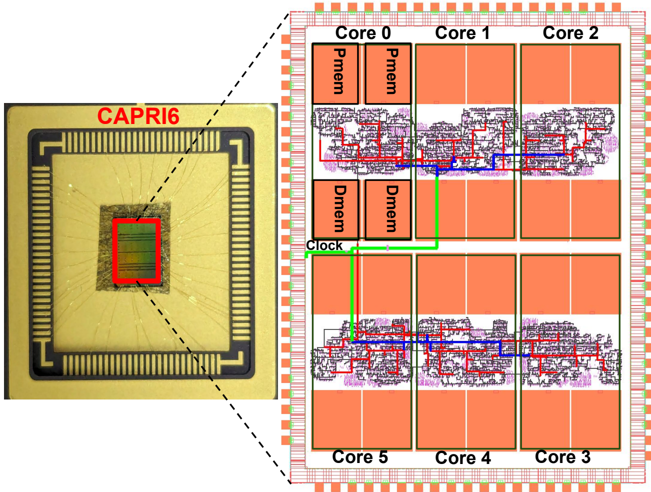


Fig. 4. The CAPRI6 ASIC architecture die photograph and floorplan reveal six independently routed cores (Core 0 to Core 5). Clock tree distribution across cores in the system. Each core's clock characteristics are represented, highlighting the variations that could affect synchronization and performance under stress experiments

reduce the clock period, are analyzed to determine whether the signals arrive too late or too early, resulting in incorrect state transitions or metastable conditions. Correlating scan chain outputs with these timing violations ensures that the observed faults align with timing-induced vulnerabilities.

Additionally, the validation incorporates spatial and temporal layout correlations during the EMFI. EMFI introduces localized disruptions in specific hardware regions by targeting temporally sensitive or spatially dense areas [9]. The scan chain captures the resulting faults and their locations are mapped to the chip layout. This methodology evaluates the physical design weaknesses that contribute to faults by examining whether the observed fault occurs in areas with high routing congestion or exposed surfaces. By combining STA and EMFI correlations, the proposed methodology ensures a robust and reliable validation process. This approach links observed anomalies to timing and physical design characteristics, enhancing the accuracy and reliability of fault diagnostics while enabling informed design improvements. Having established the analysis framework, we next discuss the implementation of this approach on our custom six-core ASIC, CAPRI6, and the experimental setup for fault injection.

III. CAPRI6 and Experiment Setup

A. CAPRI6 Architecture

The CAPRI6 ASIC is a 180nm, 12 sqmm. SoC that consists of six identical MPS430 cores. Each core has a unique layout pattern in the chip as shown in the figure 4. Each core has dedicated 2 KB program and data memory for software application. The core programming interface uses I2C. Through GPIO, an on-chip communication network between cores is created. Each core can halt the SoC upon local fault detection and notify the other cores through a redundant fault communication path. Next, the full state of registers across cores can be accessed through the scan chain for the fault root cause analysis. Each core includes on-chip (glitch) detectors

TABLE I
Gate count distribution: Minor variations can still influence timing, as illustrated in Fig. 5.

CoreID	openMSP430	GPIO	TimerA	Sensors
0	4693	1547	875	1416
1	4698	1448	881	1421
2	4693	1352	878	1417
3	4659	1472	880	1414
4	4699	1458	881	1413
5	4699	1353	880	1431

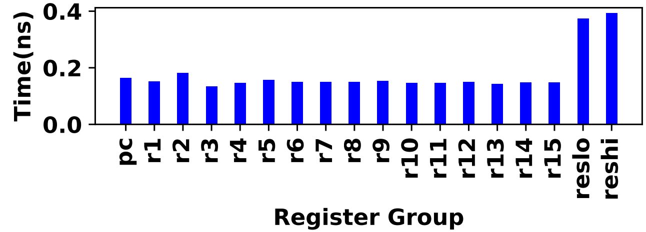


Fig. 5. Standard deviation in slack time(ns) across different system registers. The x-axis groups register, y-axis measures the timing variability across the six cores, highlighting the layout-level disparities

(three ring oscillators of 5 stages and one of 11 stages) to monitor voltage drops and timing glitches. In addition, each core has a timer and a hardware multiplier module to facilitate computations.

