ProbeShooter: A New Practical Approach for Probe Aiming[*]

Daehyeon Bae[†]
Korea University
Seoul, South Korea

Sujin Park
Korea University
Seoul, South Korea

Minsig Choi
Korea University
Seoul, South Korea

Young-Giu Jung
YM-NaeulTech.
Incheon, South Korea

Changmin Jeong
Agency for Defense Development
Seoul, South Korea

Heeseok Kim[‡]
Korea University
Sejong, South Korea

Seokhie Hong[‡]
Korea University
Seoul, South Korea

Abstract

Electromagnetic side-channel analysis is a powerful method for monitoring processor activity and compromising cryptographic systems in air-gapped environments. As analytical methodologies and target devices evolve, the importance of leakage localization and probe aiming becomes increasingly apparent for capturing only the desired signals with a high signal-to-noise ratio. Despite its importance, there remains substantial reliance on unreliable heuristic approaches and inefficient exhaustive searches. Furthermore, related studies often fall short in terms of feasibility, practicality, and performance, and are limited to controlled DUTs and low-end MCUs.

To address the limitations and inefficiencies of the previous approaches, we propose a novel methodology—**ProbeShooter**—for leakage localization and probe aiming. This approach leverages new insights into the spatial characteristics of amplitude modulation and intermodulation distortion in processors. As a result, ProbeShooter provides substantial improvements in various aspects: **1)** it is applicable to not only simple MCUs but also complex SoCs, **2)** it effectively handles multi-core systems and dynamic frequency scaling, **3)** it is adoptable to uncontrollable DUTs, making it viable for constrained real-world attacks, and **4)** it performs significantly faster than previous methods. To demonstrate this, we experimentally evaluate ProbeShooter on a high-end MCU (the NXP i.MX RT1061 featuring a single ARM Cortex-M7 core) and a complex SoC (the Broadcom BCM2711 equipped with the Raspberry Pi 4 Model B, featuring four ARM Cortex-A72 cores).

CCS Concepts

• **Security and privacy → Side-channel analysis and countermeasures**; **Embedded systems security**.

Keywords

Hardware security, Electromagnetic side-channel analysis, Leakage localization, Cartography

1 Introduction

Electromagnetic Side-Channel Analysis (EMSCA) is a widely used method due to its versatility, enabling the monitoring of processor activity [27, 30] or compromising cryptographic systems [11, 31, 34] in *air-gapped* environments. Many studies have been measuring near-field electromagnetic (EM) emissions from various sources to capture Integrated Circuits' (ICs) information leakage: ❶ **the surface of the chip (silicon die)**, ❷ **decoupling capacitors**, ❸ **power rails on the PCB or substrate**, and ❹ **power management ICs (PMICs)**. However, except for source ❶ (the surface of the silicon die), all other sources (❷–❹) share characteristics with power leakage, offering no significant advantage over power analysis beyond being air-gapped; the benefit of measuring *localized leakage* has been completely overlooked. When targeting low-end ICs, considering such factors may seem excessive. However, as target ICs become more advanced (e.g., multi-core systems, complex SoC, and more), it is essential to fully utilize all the advantages of EM side-channel characteristics at the silicon die level [2, 32].

Direct probing on the surface of the chip (silicon die) involves ***leakage localization*** and ***probe aiming*** to ensure that only the desired signal is captured with a high Signal-to-Noise Ratio (SNR) in a silicon with a diverse integration of functional components. To achieve this, the following inscrutability of EM side-channel characteristics must be considered:

✻ The leakage map can significantly differ depending on various factors such as the probe coil diameter, orientation, and measurement height even for the same chip (i.e., multi-dimensional characteristics) [12–14].

✻ Even with approximate layout obtained from decapping or X-ray imaging, it remains challenging to accurately pinpoint the source of EM side-channel leakage. This is because EM side-channel can originate from various sources within the chip, including not only logic circuits but also top metal layers (primarily involved in power distribution) and bond wires [13, 19].

Thus, the importance of methods for achieving rapid and accurate leakage localization and probe aiming *in the given setup* is underscored, with several studies having been conducted on this matter.

Previous Approaches and Its Limitations. Numerous leakage localization studies have been conducted targeting cryptographic implementations [8, 14, 15]. These methods involve executing a cryptographic algorithm with a known key in a controlled environment and evaluating key recovery performance or other metrics at location candidates on the chip surface. However, these methods are notably expensive due to the need to measure and analyze numerous time series. As the DUT becomes more advanced and the spatial granularity increases, the cost becomes unaffordable, making them applicable only to low-end MCUs and thus limiting their feasibility. Moreover, these methods have limited practicality,

[*] This is the preprint version of a paper accepted for presentation at the 20th ACM Asia Conference on Computer and Communications Security—**ASIA CCS '25**. The final version will be available in the conference proceedings.
[†] First author (dh_bae@korea.ac.kr)
[‡] Corresponding authors ({shhong, 80khs}@korea.ac.kr).

as they cannot be applied to DUTs with features like multi-core or frequency scaling, nor in uncontrollable environments. Further details on previous approaches can be found in §5.

Another approach involves analyzing the power density around the operating clock in the frequency domain. However, this method has not yet been systematically studied and tends to rely on heuristic approaches [11, 21, 34]. This approach may succeed in some instances; however, it is likely to face several limitations, including the following:

- ✻ The operating frequency of other components within the silicon die, or its harmonics and other modulation results, may overlap. This likelihood is further increased if the core clock frequency is expressed as a round number.
- ✻ When observing the power density of the core clock frequency, signals from the Phase-Locked Loop (PLL)—generating the clock—can be measured stronger than those near the core. Therefore, the leakage from the core might be negligible, making it difficult to identify.
- ✻ Specific methodologies for isolating individual core signals in multi-core systems and addressing frequency scaling have not been studied.
- ✻ In the absence of established methodologies, reliance on heuristic approaches results in reduced precision and robustness of probe aiming.

Our Approach. To address the limitations and inefficiencies of previous approaches, we propose a new practical methodology for leakage localization (cartography) and probe aiming—**ProbeShooter**. This is an end-to-end framework that systematizes the entire process from signal measurement to analysis. Additionally, this approach targets CPU cores rather than cryptographic implementations, leveraging Amplitude Modulation (AM) and Intermodulation Distortion (IMD) characteristics through frequency analysis. Recently, EM side-channel leakage has become increasingly valuable, not only for cryptanalysis but also for applications requiring CPU activity, including EM-based anomaly detection [24, 29, 30] and forensic approaches [27, 28]. Therefore, ProbeShooter enables probe aiming with greater versatility when targeting CPU cores. The advantages of the proposed method are described in the following paragraph, along with the contributions.

Contributions. We summarized our contributions as follows:

- ✻ We present new observations on the spatial characteristics of AM and IMD in processors. Based on these observations, we propose ProbeShooter, a novel methodology for leakage localization and probe aiming in two distinct thread models. This approach is, to the best of our knowledge, the first of its kind.
- ✻ While previous approaches are limited to low-end MCUs in terms of practical applicability, ProbeShooter can also be applied to complex SoCs. To emphasize high feasibility, we conduct experiments using the off-the-shelf high-end MCU and the complex SoC.
- ✻ The proposed method effectively addresses systems with multi-core and dynamic frequency scaling, unlike previous approaches. Furthermore, it can be adopted for uncontrollable DUTs, making it suitable for constrained real-world attacks. This emphasizes the practicality of ProbeShooter.
- ✻ The proposed method achieves significantly faster performance compared to exhaustive searches and previous studies. This improvement is attributed to its exclusive reliance on frequency sweeping, in contrast to existing methods that involve key recovery or other forms of time series analysis on cryptographic implementations using numerous time series.
- ✻ We release the artifacts of ProbeShooter, including the source codes, to facilitate the reproducibility of experiments and advance future research.

Organization. The remainder of this paper is organized as follows: Section 2 presents new observations and insights underlying the proposed methodology. Section 3 proposes two versions of ProbeShooter. Section 4 evaluates ProbeShooter from various perspectives. Section 5 examines related work and provides a direct comparison with ProbeShooter. Section 6 discusses the limitations, and Section 7 concludes the paper.

2 Key Observations for ProbeShooter

In this section, we present the key observations and insights about the spatial characteristics of EM side-channel leakage underlying ProbeShooter. Before presenting our new observations, we first introduce the previous work in §2.1. And we present new observations on AM and IMD in §2.2 and §2.3, respectively. Finally, §2.4 summarizes the insights gained through our new observations and presents their potential applications.

2.1 Previous Work

EM Side-Channel Leakage of Processor. Previous studies have primarily described EM side-channel leakage caused by program activity (i.e., data and instruction-dependent leakage) as unintentional *amplitude modulation* [1, 6, 30]. That is, the signal emitted from the switching activity—synchronized with the clock signal—is considered the carrier, while the program activity is regarded as the modulator. This can be reasonably explained by the fact that the current and voltage alterations is the root cause of side-channel leakages, including EM [23]. For Complementary Metal-Oxide-Semiconductor (CMOS) circuits, switching activities that consume more power—such as instructions involving data with high Hamming weight/distance or more complex circuits—result in higher current draw, which induces differences in amplitude.

Side-Band of AM. The EM signal generated at the clock period—caused by the core's switching activity—is not an ideal sinusoidal carrier. And, program activities that modulate the carrier are also arbitrary. Therefore, the sidebands of AM in processors may exhibit intriguing patterns; if a processor operating at a clock frequency f_c repeatedly executes specific instructions with period T, frequency peaks at $f_c \pm \frac{n}{T}$ Hz (where $n \in \mathbb{N}$) will appear as sidebands of AM [5, 6, 37]. This enables each code region in a software to have a distinct spectral signature, thereby contributing to various applications such as software profiling, anomaly detection, forensic approaches, and more [27–30].

2.2 New Observations on AM

Unintended High-Power Transmitting Antenna. According to A. Kumar et al. [19], EM side-channel leakage from the silicon die is predominantly caused by the top metallization layers of the on-chip Power Delivery Network (PDN). The top metal layer primarily serves for power distribution and signal routing. It is designed with the thickest and widest interconnects to handle high-current flow, thus having a dominant influence on EM side-channel leakage from the silicon die. This implies that during switching activity in CMOS circuits, the power rails of CPU core in the top metal layer will experience the most significant voltage and current variations. That is, portions of the top metal layers associated with the core's power rails function as *high-power transmitting antennas*, emitting amplitude-modulated signals (i.e., information leakage).

AM and Leakage Localization. For the reasons outlined above, we can perform probe aiming by identifying the location on the chip surface where the transmission strength of the *virtual antenna* is highest. This approach is effective because **1)** the CPU core's power domain is isolated from other components within the silicon, resulting in localized leakage with high SNR, and **2)** the power rails—being thick and subject to significant voltage and current variations—emit signals strong enough to be distinguished.

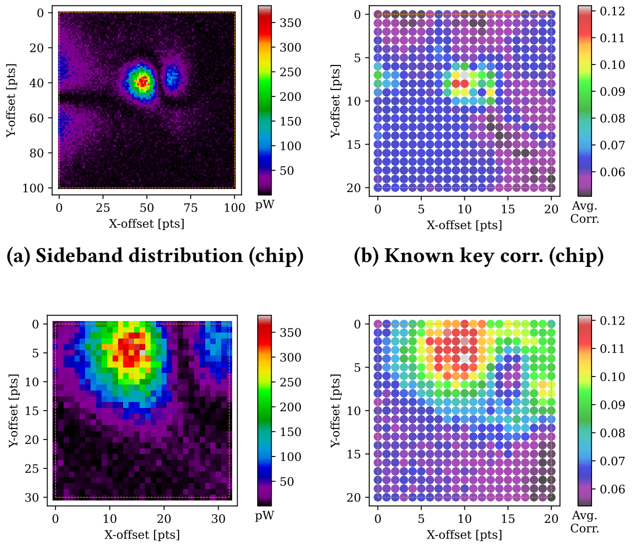
To achieve leakage localization for a virtual antenna, we first induce the antenna to radiate signals at a predetermined frequency. Here, we leverage previous research on AM sidebands of EM side-channel leakage in processors (§2.1). By manipulating instructions, we can generate sideband peaks at specific frequencies and scan them in the frequency domain. A strong sideband signal indicates intense AM, which reflects significant amplitude changes due to program activity and can indirectly reflect the information leakage. Consequently, the resulting leakage map serves as a guideline for identifying information leakage and facilitating probe aiming in the given environment.

Proof of Concept. For the Proof-of-Concept (PoC) experiment, we conduct tests on the i.MX RT1061, a high-end MCU from NXP (detailed setup is described in §4.1.1). We present the power distribution of sideband peak from AM at the chip and silicon die level in Figure 1(a) and 1(c), respectively. Here, the sideband peaks was designed to occur in the form of $f_c \pm \frac{n}{T}$ by applying findings from previous research (§2.1). To assess whether these distributions can identify sources of information leakage, we also present the actual data leakage maps during AES-128 encryptions in Figure 1(b) and 1(d). These results demonstrate the validity of our observations and suggest the potential for developing an effective probe aiming methodology.

