Power PUFs: Strengthening SRAM PUFs Against Fault Injection on Low-Cost IoT Devices

Abhinav Komanduri
Electrical Engineering and Computer Science
University of Arkansas
Fayetteville, Arkansas
Email: abhinavk@uark.edu

Alexander H. Nelson
Electrical Engineering and Computer Science
University of Arkansas
Fayetteville, Arkansas
Email: ahnelson@uark.edu

***Abstract*—Physically Unclonable Functions (PUFs) are emerging lightweight hardware security primitives that utilize inherent manufacturing variations in integrated circuits to generate unique and reproducible digital fingerprints for authorization or authentication. This paper explores the implementation and evaluation of SRAM-based PUFs on the ESP32 microcontroller under normal and fault-injected operating conditions. A baseline PUF generation method is done, and its reliability and stability are evaluated using the intra-hamming distance. Fault injection attacks using voltage anomalies are employed to study the effects on the PUF's reliability. Preliminary results show that while the PUF is stable under normal operating conditions, the reliability significantly decreases under faulted conditions. The results showcase the vulnerability of SRAM PUFs to environmental and operational factors and lay the groundwork for exploring additional fault injection methods and countermeasures for enhanced robustness.**

Keywords - Physically Unclonable Functions, SRAM, Fault Injections, IoT devices, Hardware Security

I. Introduction

The Internet of Things (IoT) is at the center of technology and connectivity and has transformed sectors like healthcare. Today, over 160 million IoT devices enhance patient care, streamline operations, and provide data to care teams [1]. However, this revolution comes with the challenge of securing devices and sensors, which often have vulnerabilities and limited resources. In healthcare, where patient confidentiality and data security are crucial, ensuring robust data protection is an essential area of research.

Many strides are being made to protect sensor and IoT data, mainly using memory-based Physically Unclonable Functions (PUFs). By exploiting unique silicon variations inherent in the manufacturing process of integrated circuits, PUFs produce unpredictable bit patterns in response to specific power-up states, forming the challenge-response pair (CRPs) [2]. These CRPs act as distinctive device hardware fingerprints, enabling secure authentication and key generation. The inherent uniqueness and randomness of PUF responses enhance data protection, making them a robust solution for safeguarding sensitive information. While their physical properties make them resilient to software-based attacks, the reliance on hardware interactions for CRP generation leaves them vulnerable to targeted hardware attacks [3].

Among various PUF types, static random access memory (SRAM) PUFs utilize the random startup values of memory in an integrated circuit (IC). SRAM PUFs are particularly appealing for IoT devices due to their simplicity and integration into existing memory structures. Despite these advantages, SRAM PUFs are not impervious to attack. Fault injection attacks, such as voltage and clock glitching, can destabilize hardware behavior, potentially undermining the reliability and uniqueness of PUFs. While fault injection effects on RAM have been studied, their specific implications for PUF-based security still need to be explored, especially in the context of low-cost IoT devices. This gap poses a critical challenge, as compromised PUF responses could jeopardize authentication mechanisms in pervasive IoT environments.

We aim to address this gap by investigating how fault injection attacks influence the core metrics of SRAM PUFs, including reliability, uniqueness, and stability, within the constraints of IoT devices. The overarching goal is to use controlled fault injection techniques and experimental evaluations on commercially available IoT platforms. We seek to uncover vulnerabilities and propose potential mitigation strategies. In this work, we present our preliminary results by showing the impact of the crowbar fault injection on an SRAM PUF with the ESP32-DOWDQ6 acting as the device under test (DUT).

This paper is organized as follows: Section 2 reviews related work on SRAM fault injections and PUF security. Section 3 outlines the experimental methodology and evaluation metrics. Section 4 presents preliminary results, and Section 5 discusses insights and future directions. Finally, Section 6 concludes the paper.