The CAPRI6 ASIC is designed to operate at a clock rate of 50 MHz. Synopsys Design Compiler was utilized for front-end synthesis. Back-end implementation, including placement and routing, was executed with the Synopsys ICC2 Compiler, while Mentor Graphics Calibre conducted design verification and sign-off to ensure manufacturability.

a) Placement and routing variation: The clock tree distribution layout Figure 4 illustrates the differences in clock routing across the six cores. Table I presents the variations in the gate count for different modules (openMSP430, GPIO, TimerA, and Sensors) within each core, which result from variations in placement and routing during the physical design process. These disparities may affect the timing performance and fault behavior. For example, in Figure 5 the standard deviation of the register slack time (ns) could potentially reduce the timing margins. Furthermore, these layout-induced variations could intensify synchronization challenges, particularly under stress conditions such as EM pulses or fault injection scenarios.

b) Compatible with CW308: We integrated the CAPRI6 chip as an add-on card on the ChipWhisperer CW308 [10] board, ensuring that our fault and side-channel analyses remain compatible with the ChipWhisperer platform. For root-cause analysis experiments which require scan chain access, we used a measurement setup as depicted in Figure 6.

Raspberry Pi (RPI) GPIOs are configured as scan chain read or write, and as program interface. CW305 generates a glitch upon a handshake signal from the RPI.

B. Experimental setup

a) Clock glitch measurement setup: The clock glitch measurement setup comprises multiple components that operate in coordination. The setup is schematically represented

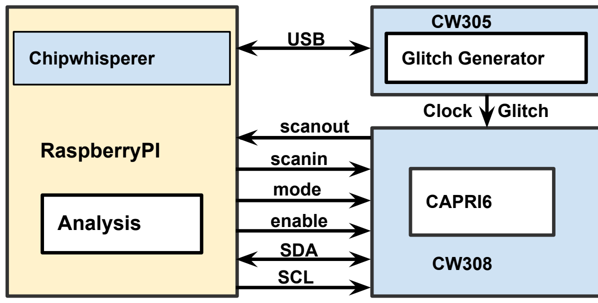


Fig. 6. Schematic diagram of the CAPRI6 (ASIC) mounted on the CW308 main board, interfacing with the CW305 FPGA for clock glitch generation and a Raspberry Pi for scan chain control.

by CAPRI6, CW305, and Raspberry Pi, which are physically interconnected for synchronization in the lower parts of the figure 6. Control signals, including scanin, scanout, clock, and glitch, were managed through these connections to ensure precise measurements. This arrangement facilitates detailed fault analysis by isolating and evaluating the effects of clock glitches on the specific components of CAPRI6. The ChipWhisperer platform interfaces with a Raspberry Pi for configuring CW305 FPGA board to achieve desired glitch width injection. CAPRI6 serves as the target device and interfaces with the CW305. It is controlled by the Raspberry Pi through scan chains. The scan chain is utilized to configure the internal state of the CAPRI6 registers in scan mode prior to transitioning to execution mode for a single clock cycle. During this execution cycle, the CW305 board injected a clock glitch of a predetermined width. The scan chain captures the impact of the glitch on specific instruction-related registers.

b) EMFI measurement setup: EMFI measurement campaigns used the same setup as the Spider EMFI pulse generator. The chip was systematically scanned across 20 rows and 15 columns, with pulse durations ranging from 4 ns to 25 ns and voltages varying from 1.8 V to -4.0 V. EMFI pulses were introduced during a single clock cycle, and the scan chain captured the faults in the key registers. A Raspberry Pi controlled the experimental setup, whereas the Spider's manipulator ensured precise control of the XY coordinates for fault injection. This configuration was designed to analyze electromagnetic vulnerabilities and identify susceptible hotspots of the chip.