2.3 New Observations on IMD

IMD in Processors. AM is the main contributor to EM side-channel leakage in processors, but the spectrum also includes additional frequencies due to various coupling and modulation effects [1]. These effects are generally less significant compared to AM, often overlap with other frequencies, or present in unpredictable forms, and thus have not garnered substantial attention. Among these, we focused on IMD.

IMD is a type of nonlinear distortion that occurs when two or more signals with different frequencies interact in a nonlinear circuit, resulting in new components that are sums and differences of the original frequencies or their harmonics (see Appendix A.1 for more details). Although harmonic distortion—another type of nonlinear distortion—is readily observed in EM side-channel leakage, the IMD caused by repeated program activity (coupling or internal control/data signals from repetitive circuit operations can intermodulate with the clock) is challenging to detect independently because its frequency coincides with the sidebands of AM—in the form $f_c \pm \frac{n}{T}$ Hz.



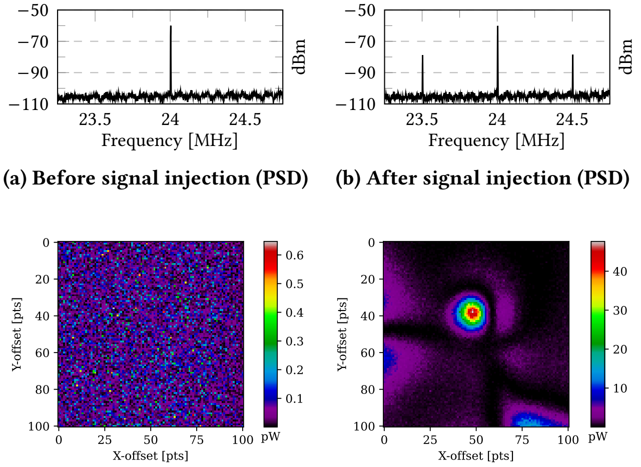
(a) Sideband distribution (chip) **(b) Known key corr. (chip)**

(c) Sideband distribution (die) **(d) Known key corr. (die)**

Figure 1: Comparison between the power distribution of the sideband—(a) and (c)—and the actual data leakage from AES-128 encryptions—(b) and (d). Data leakage refers to the averaged correlation coefficients between the measurements and the output of the 1st round `SubBytes` using a known full key.

Enforced IMD and Leakage Localization. Although inherent IMD from processor activity cannot be directly observed, we can instead leverage the characteristics of IMD to perform interesting tasks: injecting arbitrary signals into the processor to *deliberately* induce IMD. When a signal of a specific frequency is injected into the nonlinear circuits of a chip—which operates with its own clock—IMD occurs between the two frequencies. Specifically, if the signal is injected exclusively into the CPU core, IMD will occur between the CPU core's clock frequency and the injected signal. This distortion causes changes in voltage and current at the IMD frequencies in both the logic circuit of the CPU core and the connected power rails. As noted in §2.2, the power rails on the top metal layer, being the thickest, experience the most significant voltage and current changes, resulting in stronger signals compared to those from other parts. Thus, similarly to AM, we can assume the presence of a virtual antenna emitting IMD products. Based on this, we can perform leakage localization by scanning the chip surface using the intensity of the IMD products as a metric.

Proof of Concept. For the PoC experiment, we use a signal generator to inject a 500 kHz sine wave into the CPU core operating at 24 MHz, inducing IMD with the clock frequency. Figure 2(a) and 2(b) show the changes in Power Spectral Density (PSD) before and



(a) Before signal injection (PSD) **(b) After signal injection (PSD)**

(c) Before signal injection (Map) **(d) After signal injection (Map)**

Figure 2: IMD generation from injecting a 500 kHz signal into a CPU core operating at 24 MHz. The PSD is depicted in (a) and (b), while the distribution of power density at the IMD peak frequency (24.5 MHz) is shown in (c) and (d).

after signal injection, while Figure 2(c) and 2(d) show the changes in power distribution at the frequency corresponding to the IMD peak. These results also validate our observations.

2.4 Insights for ProbeShooter

Through this section, we derive the following insights:

- ✻ By leveraging the sidebands of AM and IMD products, we can capture the *dominant EM side-channel leakage of the CPU core originating from the top metal layer connected to the core's power rail.* However, this method may struggle to identify subtle EM leakage directly emitted from the CPU core's logic circuits, potentially leading to some false negatives. This direct EM leakage is not only difficult to exploit but also does not offer any significant advantages over the dominant EM leakage we have identified, and therefore, this issue can be mitigated.
- ✻ IMD occurs solely between the operating clock and the injected signal, independent of the device's operation. Therefore, leakage localization can be performed without requiring control over the chip. This indicates the potential for leakage localization on uncontrollable targets.
- ✻ Since our new approach performed in the frequency domain, it is robust to noise and can be executed faster than previous methods. Moreover, in multi-core environments, if each core operates simultaneously at a distinct frequency (period), they can be independently identified within the frequency domain. This indicates the potential for effective leakage localization in multi-core systems. The fact that each core typically has its own power rail enables this possibility.

These insights are further elaborated in §3, forming the theoretical foundation of ProbeShooter.

3 Proposed Methodology

In this section, we propose a novel approach, ProbeShooter based on the key observations and insights from §2. We introduce two versions of ProbeShooter—denoted as -P(assive) and -A(ctive)—to address various scenarios:

- ✻ **ProbeShooter-P** (§3.1 / Abbr. PS-P). A *passive* version of ProbeShooter. PS-P involves the *passive* observation of AM sidebands induced by executing *cyclic instruction gadgets* (§3.1.3). To achieve this, PS-P requires control over the chip (e.g., code execution). It may be a strong assumption from the attacker's perspective but may not be significant depending on whether the chip is off-the-shelf product or from the defender's perspective. Attacker assumptions are discussed in detail in §3.1.1, while applicable scenarios are covered in §3.4.
- ✻ **ProbeShooter-A** (§3.2 / Abbr. PS-A). An *active* version of ProbeShooter. PS-A involves *actively* injecting periodic signals into the DUT using an external signal generator. Then, the resulting IMD products are leveraged for probe aiming. It can be applied to *uncontrollable* chips, making it useful for scenarios involving constrained real-world attacks. To achieve this, PS-A requires physical access to the power rails of the target chip (CPU core). Attacker assumptions are discussed in detail in §3.2.1, while applicable scenarios are covered in §3.4.

High-level Representation. We show the high-level representation of ProbeShooter (PS-P, PS-A) in the Figure 3. The two versions generally perform similarly but show the most significant difference in the preliminary steps (P-1, A-1). And the subsequent stages operate in a broadly similar manner but differ in their details.

3.1 ProbeShooter-P(assive)

3.1.1 Threat Model. PS-P requires control over the chip to enable the evaluator to execute a *cyclic instruction gadget* (§3.1.3). This is undoubtedly a strong assumption from an attacker's perspective. Therefore, PS-P is considered a version designed more from a defensive standpoint rather than an offensive one (see §3.4 for more details). However, this strong assumption can be mitigated even from an attacker's perspective if the target chip is a commercially available (i.e., off-the-shelf). In such cases, an attacker could utilize a chip of the same model as a target to generate leakage maps and perform probe aiming. Then, the required attacker assumptions are similar to the profiling attack scenarios in typical side-channel analysis and therefore acceptable [7, 25].

For a multi-core system, it is required to ensure that the gadget remains on the same core without task migration while PS-P is running; the ability to set core affinity is required. Considering the goal of probe aiming for a specific core, it is acceptable. Here, code execution and core affinity configuration can be performed by a non-privileged user. That is, root privileges are not required. And we discuss Dynamic Voltage and Frequency Scaling (DVFS) in §3.3 and §4.5; for the rest of the sections, we describe mechanisms assuming a fixed CPU core clock frequency.

3.1.2 Procedure. PS-P consists of the following P-1–P-4 steps:

P-1 **Triggering Cyclic Instruction Gadget.** First, the evaluator must trigger a *cyclic instruction gadget* on the DUT to deliberately

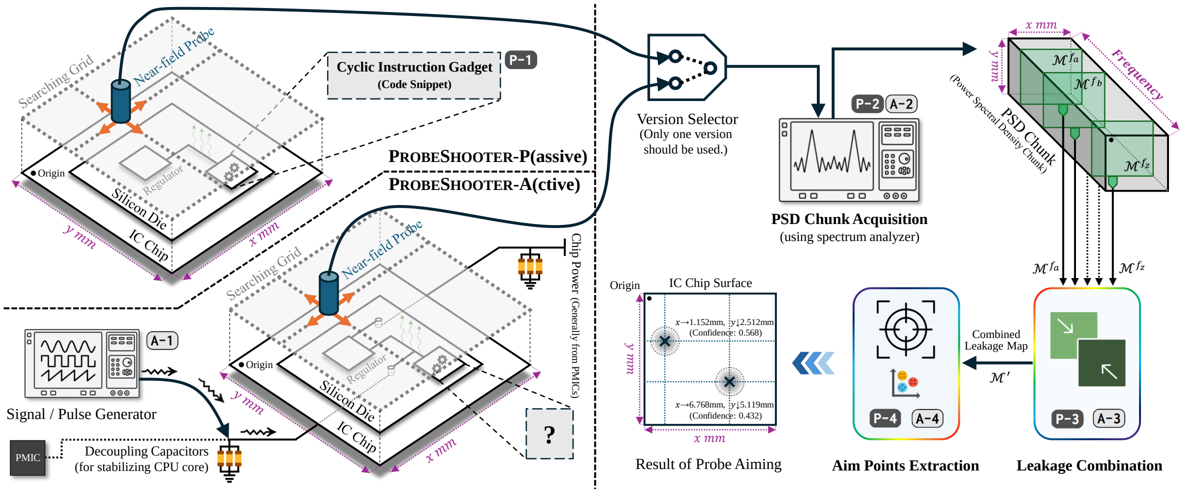


Figure 3: High-level representation of ProbeShooter (-P and -A versions).

generate the sideband peaks of the AM (§2.2). Here, a cyclic instruction gadget refers to a code snippet that repeatedly executes several instructions with a fixed clock cycle (i.e., loop) and is used to generate *intentional* frequency leakage—sidebands of AM. Note that gadgets in PS-P are not found in existing instruction chunk of a system as in ROP [26] and Spectre [17], but are executed directly by the evaluator on a controlled system. For a more detailed explanation and requirements of instruction gadgets, please refer to §3.1.3. In multi-core systems, the gadget is executed with its affinity set to the core under investigation. By simultaneously executing gadgets with different clock cycles on each core, it is possible to obtain leakage maps for all cores and perform probe aiming at once (§3.1.4).

P-2 **PSD Chunk Acquisition.** Next, the evaluator scans the chip surface using a near-field probe and a spectrum analyzer while the gadget is running. During this process, the evaluator acquires the PSD for all points on the grid to construct the PSD chunk (Appendix B). Here, the frequency range should be configured to encompass all frequencies relevant to the power density maps ($\mathcal{M}$) of the sideband peaks that will be used in the subsequent P-3 step. And the resolution bandwidth (RBW) should be set sufficiently to ensure that each frequency peak should be distinguishable. To reduce noise and improve reliability—especially in complex SoCs—it is permissible to average multiple PSDs at each point. This approach, however, must be balanced against the associated time trade-offs (§3.5). If the frequency band is noisy due to factors such as wireless communication, the evaluator can adjust the frequency peaks by modifying the core clock frequency or the gadget's instruction configuration.

P-3 **Leakage Combination.** In this step, multiple leakage maps are combined into a single map $\mathcal{M}'$ for the subsequent step. Here, at least two maps are used, and by combining data from multiple frequencies, some noise attenuation is achieved due to the effect of frequency diversity. The simplest and most reliable method for leakage combination is to compute the average of multiple maps. For instance, if the clock frequency is f_c and the inverse of the gadget's repetition period is f_{gadget}, leakage combinations can be performed simply as follows:

$$\mathcal{M}' \leftarrow (\mathcal{M}^{f_c - f_{\text{gadget}}} + \mathcal{M}^{f_c + f_{\text{gadget}}})/2$$

It is also allowed to combine more maps if additional sideband peaks are available.

P-4 **Aim Points Extraction.** Finally, the evaluator leverages the combined leakage map ($\mathcal{M}'$) to ultimately perform *probe aiming*. In this step, leaky areas of type 1–3 are extracted, and the aiming points and their confidence levels are derived. Here, type-*n* areas refers to the *humps* extracted from the $\mathcal{M}'$. As the type increases, the size of the hump decreases, and it becomes more reliable. We have detailed this process in Algorithm 1. Furthermore, to enhance clarity, we present the high-level flow of Algorithm 1 in Figure 4.

In Algorithm 1, outliers—peaks caused by intermittent activity from non-target components or by environmental noise—in $\mathcal{M}'$ are first removed by `median_filter`. And a `uniform_filter` is applied to flatten the map. The size of the `median_filter` is kept sufficiently small to avoid distorting the map—typically 2×2 or 3×3—while the `uniform_filter` is applied at a size 1.5 to 2 times larger than the median filter.