II. Related Works

Dreyer et al. introduced a novel method to leverage SRAM PUFs on the ESP32 microcontroller, demonstrating that robust PUF implementations can be achieved without additional hardware. Using the ESP32's native SRAM, a PUF challenge was extracted using a Raspberry Pi 2 via serial communication. Their proof-of-concept application implemented authentication and authorization, where the Authentication Server obtained the PUF challenge via a QR code and verified the ESP32's identity through a challenge-response protocol. This approach

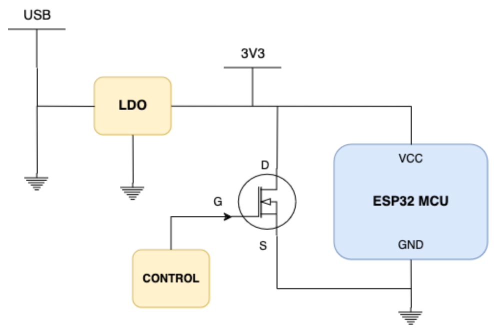

Fig. 1. Crowbar fault circuit with MOSFET to short 3V3 power supply to ground.

highlights the potential for low-cost, hardware-efficient PUF applications on commodity microcontrollers [4].

Complementing this work, Roelke et al. investigated the impact of wearout effects on the stability of SRAM-based PUFs. They conducted stress tests involving elevated temperatures and voltages to degrade SRAM fingerprints. Over time, these stressors permanently eroded the PUF's reliability, rendering it unusable for security applications. Their findings emphasize that physical degradation presents a unique threat to the long-term viability of SRAM PUFs, necessitating strategies to mitigate such wearout-induced vulnerabilities [5].

Additionally, Delvaux, et al. explored fault injection vulnerabilities in the ESP32 V3 chip using electromagnetic glitches. Their attack demonstrated bypassing critical security features like Secure Boot and Flash Encryption. By injecting a single electromagnetic glitch, they manipulated the Program Counter register to force the chip into ROM's Download Mode, enabling access to decrypted flash contents. This technique shows the susceptibility of microcontrollers to fault injection attacks [6].

Together, these works present ideas, from innovative PUF applications to vulnerabilities posed by fault injection and physical wearout, showing the capability of ensuring SRAM PUF security in modern embedded systems.

III. Methods

A. PUF Generation and Operating Conditions

Since SRAM PUFs are based on uninitialized SRAM values, the first step was to secure 16 bytes of memory before the bootloader of the ESP32 configured the memory to 0. This process was done by utilizing the `RTC_IRAM_ATTR` attribute in the ESP-IDF framework that executes the extraction of SRAM values before the boot process completes and is saved in a `NO_INIT` variable that keeps the state through boot. This process was repeated for 100 iterations for each DUT to capture the initial state frequency of SRAM cells, enabling a calculation of each bit's probabilistic state (0 or 1) that is used to create a stable PUF identifier (ID). This initial extraction stage is called *factory boot*.

The ID generated, however, cannot be stored on the device as that gives way to data leakage and loss. Hamming error correction enabled the PUF ID to be used for authentication or authorization purposes and maintain its integrity. Five parity bits for each byte of data (four data bits and one overall parity) were stored in the non-volatile storage (NVS) of the ESP32. On each subsequent boot, the parity bits of the extracted raw SRAM data are compared to the factory PUF bits in NVS. The single-error correction and double error-detection (SECDED) hamming correction scheme is applied for each byte of data. If any double bit-flips are detected, a reboot is triggered. There is always the possibility of a multi-bit flip in which case the SHA-256 hash of the raw data is compared to the hash of the factory PUF which is also stored in NVS. Due to the SRAM's entropic nature, this process ensures that only the helper data remains vulnerable while the PUF ID remains secure and non-reconstructible. The experiments were conducted under two operating conditions to evaluate the reliability and resilience of the SRAM PUF implementation. Under normal operating conditions, the device was powered at a stable supply voltage of 3.3 V, and the ambient temperature was maintained at 25°C, representing typical IoT deployment environments. During this state, the firmware executes without interruptions, ensuring that the SRAM PUF response reflects its entropy without external interference.