IV. Clock Glitch Analysis

This section describes experiments that analyze fault behavior for three instructions: MOV transfers data from register R6 to R15; ADD sums registers R6 and R15; and MUL performs multiplication using the multiplier peripheral module.

a) MOV r6, r15: The instruction "Mov r6, r15" (0x460f) is a two-operand arithmetic operation completed in one clock cycle during software execution. During execution, we assessed system performance by intentionally introducing timing glitches ranging from 1.44 ns to 30 ns and observing the results with a scanout. In this example, the MOV instruction transfers 0xFFFF from R6 to R15. Figure 7 illustrates that Core 1 experiences the highest fault impact across glitch

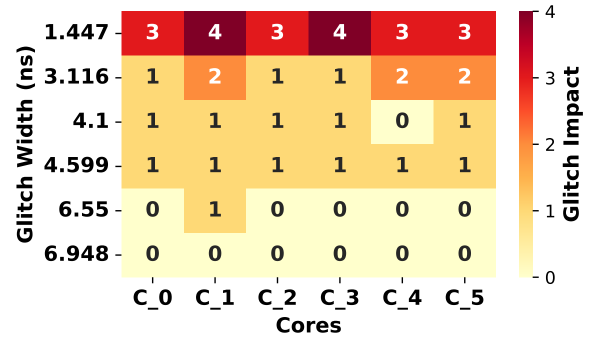


Fig. 7. MOV instruction : High glitch impacts at narrower glitch widths

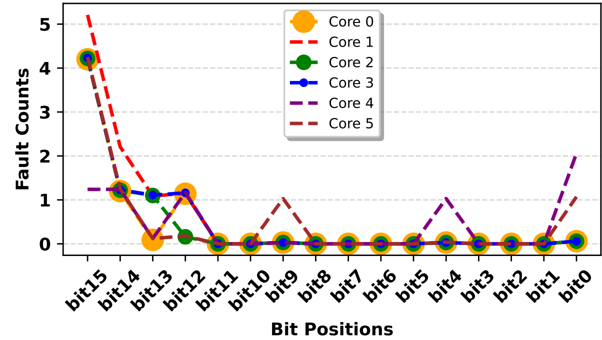


Fig. 8. MOV: R15 fault counts peak in bits 15–13

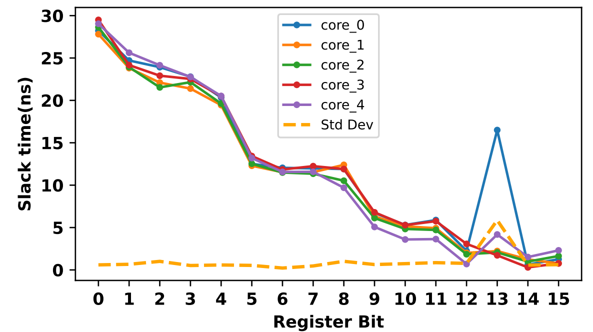


Fig. 9. STA slack analysis for R15 indicates that bits 12–15 have minimal timing margins, highlighting their criticality.

widths, with a maximum of four faults at 1.4 ns glitch width, identifying it as the most susceptible core. The origin of these faults is further localized using Figure 8 (fault counts vs. bit positions), which reveals that faults predominantly occur at bit position 15 in Core 1. Figure 8 also reveals a common pattern in which faults primarily occur in MSB (boundary) bits, attributed to inherent timing vulnerabilities. Figure 9 (STA analysis) confirms that the most significant bits (in particular, bit 15) of R15 have the least timing slack, making them most prone to timing faults. These factors likely contributed to the timing violations observed during the glitch conditions. The correlation between the STA findings and experimental outcomes supports the conclusion that a timing constraint at bit 15 is the root cause, indicating the need for focused design enhancements to improve system robustness.

b) ADD r6, r15: The instructions ADD R6 and R15 retrieve the value stored in R6(0xF00F), combine it with the existing value in R15(0X0FF0), and store the sum back in

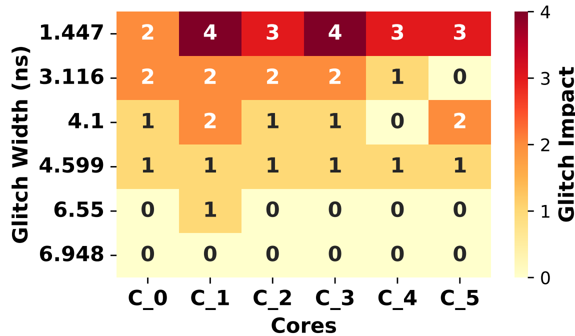


Fig. 10. ADD instruction: Core 1 is most sensitive.