Next, *type1_humps*, *type2_humps*, and *type3_hump*s are sequentially extracted, with the final aiming points and their confidence levels being calculated based on the *type3_hump*s. During this process, the DBSCAN algorithm is employed to cluster the humps and eliminate isolated humps. For this purpose, the maximum distance between points within a cluster (ϵ_{1-2}) is set to an arbitrary real value between $\sqrt{2}$ and 2. This ensures that only points that are adjacent horizontally, vertically (Δ=1), or diagonally (Δ=$\sqrt{2}$) are considered within the same cluster. Note that in environments with minimal noise (e.g., averaged PSD or MCU environments), the *type2_humps* and *type3_hump*s are generally identical. However, if the *type2_humps* are fragmented, we select the dominant cluster and extract it as a *type3_hump*. Finally, the aim points are determined using the *type3_hump*, and the confidence level is calculated based on its power density.

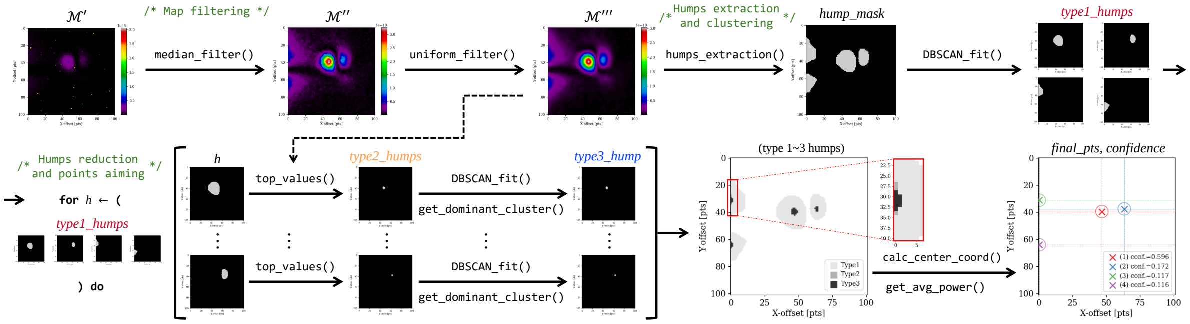


Figure 4: Detailed flow of the *Aim Points Extraction* step as described in Algorithm 1.

Algorithm 1: Pseudo-code for aim points extraction using a combined leakage map.

```
Input: Combined leakage map ($\mathcal{M}'$), Filter sizes ($\kappa_m$, $\kappa_u$), DBSCAN params
       ($Ms_{1-2}$, $\epsilon_{1-2}$), Quantile thresholds ($\tau$, $r$)
Output: final_pts, confidence
/* ===== Map filtering ================================== */
$\mathcal{M}''$ ← median_filter($\mathcal{M}'$, $\kappa_m$)              // Outlier Removal
$\mathcal{M}'''$ ← uniform_filter($\mathcal{M}''$, $\kappa_u$)            // Map Flattening
/* ===== Type1-humps extraction and clustering =========== */
hums_mask ← humps_extraction($\mathcal{M}'''$, $\tau$)
type1_humps ← DBSCAN_fit(humps_mask, $Ms_1$, $\epsilon_1$)
/* ===== Humps reduction and points aiming =============== */
final_pts, avg_power ← [ ], [ ]
for h ← type1_humps do
    type2_humps ← top_values($\mathcal{M}''' \odot h$, r)
    /* ⊙: Selective extraction using hump mask h */
    temp_humps ← DBSCAN_fit(type2_humps, $Ms_2$, $\epsilon_2$)
    type3_hump ← get_dominant_cluster(temp_humps)
    /* i ← Current iteration of the loop */
    final_pts [i] ← calc_center_coord(type3_hump)
    avg_power [i] ← get_avg_power(type3_hump)
confidence ← avg_power / sum(avg_power)           // $\Sigma confidence$=1
return final_pts, confidence
```

```
;7cycles/iter. in i.MX RT1061
gadget:
push {r1}
pop  {r1}
b.n  gadget
;
```

Listing 1: Example of a gadget for ARMv7-M.

```
;7cycles/iter. in BCM2711
mov   x1, #0x1000
mov   x2, #0x3
gadget:
udiv  x0, x1, x2
b     gadget
```

Listing 2: Example of a gadget for ARMv8-A.

3.1.3 *Cyclic Instruction Gadget.*

A cyclic instruction gadget is an instruction snippet designed to generate intentional frequency leakage (sideband peaks), which the evaluator executes directly. It should take the form of an *infinite loop*, with the number of clock cycles taken by each iteration being fixed. Additionally, it is advisable to design the instructions comprising the gadget in a manner that allows for a rough prediction of the required clock cycles, aiming for simplicity. For example, we utilized simple gadgets as described in Listing 1 and Listing 2 for ARMv7-M and ARMv8-A, respectively.

Type of Instruction. As described in §2, our approach localizes EM leakage near the top metal layer of the core power rail; it focuses on voltage and current fluctuation in the power rail due to program activity. Thus, the fact that different computational circuits operate at various locations depending on the types of executed instructions does not affect the results of ProbeShooter. Instead, the use of power-intensive instructions maximizes the fluctuations in current and voltage, leading to improved results.

Clock cycles of the gadget. Accurate knowledge of the gadget's clock cycles is crucial for leakage combination (P-3). In the case of MCUs, assuming that PS-P has the capability to develop and upload firmware (§3.2.1), clock cycles can be easily determined through debugging, performance counter, or reference manuals. For SoCs, clock cycles can be obtained by accessing the performance counters within the chip. This approach remains effective even in the presence of various performance enhancement techniques such as branch prediction and instruction-level parallelism. We have detailed this process in Appendix C.1. By leveraging this approach, we can achieve interference-free results by precisely adjusting the clock cycles of the gadgets. We recommend selecting gadget clock cycles that are coprime with the core clock frequency. This is because if they are not coprime, sideband peaks are likely to overlap with commonly used frequencies or their harmonics, increasing the likelihood of interference from other components.

3.1.4 *Targeting Multiple Cores Simultaneously.*

PS-P can target multiple cores simultaneously by executing different gadgets on each core. This implies that it is not necessary to complete the preceding P-1–P-4 steps for each core individually; a single acquisition of a PSD chunk is sufficient to obtain the leakage maps of all cores. To achieve this, the clock cycles of each gadget must be *pairwise coprime*. This ensures that frequency leakage (sideband peaks) from individual cores does not overlap, allowing all cores' leakage to be independently present within a single PSD chunk. In other words, by operating P-3 and P-4 on each core within one PSD chunk, multiple cores can be targeted at once. We demonstrate this through experiments in §4.4.

3.2 ProbeShooter-A(ctive)

3.2.1 *Threat Model.*

This version does not require control over the DUT; that is, it does not necessitate executing cyclic instruction gadgets as in PS-P. Instead, the cores under investigation must be in an active state rather than IDLE state (see §3.2.4 for more details).

PS-A requires inducing IMD by injecting periodic signals into the power rails of the target chip (CPU core) using an external signal generator. To achieve this, the evaluator must have physical access to the power rails of the CPU core. In most chips that are not low-end, the power domain of the CPU core is isolated from other components within the silicon die, and the power rails are connected to external decoupling capacitors for voltage stabilization. Furthermore, it is common for the power to be supplied directly to the CPU core from the on-board PMIC, bypassing the chip's internal regulators. In such cases, numerous decoupling capacitors are also connected to the power rails. Therefore, in most cases, the evaluator can access the power rails of the CPU core through the specific decoupling capacitors (or their associated pads). These attacker assumptions about the physical access are similar to those required in impedance side-channel analysis [3, 23].

3.2.2 Procedure. PS-A consists of the following A-1–A-4 steps:

A-1 **Periodic Signal Injection.** To generate artificial IMD products, the evaluator connects a signal generator to the power rails of the target chip (CPU core) and injects a signal at a specific frequency f_{inj}. As noted in §3.2.1, power rails are generally accessible through decoupling capacitors. While decoupling capacitors are typically integrated onto the main PCB, in high-performance chips or SoCs, they may also be located on the substrate in flip-chip packages. If the injected signal is significantly attenuated, it is acceptable to remove a small number of capacitors, as long as it does not impact the chip's operation. Typically, removing a small number of decoupling capacitors does not impact the chip's operation. Injecting the signal in this manner will result in intermodulation distortion between the core's clock frequency and f_{inj}, creating IMD peaks. Here, IMD generation is independent of the DUT's operations, making it applicable to uncontrollable DUTs. The power and frequency of the injected signal are discussed in §3.2.3.

A-2 **PSD Chunk Acquisition.** Next, the evaluator obtains a PSD chunk. This step is identical to P-2 of PS-P, with only minor differences in parameters such as frequency range and RBW. In PS-P, the frequency range is determined based on the gadget, whereas in PS-A, it is determined solely based on the frequency of the injected signal (f_{inj}). That is, the sweep range should be set to include frequencies combined from f_c and f_{inj}, which will be used in the subsequent step (A-3).

A-3 **Leakage Combination.** This step is also very similar to P-3 of PS-P. The difference is that, rather than being dependent on the gadget, it combines leakages using maps corresponding to the frequencies of IMD peaks generated by signal injection. For instance, if the clock frequency is f_c and injected frequency is f_{inj}, leakage combinations can be performed simply as follows:

$$\mathcal{M}' \leftarrow (\mathcal{M}^{f_c - f_{\text{inj}}} + \mathcal{M}^{f_c + f_{\text{inj}}})/2$$

However, in the case of PS-A, the IMD peaks is generally lower and narrower compared to sideband peaks of PS-P, requiring the use of a finer (lower) RBW. In such cases, due to the longer sweep time, it may be difficult to sweep a frequency range wide enough to include two or more IMD peaks; therefore, observing only one IMD peak may suffice, allowing the A-3 step to be omitted. If step A-3 is omitted, it is recommended to perform noise reduction through PSD averaging at A-2.

A-4 **Aim Points Extraction.** This step is exactly identical to P-4 step in PS-P. Therefore, please refer to Algorithm 1 and Figure 4 presented in P-4 (§3.1.2). One important consideration is the potential for intermodulation to occur in the voltage regulators connected to the power rails in PS-A; false positive humps may appear. However, this occurs at consistent locations and generally exhibits a lower confidence level compared to true positive humps, making it feasible to eliminate through post-processing.

3.2.3 Power and Frequency of Injected Signals. Injecting signals into the power rails has the potential to induce unintended voltage fault attack (voltage glitch) [4]. Therefore, the signal must be applied at a power level that does not affect the behaviour of the chip. For f_{inj}, it is sufficient to ensure that IMD peaks do not overlap with the hump of the clock frequency. This also varies by chip and therefore must be determined experimentally. Our signal injection environment, as well as the signal power and frequency, is detailed in §4.1.2.

3.2.4 Additional Requirements for CPU Core Status. Most SoCs employs *clock gating* to prevent supplying clock signal to the CPU cores when they are not needed for power efficiency; as a result, operational circuits—such as ALU—within the CPU core can remain inactive. A state where the clock is not supplied to the core is referred to as an IDLE state. In IDLE state, the core cannot generate IMD products with the injected signal (f_{inj}) because the CPU core is not operating. Therefore, to perform PS-A on an SoC with clock gating, a dominant process (software) must be active within the core. However, *there is no need for specific knowledge about this process.* Note that most MCUs and several SoCs lacking clock gating features are not relevant to this subsection.

3.3 ProbeShooter on DVFS-enabled Systems

In numerous application processor families (e.g., ARM Cortex-A profile), the voltage and clock frequency of the cores are dynamically adjusted to optimize power efficiency, a process known as Dynamic Voltage and Frequency Scaling (DVFS). For ARM cores, pre-defined pairs of voltage and core clock frequency—Operating Performance Points (OPPs)—are included in the kernel, which correspond to different levels of CPU workload. Then, the voltage and clock frequency are adjusted in real-time according to the workload on the cores, based on these predefined OPPs. When there is no workload—distinct from a low workload—the cores may enter an IDLE state, as mentioned in §3.2.4. To apply ProbeShooter to a DVFS-enabled system, two major modifications are required:

#1. Broadband frequency sweep in P-2 and A-2 steps.
#2. Replacement of the leakage combination steps (P-3, A-3) with Algorithm 2.