In contrast, a voltage fault injection was introduced on the VDD power line under faulted conditions. This fault was implemented by shorting the VDD line to ground briefly during the power-up sequence. This induced unpredictable behavior in the SRAM, potentially altering the initial power-up state of memory cells.

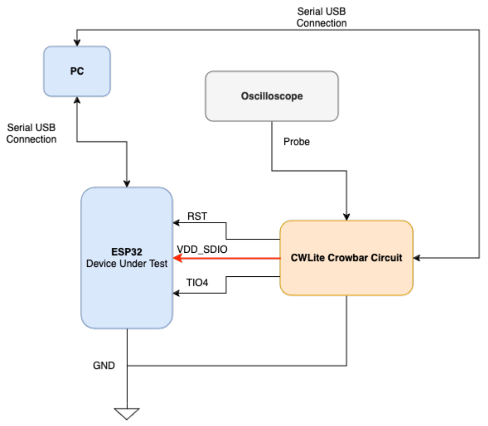

Fig. 2. Experimental setup with ESP32 DUT connected to CW Lite.

B. Target Preparation

To investigate SRAM PUF vulnerabilities under fault conditions, the device under test (DUT) was prepared for controlled voltage glitching attacks. The chosen attack paradigm was the crowbar fault injection attack, which temporarily shorts the power supply of a load to ground for a few nanoseconds with the intention of bypassing the system reset and corrupting internal RAM. This, in turn, will cause the data to be vastly

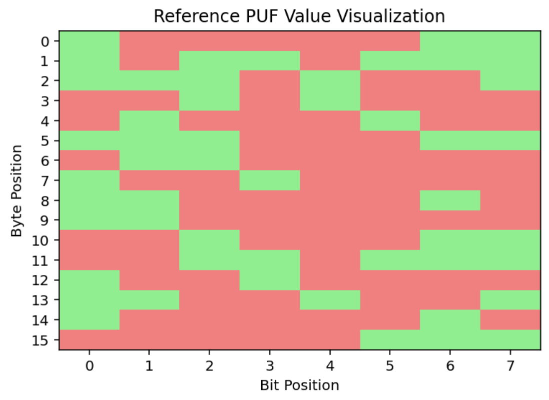

Fig. 3. Visualization of reference PUF value generated where green and red represent bit states 1 or 0, respectively.

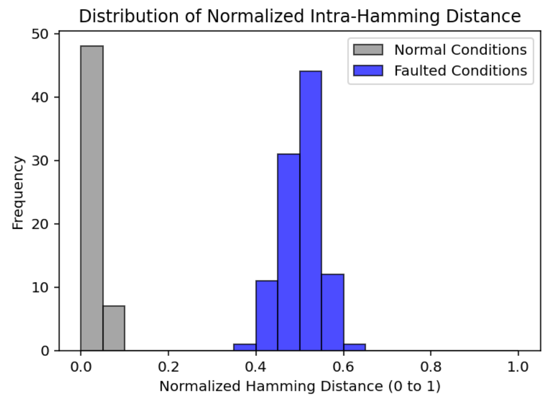

Fig. 4. Distribution of HD_{intra} over 100 PUF iterations.

different and the error correction of the PUF to fail. Figure 1 shows how the crowbar attack works in relation to the DUT.

The DUT was prepared by isolating the power supply trace `VDD_SDIO`, which directly powers the internal memory of the ESP32 [7]. The PCB trace for this pin was carefully cut, and a wire was broken out to a breadboard, allowing for external manipulation of the power supply into the pin.

The ChipWhisperer Lite was employed to perform the crowbar fault injection and was configured to rapidly short the VDD line to the ground for durations in the nanosecond range. This brief power disruption is designed to bypass the DUT's internal power loss protections, exploiting the momentary instability in voltage regulation to induce unpredictable behavior in the SRAM.