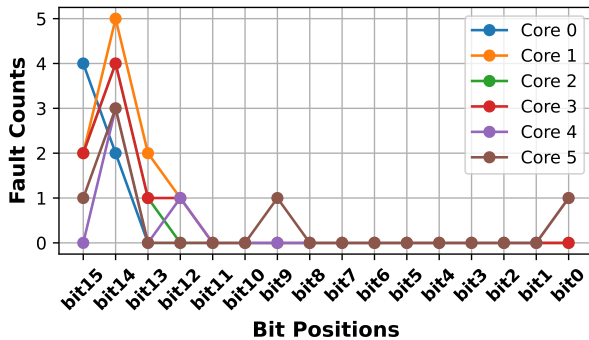


Fig. 11. ADD Inst: Fault distribution across bit positions for the ADD instruction, highlighting higher fault susceptibility in significant bits (bit15 to bit12) across the six cores

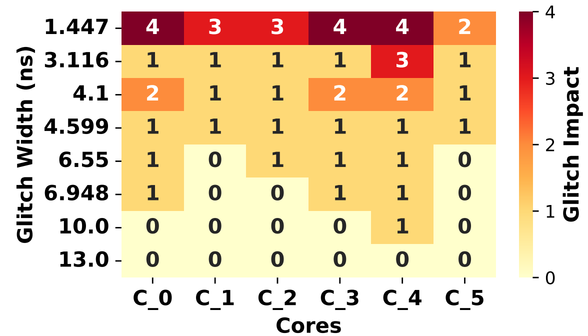


Fig. 12. MULT: Core 4 is more susceptibility at narrow glitch widths.

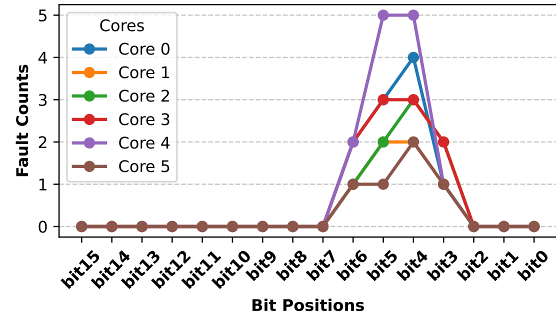


Fig. 13. Fault characterization across bit positions reveals Core 4 as the most vulnerable, with the highest fault counts at bit4 and bit5

R15(0XFFFF). Figure 10 demonstrate that Core 1 exhibits high sensitivity to clock glitch injection across glitch widths ranging from 1.4 ns to 6.5 ns. The fault localization in Figure 11 identifies bit position 14 in Core 1 as the primary source of observed faults. STA confirms that bit 14 in Core 1 experiences timing constraints under stress conditions.

c) MULT op1, op2: When (0×0F00) and (0×0800) are provided as inputs for op1 and op2, respectively to multiplication modules, the resulting outputs are RESHI(0×0078) and RESLO(0×0000), respectively. The experimental findings indicate that Core 4 exhibits a high susceptibility to glitches, as evidenced by the heatmap 12 demonstrating its increased fault vulnerability. A substantial proportion of the multiplier module bits are situated on critical paths, rendering them susceptible to timing issues. Fault detection is contingent upon the specific multiplier and multiplicand operands because varying input combinations activate distinct components of the circuit. Figure 13 fault characterization reveals that RESHI bits 4 and 5 are critical because of their role in the intermediate stage of multiplication carry propagation.

V. EMFI ANALYSIS

Next, we examine the effects of EMFI on CAPRI6, using the same test instructions. This section presents the spatial fault patterns observed and correlates them with the chip layout.
To assess EMFI sensitivity, we varied the pulse durations to 4 ns, 15 ns, and 25 ns, and adjusted the voltage levels to 1.8 V and -4 V for each instruction. This approach enabled us to observe the effects of different pulse parameters on the susceptibility of the instructions to EMFI-induced faults.