#1. Broadband frequency sweep. When the core's clock frequency is fixed, acquiring PSD chunks over a narrow bandwidth around the specified frequency was sufficient. However, for DVFS-enabled systems, it is necessary to measure a sufficiently broad bandwidth to encompass all frequencies included in the OPPs. For example, in the Broadcom BCM2711—which will be tested in §4—there are a total of 10 OPP pairs ranging from 600 MHz to 1.5 GHz in 100 MHz increments (voltage values are omitted). Then, we need to sweep a bandwidth of approximately 1 GHz across all points on the chip surface. In this case, due to the use of a relatively coarse RBW, it is essential to carefully adjust the frequency of gadgets or injected

Algorithm 2: Pseudo-code for leakage combination to apply ProbeShooter on DVFS-enabled systems.

```
Input: PSD chunk ($\mathcal{D}$), Clock cycle of gadget ($c$) or frequency of injected
       signal ($f_{\text{inj}}$)
Output: Combined leakage map ($\mathcal{M}'$)
$\mathcal{M}_{t1}, \mathcal{M}_{t2} \leftarrow [], []$        // Temporary map for sideband/IMD peaks
for $psd \leftarrow \mathcal{D}$ do
    /* Get a single PSD for all points on the chip surface   */
    $f_{\text{clock}} \leftarrow$ estimate_core_clock($psd$)
    $f_{\text{leakage}} \leftarrow f_{\text{clock}}/c$                 // In PS-A, ($f_{\text{leakage}} \leftarrow f_{\text{inj}}$)
    /* $(\alpha, \beta)$: Coordinates of the corresponding PSD          */
    $\mathcal{M}_{t1}[\alpha, \beta] \leftarrow$ get_power_density($psd$, $f_{\text{clock}} - f_{\text{leakage}}$)
    $\mathcal{M}_{t2}[\alpha, \beta] \leftarrow$ get_power_density($psd$, $f_{\text{clock}} + f_{\text{leakage}}$)
$\mathcal{M}_{leakage} \leftarrow (\mathcal{M}_{t1} + \mathcal{M}_{t2})/2$                    // Point-wise
$\mathcal{M}' \leftarrow$ uniform_filter($\mathcal{M}_{leakage}$)
return $\mathcal{M}'$
```

signals to ensure that sideband/IMD peaks can be distinguished from other peaks.

#2. Leakage combination for DVFS-enabled systems. To apply ProbeShooter to a DVFS-enabled system, we have developed a new leakage combination algorithm that replaces P-3 and A-3, as described in Algorithm 2. We estimate the variable clock frequencies using wide-band PSDs measured at all points of the chip surface. Then we identify the sideband/IMD peaks and generate a new leakage map based on the clock estimation. Therefore, the accuracy of the `estimate_core_clock` has a significant impact on the results. Clock estimation is feasible due to the discrete and fixed number of OPPs, which allows for various methods such as clustering PSDs or analyzing a single PSD. Nevertheless, we have confirmed that simply selecting the frequency with the highest power density among those available in the OPP achieves over 75% accuracy; it is sufficient for our purpose. Results could be improved with a more refined clock estimation function. We evaluate the ProbeShooter for variable frequencies in §4.5.

3.4 Real-World Scenarios and Use Cases

In most real-world scenarios, probe aiming can be effectively covered by PS-P alone. The scenarios where PS-A needs to be considered are limited to cases where the DUT is uncontrollable and the chip is not commercially available. This is because, as mentioned in §3.1.1, the evaluator can obtain another device equipped with the same chip model and perform leakage localization and probe aiming, similar to a profiling attack scenario. Therefore, the scenarios requiring PS-A are limited to real-world hacking or penetration testing contexts with constrained environment. We present the decision tree for ProbeShooter version selection in the Figure 5. And we have listed several representative use cases as follows:

- **Academic research.** ProbeShooter can be employed for probe aiming in wide range of EMSCA-related studies targeting CPU cores. Particularly, since most experiments are conducted in unconstrained environments where code execution privileges are readily available, PS-P can be effectively utilized.
- **Vulnerability analysis in a less constrained environments.** For example, security engineers analyze EM side-channel vulnerabilities of their products to ultimately enhance the security. In this case, they often have control at the code execution level on their products, making it possible to use PS-P.
- **EM-based anomaly detection.** This emerging research area leverages EM side-channels to monitor program activity in air-gapped environments for anomaly detection [24, 29]. Profiling is usually performed on systems under the operators' control, facilitating the effective use of PS-P. A notable application is in military weapon systems. A similar approach can be found from a forensic perspective [27, 28].
- **Penetration testing or real-world hacking in constrained environments.** In highly constrained environments without code execution privilege, PS-A can be effectively utilized.

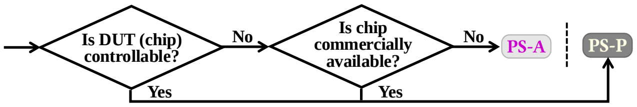


Figure 5: Decision tree for ProbeShooter version selection based on the environmental constraints. The second conditional branch evaluates whether the attacker's assumptions in a typical profiling attack scenario are satisfied.

3.5 Complexity

ProbeShooter spends nearly all of its time during the PSD chunk acquisition phase (P-2, A-2); the costs of all other steps are negligible. Let T_{sweep} be the sweep time of the spectrum analyzer, N_{mean} be the number of PSDs to average, T_{comm} be the time for transferring PSD data from the spectrum analyzer to the PC, T_{move} be the time for the XYZ table to move between points, and (m, n) be the size of the grid on the chip's surface. And, let the time required for the post-processing (P-3–P-4 and A-3–A-4) be denoted as T_{proc}. Then, the operating time of ProbeShooter (T_{ps}) is as follows:

$$T_{ps} = \overbrace{(T_{sweep} \times N_{mean} + T_{comm} + T_{move})}^{T_{point}} \times m \times n + T_{proc}$$

Here, T_{point} denotes the processing time for a single point. The *theoretical* complexity may appear high, as this is based on cartography of the entire chip surface. However, unlike previous studies that involve *collecting numerous time-series signals* at each point, we perform only *frequency sweep*, which results in significantly faster execution. And note that, PS-P and PS-A can be considered to have the same time complexity under the same parameters.

4 Evaluation

In this section, we evaluate ProbeShooter across various aspects: soundness (§4.2), robustness (§4.3), and applicability—frequency scaling (§4.5) and multi-core system (§4.4). And we evaluate the PS-A for the uncontrollable DUT in §4.6.

4.1 Experimental Setup

4.1.1 Device-Under-Test Setup. To emphasize the applicability and practicality of ProbeShooter, we selected one high-end MCU and one complex SoC with different ARM core profiles for experimentation: NXP i.MX RT1061 and Broadcom BCM2711.

NXP i.MX RT1061. The NXP i.MX RT1061 is a high-end MCU equipped with a single-core ARM Cortex-M7 (32-bit ARMv7-M). For

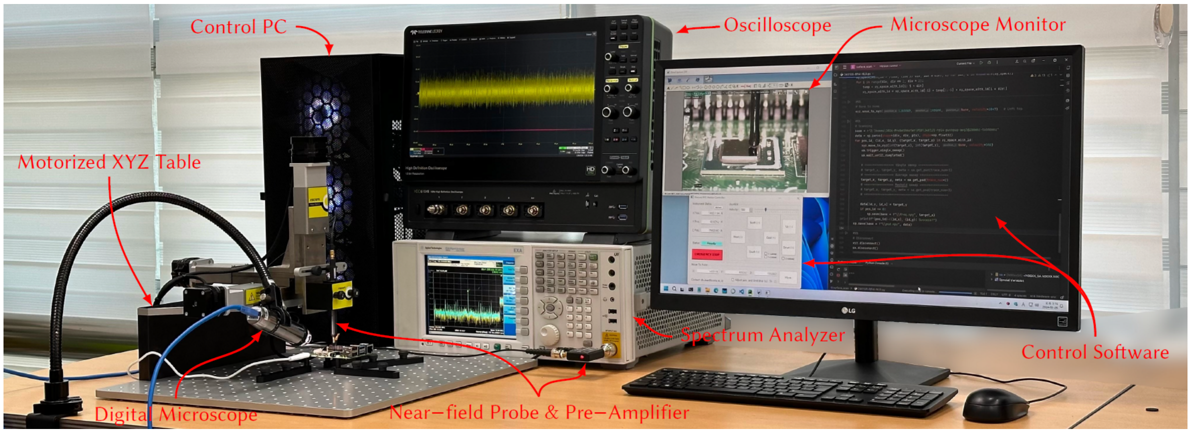


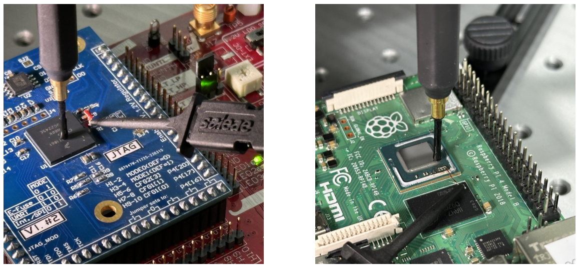
Figure 6: Test bench for ProbeShooter-P. The oscilloscope is only for the evaluation and is not required for ProbeShooter.

testing i.MX RT1061, we utilized test boards that we fabricated ourselves, based on open-source PCB designs from the ChipWhisperer project [33]. We also developed and uploaded bare-metal firmware using the MCUXpresso IDE provided by NXP. The i.MX RT1061, as a high-end MCU, supports a core clock frequency of up to 588 MHz. The core clock frequencies used in the experiments vary and will be detailed in the respective subsections and Appendix D.

Broadcom BCM2711. The Broadcom BCM2711 is a complex SoC equipped with quad-core ARM Cortex-A72 (64-bit ARMv8-A). For testing BCM2711, we utilized the off-the-shelf Raspberry Pi 4 Model B released in 2019. And we deployed *Raspberry Pi OS with Desktop* (Debian Bullseye / Linux kernel v5.13.0-1031) on the BCM2711. Furthermore, to operate the GPU (VideoCore IV), we kept the monitor continuously on while ProbeShooter was running and launched a `gnome-system-monitor` that displayed the core utilization in real-time graphs (GUI). The BCM2711 supports a core clock frequency ranging from 600 MHz to 1.5 GHz in an OS-based environment. The core clock frequency and affinity configuration varied across experiments and will be detailed in the respective subsections and Appendix D.

4.1.2 ProbeShooter Setup. To precisely scan the chip surface with the near-field EM probe, we used the motorized XYZ table from Riscure. And we used the N9010A spectrum analyzer from Agilent (Keysight), which supports frequencies up to 3.6 GHz, to acquire the PSD chunk. Lastly, we used the RF-B 0.3-3 near-field probe (ϕ≈2*mm*, Δ<1*mm*) and the 30dB pre-amplifier from Langer. We show our test bench for PS-P in Figure 6. Note that the oscilloscope was only used for evaluation (§4.2) and is *not required* for ProbeShooter.

For signal injection during the A-1 step of PS-A, we utilized a Anritsu MG36221A signal generator. The signal injection setups for the i.MX RT1061 and BCM2711 are shown in Figure 7. Here, we removed some decoupling capacitors connected to the CPU cores of both the i.MX RT1061 (capacitors on main PCB) and BCM2711 (capacitors on substrate of flip-chip BGA package). And we injected the signal through the pads of removed capacitors. Unless otherwise specified, the default configuration for the injected signal's shape, frequency, power, and impedance is as follows:



(a) Case: i.MX RT1061

(b) Case: BCM2711

Figure 7: Signal injection setup for ProbeShooter-A.

✻ **i.MX RT1061:** Sine wave (500 *kHz*, 25 *dBm*, *Z*=50Ω)
✻ **BCM2711:** Sine wave (10 *MHz*, 30 *dBm*, *Z*=50Ω)

Although V_{pp} is significantly higher than the voltage of the power rail, in practice, the actual voltage does not fluctuate severely due to parasitic capacitance and the remaining decoupling capacitors. In fact, we did not observe any malfunctions or errors in the chip during the signal injection.

4.2 Soundness

In this subsection, we demonstrate the following: *"Among the aiming points extracted by ProbeShooter, the one with the* ***highest confidence level*** *exhibits the* ***most significant and exploitable EM side-channel leakage*** *on the chip surface."*

The side-channel leakage of the processor (CPU core) can be classified into two types: data leakage and instruction leakage [22]. To demonstrate that two significant and exploitable leakages are emitted at the targeted points extracted through ProbeShooter, this subsection performs following two experiments: Correlation EM Analysis (CEMA) on i.MX RT1061 (data leakage / Exp.#S1) and EM-based anomaly detection [29] on BCM2711 (instruction leakage / Exp.#S2).

[Exp.#S1: CEMA on i.MX RT1061] For the experiment, we executed the unprotected `tiny-AES` [18] at 24 MHz. And, we collected 9,000 time series measurements from every point on a 7×7 grid across the chip surface (detailed setup: Table 2 of Appendix D). Figure 8 presents the CEMA results for the time series measured at all grid points, as well as from the targeted point using ProbeShooter. The significant difference in the CEMA results demonstrates that ProbeShooter is capable of performing precise probe aiming to capture exploitable EM leakage. Noteworthy is that the time series measurements and CEMA for 49 points took 20 hours, while ProbeShooter—even having a much higher spatial resolution (101×101)—took only 1.6 hours; the time required for probe aiming has been reduced by 92%, while the key recovery performance at the extracted points has improved by 86.6% (time: 20→1.6 hours, GE: 41.9375→5.625).