Figure 2 shows the experimental setup to apply the fault on the ESP32 DUT from the CW Lite device. The RST line and TIO4 line are connected to the ESP32 and when the signal goes high, the CW initiates a reboot to get the next PUF iteration (of the 100). The SDIO line (in red) was connected via a breadboard as the wire is a "green-wire" on the PCB.

C. Evaluation

This setup enabled precise control over fault injections, allowing reproducible testing of the SRAM PUF's behavior under crowbar fault conditions. The results from these tests provided critical insights into the robustness of the PUF implementation and the effectiveness of error-correcting measures in mitigating fault-induced vulnerabilities.

Metric	α_{NOC}	α_{FOC}
Reliability	0.037	0.468
HD_{intra}	0.963	0.531

TABLE I
AVERAGE INTRA-HAMMING DISTANCE AND RELIABILITY FOR NORMAL AND FAULTED OPERATING CONDITIONS.

This work uses reliability to assess the effectiveness and behavior of the SRAM PUF implementation. This provides insights into the PUF's stability, resistance to environmental variations, and ability to generate distinctive device identifiers [8].

Reliability represents the ability of the PUF to produce the same response consistently under varying environmental conditions. It can be expressed mathematically as

$$\text{Reliability} = 1 - HD_{intra} \tag{1}$$

where HD_{intra} is the intra-Hamming distance defined as

$$HD_{intra} = \frac{1}{k}\sum_{i=1}^{k} HD(R_i(n), R_i'(n)) \tag{2}$$

where k is the total number of bits in a string (in this case 128), with $R_i(n)$ as the baseline PUF response and $R_i'(n)$ as the PUF response under faulted conditions. The intra-hamming distance (HD_{intra}) measures similarity between bit strings, realized by an XOR operation, where a lower value indicates higher similarity. For this work, aiming to undermine SRAM PUF reliability, the ideal result is a reliability approaching zero.

D. Attacker Model

For the attack outlined in this work, we consider adversaries that:

- are in physical proximity of the target device
- are in a non-privileged position and do not require any wireless connection to the device or its software
- are able to inject faults at a coordinated time to trigger repeatable destructive outcomes

IV. Preliminary Results

With the ESP32-DOWDQ6 IC as the DUT, preliminary results were obtained. When the baseline PUF is generated, it is a bit string of 128 bits (16 bytes). To visualize this, Figure 3 shows the reference PUF where each cell is a bit position and a green cell and red cell indicates a bit state of 1 or 0, respectively.

Figure 3 presents the frequency distribution of intra-hamming values calculated from Equation 2 across 100 generated PUFs, comparing responses under normal operating conditions (α_{NOC}) to those under faulted operating conditions (α_{FOC}). The distribution shifts rightward toward the 0.5

mark in faulted conditions, indicating a higher mean intra-hamming distance. This suggests decreased reliability, as the PUF responses become less consistent and less reproducible when the DUT is under voltage anomalies. This behavior aligns with my predictions, confirming that faulted conditions can effectively undermine the stability of the SRAM PUF.

Table I shows the mean intra-hamming distance for each distribution, α_{NOC} and α_{FOC}, and calculates the reliability coefficient using Equation 1. As expected, under normal operating conditions (α_{NOC}), the PUF demonstrates high reliability and reproducibility, with a reliability rating of 0.963. Under faulted conditions (α_{FOC}), the distribution shifts rightward, and the mean intra-hamming distance is 0.468, which yields a reliability of 0.531. This drop proves the impact of voltage anomalies on the PUF's stability.

V. Discussion and Future Work

These preliminary results show that SRAM PUFs, while a viable option for lightweight security, are still vulnerable to system or hardware-level attacks, particularly voltage-glitching attacks. Under normal operating conditions, the generated PUF showed high reliability and stability, as shown by a mean intra-hamming distance of 0.037 and a reliability coefficient of 0.963. The values show that the PUF can produce reproducible identifiers. In contrast, under faulted conditions induced by the crowbar circuit, the reliability significantly dropped to 0.531, shifting the intra-hamming distance distribution towards 0.5. The behavior shows the destabilizing effect and decreased response consistency of voltage anomalies on SRAM-based PUFs

This work sets the groundwork for future studies into SRAM PUF resiliency. Future directions to be explored are varied attack types and alternative PUF architectures.