The MOV experiment demonstrates a strong correlation between R15 layout coordinates and the EMFI heatmap, as shown in the top side of the Figure 14. This relationship highlights the influence of specific layout features on glitch sensitivity. Figure 15 shows that the EMFI localization effects are more pronounced on bit 15 in Core 0 and bit 11 in Core 1, making them particularly fault-sensitive compared with other bits. Additionally, the figure indicates that Core 4 has six distinct fault positions, underscoring its higher vulnerability to the EMFI. The heightened sensitivity observed can be attributed to the ability of pulses to traverse metal layers and power delivery network (PDN) design components, thereby inducing both spatial and temporal susceptibilities in Core 4. Figure 14a overlays the EMFI fault heatmap(top right) on the chip layout(top left) for the MOV instruction (showing the clusters of faults, notably in Core 4's region), while Figure 14b shows that the ADD(bottom left) and MULT(bottom right) instructions produce similar fault-location patterns. This consistency across instructions reinforces that certain physical regions (Core 4's region has more metal layers exposed) are inherently more EMFI-sensitive. The results underscore the main idea of the study, which is fault root cause identification through precise EMFI localization. These insights pave the way for targeted design interventions to improve the fault resilience in critical chip regions.

Figure 15 demonstrates a strong correlation between pulse duration and fault count, with longer durations yielding increased faults across all instructions. Moreover, faults induced at –4.0 V consistently exhibit greater impact than those at 1.8 V, indicating that negative voltage pulses couple more effectively into transistor wells and increase bit flips.

The study evaluated clock glitch widths from 1 ns to 20 ns

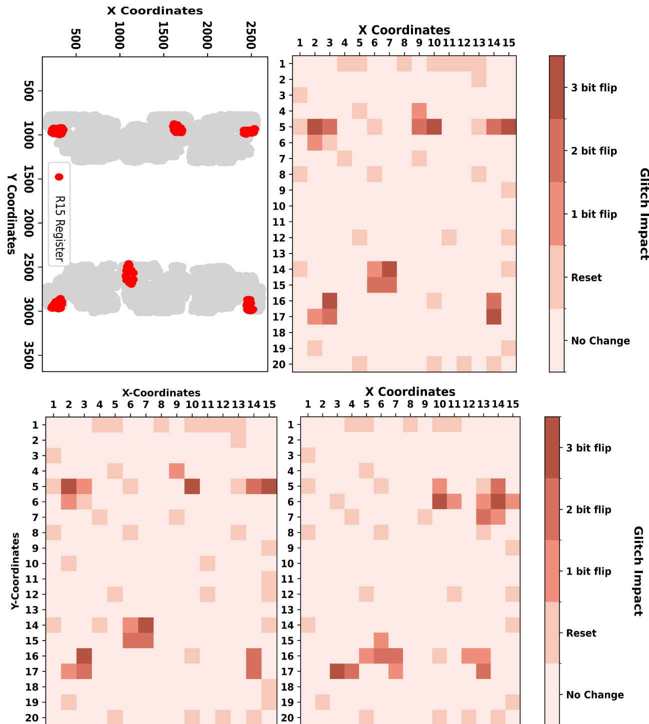


Fig. 14. EMFI-induced fault mappings onto the die layout identify R15 failure hotspots during MOV operations. Comparable spatial distributions in ADD and MULT heatmaps confirm persistent layout-dependent vulnerability.

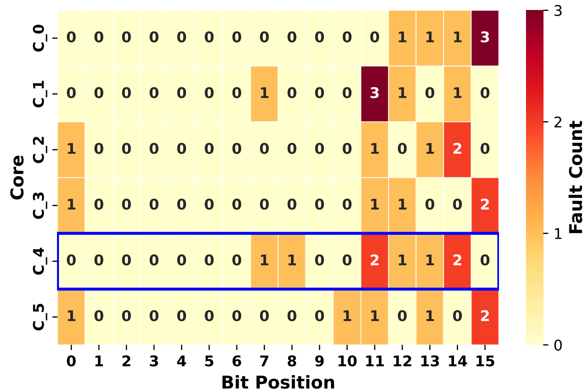


Fig. 15. EMFI Sensitive Bits for MOV Instruction: Bit 15 for Core 0 and bit 11 for Core 1 shows high sensitivity. Core 4 exhibits a six vulnerable locations due to reduced protective shielding.

and found that pulse widths over 10 ns produced minimal impact, likely due to the system's maximum clock frequency. Similarly, EMFI experiments covered pulse durations from 1 ns to 25 ns with effective voltages between –4 V and 1 V, parameters determined by the resolution of the test apparatus (the Spider EMFI pulse generator). These boundaries establish the practical limits for fault injection and provide a foundation for future research to refine these parameters for more precise fault characterization across different design configurations.