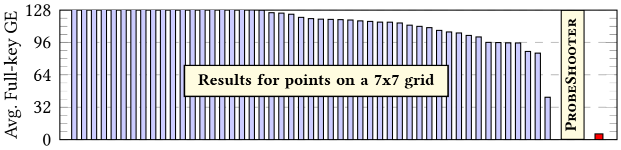


Figure 8: Comparison of CEMA results for the i.MX RT1061 (7x7 grid vs. ProbeShooter). The bars (in descending order) represent the full-key guessing entropy of AES-128, with lower values indicating better performance. The red bar on the far right represents the results of ProbeShooter.

[Exp.#S2: Remote [29] on BCM2711] For the experiment, we defined the `BasicMath` benchmark software from MiBench [10] as normal behavior and executed it at 1.2 GHz on the BCM2711 (Table 3 of Appendix D). Then, we measured the EM signals and profiled (trained) them using the Remote framework. During the execution of `BasicMath`, we invoked an unprofiled function (i.e., abnormal) and evaluated the F1 score of a binary classifier to determine the presence of anomalies. Figure 9 shows the F1 scores for all points on the 7×7 surface and those extracted using ProbeShooter. The results also demonstrates that ProbeShooter is capable of performing precise probe aiming to capture exploitable EM leakage. And we present the F1 score map evaluated with a finer granularity of 21×21 for each core in Figure 15 of Appendix E.

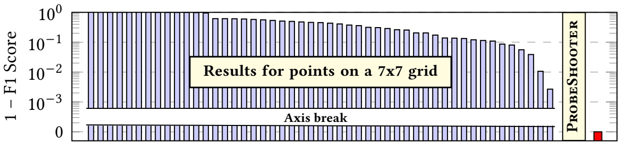


Figure 9: F1 score comparison of the Remote on BCM2711 (7x7 grid vs. ProbeShooter). The bars represent the 1 − F1 Score on a log scale, with lower values indicating better performance. The red bar on the far right represents the results of ProbeShooter.

4.3 Robustness

In this subsection, we demonstrate the following: *"ProbeShooter is **not susceptible** to various factors and can provide consistent results."*

[Exp.#R1: Reproducibility] To demonstrate the reproducibility of ProbeShooter, we conducted probe aiming five times under identical conditions on both the i.MX RT1061 and BCM2711 (detailed setup: Table 4 of Appendix D). As a result, we present the deviations of aiming points in Figure 10, and the detailed distribution in Figure 16 of Appendix E. The deviations resulting from this experiment will serve as a reference for analyzing the impact of other factors.

[Exp.#R2: Clock Frequency] For the experiment, we perform probe aiming on both the i.MX RT1061 and BCM2711 operating at various frequencies: 24, 73.5, 196, 294, and 588 MHz for the i.MX RT1061 and 0.6, 0.8, 1.0, 1.2, and 1.4 GHz for the BCM2711 (detailed setup: Table 5 of Appendix D). As a result, we present the distribution of aiming points in Figure 17 of Appendix E, and the deviations in Figure 10. This indicates that there is a deviation in the aiming results that exceeds the level of re-attempts, depending on the operational clock of the DUT. However, as this deviation still averages within a Euclidean distance of 0.5 points (i.MX RT1061 < 60 μm, BCM2711 < 35 μm) and remains within the *type3_hump*, it is considered acceptable.

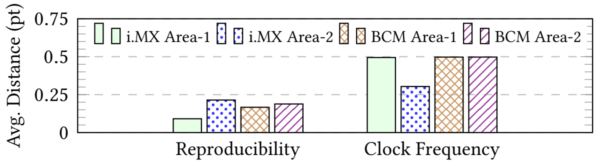


Figure 10: Comparison of probe aiming deviations for each DUT and robustness evaluation. The bars represent the average Euclidean distance from centroid.

4.4 Multi-Core System

[Exp.#M1: Multi-Core System] For the experiment on the multi-core system, we *simultaneously executed* different gadgets on *all cores* of the BCM2711 through affinity configuration. Gadgets were executed on `core0` to `core3` with iterations of 7, 11, 13, and 17 cycles, respectively. The detailed code for the gadgets and the experimental setups are provided in Table 6 of Appendix D. We present the results of leakage combination and probe aiming using a single PSD chunk in Figure 11. This demonstrates that PS-P can simultaneously identify leakage from each core in a multi-core system.

To substantiate the accuracy of probe aiming for each core, experimental results from the SpectrEM [9] can be referenced. The authors have presented the results of SpectrEM attacks on the surface of the same BCM2711 chip, depicted as BER (Bit Error Rate) maps for each core. Our probe aiming results show a close similarity to the BER maps from SpectrEM (keep in mind the difference in the boundaries of the scans). Additionally, the results in §4.2 demonstrate the applicability of our method to multi-core systems through the F1 score maps for all cores presented in Figure 15 of Appendix E. However, due to the lack of refinement and significant variance in these F1 score maps, it may be more effective to compare the probe aiming results with SpectrEM's BER maps.

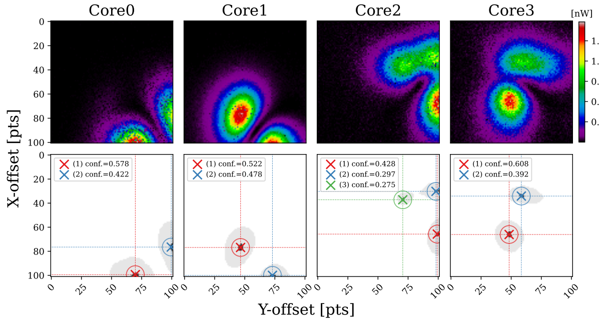


Figure 11: Leakage maps and probe aiming results for all cores (Core0 to Core3) of BCM2711. The four cores were simultaneously targeted using a single psd chunk.

4.5 Dynamic Frequency Scaling

[Exp.#F1: Frequency Scaling] To demonstrate the applicability of ProbeShooter to systems with dynamic frequency scaling, we create a simulated PSD chunk by combining PSD measurements taken with a fixed clock. The reason is that in our experimental setup (BCM2711 / Dedicated Linux), high core utilization often leads to higher clock frequencies, reducing the effectiveness of DVFS; it is challenging to have frequency scaling features fully utilized in our environment. Nevertheless, it remains meaningful as it may vary depending on the frequency scaling policies or architecture.

To combine the simulated PSD chunk, we repeatedly measured each PSD chunk while fixing the clock to all frequencies defined in the valid OPPs—ranging from 600 MHz to 1.5 GHz in 100 MHz increments. Subsequently, we replaced the PSDs for all locations on the chip surface with those corresponding to random clock frequencies. We present the leakage maps and probe aiming results based on the accuracy of the `estimate_clock_freq` (Algorithm 2) in Figure 12. Our results show that ProbeShooter can be effectively applied to systems with frequency scaling, provided that the clock estimation accuracy is approximately 75–80% or higher. The detailed setup is described in Table 7 of Appendix D.

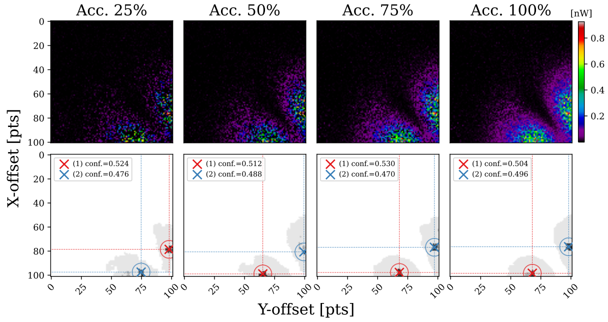


Figure 12: Leakage map and probe aiming results for Core0 of BCM2711 with frequency scaling, based on the accuracy of the clock guessing function (estimate_clock_freq).

4.6 Uncontrollable DUT

For the experiment, the inherent clock gating feature of the BCM2711 requires the assumption that a dominant process is running on the DUT to successfully perform PS-A (§3.2.4). Therefore, we conducted experiments with `Core1` of the uncontrollable BCM2711 running repeated AES-128 encryptions to increase core utilization. It is important to note that no knowledge of the internal software (AES-128) is required, nor is there any need for control to execute dedicated code, such as gadgets for PS-P. In contrast, the i.MX RT1061 does not require such considerations.

[Exp.#A1: Soundness & Robustness] For the experiment, we operated the i.MX RT1061 at 24 MHz and the BCM2711 at 1.2 GHz, injecting signals of 500 kHz and 10 MHz, respectively (Table 8 of Appendix D). We then performed probe aiming five times for each case in identical environments. As a result, we present the deviations of aiming points in Figure 13, and the detailed distribution in Figure 18 of Appendix E. The reproduction error and the deviation from the PS-P results in the area showing the highest confidence are both less than 0.5 points—similar to the results in §4.3—and are therefore acceptable (i.MX RT1061 < 60 μm, BCM2711 < 35 μm).

[Exp.#A2: Robustness to Injected Signal Frequency] For the experiment, we injected signals of different frequencies into the i.MX RT1061 and BCM2711, respectively : 0.1, 0.5, 3, 5, and 10 MHz for the i.MX RT1061 and 5, 10, 20, 30, and 40 MHz for the BCM2711 (detailed setup: Table 9 of Appendix D). As a result, we present the deviations of aiming points in Figure 13, and the detailed distribution in Figure 19 of Appendix E. The deviation corresponding to the frequency of the injected signals is within 0.3 points—which is at a level comparable to the reproduction error of the PS-P—and is thus acceptable (i.MX RT1061 < 36 μm, BCM2711 < 21 μm).

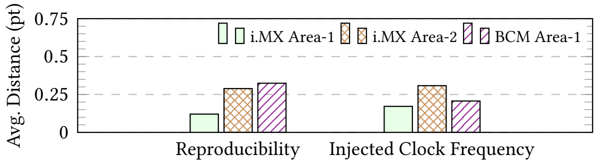


Figure 13: Comparison of deviations in probe aiming results for uncontrollable DUT. The bars represent the average Euclidean distance from centroid.

[Exp.#A3: Multi-Core System] We first presented the probe aiming results for a dominant process operating on a single core of the BCM2711 in Figure 20(a) of Appendix E. And the probe aiming results for dominant processes running simultaneously on both `Core1` and `Core2` are shown in Figure 20(b) and Figure 20(c).

In PS-A, the IMD peak frequencies of all cores are identical, resulting in overlapping leakage maps. Consequently, subsequent efforts are required to identify each individual core from the candidate points in the probe aiming results. Additionally, for specific chips, injecting a signal can induce intermodulation distortion near the voltage regulator, potentially leading to false positives. Since this occurs consistently at the same location, it can be easily identified and mitigated through post-processing techniques, such as removing these peaks.

Table 1: Comparison with related work.

Reference	Underlying Methods	Target	Feasibility	Multi-Core*	Freq. Scaling	Full coverage	No Control Required	Performance Improvement†
Exhaustive Search	🔑	🔒	○	✗	✗	✓	✗	1×
Daniel et. al [8]	🔑+ Greedy alg.	🔒	◔	✗	✗	✗	✗	$<$ 100×
Iyer et. al [14]	σ + Greedy alg.	🔒	◑	✗	✗	✗	✗	$<$ 100×
Jiang et. al [15]	$\hat{\sigma}$ + Gradient desc.	🔒	◑	✗	✗	✗	✗	$<$ 100×
ProbeShooter-P	📊 + AM	♥	●	✓	✓	✓	✗	530×
ProbeShooter-A	📊 + IMD	♥	●	✓‡	✓	✓	✓	470×

The symbols in the table include: [🔑] Known key recovery, [σ] Analysis of Variance (ANOVA) with $F-$test, [$\hat{\sigma}$] A variance-based approach, [📊] Frequency sweep; [🔒] Cryptographic implementation, [♥] CPU core; [○–●] Feasibility level; [✓] Supported, [✗] Unsupported.

* Leakage localization at the individual core level when two or more cores operate simultaneously.

† Targeting the i.MX RT1061 with 5k time series at a granularity of 101×101.

‡ Leakage localization is performed by overlaying individual leakage maps of all active cores.

5 Related Work

Previous studies have generally adopted two main approaches: reducing the number of location candidates and decreasing the cost of quantifying leakage at a single point [8, 14, 15]. J. Danial et al. [8] utilized key recovery performance as a leakage metric and reduced location candidates using a greedy-gradient search algorithm. V. Iyer et al. [14] also reduced location candidates using a greedy search algorithm. They further improved performance by replacing the leakage metric with an Analysis of Variance (ANOVA) approach. M. Jiang et al. [15] reduced location candidates using gradient descent, similar to [14], while applying an ANOVA approach; however, they used standard deviation values directly instead of an F-test. Greedy search and gradient descent do not guarantee full coverage and may get stuck in local minima. Moreover, in cases with highly localized leakage—such as our i.MX RT1061 target—performance improvements are limited. Targeting data leakage in cryptographic implementations is also not feasible for complex SoCs, as identifying leakage at a single point can be highly time-consuming.