Crowbar fault injection can cause significant damage but requires physical access. In contrast, electromagnetic (EM) fault injection disrupts circuit operations using electromagnetic pulses, affecting SRAM initialization without needing invasive methods. Other attack types include clock glitching and laser fault injection. This research aims to explore these diverse attacks to enhance understanding of how fault injections compromise PUF reliability and to aid in developing mitigation strategies.

SRAM PUFs work best on resource-constrained IoT devices where memory is in abundance. Despite this, they are still classified as *weak PUFs*. Extensions on the SRAM PUF are other memory-based PUFs, such as latch-based or flip-flop-based memory PUFs, that are considered stronger as they are more reproducible. Deeper studies into PUF behavior can be better conducted when Field Programmable Gate Arrays (FPGAs) are used. FPGAs allow for the realization of the ring oscillator PUF, butterfly PUF, and hybrid PUF, all of which utilize the simulation of SRAM to generate PUFs. This ensures less randomness, which is detrimental for PUFs in authorization or authentication applications. Reproducibility must be possible, but reverse engineering must be difficult.

VI. Conclusion

This paper examined the effectiveness and drawbacks of SRAM PUFs as a lightweight security solution for IoT devices, particularly in pervasive healthcare technology. Our evaluation of an SRAM PUF on the ESP32 revealed a high reliability coefficient of 0.963 under normal conditions, but this dropped to 0.531 under fault injection attacks. These results indicate that SRAM PUFs are vulnerable to voltage glitches, compromising their stability and reproducibility.

This study sets the stage for exploring advanced fault injection techniques like electromagnetic and clock glitching, along with developing error correction methods. Future work will focus on alternative PUF architectures and environmental stressors to strengthen memory-based PUFs for resource-constrained IoT applications.

Acknowledgment

This work is funded by the University of Arkansas Honors College Research Grant. This material is based upon work supported by the National Science Foundation under Grant No. 2237945.

References

[1] K. Lopatina, V. A. Dokuchaev, and V. V. Maklachkova, "Data risks identification in healthcare sensor networks," in *2021 International Conference on Engineering Management of Communication and Technology (EMCTECH)*, Vienna, Austria: IEEE, Oct. 20, 2021, pp. 1–7.

[2] O. Staníček, "Physical unclonable functions on ESP32," Ph.D. dissertation, Czech Technical University, Prague, Czech Republic, May 5, 2022, 69 pp.

[3] J. v. Woudenberg and C. O'Flynn, *The hardware hacking handbook: breaking embedded security with hardware attacks*. San Francisco, CA: No Starch Press, 2022, 479 pp.

[4] J. Dreyer, R. Tönjes, and N. Aschenbruck, "ESPuF – enabling SRAM PUFs on commodity hardware," in *2023 16th International Conference on Signal Processing and Communication System (ICSPCS)*, Bydgoszcz, Poland: IEEE, Sep. 6, 2023, pp. 1–10.

[5] A. Roelke and M. R. Stan, "Attacking an SRAM-based PUF through wearout," in *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, Pittsburgh, PA, USA: IEEE, Jul. 2016, pp. 206–211.

[6] J. Delvaux, C. Mune, M. Romero, and N. Timmers, "Breaking Espressif's ESP32 v3: Program counter control with computed values using fault injection," in *18th USENIX WOOT Conference on Offensive Technologies (WOOT 24)*, Philadelphia, PA: USENIX Association, Aug. 2024, pp. 229–243.

[7] E. Systems, *Esp32 series datasheet*, version V3.9, Accessed: 2024-11-27, Jul. 2023.

[8] P. Ahr, J. Dreyer, M. Reski, *et al.*, *Industry 4.0 security trust anchors: Considering supply voltage effects on sram-puf reliability*, May 2023.