VI. Conclusion

CAPRI6, a six-core redundant microcontroller design, is proposed as a platform for fault root-cause analysis. By leveraging scan chain-based fault observability and redundancy, we demonstrated the precise localization and characterization of faults induced by clock glitching and EMFI. The experimental results validated our approach by correlating the observed faults with pre-silicon design data and highlighting layout-level vulnerabilities and timing constraints. This methodology provides actionable insights for designing fault-resilient systems, thus paving the way for enhanced security in critical hardware applications. Future work may extend CAPRI6 insights to other architectures, refine fault countermeasures, and investigate machine-learning methods for efficient scan-chain observability classification.

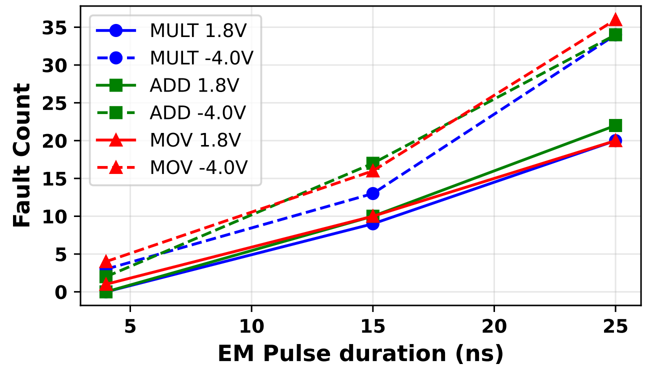


Fig. 16. EMFI pulse duration vs number of faults observed. Longer pulse durations cause more faults up to a certain limit, and negative polarity pulses (-4.0 V) consistently induce more faults than positive pulses (1.8 V)

Acknowledgments

This research was supported in part by NSF Award 2219810

References

[1] O. Bittner, T. Krachenfels, A. Galauner, and J.-P. Seifert, "The forgotten threat of voltage glitching: A case study on nvidia tegra x2 socs," in *2021 Workshop on Fault Detection and Tolerance in Cryptography (FDTC)*, 2021, pp. 86–97.

[2] T. Dullien, "Weird machines, exploitability, and provable unexploitability," *IEEE Trans. Emerg. Top. Comput.*, vol. 8, no. 2, pp. 391–403, 2020. [Online]. Available: https://doi.org/10.1109/TETC.2017.2785299

[3] O. Girard. (2024) openmsp430. Accessed: 2024-12-13. [Online]. Available: https://opencores.org/projects/openmsp430

[4] Z. Liu, D. Shanmugam, and P. Schaumont, "Faultdetective explainable to a fault, from the design layout to the software," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2024, no. 4, pp. 610–632, 2024. [Online]. Available: https://doi.org/10.46586/tches.v2024.i4.610-632

[5] J. Richter-Brockmann, P. Sasdrich, and T. Güneysu, "Revisiting fault adversary models–hardware faults in theory and practice," *IEEE Transactions on Computers*, vol. 72, no. 2, pp. 572–585, 2022.

[6] S. Tollec, V. Hadži ´c, P. Nasahl, M. Asavoae, R. Bloem, D. Couroussé, K. Heydemann, M. Jan, and S. Mangard, "Fault-resistant partitioning of secure cpus for system co-verification against faults," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2024, no. 4, p. 179–204, Sep. 2024. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/11788

[7] V. Bansal, O. A. Mohamed, and F. Ghaffari, "Layout-based reliability analysis of openmsp430 register file under external radiations," in *2023 International Conference on Microelectronics (ICM)*, 2023, pp. 294–297.

[8] A. Marotta, R. Lashermes, G. Bouffard, O. Sentieys, and R. Dafali, "Characterizing and modeling synchronous clock-glitch fault injection," in *International Workshop on Constructive Side-Channel Analysis and Secure Design*. Springer, 2024, pp. 3–21.

[9] M. Dumont, M. Lisart, and P. Maurine, "Electromagnetic fault injection: How faults occur," in *2019 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*. IEEE, 2019, pp. 9–16.

[10] C. O'Flynn. Chipwhisperer cw308 ufo target documentation. Accessed: 2024-12-12. [Online]. Available: https://rtfm.newae.com/Targets/CW308%20UFO/