F. Werner et al. [35, 36] modeled approximate leakage sources based on signals measured at the edges of the PCB using Nelder-Mead simplex optimization. However, because this approach focuses on EMI/EMC, it is conducted at the PCB level rather than the chip level, limiting its applicability in the context of EMSCA. B. Hou et al. [13] identified leakage sources on the chip by clustering frequencies with similar spatial coherence using the DBSCAN algorithm. However, this approach is impractical for complex SoCs where numerous components operate at diverse frequencies within a silicon die. Moreover, it has not been demonstrated whether exploitable leakage can be identified within the data processing logic. We compare ProbeShooter with the literature in the same research category, as shown in Table 1.

6 Discussion

Applicability to Hardware Cryptographic Implementation. ProbeShooter enables EM leakage cartography and probe aiming for *CPU cores*. However, in the context of cryptanalysis, further research is required to identify leakages in hardware implementations such as FPGAs, ASICs, and cryptographic co-processors. Nevertheless, many cryptographic algorithms are still executed on CPU cores—particularly new schemes, public key cryptography, and proprietary standard schemes of specific nations—and EM side-channel has numerous applications beyond cryptanalysis, which makes ProbeShooter highly valuable. In fact, numerous studies on optimized cryptographic implementations for specific CPU architecture provide evidence supporting this claim [16, 20].

Multi-core Discernibility of PS-A. PS-P allows the evaluator to directly execute gadgets, enabling the acquisition of leakage maps for all individual cores in a multi-core system by running gadgets on each core separately. However, for PS-A, if a dominant process is active on two or more cores simultaneously, the leakage maps for these cores are obtained in an overlapping manner, which may make it difficult to separate the leakage maps for individual cores. This is because we cannot access the individual power rails of each core separately from the others. Therefore, further work is required to achieve spatial separation of individual cores' leakage.

High Budgetary Costs of Test Bench. ProbeShooter requires a spectrum analyzer, a motorized XYZ table, and a signal generator. These instruments are generally costly, which can make it challenging to set up a test bench. However, it is possible to mitigate the cost constraints by setting up a test bench using relatively inexpensive equipments. For example, a spectrum analyzer and a motorized XYZ table can be replaced by a Software-Defined Radio (SDR) and 3D printer-based equipment [8], respectively. Such alternative equipment must take into account their lower performance. In particular, SDRs may be limited by their RBW and bandwidth, while non-SCA-specific motorized XYZ tables may exhibit restricted precision and position reproducibility.

7 Conclusion

In this paper, we proposed ProbeShooter, a novel methodology for leakage localization and probe aiming that leverages the spatial characteristics of AM and IMD. Our approach addresses the limitations of previous methods by providing a practical solution applicable to both high-end MCUs and complex SoCs. Through experimental validation, we demonstrated that ProbeShooter outperforms previous approaches in terms of feasibility, practicality, and performance.

Our findings highlight the potential of ProbeShooter to significantly advance EMSCA-related research by enabling precise capture of information leakage from CPU cores with high SNR. This capability not only facilitates academic research but also enhances the feasibility of real-world attacks in constrained environments. Furthermore, the versatility of ProbeShooter allows for its application across various systems, including those with multi-core architectures and dynamic frequency scaling. Future work could explore extending this methodology to other hardware targets, such as FPGAs, GPUs, NPUs and cryptographic co-processors.

Availability

ProbeShooter is available at:
https://github.com/ProbeShooter/AsiaCCS25-ProbeShooter

Acknowledgments

This work was supported by the Agency for Defense Development by the Korean government (No. UG233015TD).

References

[1] Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao, and Pankaj Rohatgi. 2003. The EM Side—Channel(s). In *Proceedings of the Cryptographic Hardware and Embedded Systems* (Redwood Shores, CA, USA) *(CHES '02)*. Springer, Berlin, Heidelberg, 29–45. doi:10.1007/3-540-36400-5_4

[2] Christos Andrikos, Lejla Batina, Lukasz Chmielewski, iran Lerman, Vasilios Mavroudis, Kostas Papagiannopoulos, Guilherme Perin, Giorgos Rassias, and Alberto Sonnino. 2019. Location, Location, Location: Revisiting Modeling and Exploitation for Location-Based Side Channel Leakages. In *Advances in Cryptology – ASIACRYPT 2019* (Kobe, Japan). Springer International Publishing, Cham, Switzerland, 285–314. doi:10.1007/978-3-030-34618-8_10

[3] Md Sadik Awal and Md Tauhidur Rahman. 2023. Disassembling Software Instruction Types through Impedance Side-channel Analysis. In *2023 IEEE International Symposium on Hardware Oriented Security and Trust* (San Jose, CA, USA) *(HOST '23)*. IEEE, New York, NY, USA, 227–237. doi:10.1109/HOST55118.2023.10133318

[4] Jakub Breier and Xiaolu Hou. 2022. How Practical Are Fault Injection Attacks, Really? *IEEE Access* 10 (2022), 113122–113130. doi:10.1109/ACCESS.2022.3217212

[5] Robert Callan, Alenka Zajić, and Milos Prvulovic. 2014. A Practical Methodology for Measuring the Side-Channel Signal Available to the Attacker for Instruction-Level Events. In *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture* (Cambridge, United Kingdom) *(MICRO '14)*. IEEE Computer Society, USA, 242–254. doi:10.1109/MICRO.2014.39

[6] Robert Callan, Alenka Zajić, and Milos Prvulovic. 2015. FASE: Finding Amplitude-Modulated Side-Channel Emanations. In *Proceedings of the 42nd Annual International Symposium on Computer Architecture* (Portland, Oregon) *(ISCA '15)*. Association for Computing Machinery, New York, NY, USA, 592–603. doi:10.1145/2749469.2750394

[7] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. 2003. Template Attacks. In *Cryptographic Hardware and Embedded Systems* (Redwood Shores, CA, USA) *(CHES '02)*. Springer Berlin Heidelberg, Berlin, Heidelberg, 13–28. doi:10.1007/3-540-36400-5_3

[8] Josef Danial, Debayan Das, Santosh Ghosh, Arijit Raychowdhury, and Shreyas Sen. 2020. SCNIFFER: Low-Cost, Automated, Efficient Electromagnetic Side-Channel Sniffing. *IEEE Access* 8 (2020), 173414–173427. doi:10.1109/ACCESS.2020.3025022

[9] Jesse De Meulemeester, Antoon Purnal, Lennert Wouters, Arthur Beckers, and Ingrid Verbauwhede. 2023. SpectrEM: Exploiting Electromagnetic Emanations During Transient Execution. In *32nd USENIX Security Symposium* (Anaheim, CA, USA) *(USENIX Security '23)*. USENIX Association, Berkeley, CA, USA, 6293–6310.

[10] M.R. Guthaus, J.S. Ringenberg, D. Ernst, T.M. Austin, T. Mudge, and R.B. Brown. 2001. MiBench: A free, commercially representative embedded benchmark suite. In *Proceedings of the Fourth Annual IEEE International Workshop on Workload Characterization. WWC-4 (Cat. No.01EX538)* (Austin, TX, USA) *(WWC '01)*. IEEE Computer Society, USA, 3–14. doi:10.1109/WWC.2001.990739

[11] Gregor Haas and Aydin Aysu. 2022. Apple vs. EMA: Electromagnetic Side Channel Attacks on Apple CoreCrypto. In *Proceedings of the 59th ACM/IEEE Design Automation Conference* (San Francisco, CA, USA) *(DAC '22)*. Association for Computing Machinery, New York, NY, USA, 247–252. doi:10.1145/3489517.3530437

[12] Johann Heyszl, Dominik Merli, Benedikt Heinz, Fabrizio De Santis, and Georg Sigl. 2013. Strengths and Limitations of High-Resolution Electromagnetic Field Measurements for Side-Channel Analysis. In *Smart Card Research and Advanced Applications* (Berlin, Germany) *(CARDIS '12)*. Springer Berlin Heidelberg, Berlin, Heidelberg, 248–262. doi:10.1007/978-3-642-37288-9_17

[13] Bo Hou, Rui Ding, Weiheng Shao, Siyang Liu, and Liwei Wang. 2024. Pattern clustering method of magnetic near-field radiation emissions based on DBSCAN algorithm. *IET Science, Measurement & Technology* (2024), 14. doi:10.1049/smt2.12182

[14] Vishnuvardhan V. Iyer and Ali E. Yılmaz. 2022. An ANOVA Method to Rapidly Assess Information Leakage Near Cryptographic Modules. *IEEE Transactions on Electromagnetic Compatibility* 64, 4 (2022), 915–929. doi:10.1109/TEMC.2022.3157664

[15] Minmin Jiang. 2023. *Analysis and Mitigation of EM Side-Channel Attacks on Chip-to-Chip Interconnects*. Ph. D. Dissertation. University of Manchester, UK.

[16] Matthias J. Kannwischer, Joost Rijneveld, and Peter Schwabe. 2019. Faster Multiplication in Z2m[x] on Cortex-M4 to Speed up NIST PQC Candidates. In *Applied Cryptography and Network Security* (Bogota, Colombia) *(ACNS '19)*, Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung (Eds.). Springer International Publishing, Cham, 281–301. doi:10.1007/978-3-030-21568-2_14

[17] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. 2019. Spectre Attacks: Exploiting Speculative Execution. In *2019 IEEE Symposium on Security and Privacy* (San Francisco, CA, USA) *(S&P '21)*. IEEE, New York, NY, USA, 1–19. doi:10.1109/SP.2019.00002

[18] kokke. 2024. tiny-AES-c. https://github.com/kokke/tiny-AES-c [Accessed: Aug. 2024].

[19] Amit Kumar, Cody Scarborough, Ali Yilmaz, and Michael Orshansky. 2017. Efficient simulation of EM side-channel attack resilience. In *Proceedings of the 36th International Conference on Computer-Aided Design* (Irvine, CA, USA) *(ICCAD '17)*. IEEE Press, NY, USA, 123–130. doi:10.1109/ICCAD.2017.8203769

[20] Hyeokdong Kwon, Hyunjun Kim, Siwoo Eum, Minjoo Sim, Hyunji Kim, Wai-Kong Lee, Zhi Hu, and Hwajeong Seo. 2022. Optimized Implementation of SM4 on AVR Microcontrollers, RISC-V Processors, and ARM Processors. *IEEE Access* 10 (2022), 80225–80233. doi:10.1109/ACCESS.2022.3195217

[21] J. Longo, E. De Mulder, D. Page, and M. Tunstall. 2015. SoC It to EM: Electro-Magnetic Side-Channel Attacks on a Complex System-on-Chip. In *Proceedings of the Cryptographic Hardware and Embedded Systems* (Saint-Malo, France) *(CHES '15)*. Springer, Berlin, Heidelberg, 620–640. https://doi.org/10.1007/978-3-662-48324-4_31

[22] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. 2008. *Power analysis attacks: Revealing the secrets of smart cards*. Vol. 31. Springer Science & Business Media, New York, NY, USA.

[23] Saleh Khalaj Monfared, Tahoura Mosavirik, and Shahin Tajik. 2023. LeakyOhm: Secret Bits Extraction using Impedance Analysis. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security* (Copenhagen, Denmark) *(CCS '23)*. Association for Computing Machinery, New York, NY, USA, 1675–1689. doi:10.1145/3576915.3623092

[24] Alireza Nazari, Nader Sehatbakhsh, Monjur Alam, Alenka Zajic, and Milos Prvulovic. 2017. EDDIE: EM-Based Detection of Deviations in Program Execution. In *Proceedings of the 44th Annual International Symposium on Computer Architecture* (Toronto, ON, Canada) *(ISCA '17)*. Association for Computing Machinery, New York, NY, USA, 333–346. doi:10.1145/3079856.3080223

[25] Stjepan Picek, Annelie Heuser, Guilherme Perin, and Sylvain Guilley. 2022. Profiled Side-Channel Analysis in the Efficient Attacker Framework. In *Smart Card Research and Advanced Applications* (Lübeck, Germany) *(CARDIS '21)*. Springer International Publishing, Cham, 44–63. doi:10.1007/978-3-030-97348-3_3

[26] Ryan Roemer, Erik Buchanan, Hovav Shacham, and Stefan Savage. 2012. Return-Oriented Programming: Systems, Languages, and Applications. *ACM Trans. Inf. Syst. Secur.* 15, 1, Article 2 (mar 2012), 34 pages. doi:10.1145/2133375.2133377

[27] Asanka Sayakkara, Nhien-An Le-Khac, and Mark Scanlon. 2019. A survey of electromagnetic side-channel attacks and discussion on their case-progressing potential for digital forensics. *Digital Investigation* 29 (2019), 43–54. doi:10.1016/j.diin.2019.03.002

[28] Asanka P. Sayakkara and Nhien-An Le-Khac. 2021. Electromagnetic Side-Channel Analysis for IoT Forensics: Challenges, Framework, and Datasets. *IEEE Access* 9 (2021), 113585–113598. doi:10.1109/ACCESS.2021.3104525

[29] Nader Sehatbakhsh, Alireza Nazari, Monjur Alam, Frank Werner, Yuanda Zhu, Alenka Zajic, and Milos Prvulovic. 2020. REMOTE: Robust External Malware Detection Framework by Using Electromagnetic Signals. *IEEE Trans. Comput.* 69, 3 (2020), 312–326. doi:10.1109/TC.2019.2945767

[30] Nader Sehatbakhsh, Alireza Nazari, Alenka Zajic, and Milos Prvulovic. 2016. Spectral profiling: Observer-effect-free profiling by monitoring EM emanations. In *The 49th Annual IEEE/ACM International Symposium on Microarchitecture* (Taipei, Taiwan) *(MICRO '16)*. IEEE Press, NY, USA, Article 59, 11 pages. doi:10.1109/MICRO.2016.7783762

[31] Carlton Shepherd, Konstantinos Markantonakis, Nico van Heijningen, Driss Aboulkassimi, Clément Gaine, Thibaut Heckmann, and David Naccache. 2021. Physical fault injection and side-channel attacks on mobile devices: A comprehensive analysis. *Computers & Security* 111 (2021), 102471. doi:10.1016/j.cose.2021.102471

[32] Takeshi Sugawara, Daisuke Suzuki, Minoru Saeki, Mitsuru Shiozaki, and Takeshi Fujino. 2013. On Measurable Side-Channel Leaks Inside ASIC Design Primitives. In *Cryptographic Hardware and Embedded Systems* (Santa Barbara, CA, USA) *(CHES '13)*, Guido Bertoni and Jean-Sébastien Coron (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 159–178. doi:10.1007/978-3-642-40349-1_10

[33] NewAE Tech. 2022. ChipWhisperer CW308 Targets. https://github.com/newaetech/chipwhisperer-target-cw308t [Accessed: Aug. 2024].

[34] Aurélien Vasselle, Philippe Maurine, and Maxime Cozzi. 2019. Breaking Mobile Firmware Encryption through Near-Field Side-Channel Analysis. In *Proceedings of the 3rd ACM Workshop on Attacks and Solutions in Hardware Security Workshop* (London, United Kingdom) *(ASHES '19)*. Association for Computing Machinery, New York, NY, USA, 23–32. doi:10.1145/3338508.3359571

[35] Frank Werner, Derrick Albert Chu, Antonije R. Djordjević, Dragan I. Olćan, Milos Prvulovic, and Alenka Zajić. 2018. A Method for Efficient Localization of Magnetic Field Sources Excited by Execution of Instructions in a Processor. *IEEE Transactions on Electromagnetic Compatibility* 60, 3 (2018), 613–622. doi:10.1109/TEMC.2017.2742501

[36] Frank T. Werner, Jelena Dinkić, Dragan Olćan, Antonije Djordjević, Milos Prvulović, and Alenka Zajić. 2021. An Efficient Method for Localization of Magnetic Field Sources That Produce High-Frequency Side-Channel Emanations. *IEEE Transactions on Electromagnetic Compatibility* 63, 6 (2021), 1799–1811. doi:10.1109/TEMC.2021.3063657

[37] Alenka Zajić and Milos Prvulovic. 2014. Experimental Demonstration of Electromagnetic Information Leakage From Modern Processor-Memory Systems. *IEEE Transactions on Electromagnetic Compatibility* 56, 4 (2014), 885–893. doi:10.1109/TEMC.2014.2300139

A Nonlinear Distortion (NLD)

Transistors—the building blocks of digital processors—are representative *nonlinear* electronic components. This implies that the relationship between input and output variables deviates from linearity in terms of voltage, current, or other relevant parameters. When a radio frequency signal passes through a nonlinear component (or circuit), additional frequencies beyond the fundamental frequency are introduced as a consequence of signal deformation, known as nonlinear distortion (NLD). The most prevalent forms of NLD include intermodulation distortion (IMD).

A.1 Intermodulation Distortion (IMD)

When two or more frequencies of signals interact in a nonlinear system, integer multiples of each fundamental frequency are generated. And the sum and difference frequencies between the fundamental frequencies and their integer multiples are generated. All these signals superposed with the fundamental signal, causing distortion, and this is referred to as IMD. The time-varying signal $V_{\text{IMD}}(t)$ distorted by intermodulation products (IMPs) of fundamental signals $(f_1, f_2, \cdots, f_n)$ is as follows:

$$V_{\text{IMD}}(t) = \overbrace{V_{f_1}(t) + V_{f_2}(t) + \cdots + V_{f_n}(t)}^{\text{Fundamental signals}}$$
$$+ \overbrace{\sum_{o=2}^{\infty}\left(\sum_{\mathbf{k}\in\mathcal{K}_o} V_{k_1 f_1 + k_2 f_2 + \cdots + k_n f_n}(t)\right)}^{\text{Intermodulation products}}$$

where o and $\mathbf{k}$ denote the intermodulation order and the coefficient vector ($\mathbf{k}$=$(k_1, k_2, \cdots, k_n)$), respectively. And, $\mathcal{K}_o$ is defined as follows for a given intermodulation order o:

$$\mathcal{K}_o = \left\{\mathbf{k}\,\middle|\, |k_1| + |k_2| + \cdots + |k_n| = o, k_i \in \mathbf{Z}\right\}$$

For instance, when considering two frequencies, f_1 and f_2, all (k_1, k_2) pairs representing the 2nd order (o=2) IMPs are (0, 2), (0, -2), (2, 0), (-2, 0), (1, 1), (1, -1), (-1, 1), and (-1, -1). For these pairs, taking the absolute value of negative frequencies and removing duplicate frequencies yields the 2nd order IMPs: $2{\cdot}f_1$, $2{\cdot}f_2$, $f_1{+}f_2$, $f_1{-}f_2$, and $f_2{-}f_1$.

Commonly encountered form of IMD often reflects scenarios in the field of mobile communications, where two frequencies are in close proximity. In the case of IMD between adjacent frequencies, odd-order IMDs may affect the quality of the signal, while even-order IMDs—being far away in frequency domain—are out of interest. In contrast to adjacent fundamental frequencies, we intend to analyze the IMD that occurs between the *low-frequency* signals—generated by an external signal generator—and the *high-frequency* clock signals. In this case, all intermodulation products occur close to the fundamental frequencies, regardless of whether the order is even or odd. Figure 14 shows the frequency analysis results of the intermodulation-distorted signal $V_{\text{IMD}}(t)$ caused by interaction between the low (f_1) and the high frequency (f_2) signals.

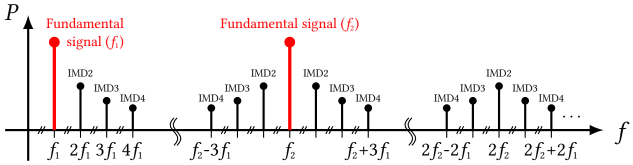


Figure 14: The periodogram of intermodulation-distorted signal (up to 4th order IMD). The humps—consisting of frequency peaks—centered around 3·f$_2$ and 4·f$_2$ are omitted.

B PSD Chunk and Spectrum Cartography

To understand the mechanism of the ProbeShooter, we introduce PSD chunk and spectrum cartography. A PSD chunk ($\mathcal{D}$) refers to the frequency domain data of the entire chip surface measured at both PS-P and PS-A, and is obtained using a spectrum analyzer. To acquire the PSD chunk, first we divide the chip surface into an $m{\times}n$ grid of points. Here, the grid must include the vertices and edges of the chip (die) to ensure full coverage. For instance, dividing a 12×12mm surface into an 11×11 grid requires an axis spacing of 1.2mm for each point. And we obtain a PSD chunk by measuring the PSD within the frequency range of f_a to f_b for all points. In this way, the PSD chunk is composed of spectral maps ($\mathcal{M}$) corresponding to all valid discrete frequencies between f_a to f_b. That is, the map $\mathcal{M}^{f_t}$ for the frequency f_t—within the range f_a to f_b—is defined as follows:

$$\mathcal{M}^{f_t}_{m\times n} = \begin{bmatrix} S^{1,1}_{xx}(f_t) & S^{1,2}_{xx}(f_t) & \dots & S^{1,n}_{xx}(f_t) \\ S^{2,1}_{xx}(f_t) & S^{2,2}_{xx}(f_t) & \dots & S^{2,n}_{xx}(f_t) \\ \vdots & \vdots & \ddots & \vdots \\ S^{m,1}_{xx}(f_t) & S^{m,2}_{xx}(f_t) & \dots & S^{m,n}_{xx}(f_t) \end{bmatrix}$$

where $S^{\alpha,\beta}_{xx}$ denotes the power spectral density—measured from a spectrum analyzer—of the time-varying signal $x(t)$ at the point (α, β). The $S^{\alpha,\beta}_{xx}(f_t)$ represents power density of the frequency f_t in a *linear scale* typically at levels ranging from several pW to μW. And we represent the map with the point $(1, 1)$ always facing towards the *origin* of the chip.

C Cyclic Instruction Gadget

C.1 Measuring Clock Cycles of a Gadget

This appendix presents a method for measuring the clock cycles of a gadget used on the BCM2711. To measure the clock cycles of the gadget, we utilized the Listing 3. This code accesses the ARM core's PMU (Performance Monitoring Unit) to obtain the clock cycles after executing the gadget in a loop one million times. Although conditional branches are present, the branch prediction and various optimization features of the BCM2711 enable them to execute almost like unconditional branches. In other words, the resulting clock cycles are similar to those obtained when using an unconditional branch (i.e., infinite loop). In practice, if the branch prediction accuracy is high, the difference is only a few dozen cycles; even if the accuracy is low, the difference does not exceed one million cycles. Therefore, by dividing the result from Listing 3 by 1 million and applying the floor function, we can estimate the clock cycles of a gadget in an infinite loop configuration.

```
void print_gadget_cycles() {
    uint64_t diff;
    asm volatile (  // Initialize iteration (1,000,000)
        "mov x4, #1000   \n"
        "mul x4, x4, x4 \n");
    asm volatile (  // Initialize operands of the gadget
        "mov x7, #0xffffffffffffffff \n"
        "mov x8, #0xff11              \n");
    asm volatile ("msr pmcr_el0, %0" : : "r"(0x00000001));
    asm volatile ("mrs x0, pmccntr_el0");
    asm volatile (
        "gadget:            \n"
// ==================== Gadget below ====================
        "udiv x6, x7, x8 \n"
// ======================================================
        "subs x4, x4, #1 \n"
        "bne gadget       \n");
    asm volatile (
        "mrs x1, pmccntr_el0 \n"
        "sub x2, x1, x0       \n"
        "mov %0, x2" : "=r"(diff));
    printf("Cycles: %llu\n", diff);
}
```

Listing 3: Code for measuring the gadget's clock cycles. The value obtained by dividing the result by 1 million and applying a floor function is used as the clock cycle of the gadget.

C.2 Gadgets for NXP i.MX RT1061

This appendix presents the source code for the gadgets used in the experiments on the NXP i.MX RT1061.

```
gadget:
push {r1}
push {r2}
b.n  gadget
```

Listing 4: Code for the `IMXRT1061-PUSHPOP-7C` gadget.

C.3 Gadgets for Broadcom BCM2711

This appendix presents the source code for the gadgets used in the experiments on the Broadcom BCM2711.

```
mov x1, #0x1000
mov x2, #0x3
gadget:
udiv x0, x1, x2
b gadget
;
;
;
```

Listing 5: Code for the `BCM2711-UDIV-7C` gadget.

```
mov   x1, #0
movk  x1, #0x0078, LSL #32
movk  x1, #0x1234, LSL #16
movk  x1, #0x5678
mov   x2, #0x1234
gadget:
udiv  x0, x1, x2
b gadget
```

Listing 6: Code for the `BCM2711-UDIV-11C` gadget.

```
mov   x1, #0
movk  x1, #0x5678, LSL #32
movk  x1, #0x1234, LSL #16
movk  x1, #0x5678
mov   x2, #0x1234
gadget:
udiv  x0, x1, x2
b gadget
```

Listing 7: Code for the `BCM2711-UDIV-13C` gadget.

```
mov x1, #0xffffffffffffffff
mov x2, #0xff11
gadget:
udiv x0, x1, x2
b gadget
;
;
;
```

Listing 8: Code for the `BCM2711-UDIV-17C` gadget.

D Detailed Experimental Setups

This appendix presents a detailed summary of the experimental setups for each evaluation, organized in tabular form. Each table is distinguished by an experiment identifier in the format **Exp.#XX**.

Table 2: Detailed experimental setup for Exp.#S1.

Exp.#S1: CEMA on i.MX RT1061	
DUT	✻ Core clock freq.: 24 MHz
Gadget	`IMXRT1061-PUSHPOP-7C` (Listing 4)
SA Setup	✻ Sweep range: 1–48 MHz (Δ47 MHz) ✻ RBW / Freq. bin: 50 kHz / 9,030 bins ✻ PSD Averaging: No
Granularity	12×12 mm ⇒ 101×101 grid (↔120 μm, ↕120 μm) (⇒ PSD chunk of size 101×101×9030 / ≈351 MiB)
Duration	95.64 minutes
Evaluation (CEMA)	✻ Instrument: Teledyne LeCroy HDO6104B ✻ Implementation: `tiny-AES` [18] ✻ Eval. granularity: 7×7 (↕↔1, 714μm) ✻ #Measurements per location: 9,000 ✻ Sampling rate: 250 MS/s (AES-R1 / 29,137 samples) ✻ Leakage modeling: `HammingWeight(SBox[`$p \oplus k$`])` ✻ Duration: ≈20 hours

Table 3: Detailed experimental setup for Exp.#S2.

Exp.#S2: Remote on BCM2711	
DUT	✻ Core clock freq.: 1.2 GHz
Gadget	**`[Core1]`** `BCM2711-UDIV-17C` (Listing 8)
SA Setup	✻ Sweep range: 1.0–1.4 GHz (Δ400 MHz) ✻ RBW / Freq. bin: 50 kHz / 40,001 bins ✻ PSD Averaging: No
Granularity	7.48×6.64 mm ⇒ 101×101 grid (↕66.4μm, ↔74.8μm) (⇒ PSD chunk of size 101×101×40001 / ≈1.52 GiB)
Duration	245.7 minutes
Evaluation (Remote)	✻ Instrument (SDR): Aaronia SPECTRAN V6 Plus ✻ Center freq. (SDR): 1.225 GHz ✻ Sampling clock (SDR) : 46.08 MHz (upper sideband) ✻ Profiled SW: MiBench BasicMath [10] (on `Core1`) ✻ Eval. granularity: 7×7 (↕1.069mm, ↔0.948mm) ✻ Duration: ≈9 hours / Core

Table 4: Detailed experimental setup for Exp.#R1.

Exp.#R1: Reproducibility		
	i.MX RT1061	**BCM2711**
DUT	✻ Core clock freq.: 24 MHz	✻ Core clock freq.: 1.2 GHz
Gadget	✻ `IMXRT1061-PUSHPOP-7C`	✻ **`[Core0]`** `BCM2711-UDIV-17C`
SA Setup	✻ 1–48 MHz (Δ47 MHz) ✻ 50 kHz RBW / 9,030 bins ✻ PSD Averaging: No	✻ 1.0–1.4 GHz (Δ400 MHz) ✻ 50 kHz RBW / 40,001 bins ✻ PSD Averaging: No
Granularity	✻ 101×101 grid	✻ 101×101 grid
Duration	✻ 95.64m / PSD chunk	✻ 245.7m / PSD chunk

Table 5: Detailed experimental setup for Exp.#R2.

Exp.#R2: Clock Frequency		
	i.MX RT1061	**BCM2711**
DUT	✻ 24, 73.5, 196, 294, 588 MHz	✻ 0.6, 0.8, 1.0, 1.2, 1.4 GHz
Gadget	✻ `IMXRT1061-PUSHPOP-7C`	✻ **`[Core0]`** `BCM2711-UDIV-17C`
SA Setup	✻ Variable range (Δ47–192 MHz) ✻ 50 kHz RBW / 9,030–37,660 bins ✻ PSD Averaging: No	✻ Variable range (Δ200 MHz) ✻ 50 kHz RBW / 40,001 bins ✻ PSD Averaging: No
Granularity	✻ 101×101 grid	✻ 101×101 grid
Duration	✻ 95.64–231.8m / PSD chunk	✻ 229.49–245.7m / PSD chunk

Table 6: Detailed experimental setup for Exp.#M1.

Exp.#M1: Multi-Core System	
DUT	✻ Core clock freq.: 1.2 GHz
Gadgets	**`[Core0]`** `BCM2711-UDIV-7C` (Listing 5) **`[Core1]`** `BCM2711-UDIV-11C` (Listing 6) **`[Core2]`** `BCM2711-UDIV-13C` (Listing 7) **`[Core3]`** `BCM2711-UDIV-17C` (Listing 8)
SA Setup	✻ Sweep range: 1.0–1.4 GHz (Δ400 MHz) ✻ RBW / Freq. bin: 50 kHz / 40,001 bins ✻ PSD Averaging: No
Granularity	7.48×6.64 *mm* ⇒ 101×101 grid (↕66.4μm, ↔74.8μm) (⇒ PSD chunk of size 101×101×40001 / ≈1.52 GiB)
Duration	245.7 minutes

Table 7: Detailed experimental setup for Exp.#F1.

Exp.#F1: Frequency Scaling	
DUT	✻ Core clock freq.: 0.6–1.5 GHz (0.1 GHz increments)
Gadget	**`[Core0]`** `BCM2711-UDIV-17C` (Listing 8)
SA Setup	✻ Sweep range: 0.5–1.6 GHz (Δ1.1 GHz) ✻ RBW / Freq. bin: 50 kHz / 40,001 bins ✻ PSD Averaging: No
Granularity	7.48×6.64 *mm* ⇒ 101×101 grid (↕66.4μm, ↔74.8μm) (⇒ PSD chunk of size 101×101×40001 / ≈1.52 GiB)
Duration	303.7 minutes / PSD chunk

Table 8: Detailed experimental setup for Exp.#A1.

Exp.#A1: Soundness & Robustness		
	i.MX RT1061	**BCM2711**
DUT	✻ Core clock freq.: 24 MHz	✻ Core clock freq.: 1.2 GHz
SA Setup	✻ 23.4–23.6 MHz (Δ200 kHz) ✻ 1 kHz RBW / 2,010 bins ✻ PSD Averaging: No	✻ 1209.975–1210.025 MHz (Δ5 kHz) ✻ 0.51 kHz RBW / 1,000 bins ✻ PSD Averaging: 5
Granularity	✻ 101×101 grid	✻ 101×101 grid
Signal Injection	✻ Sine wave ✻ 500 *kHz* / 25 *dBm*	✻ Sine wave ✻ 10 *MHz* / 30 *dBm*
Duration	✻ 95.92m / PSD chunk	✻ 339.6m / PSD chunk

Table 9: Detailed experimental setup for Exp.#A2.

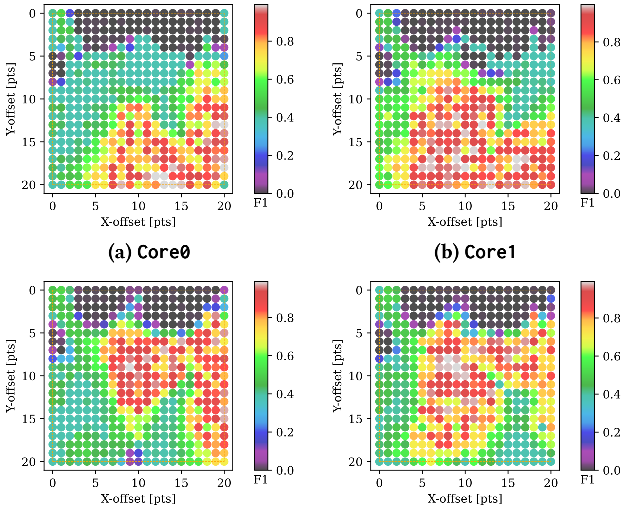
Exp.#A2: Robustness to Injected Signal Frequency		
	i.MX RT1061	**BCM2711**
DUT	✻ Core clock freq.: 24 MHz	✻ Core clock freq.: 1.2 GHz
SA Setup	✻ $f_c - f_{inj}$ ± 100 kHz (Δ200 kHz) ✻ 1 kHz RBW / 2,010 bins ✻ PSD Averaging: No	✻ $f_c - f_{inj}$ ± 5 kHz (Δ10 kHz) ✻ 0.51 kHz RBW / 210 bins ✻ PSD Averaging: 5
Granularity	✻ 101×101 grid	✻ 101×101 grid
Signal Injection	✻ 0.1, 0.5, 1, 3, and 10 *MHz* ✻ 25 *dBm* / Sine wave	✻ 5, 10, 20, 30, and 40 *MHz* ✻ 30 *dBm* / Sine wave
Duration	✻ 95.92m / PSD chunk	✻ 129.7m / PSD chunk

Table 10: Detailed experimental setup for Exp.#A3.

Exp.#A3: Multi-Core System	
DUT	✻ Core clock freq.: 1.2 GHz
SA Setup	✻ Sweep range: $f_c - f_{inj}$± 5 kHz (Δ10 kHz) ✻ RBW / Freq. bin: 0.51 kHz / 210 bins ✻ PSD Averaging: 5
Granularity	7.48×6.64 *mm* ⇒ 101×101 grid (↕66.4 μm, ↔74.8 μm) (⇒ PSD chunk of size 101×101×40001 / ≈8.17 MiB)
Signal Injection	Sine wave (10 *MHz*, 30 *dBm*, Z=50Ω)
Duration	129.7 minutes / PSD Chunk

E Detailed Experimental Results

This appendix presents detailed experimental results.



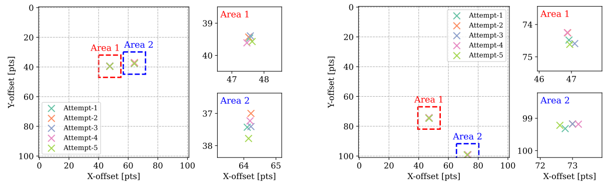
(a) `Core0`

(b) `Core1`

(c) `Core2`

(d) `Core3`

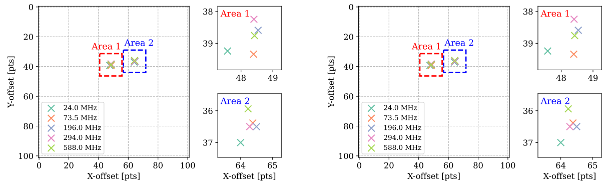
Figure 15: F1 score maps for all cores of the BCM2711 evaluated with a granularity of 21x21, showing a significant variance in the values. Additionally, the prominent areas generally appear larger because they utilize instruction leakages, which can be captured more easily than data leakages (Exp.#S2: Remote on BCM2711).



(a) Case: i.MX RT1061

(b) Case: BCM2711

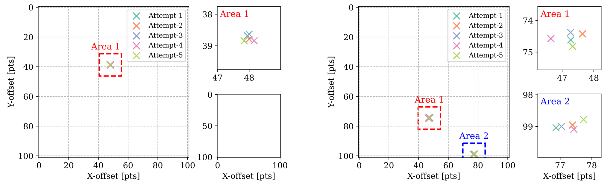
Figure 16: Results from repeatedly performing probe aiming in the same environment to evaluate robustness in terms of reproducibility (Exp.#R1: Reproducibility).



(a) Case: i.MX RT1061

(b) Case: BCM2711

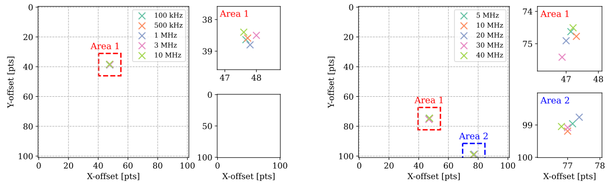
Figure 17: Probe aiming results for different core frequencies supported by each chip (Exp.#R2: Clock Frequency).



(a) Case: i.MX RT1061

(b) Case: BCM2711

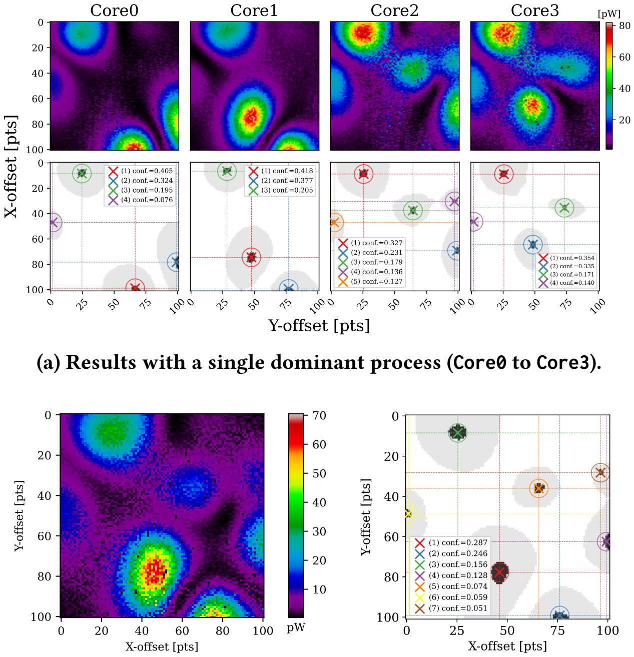
Figure 18: Results from repeatedly performing probe aiming using ProbeShooter-A in the same environment to evaluate robustness in terms of reproducibility (Exp.#A1: Soundness & Robustness).



(a) Case: i.MX RT1061

(b) Case: BCM2711

Figure 19: Probe aiming results for different injected signal frequencies of ProbeShooter-A (Exp.#A2: Robustness to Injected Signal Frequency).



(a) Results with a single dominant process (`Core0` to `Core3`).

(b) Overlapped leakage map

(c) Probe aiming result

Figure 20: ProbeShooter-A results for a multi-core system. (a) shows a single dominant process running on each core, while (b) and (c) show dominant processes running simultaneously on `Core1` and `Core2` (Exp.#A3: Multi-Core System).